Caché Release Notes and Upgrade Checklist Archive

Version 2018.1
2024-04-03

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

Caché Release Notes and Upgrade Checklist Archive
Caché Version 2018.1 2024-04-03

For Support questions about any InterSystems products, contact:

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

Table of Contents

List of Tables

About This Book

This book collects information on versions of Caché before the current release. This information includes both the new features in those versions and information needed for upgrading to those versions.

This book contains the following sections:

- Caché 2017.2
- Caché 2017.1
- Caché 2016.2
- Caché 2016.1
- Caché 2015.2
- Caché 2015.1
- Caché 2014.1
- Caché 2013.1
- Caché 2012.2
- Caché 2012.1
- Caché 2011.1
- Caché 2010.2
- Caché 2010.1
- Caché 2009.1
- Caché 2008.2
- Caché 2008.1
- Caché 2007.1
- Caché 5.2
- Caché 5.1
- Pre-2014.1 Upgrade Information

Important: Each of the release-specific checklists provides information on moving to that release from the previous release and a more detailed table of contents. Together these form a chronology of changes across releases. It is important to remember that items mentioned in the sequence may be invalidated by later changes. When an upgrade operation crosses multiple versions, it is important to read all the intervening material.

The following books provide related information:

- Caché Release Notes and Upgrade Checklist provides information on the current Caché release. The book includes the chapter General Upgrade Information.
- The online *InterSystems Supported Platforms* document for this release lists the technologies 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

Caché 2017.2

This chapter provides the following information for Caché 2017.2:

1.1 New and Enhanced Features for Caché 2017.2

This section includes:

1.1.1 Parallel Dejournaling For Mirroring and Journal Restore

In this release, the throughput of dejournaling is improved in Mirroring and Journal Restore. This is aimed at improving the scalability of mirrored systems that incur high rates of database updates.

Dejournaling is the process of applying records from journal files to the databases. Prior to this release, a single job performed all of the database updates in a dejournal session, aided by a series of prefetcher jobs. In this release, multiple updater jobs (up to four of them) can work in parallel to apply records to different databases. This feature is used if there are sufficient CPUs and shared memory heap available .

In Mirroring, use of parallel dejournaling can be restricted by member type with a new configuration setting. By default, the feature is off for Reporting Async members. If reports include data from multiple databases that is in the process of being updated by dejournaling, the differences in timing of updates to different databases could result in more variability in the results of those reports than in previous versions. The character of that variability is not substantially different than what could be expected in any report run against changing data, so it is expected that most reporting applications could use this feature with no negative impact.

For details on parallel dejournaling, see "Configuring Parallel Dejournaling" in the "Mirroring" chapter of the *Caché High Availability Guide*.

For journal restore, parallel dejournaling is not used if certain non-default options are selected. In particular, it is not used in conjunction with the option to abort after any error, nor with the option to journal applied updates. For more information, see "Restore Globals From Journal Files Using ^JRNRESTO" in the "Journaling" chapter of the *Caché Data Integrity Guide*.

Neither shadowing nor system startup make use of parallel dejournaling.

Finally, this release includes two improvements that apply regardless of whether parallel dejournaling is used. This release includes internal improvements to dejournal prefetching efficiency and it limits the memory consumption for the "dejournal queue" to 50MB per updater; prior releases may have used substantially more memory for the dejournal queue.

1.1.2 DeepSee New Features

DeepSee Folder Manager — To simplify the deployment of DeepSee components, there is a new Folder Manager option to export related items. When exporting to a container class, this new option will not only export the selected items but also other items that are related to the selected items. Related items for a dashboard include the pivot tables and termlists used by the dashboard. Related items for a pivot table include named filters, pivot variables, and shared calculated members.

Prior to this release the Folder Manager always used the server file system for exporting/importing files. Now folder manager provides the option of using the local file system or the server file system

Dashboard Filters — Named filters can now be used as the default value for dashboard filters.

1.1.3 iFind and iKnow New Features

iFind now supports cooccurrence search, allowing you to look for records where the search terms appear close to one another, but not necessarily in a particular order, as with the more strict positional search option. Use square brackets around a comma-separated list of search terms to enable cooccurrence search, optionally including a range within which the search terms need to occur. For example, the query [Boston, New York, 5] will look for records where "Boston" and "New York" appear within at most 5 positions of one another.

This change also includes a couple of performance improvements for building iKnow domains, especially when leveraging systems with high core counts. Depending on your hardware and dataset, you may see improvements from 10 to 30% in overall processing time.

There are a few extensions to the iKnow REST APIs. For example, you can now retrieve just the result count or just the count along with full results for most endpoints querying domain data.

1.1.4 SQL Enhancements

This release includes the following SQL enhancements:

- SQL Query Auditing
- Query Optimizations
- Optional ANSI SQL Operator Precedence
- Frozen Plan Evolution

1.1.4.1 SQL Query Auditing

This release adds the ability to audit the execution of SQL queries. There are three new system events for auditing SQL queries:

- %System/%SQL/DynamicStatement — for Dynamic SQL Queries
- %System/%SQL/EmbeddedStatement — for Embedded SQL Queries
- %System/%SQL/XDBCStatement - for xDBC Queries

To enable auditing for these events, go to the **System Audit Events** portal page by selecting **System Administration** > **Security** > **Auditing** > **Configure System Events**.

1.1.4.2 Query Optimizations

SQL queries have several improved optimizations. The query optimizer now considers outlier selectivity when calculating the selectivity of join conditions, resulting in improved plans for some queries. Outer joins can now take advantage of all optimizations available to inner joins. In particular, outer join conditions that could only partially be satisfied by an index no longer require the construction of a temporary file, often significantly improving query performance.

1.1.4.3 Optional ANSI SQL Operator Precedence

There is a new option to set SQL operator precedence to the ANSI SQL standard instead of the left-to-right CACHE SQL operator precedence. CACHE SQL operator precedence is still the default. You can change SQL operator precedence to ANSI SQL via an API or in the **General SQL Settings** portal page. Select **System Administration** > **Configuration** > **SQL and Object Settings** > **General SQL Settings**.

1.1.4.4 Frozen Plan Evolution

Previous releases of Caché included the ability to freeze SQL query plans and to automatically freeze query plans when you upgrade to a new version. If a new release provided improved optimization for a query plan, it would not be applied when you upgraded. With this release, Caché identifies any query with a frozen plan that might benefit from new optimizations and flags queries where the frozen plan may benefit from a new optimization. On the SQL Statement index page, these queries are identified in the New Plan column. You can then unfreeze these plans to take advantage of the new optimizations.

1.1.5 Where to Find Information on New Features in Atelier, the Eclipse-Based IDE

Atelier, the Eclipse-Based IDE for Caché, is available on an independent release cycle from Caché. Consequently, new features are described in the Atelier documentation provided with each new Atelier release. The Atelier IDE brings together the powerful and popular Eclipse development environment and the InterSystems Caché database. Atelier allows you to develop Caché applications using a modern file-based IDE on a client system. Atelier handles uploading the application to the Caché server where it can be run and debugged.

The focus of future development will be on the new Eclipse based IDE. Studio will remain an option to install and developers can continue to develop code with it. However, it will be treated as a maintenance product and will not see new functionality added as we move forward with Atelier. Some minor bugs may not be addressed either depending on resources required versus the severity of the issue.

Atelier is available as a separate download in addition to Caché or Ensemble. You can choose to install either a stand-alone Rich Client Platform (RCP) application, or a plug-in that can be added to an existing Eclipse installation. Users of the RCP application can add additional Eclipse plug-ins. Atelier uses the Eclipse auto-update mechanism to help users get the latest changes. For information on downloading Atelier and for the Atelier documentation, see http://www.intersystems.com/atelier, the Atelier home page.

1.1.6 Other Items of Note

In addition, many more minor improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

Areas of improvement include:

- XML Schema Wizard provides an option allowing cascading delete of subclasses when the instance of the superclass is deleted.
- Caché Nodejs supports node 7.
- New option to force CSP buffer to flush even if the buffer is not yet filled.

1.2 Caché 2017.2 Upgrade Checklist

Purpose

The purpose of this chapter is to highlight those features of Caché 2017.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2017.1 and 2017.2.

Important: If you are upgrading from Caché 2008.2 or an earlier version, you cannot upgrade directly to Caché 2017.2, but must first upgrade to a version between Caché 2009.1 and Caché 2016.2. For details, see Supported Upgrade Paths in the *Caché Installation Guide*.

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

1.2.1 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 2017.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

1.2.1.1 Operational Changes

This section details changes that have an effect on the way the system operates.

Changes to Permissions Required for Studio Access to a Namespace

In this release, Studio must have a login role that allows database access to the default database for the namespace in order to connect to a namespace. If Studio does not have such a role, it will get an error when it attempts to connect to the namespace. In previous releases, Studio access was allowed if the login role had any SQL privilege to the default database for the namespace.

Feature Tracker Enabled by Default for New Installs

Feature Tracker is a software utility in Caché that gathers statistics on software module usage. The statistics track whether or not software modules are present and used in a given Caché instance. Feature Tracker sends this information via https to InterSystems weekly. These statistics help us plan development and support. The information gathered does not include any application data. In previous releases, Feature Tracker was disabled by default. In this release, Feature Tracker is enabled by default for new installs. If you are upgrading an existing installation, the upgrade preserves the existing state of Feature Tracker.

If you wish to disable Feature Tracker, you suspend the task that runs it. For details, see "How to Deactivate Feature Tracker" in the *Caché System Administration Guide*.

Parallel Dejournaling Changes the Order of Dejournaling

This release includes parallel dejournaling, which makes it possible for up to four jobs to perform the dejournaling. A database is always dejournaled by a single job, so parallel dejournaling does not impact the order in which a single database is dejournaled. But it can change the order in which one database is dejournaled relative to the dejournaling in another database. This is because databases being updated by separate dejournaling jobs are likely to be at slightly different places in the dejournaling sequence; for example, database A may contain updates made on the primary at 11:45:30 when database B is only up to the updates from 11:45:28. For details, see "Configuring Parallel Dejournaling" in the "Mirroring" chapter of the *Caché High Availability Guide*.

Changes to Journal Buffer Pool Size

In this release, the journal buffer pool size is configurable and the default has been increased to 64MB. In previous releases, the default was 8MB on non-Unicode systems and 16MB on Unicode systems. An increase in this buffer pool size can improve efficiency, but it also increases the amount of data that may be lost if a system crash occurs.

1.2.1.2 Platform-Specific Items

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

There are no known platform-specific changes.

1.2.2 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.

1.2.2.1 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release:

- %Library.RunJava

Removed Methods

The following methods were present in the previous version and have been removed in this release:

- %DeepSee.UI.Dialog.ShowQuery
  - EncryptQuery
- %DeepSee.Utils
  - %ClearUpdateBuffer
- %Library.IResultSet
  - %ToDynamicArray
  - %ToDynamicObject
  - %ToJSONValue

- %Net.Remote.Utility
  - ClearPassphrase
  - GetPassphrase
  - RecordPassphrase
- %SQL.IResultSet
  - %ToDynamicArray
  - %ToDynamicObject
  - %ToJSONValue
- %SQL.StatementMetadata
  - %ToJSONValue
- Config.Shadows
  - SourceDatabasesClose
  - SourceDatabasesExecute
  - SourceDatabasesFetch

Removed Properties

The following properties were present in the previous version and have been removed in this release:

- %DeepSee.Component.Widget.controlPanel
  - isPdfEnabled
- %DeepSee.Component.Widget.widget
  - isPdfEnabled

Removed Parameters

No parameters have been removed in this release that were present in the previous version.

Removed Indices

No indices have been removed in this release that were present in the previous version.

Modified Methods

The following methods have different signatures in this version of Caché:

- %Atelier.v1.Utils.MetaData
  - old method: `Build (pDataBaseName:%String) As (none)`
  - new method: `Build (pDataBaseName:%String) As %Status`
- %CSP.UI.Portal.Dialog.EncAddAdmin
  - old method: `SaveData (What, File, Username1, Password1, Username2, Password2, Description) As %String`
  - new method: `SaveData (What, File, Username1, Password1, Username2, Password2, Description, KeyLen) As %String`

- %CSP.UI.Portal.ViewLog
  - old method: `DrawLogContent (filename) As %Status`
  - new method: `DrawLogContent (tmp) As %Status`
- %DeepSee.AbstractKPI
  - old method: `%GetKPIValue (pKPIName:%String, *pValue:%String, pKPIProperty:%String="", pSeries:%String="", &pFilters:%String, pCellContext:%String="", &pCacheKey:%String, *pPctComplete:%Integer, pParentQueryKey:%String="") As %Status`
  - new method: `%GetKPIValue (pKPIName:%String, *pValue:%String, pKPIProperty:%String="", pSeries:%String="", &pFilters:%String, pCellContext:%String="", &pCacheKey:%String, *pPctComplete:%Integer, pParentQueryKey:%String="", *pKPIStatus:%Status) As %Status`
- %DeepSee.Component.pivotTable
  - old method: `getFilterInfo (fnames, fvalues) As (none)`
  - new method: `getFilterInfo (fnames, fvalues, flabels) As (none)`
- %DeepSee.PMML.Model.AbstractModel
  - old method: `%ExecuteModelInternal (pInput:%DeepSee.PMML.ModelInput, *pOutput:%DeepSee.PMML.ModelOutput) As %Status`
  - new method: `%ExecuteModelInternal (&pInput:%DeepSee.PMML.ModelInput, *pOutput:%DeepSee.PMML.ModelOutput) As %Status`
- %DeepSee.PMML.Model.GeneralRegression
  - old method: `CalculateXBeta (pObservation:%DeepSee.PMML.ModelInput, *pXBeta, *pBestTarget:%String="", *pBestScore:%Double="", pAddZero:%String="") As %Status`
  - new method: `CalculateXBeta (&pObservation:%DeepSee.PMML.ModelInput, *pXBeta, *pBestTarget:%String="", *pBestScore:%Double="", pAddZero:%String="") As %Status`
- %DeepSee.ResultSet
  - old method: `%PrepareObject (pQuery:%DeepSee.Query.query) As %Status`
  - new method: `%PrepareObject (pQuery:%DeepSee.Query.query, &pVariables) As %Status`
- %DeepSee.Utils
  - old method: `%GetDimensionMembers (pCubeName:%String, pSpec:%String, pContext:%List, *pMembers, pMaxMembers:%Integer=100, *pMemberClass:%String, &pRelatedFilters, pCalcMode:%Integer=0, pSearchKey:%String="") As %Status`
  - new method: `%GetDimensionMembers (pCubeName:%String, pSpec:%String, pContext:%String="", *pMembers, pMaxMembers:%Integer=100, *pMemberClass:%String, &pRelatedFilters, pCalcMode:%Integer=0, pSearchKey:%String="") As %Status`
- %Exception.SystemException
  - old method: `%OnNew (pName:%String="", pCode:%String="", pLocation:%String="", pData:%String="", pInnerException:%Exception.AbstractException=$$$NULLOREF, pStack:%String="") As %Status`

  - new method: `%OnNew (pName:%String="", pCode:%String="", pLocation:%String="", pData:%String="", pInnerException:%Exception.AbstractException=$$$NULLOREF, pStack:%String) As %Status`

- %Library.GlobalEdit
  - old method: `GetGlobalSizeBySubscript (Directory:%String, StartNode:%String, EndNode:%String="", &Size:%String=0) As (none)`
  - new method: `GetGlobalSizeBySubscript (Directory:%String, StartingNode:%String, EndingNode:%String="", &Size:%String=0) As (none)`

- %Library.Routine
  - old method: `CompileAll (Flags:%String=0, IO:%String=$p, &Count:%Integer, &Errors:%Integer, MultiCompile:%Integer=1, Journal:%Integer=1, KeepSource:%Boolean=1) As %Status`
  - new method: `CompileAll (Flags:%String=0, IO:%String=$p, &Count:%Integer, &Errors:%Integer, MultiCompile:%Integer, Journal:%Integer, KeepSource:%Boolean=1) As %Status`
  - old method: `CompileSelected (Mask:%String="*.*", Flags:%String=0, IO:%String=$p, &Count:%Integer, &Errors:%Integer, MultiCompile:%Integer=1, Journal:%Integer=1, KeepSource:%Boolean=1) As %Status`
  - new method: `CompileSelected (Mask:%String="*.*", Flags:%String=0, IO:%String=$p, &Count:%Integer, &Errors:%Integer, MultiCompile:%Integer, Journal:%Integer, KeepSource:%Boolean=1) As %Status`

- %Library.SQLCatalog
  - old method: `SQLClassname (qh:%Library.SQLProcContext, table:%String) As %Library.String`
  - new method: `SQLClassname (qh:%Library.SQLProcContext, table:%String(MAXLEN=257)) As %Library.String`

- %SYSTEM.SQL
  - old method: `SetSQLStats (flag:%Library.Integer=1) As %Library.Integer`
  - new method: `SetSQLStats (flag:%Library.Integer=0) As %Library.Integer`

- %ZEN.ComponentEx.svgImageProvider
  - old method: `dumpSVGNode (e, svgDoc, src, intro, coda, NSPrefix, maxWidth) As (none)`
  - new method: `dumpSVGNode (e, svgDoc, src, intro, coda, NSPrefix, maxWidth, maxHeight) As (none)`
  - old method: `extractXSLFOSource (svg, intro, coda, maxWidth) As (none)`
  - new method: `extractXSLFOSource (svg, intro, coda, maxWidth, maxHeight) As (none)`

- %ZEN.Template.AddInWizard.SOAPWizard
  - old method: `GetSRC (filetype:%String, url:%String, sslConfig:%String="", username:%String="", password:%String="") As %String`

  - new method: `GetSRC (filetype:%String, url:%String, sslConfig:%String="", sslCheckServerId:%Boolean, username:%String="", password:%String="") As %String`
  - old method: `SaveLast (filetype, url, sslConfig="") As (none)`
  - new method: `SaveLast (filetype, url, sslConfig="", sslCheckServerId) As (none)`

- SYS.Stats.WriteDaemon
  - old method: `Sample (DaemonID:%Integer) As SYS.Stats.Resource`
  - new method: `Sample (DaemonID:%Integer=1) As SYS.Stats.WriteDaemon`

1.2.2.2 Class Compiler Changes

Hard Errors in delete() are Reported with %Status Value

In previous releases, if the delete() method in the class compiler encountered a hard error, it would throw the error to the caller. In this release, these errors are handled by the delete() method, which reports the error to the caller with a %Status value.

1.2.2.3 DeepSee Changes

Change Default to Display Mode in DeepSee Listing Selection

In previous releases, if a DeepSee field was not explicitly declared %EXTERNAL(field) or %INTERNAL(field), DeepSee would display its internal value in the listing output. In this release DeepSee will use its external value. To have the internal value used, you must wrap the field in %INTERNAL().

1.2.2.4 Java and Gateway Changes

Do not use JMS Gateway as Message Listener

This release removes the JMSGateway class from the Java Gateway. This class was intended as an example on how to use the JMS listener in Ensemble. If you have used this class to implement a JMS listener, you should replace this with the mechanism demonstrated in the EnsLib.JavaGateway.JMSTest class.

Object Gateway Load^%apiGTW Does Not Make err Variable Public

In previous releases, the call to Load^%apiGTW would deliberately make the variable `err` public so that it could be used in the Management Portal and other InterSystems APIs. In this release, it does not automatically make it public but instead returns a status value. If your code depended on this previous behavior, you must explicitly define `err` as a public variable by calling:

```
Set err=$$Load^%apiGTW
```

1.2.2.5 .NET Language Bindings

Replace SysList, SysListUtil, and CacheListRO Classes Used With ODBCx APIs

If you use the SysList, SysListUtil, and CacheListRO Classes directly with the ODBCx APIs, replace them with other classes that provide similar functionality. Although the classes have been deprecated, they have not yet been deleted. If you use these classes but not use them with any other ADO APIs, you can defer replacing them until it is convenient. But if you use them with the ODBCx APIs, you must replace them before using with this release.

1.2.2.6 Object Library

%RunJava Utility Class Removed

In this release, the %RunJava utility class has been removed. If you call methods in this class, you must replace them with calls to the utility methods in %Net.Remote.Service.

1.2.2.7 ObjectScript Changes

Limit Number of Subscripts for Indirection

In this release, applications are limited to 254 subscripts for indirection. It is unlikely that there is any existing code doing indirection with more than 254 subscripts. Such code would create nodes that are not easily accessible.

1.2.2.8 SQL Changes

Changes in API to Create QuickStatement

The QuickStatement class syntax has changed and the class was moved from cache-jdbc to cache-db jar. The new syntax is

```
QuickStatement qs = QuickStatement.createQuickStatement((CacheConnection) connection);
```

If you use the previous syntax, "`QuickStatement qs = (QuickStatement) rs.getObject("**QuickStatement**");`", you must update your code to the new syntax.

Changes in %Date LogicalToOdbc and LogicalToDisplay

In previous releases if a timestamp value was passed through %Date LogicalToOdbc or LogicalToDisplay, it was not altered. As part of a fix to support dates prior to December 31, 1840, this behavior has changed. In this release if a timestamp value was passed through %Date LogicalToOdbc or LogicalToDisplay, it removes the time portion of the value and returns only the date portion. If your application logic depends on the previous behavior, you may need to modify your code. For example, if you query data in Display or Odbc mode from a table linked to Oracle and use a function like CAST(field AS DATE) or TO_DATE(...), Oracle returns a datetime value. In this release, Caché converts it to a DATE.

Enforce REFERENCES Privileges for Foreign Keys

In previous releases, it was possible to create a foreign key constraint through a DDL statement without holding the REFERENCES privilege for the referenced table. In this release, for a user to define a foreign key constraint that references table T, the user must hold the REFERENCE privilege on table T, or the REFERENCE privilege on the column(s) of table T that are being referenced by the foreign key. If an application creates foreign keys without holding REFERENCES privileges, it will encounter errors.

1.2.2.9 System Changes

Change in oref Numbers Ordering

In this release, we have improved the efficiency of processing OREFs. As a consequence, the OREF number assignments, which were previously assigned in the order created are now assigned in an independent way. Consequently, if your code relies on the way in OREF numbers were assigned, you must modify the logic. The previous ordering was an implementation detail and was not a documented feature. For example, if OREF string values are used to index a Caché multidimensional array, the subscript order of the OREF string values will no longer have any relation to the order in which those OREF values were created. If an application is using OREF value order to control order of execution or order of data handling, then that application will need to be modified to use other techniques, like $INCREMENT or $SEQUENCE, to generate the desired ordering.

Similarly, this change affects the order of items in relationships (for example, the order of children within a parent). Note, however, that it is not supported to rely on the order of items in relationships.

Console Log Changes

In some cases, long console log messages that were broken into two messages in previous releases will now be written as a single message. If you have code that parses for specific messages in the console log, you should update it.

1.2.2.10 Web Services

SOAP Wizard Includes Option to Generate Shorter Array Names

In previous releases, the default type name for an array item includes both the item name and type name or the key name and type name, even when the names are identical. In this release, by default, if the item name and type name are the same, only one is included in the default array type name. Similarly, if the key name and type name are identical, only one is included in the default array type name. For example, in previous releases, an XML schema (or a WSDL for a web service) might include something like the following:

```
<element minOccurs="0" name="PropName" type="s01:ArrayOfSimpleObjectSimpleObject" xmlns:s01="mytypes"/>
```

As of this release, the schema or WSDL will instead include this:

```
<element minOccurs="0" name="PropName" type="s01:ArrayOfSimpleObject" xmlns:s01="mytypes"/>
```

In some cases, you can revert to the previous default behavior:

- The AllowRedundantArrayName property of %XML.Schema allows the old form in an XML schema. If true, the previous longer type names are generated.
- The *ALLOWREDUNDANTARRAYNAME* parameter of a web service class allows the old form of array name in the XML schemas in the WSDL for the web service. If this parameter is true, then the previous longer type names are generated.
- There is no way to restore the previous default of long array names, if you are generating the schema by using the **XMLSchema()** method of %XML.Adaptor. The schema would need to be created using the %XML.Schema class.

Change to Invoking Web Methods Using HTTP Requests

In previous releases you could call a Caché SOAP web service directly without making an HTTP SOAP request. This shortcut avoids using a SOAP client but it bypasses the security available using HTTP SOAP requests. This shortcut is used by the catalog and test pages that Caché generates for a SOAP web service class. InterSystems recommends that you not use this shortcut but always use a SOAP web client to generate an HTTP SOAP request to call Caché SOAP web services. If you have code that uses the %SOAP.WebServiceInvoke class in a URL to invoke a SOAP web service, InterSystems recommends that you replace this with an HTTP SOAP request generated by a SOAP client.

Although it is not recommended, you can continue to use this shortcut mechanism, but you must explicitly enable this access. To use the %SOAP.WebServiceInvoke class and the soap_method query parameter, you must:

- Enable %SOAP.WebServiceInfo and WebServiceInvoke with commands such as:

  ```
  set ^SYS("Security","CSP","AllowClass",webapplicationname,"%SOAP.WebServiceInfo")=1
  set ^SYS("Security","CSP","AllowClass",webapplicationname,"%SOAP.WebServiceInvoke")=1
  ```

  Where *webapplicationname* is the web application name with a trailing slash, for example, `"/csp/mynamespace/"`.

- Ensure that the user accessing the CSP page has USE permission for the %Development resource.

1.2.2.11 XML Changes

XSLT2 Transformation Returns Error Status for Fatal Errors

In previous releases, when an XSLT2 transformation encountered a fatal error, it returned $$$OK. In this release, it correctly returns an error status. If your code relies on the default error handling, you may have to modify it to handle this change.

If you have overridden the default with custom error handling, the custom error handling will continue to work as it did in previous releases.

Change in XML Output for Property with Element Qualified

In previous releases, if a class with ELEMENTQUALFIED=1 had a property which is a literal datatype with ELEMENTQUALIFIED, the property was not output with a namespace specified. This change corrects this, and, in Caché 2017.2 and later releases, the property is output by XMLExport or %XML.Writer with a namespace qualification.

For example, in previous releases, the XML output for a property would be:

```
<MyProp xmlns="">0
```

In Caché 2017.2, the XML output for this property is:

```
<s01:MyProp xmlns="" xmlns:s01="http://my.namespace.com/test">
```

2

Caché 2017.1

This chapter provides the following information for Caché 2017.1:

- New and Enhanced Features for Caché 2017.1
- Caché 2017.1 Upgrade Checklist

2.1 New and Enhanced Features for Caché 2017.1

This section includes:

- FIPS 140-2 Validated Cryptography for Caché Database Encryption
- Enhancements to Support for OAuth 2.0 and OpenID Connect
- DeepSee Improvements
- Mirroring Improvements
- Improved DocBook Search
- Feature Tracker
- iKnow REST API and Other Enhancements
- New Features in Atelier, the Eclipse-Based IDE
- Other Items of Note

2.1.1 FIPS 140-2 Validated Cryptography for Caché Data-at-Rest Encryption

This release allows you to configure Caché running on Red Hat Enterprise Linux 64-bit to use a FIPS 140-2 validated library for Caché data-at-rest encryption. For details, see the article "FIPS 140–2 Compliance for Caché Database Encryption".

2.1.2 Enhancements to Support for OAuth 2.0 and OpenID Connect

Caché support for OAuth 2.0 and OpenID Connect now includes the following features:

- Support for discovery (OpenID Connect Discovery 1.0), as described in https://openid.net/specs/openid-connect-discovery-1_0.html. When you configure a client, the Management Portal provides a new discovery option that you can

use rather than manually entering all the information about the authorization server. The %SYS.OAuth2.Registration class now provides the **Discover()** method.

- Support for dynamic client registration (OpenID Connect Dynamic Client Registration), as described in http://openid.net/specs/openid-connect-registration-1_0-19.html. When you configure a client, the Management Portal provides a new registration option that you can use rather than manually entering all the information about the client on the authorization server. The %SYS.OAuth2.Registration class now provides the **RegisterClient()** method.
- Support for JSON Web Key (JWK) and JSON Web Key Sets (JWKS), as described in https://data-tracker.ietf.org/doc/rfc7517, for encrypting, decrypting, signing, and verifying signatures of JSON web tokens (JWTs).

  Because JWKS provides a means of representing a public/private key pair, it is no longer necessary to use certificate/private key pairs for these purposes. The option to use certificate/private key pairs is still included for backward compatibility.
- Key rotation (in the cases when you are using JWKS). The OAuth 2.0 configuration pages of the Management Portal provide options to rotate keys; these options add a new key pair to the JWKS and save the JWKS.
- Management Portal option to revoke tokens for a given user. This option is provided for convenience; it supplements the existing client API.

All features that are supported in 2016.2 will interoperate with 2017.1. For example, you can have a client and an authorization server at different Caché versions. The new features, however, will not work with a partner that is at version 2016.2.

In release 2017.1, there are changes in the configuration classes for OAuth 2.0 and OpenID Connect. Caché automatically updates your saved configurations.

2.1.3 DeepSee Improvements

This release includes the following DeepSee improvements:

- New controls widget for dashboard.
- Improved export of DeepSee components.
- New DayOfWeek time function.
- New CreateTable plug-in.
- Improvements to setting default filter values.
- Improvements to setting advanced filter values.
- Drilldown expression now uses expressionBuilder.
- SQL-based listings now support dynamic sorting.

2.1.4 Mirroring Improvements

This release includes the following Mirroring improvements:

- Significant performance enhancement on backup and async members for writes to the local copy of mirror journal files for updates received from the primary. This has shown as much as a 10 times improvement in journal transfer throughput, thereby extending scalability of mirrored systems. Systems utilizing a backup failover member may also see improved application performance and responsiveness on the primary as a result of these improvements.
- Promotion for Disaster Recovery async members has been enhanced. When promoting a Disaster Recovery member in a disaster scenario, before becoming primary, the promoted member now surveys all other reachable mirror members to retrieve any newer journal data that those members may have retrieved just prior to the disaster. This ensures that the Disaster Recovery cutover includes all available data and that other surviving async members can rejoin the new

primary. Additionally, an operational step is supported to allow you to perform any manual validation you wish on the promoted member before allowing it to become primary.

- ccontrol list and ccontrol qlist now include mirror member type and status.

2.1.5 Improved DocBook Search

DocBook search has improvements that can help you find information faster. Search now puts matches found in the most commonly used documents first. The DocBook search page has new options that allow you to limit the scope of your search. To reach the DocBook search page, select **Search Page** from the documentation home page. It has the following new filters:

- Find topics with *all* or *any* of the specified words. In previous releases, search was always for *all* words.
- Use free-text search or exact match with or without case sensitivity. In previous releases, search always used free-text search, which removed some special symbols and searched for all words that shared the same root. Exact match retains special symbols and only finds the exact word specified. For example, with the search word "^global", free text search finds all occurrences of the word "global", but exact match searches will only find "^global" with the caret character.
- Limit search to product allows you to search for words in either the Caché or Ensemble documentation.
- Search in a specific book.
- Search for items associated with a tag. This allows you to search for words tagged with terms such as DeepSee, Zen, HL7, and BPL.

2.1.6 Feature Tracker

In 2017.1 we are introducing a safe and trustworthy approach for InterSystems to periodically collect information about the Caché features being used while maintaining the integrity of our relationship with our clients. The Feature Tracker is controlled by a task. By default the task is suspended and Feature Tracker is disabled, but you can easily resume the task. Feature Tracker will help us prioritize which Caché features to enhance based on the features that are used by customers.

The Feature Tracker collects information about instance attributes (product type, version, license, platform, etc.) and use of technology, including ECP, Mirroring, DB Encryption, and SQL. It does not collect information about license utilization, database attributes, applications, errors, authentication, client data, client-specific configuration. Weekly, the Feature Tracker sends the collected data (XML file) to an InterSystems Caché instance using SSL. You may view the most recent collected data to see that this information is trustworthy. If for any reason the Feature Tracker cannot transmit, it fails silently with no impact to the instance.

More information about Feature Tracker can be found in Feature Tracker Collects Usage Statistics, in the *Caché System Administration Guide*.

Note: During field test, the Feature Tracker task is active and Feature Tracker is enabled.

2.1.7 iKnow REST API and Other Enhancements

This release introduces a REST API to access iKnow domain information directly from a RESTful client. Many common query API methods are supported through a simple, consistent interface where individual requests can be configured to retrieve rich query results or just the basic data, depending on your application requirements. The iKnow REST API provides reference documentation using the Swagger Specification, which is the foundation of the OpenAPI Specification.

Note: One method to view the iKnow REST API documentation, is to use Swagger UI, either by directing a web browser to http://petstore.swagger.io/ or by downloading the Swagger UI toolkit. Enter one of the following URLs in the Swagger UI form and then select the **Explore** button. For a Caché development instance with minimal security, you can enter:

`http://localhost:`*port-number*`/api/iknow/v1/user/swagger`

For a Caché instance with password security, you can enter:

`http://localhost:`*port-number*`/api/iknow/v1/user/swagger?CacheUserName=`*user-name*`&CachePassword=`*password*

In addition, this release includes a significant overhaul of our Knowledge Portal demo interface for exploring your domain's contents and a number of extensions to the iKnow Architect interface for managing a domain definition.

2.1.8 New Features in Atelier, the Eclipse-Based IDE

Atelier, the Eclipse-Based IDE for Caché, is available on an independent release cycle from Caché. Consequently, new features are described in the Atelier documentation provided with each new Atelier release. The Atelier IDE brings together the powerful and popular Eclipse development environment and the InterSystems Caché database. Atelier allows you to develop Caché applications using a modern file-based IDE on a client system. Atelier handles uploading the application to the Caché server where it can be run and debugged.

The focus of future development will be on the new Eclipse based IDE. Studio will remain an option to install and developers can continue to develop code with it. However, it will be treated as a maintenance product and will not see new functionality added as we move forward with Atelier. Some minor bugs may not be addressed either depending on resources required versus the severity of the issue.

Atelier is available as a separate download in addition to Caché or Ensemble. You can choose to install either a stand-alone Rich Client Platform (RCP) application, or a plug-in that can be added to an existing Eclipse installation. Users of the RCP application can add additional Eclipse plug-ins. Atelier uses the Eclipse auto-update mechanism to help users get the latest changes. For information on downloading Atelier and for the Atelier documentation, see http://www.intersystems.com/atelier, the Atelier home page.

2.1.9 Other Items of Note

In addition, many more minor improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in "Caché 2017.1 Upgrade Checklist."

Areas of improvement include:

- OpenSSL library—In this release the OpenSSL library has been upgraded to v1.0.2h.
- Thai (Buddhist Era) date format support extended and enhanced. Please consult the Reference guide for the functions $ZDATEH() and $ZDATETIMEH() for details.
- Lightweight statistics provide information on SQL query performance.
- SQL Read Verified provides access to uncommitted changes as Read Uncommitted does but also rechecks query conditions to verify that they are still met.
- Improved work queue manager.
- Improvements to the XSLT Gateway.
- Improvements to the JDBC driver including a faster and more robust preparser.
- Support for .NET 4.5 in .NET Gateway, Caché eXtreme, and ADO.NET.

2.2 Caché 2017.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2017.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2016.2 and 2017.1.

Important: If you are upgrading from Caché 2008.2 or an earlier version, you cannot upgrade directly to Caché 2017.1, but must first upgrade to an intermediate version. For details, see Supported Upgrade Paths in the *Caché Installation Guide*.

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

2.2.1 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 2017.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

2.2.1.1 Operational Changes

This section details changes that have an effect on the way the system operates.

Increased Default Frame Stack Memory Use on Non-Unicode Systems

In this release, we have increased the frame stack size for non-Unicode instances. This change reduces the possibility that an SQL query that works on a Unicode instance would encounter a frame stack error on a non-Unicode system. This increase typically does not have a large impact on system memory usage, but may cause problems on systems with limited memory. If you encounter problems, you can return to the previous allocation for frame stack size by calling the SetFrameStackSize method during system startup.

On a 64–bit system, enter the following:

```
Do $SYSTEM.Util.SetFrameStackSize(1024*112)
```

On a 32–bit system, enter the following:

```
Do $SYSTEM.Util.SetFrameStackSize(1024*72)
```

Must Purge Journal Log if Re-Adding Failover Member to Mirror Set

In previous releases, when a failover member was removed from a mirror set, the journal.log was deleted for the failover member. In this release, this file is not deleted. Consequently, you must run the purge utility before adding the failover member back to the same mirror set. This situation is very unusual for production systems, but you may encounter it on a test system.

Change to Stop Mirroring behavior

Mirroring provides a capability to Stop Mirroring on this member, which causes a non-primary member to temporarily disconnect from the primary, stop dejournaling, etc.

In previous versions mirroring also implicitly restarts (reconnects) if the Caché instance restarts. With this version we change this behavior and users would have to manually start mirroring (reconnect) using either ^MIRROR or the management portal.

Changes in Console Log Messages

In this release the console log contains messages when databases are mounted and unmounted. In addition, some messages have had their wording changed.

Change in How Ensemble Compiles HL7.2 Schemas

In Ensemble 2016.2, Ensemble changed the way it imported and compile HL7.2 schemas. In previous versions, if you imported an HL7.2 schema while the schema was actively being used by a business host, the host could encounter an error during the period between when the old schema was deleted and the new one compiled. The 2016.2 change avoided this error, but, unfortunately, the 2016.2 change caused other problems and Ensemble 2017.1 reverts it. With Ensemble 2017.1 and future versions of Ensemble, the existing HL7.2 schema is first deleted and then the new one is compiled.

With this release the HL7.2 compilation behavior reverts to its behavior in Ensemble 2016.1 and earlier. That is, you must stop any host items that use the HL7.2 schema before replacing the HL7.2 schema with a new version. You can replace a schema either by compiling the schema or by using the Management Portal HL7 schema editor.

Changes in Location of Java JAR Files

The location and organization of the JAR files has changed in this release. See Java and Gateway Changes for details on these changes.

/csp/user Application Settings are Left Unchanged by Upgrade

In previous releases, installing a Caché upgrade would overwrite custom /csp/user application settings and you would have to reapply the settings. In this release, the upgrade does not change these custom settings. If you have automated the process of re-applying these settings after an upgrade, you can remove this part of the process.

2.2.1.2 Platform-Specific Items

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

UNIX®

- Change in License Type for SUSE and Ubuntu systems

  In previous releases a single license type worked for 64–bit Red Hat, SUSE, and Ubuntu systems. This license continues to work for Red Hat systems, but if you have a SUSE or Ubuntu system, you should contact the InterSystems WRC to get a replacement license before upgrading to this release.

- Install Prompt Changes

  The UNIX install no longer prompts you for loading the Manager utility source code in a custom install. It always loads this code. If you have a script that runs the install, you may need to modify it so that it does not attempt to respond to the missing prompt.

- Java File Location Changed in Distribution Kit

  This change only impacts file locations in the distribution kit, not in the installed product. Java client install information is now provided as java_client.tar.gz compressed archive instead of a flat file structure under dev/java. If you have scripts that directly access these files in the distribution kit, you will have to modify the script.

- Solaris UFS Users Can Improve Journal Performance with ZFS

  For optimal performance, do not use the UFS file systems for journal files on Oracle Solaris platforms for journal. We strongly recommend that you use ZFS for journaling.

- Improved Error Detection in UNIX Installation

  In Caché 2017.1.2 and later releases, the UNIX installation has improved error detection. If your environment relies on the previous behavior, that is the installation continuing through errors, you can return to this behavior by setting "ISC_PACKAGE_SHELL_COMMAND=sh" before running the install. This disables error checking.

2.2.2 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.

2.2.2.1 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release:

- %Library.ListOfPersistentChildObjects
- %iKnow.Objects.SQLUtils
- %iKnow.Objects.StoredProcBase
- %iKnow.ont.temp.SearchResult
- Ens.Network

Removed Methods

The following methods were present in the previous version and have been removed in this release:

- %CSP.UI.Portal.Applications.Web
  - enableLDAP
- %CSP.UI.Portal.OAuth2.Client.Configuration
  - showJWT
- %OAuth2.Server.JWT
  - CreateJWT
- %SYS.OAuth2.Validation
  - JWTToObject
- %SYSTEM.iKnow
  - mapToUMLS
- %Stream.FileBinary
  - GetReadOpenParameter
- %Stream.FileCharacter
  - GetReadOpenParameter
- %UnitTest.TestSqlScript
  - runSqlStatement
- %ZEN.Template.AddInWizard.SOAPWizard

  - changePersistent
- %iFind.Utils
  - RebuildFullIndicesInNamespace
- %iKnow.UI.KnowledgePortal
  - BuildSummary
  - DrawCC
  - DrawCRC
  - DrawDrillCC
  - DrawDrillCRC
  - DrawPath
  - DrawSource
  - DrawSourceDrill
  - DrillToSources
  - GetInfoPaneText
  - InitPages
  - MakeLink
  - OnChangeLanguageMode
  - RefreshRadialContent
  - buildSummaryClient
  - displaySourceClient
  - drillToSourcesClient
  - onUpdateFilterClient
  - onUpdateTermClient
  - refreshRadial
  - selectNodeClient
  - toggleRelated
  - toggleTop
  - useTermClientRow
- Security.SSLConfigs
  - GetCRLFile
- OAuth2.Server.AccessToken
  - IsOpenID
- OAuth2.Server.Client
  - Delete

Removed Properties

The following properties were present in the previous version and have been removed in this release:

- %CSP.UI.Portal.Dialog.SQLStatementDetails
  - FrozenSettingHasChanged
- %CSP.UI.Portal.JDBCGatewayServer
  - JAVAHOMEValue
- %CSP.UI.Portal.ObjectGateway
  - JAVAHOMEValue
- %CSP.UI.Portal.XSLTGatewayServer
  - JAVAHOMEValue
  - XSLTGatewayJDKVersion
- %CSP.UI.Portal.ZenReportPrintServer
  - hintForAdobe
- %Net.Remote.ObjectGateway
  - JDKVersion
- %iKnow.Queries.SourceWSAPI.GetSummary
  - domainid
  - sourceid
- %iKnow.UI.KnowledgePortal
  - drillMaster
  - drillRelation
  - drillSlave

Removed Parameters

The following parmeters were present in the previous version and have been removed in this release:

- %Net.Remote.Java.XSLTGateway
  - JDKVersion
- %UnitTest.TestScript
  - CORRELATIONLIST
  - DATACLASS
  - DATAFILE
  - DATATAG
  - SHOWPLAN
- OAuth2.Server.REST

  - CONTENTTYPE

Removed Indices

The following indices were present in the previous version and have been removed in this release:

- OAuth2.Server.Client
  - NameIndex

Modified Methods

The following methods have different signatures in this version of Caché:

- %CSP.UI.Portal.Dialog.MirrorPromote
  - old method: `PromoteToFailover (virtualinterface, partner, standalone) As %Integer`
  - new method: `PromoteToFailover (virtualinterface, partner, standalone, NoFailover) As %Integer`
- %CSP.UI.Portal.JDBCGatewayServer
  - old method: `CheckForChanges (pOnlyPort:%Boolean=0, pPort:%String="", pLogFile:%String="", pJavaHome:%String="") As %Boolean`
  - new method: `CheckForChanges (pOnlyPort:%Boolean=0, pPort:%String="", pLogFile:%String="", pJavaHome:%String="", pUsePassphrase:%String) As %Boolean`
  - old method: `StartJDBCGateway (pHost:%String, pPort:%String) As %String`
  - new method: `StartJDBCGateway (pHost:%String, pPort:%String, pUsePassphrase:%String) As %String`
- %CSP.UI.Portal.Mirror.Dialog.SSL
  - old method: `SaveData (CAFile, CRLFile, CertificateFile, PrivateKeyFile, PrivateKeyType, PrivateKeyPassword, Protocols1, Protocols2, Protocols4, Protocols8, Protocols16, CipherList, pwModified, PWOptions) As %ZEN.proxyObject`
  - new method: `SaveData (CAFile, CRLFile, CertificateFile, PrivateKeyFile, PrivateKeyType, PrivateKeyPassword, Protocols2, Protocols4, Protocols8, Protocols16, CipherList, pwModified, PWOptions) As %ZEN.proxyObject`
- %CSP.UI.Portal.OAuth2.Client.Configuration
  - old method: `LoadData (PID:%String) As %Status`
  - new method: `LoadData (PID:%String, sys:OAuth2.Client="") As %String`
- %CSP.UI.Portal.OAuth2.Client.ConfigurationList
  - old method: `doEdit (ApplicationName) As (none)`
  - new method: `doEdit (ApplicationName, IssuerEndpointID, IssuerEndpoint) As (none)`
  - old method: `doNew () As (none)`
  - new method: `doNew (IssuerEndpointID, IssuerEndpoint) As (none)`
- %CSP.UI.Portal.OAuth2.Client.ServerConfiguration
  - old method: `LoadData (PID:%String) As %Status`

  - new method: `LoadData (PID:%String, *sys) As %Status`
  - old method: `doSave () As (none)`
  - new method: `doSave (isDiscovery) As (none)`

- %CSP.UI.Portal.OAuth2.Server.Client
  - old method: `SaveData (Name, ClientId, ClientSecret, ClientType, DefaultScope, Description, RedirectURL, SupportedGrantTypes, LaunchURL, ClientName, LogoUri, ClientUri, PolicyUri, TosUri, ClientCredentials) As %ZEN.proxyObject`
  - new method: `SaveData (pValueProxy:%ZEN.proxyObject) As %ZEN.proxyObject`
- %CSP.UI.Portal.ObjectGateway
  - old method: `SaveData (PID, Type, Name, Server, Port, LogFile, HeartbeatInterval, HeartbeatFailureTimeout, HeartbeatFailureAction, HeartbeatFailureRetry, InitializationTimeout, ConnectionTimeout, J1, J2, J3, J4) As %ZEN.proxyObject`
  - new method: `SaveData (PID, Type, Name, Server, Port, UsePassphrase, LogFile, HeartbeatInterval, HeartbeatFailureTimeout, HeartbeatFailureAction, HeartbeatFailureRetry, InitializationTimeout, ConnectionTimeout, J1, J2, J3, J4) As %ZEN.proxyObject`
- %CSP.UI.Portal.SSL
  - old method: `SaveData (PID, isTest, Name, Description, Enabled, Type, VerifyPeer, CAFile, CRLFile, CertificateFile, PrivateKeyFile, PrivateKeyType, PrivateKeyPassword, Protocols1, Protocols2, Protocols4, Protocols8, Protocols16, CipherList, TestHost, TestPort, pwModified, PWOptions) As %ZEN.proxyObject`
  - new method: `SaveData (PID, isTest, Name, Description, Enabled, Type, VerifyPeer, CAFile, CRLFile, CertificateFile, PrivateKeyFile, PrivateKeyType, PrivateKeyPassword, Protocols2, Protocols4, Protocols8, Protocols16, CipherList, TestHost, TestPort, pwModified, PWOptions) As %ZEN.proxyObject`
- %CSP.UI.Portal.XSLTGatewayServer
  - old method: `CheckForChanges (pOnlyPortOrHost:%Boolean=0, pHost:%String, pPort:%String, pJDKVersion:%String, pLogFile:%String, pJavaHome:%String, pJVMArgs:%String) As %Boolean`
  - new method: `CheckForChanges (pOnlyPort:%Boolean=0, pPort:%String, pLogFile:%String, pJavaHome:%String, pJVMArgs:%String, pUsePassphrase:%String) As %Boolean`
  - old method: `StartXSLTGateway (pHost:%String, pPort:%String, pJDKVersion:%String, pLogFile:%String, pJavaHome:%String, pJVMArgs:%String) As %String`
  - new method: `StartXSLTGateway (pHost:%String, pPort:%String, pLogFile:%String, pJavaHome:%String, pJVMArgs:%String, pUsePassphrase:%String) As %String`
- %DeepSee.CubeManager.RegistryMapGroup
  - old method: `BuildGroup (pGroupIndex="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0, pUpdateAll=0) As %Status`

  - new method: `BuildGroup (pGroupIndex="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0, pUpdateAll=0, &pBuildStats) As %Status`
  - old method: `SynchronizeGroup (pGroupIndex="", pVerbose:%Boolean=1, *pFactsUpdated:%Integer, pUpdateAll=0) As %Status`
  - new method: `SynchronizeGroup (pGroupIndex="", pVerbose:%Boolean=1, *pFactsUpdated:%Integer, pUpdateAll=0, &pSynchronizeStats) As %Status`

- %DeepSee.CubeManager.Utils
  - old method: `BuildAllRegisteredGroups () As %Status`
  - new method: `BuildAllRegisteredGroups (pMap:%DeepSee.CubeManager.RegistryMap="", &pBuildStats) As %Status`
  - old method: `BuildCube (pCubeName:%String="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0, pRepair:%Boolean=0) As %Status`
  - new method: `BuildCube (pCubeName:%String="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0, pRepair:%Boolean=0, pMapCube:%DeepSee.CubeManager.RegistryMapCube="", &pBuildStats) As %Status`
  - old method: `BuildOneRegisteredGroup (pGroupName="") As %Status`
  - new method: `BuildOneRegisteredGroup (pGroupName="", pMap:%DeepSee.CubeManager.RegistryMap="", &pBuildStats) As %Status`
  - old method: `GetDependentCubes (pCubeName="", pMap:%DeepSee.CubeManager.RegistryMap="", *pDependentCubes) As %Status`
  - new method: `GetDependentCubes (pCubeName="", pMap:%DeepSee.CubeManager.RegistryMap="", *pDependentCubes, &pGroups, &pCubes, &pBuildOrders) As %Status`
  - old method: `IsValidGroup (&pGroup:%DeepSee.CubeManager.RegistryMapGroup, &pStatus:%Status) As %Boolean`
  - new method: `IsValidGroup (&pGroup:%DeepSee.CubeManager.RegistryMapGroup, &pStatus:%Status, &pGroups, &pCubes, &pBuildOrders) As %Boolean`
  - old method: `RepairBuild (pCubeName:%String, tMap:%DeepSee.CubeManager.RegistryMap="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0) As %Status`
  - new method: `RepairBuild (pCubeName:%String, pMap:%DeepSee.CubeManager.RegistryMap="", pAsync:%Boolean=1, pVerbose:%Boolean=1, pIndexOnly:%Boolean=0, pMaxFacts:%Integer=0, pTracking:%Boolean=0) As %Status`
  - old method: `SynchronizeCube (pCubeName:%String="", pVerbose:%Boolean=1, *pFactsUpdated:%Integer, pReadCommitted:%Boolean=1, pCheckReferences:%Boolean=1, pAsync:%Boolean=0) As %Status`
  - new method: `SynchronizeCube (pCubeName:%String="", pVerbose:%Boolean=1, *pFactsUpdated:%Integer, pReadCommitted:%Boolean=1, pCheckReferences:%Boolean=1, pAsync:%Boolean=0, pMapCube:%DeepSee.CubeManager.RegistryMapCube="", &pSynchronizeStats) As %Status`

- %DeepSee.UserLibrary.Utils
  - old method: `%Export (pFullName:%String, pFile:%String="", pVerbose:%Boolean=1) As %Status`
  - new method: `%Export (pFullName:%String, pFile:%String="", pVerbose:%Boolean=1, *pXMLName:%String) As %Status`
  - old method: `%ImportContainer (pClass:%String="", pReplace:%Boolean=0, pVerbose:%Boolean=1) As %Status`
  - new method: `%ImportContainer (pFileName:%String="", pReplace:%Boolean=1) As %Status`
- %Installer.Component
  - old method: `ConfigureComponent (pDB="", pNamespace="", pVerbose=0) As %Status`
  - new method: `ConfigureComponent (pDB="", pNamespace="", pVerbose=0, &pVars) As %Status`
  - old method: `UnconfigureComponent (pDB="", pNamespace="", pVerbose=0) As %Status`
  - new method: `UnconfigureComponent (pDB="", pNamespace="", pVerbose=0, pPurgeFiles=0, &pVars) As %Status`
- %Library.Storage
  - old method: `%ValidateIndices (idxList:%List="", autoCorrect:%Boolean=0, lockOption:%Boolean=1, multiProcess:%Boolean=1) As %Status`
  - new method: `%ValidateIndices (idxList:%List="", autoCorrect:%Boolean=0, lockOption:%Integer=1, multiProcess:%Boolean=1) As %Status`
- %Net.Charset
  - old method: TranslateTableExists (charset:%String) As %Boolean
  - new method: `TranslateTableExists (charset:%String, *table) As %Boolean`
- %Net.Remote.Java.XSLTGateway
  - old method: `GetObjectGateway (*status:%Status, port:%String, host:%String, jdk:%String, javahome:%String, logfile:%String, jvmargs:%String) As %Net.Remote.ObjectGateway`
  - new method: `GetObjectGateway (*status:%Status, port:%String, host:%String, jdk:%String, javahome:%String, logfile:%String, jvmargs:%String, usepassphrase:%Boolean) As %Net.Remote.ObjectGateway`
  - old method: `StartGateway (*status:%Status, port:%String, host:%String, jdk:%String, javahome:%String, logfile:%String, jvmargs:%String) As %Net.Remote.Gateway`
  - new method: `StartGateway (*status:%Status, port:%String, host:%String, jdk:%String, javahome:%String, logfile:%String, jvmargs:%String, usepassphrase:%String) As %Net.Remote.Gateway`
  - old method: `StopGateway (gateway:%Net.Remote.Gateway) As (none)`
  - new method: `StopGateway (gateway:%Net.Remote.Gateway) As %Status`
- %OAuth2.JWT

  - old method: `JWTToObject (JWT:%String, &Local:%String, &Remote:%String, *JOSE:%String, *Body:%RegisteredObject) As %Status`
  - new method: `JWTToObject (JWT:%String, LocalPrivate:%String, RemotePublic:%String, *JOSE:%String, *Body:%DynamicObject) As %Status`
  - old method: `ObjectToJWT (&JOSE:%String, Body:%DynamicAbstractObject, &Local:%String, &Remote:%String, *JWT:%String) As %Status`
  - new method: `ObjectToJWT (&JOSE:%String, Body:%DynamicObject, LocalPrivate:%String, RemotePublic:%String, *JWT:%String) As %Status`

- %OAuth2.Server.Authenticate
  - old method: `DrawLoginHead (LOGINHEADTITLE) As (none)`
  - new method: `DrawLoginHead (LOGINHEADTITLE, MSGUSERNAME, MSGPASSWORD, MSGUSERNAMEPASSWORD) As (none)`

- %ResultSet.Custom
  - old method: `%OnNew (&returnError:%SYSTEM.Error, pRuntimeMode:%Integer=$zu(115,5)) As %Library.Status`
  - new method: `%OnNew (&returnError:%SYSTEM.Error, pRuntimeMode:%Integer={$zu(115,5)}) As %Library.Status`

- %SQL.Statement
  - old method: `%GetImplementationDetails (*pClassName:%Library.String(MAXLEN=300), *pStatementText:%Library.String(MAXLEN=""), *pArguments:%Library.List) As %Integer`
  - new method: `%GetImplementationDetails (*pClassName:%Library.String(MAXLEN=300), *pStatementText:%Library.String(MAXLEN=""), *pArguments:%Library.List, *pStatementType:%Integer) As %Integer`

- %SYSTEM.SQL
  - old method: `SetSQLStats (flag:%Library.Integer=0) As %Library.Integer`
  - new method: `SetSQLStats (flag:%Library.Integer=1) As %Library.Integer`

- %UnitTest.Manager
  - old method: `Cleanup () As %Status`
  - new method: `Cleanup (&State:%String) As %Status`
  - old method: `GetSubDirectories (root, topdir:%String, level:%Integer, suite:%String, &subdirs:%String) As %Status`
  - new method: `GetSubDirectories (root, topdir:%String, level:%Integer, suite:%String, &subdirs:%String, qspec:%String) As %Status`

- %XML.Node
  - old method: `InsertElement (localName:%String, namespace:%String="", &child:%Integer, text:%String, qname:%String="") As %String`
  - new method: `InsertElement (localName:%String, namespace:%String="", &child:%String, text:%String, qname:%String="") As %String`

  - old method: `InsertNode (sourceNode:%XML.Node, &child:%Integer, *sc:%Status) As %String`
  - new method: `InsertNode (sourceNode:%XML.Node, &child:%String, *sc:%Status) As %String`
  - old method: `InsertTree (sourceNode:%XML.Node, &child:%Integer, *sc:%Status) As %String`
  - new method: `InsertTree (sourceNode:%XML.Node, &child:%String, *sc:%Status) As %String`

- %iKnow.Queries.SourceAPI
  - old method: `GetSummary (&result, domainid:%Integer, sourceid:%Integer, length:%Integer=5, summaryConfig:%String="") As %Status`
  - new method: `GetSummary (&result, domainId:%Integer, srcId:%Integer, length:%Integer=5, summaryConfig:%String="") As %Status`

- %iKnow.Queries.SourceWSAPI
  - old method: `GetSummary (domainid:%Integer, sourceid:%Integer, length:%Integer=5, summaryConfig:%String="") As %XML.DataSet`
  - new method: `GetSummary (domainId:%Integer, srcId:%Integer, length:%Integer=5, summaryConfig:%String="") As %XML.DataSet`

- %iKnow.UI.Architect
  - old method: `addIKnowElement (type) As (none)`
  - new method: `addIKnowElement (type, parent) As (none)`

- Ens.Util.Statistics
  - old method: `RecordStats (pConfigType=0, pConfigName:%String="", pUserDimension:%String="", pCleanupAfter:%Boolean=1, pCount:%Integer=1, pDuration:%Numeric) As (none)`
  - new method: `RecordStats (pConfigType:%Integer=0, pConfigName:%String="", pUserDimension:%String="", pCleanupAfter:%Boolean=1, pCount:%Integer=1, pDuration:%Numeric) As (none)`

- Ens.VDoc.SearchTableGenerator
  - old method: `getPropId (pSearchTableClass:%String, pPropName:%String, *pPropType, *pStoreNulls) As %Integer`
  - new method: `getPropId (pSearchTableClass:%String, pPropName:%String, *pPropType, *pStoreNulls, *pUnselective) As %Integer`

- OAuth2.Server.Auth
  - old method: `GetQueryParameters (client:OAuth2.Client, *requestParameters) As %String`
  - new method: `GetQueryParameters (client:OAuth2.Server.Client, *requestParameters) As %String`

- SYS.History.SysData

- old method: `Summary (Day:%Integer=0) As (none)`
- new method: `Summary (Day:%Integer=0) As %Status`

2.2.2.2 Backup and Restore Changes

DBREST Restores CACHE.DAT Even If the File is Deleted

In previous releases, when you restored a database, EXTALL^DBREST would only restore files that were present in the database directory. Consequently, if you had done a backup and then deleted the CACHE.DAT file or another database from the directory, restoring the backup would not restore the deleted CACHE.DAT or other deleted database. With this release, all databases in the backup will be restored even if the files have been deleted from the directory. If you have a script or code that depends on the previous behavior and assumes that these databases will not be restored, you must modify it.

2.2.2.3 Class Compiler Changes

Report Error when Noncompiled Class Is Deployed

In this release calling $SYSTEM.OBJ.MakeClassDeployed on a class which is not compiled reports an error. In previous releases, it did not report an error but left the class in a state where it could not be compiled. Any subsequent attempt to use or compile this class would cause an error. However, it you have code that calls MakeClassDeployed and does not attempt to access this class, the code will now encounter an error, while it would have continued executing in previous releases.

Change to Journaling Default Behavior for Class Compiler

In this release if you call the class compiler, the compiler respects the process's default journaling environment. In previous releases, the class compiler would always journal the change unless you explicitly specified the /journal=0 qualifier. In this release, the default behavior depends on the process environment setting. You can override this setting by specifying the /journal=0 or /journal=1 qualifier.

2.2.2.4 CSP Changes

Changes to %CSP.REST DispatchRequest and AccessCheck Return Status

In previous releases errors in the AccessCheck and DispatchRequest methods could be ignored. They will now result in an Http500 response.

Handling Hyperevent Status of 0

In this release CSP ignores a return status of 0 after a hyperevent completes. In most cases this condition is usually caused by a page being reloaded before the hyperevent completes. It is possible that CSP may skip displaying a dialog box indicating a network error. In many cases the network error will repeat on the next page load.

2.2.2.5 DeepSee Changes

Dashboard Widgets Now Required to Have Names

As of 2017.1, DeepSee require each dashboard widget to have a name.

When a developer with editing privileges opens the dashboard, an alert will notify them of the fact that this auto-assign of widget names has taken place and the Modified indicator will appear on the toolbar. Once the dashboard is saved with all of its widgets named, this message will not appear again.

Pivot Table Displays Results in Correct Order

In previous versions, if you specified a crossjoin with a calculated measure in the first position and a setting in the second, CROSSJOIN(M,S), DeepSee would reverse them and display the results as if you had specified CROSSJOIN(S,M). This

had the correct values but displayed them in the wrong order. With this change, the values will be displayed in the correct order.

Note: Since this change alters how the axes are stored in the axis cache, you should clear any queries containing %MDX or %KKPI from the cache. You can clear the cache by using:

```
Do $SYSTEM.DeepSee.Reset()
```

Action Required to Benefit from DeepSee Task Licensing Improvements

In previous releases, the two tasks created for the cube registry consumed a Caché license each. In this release, these tasks do not consume a license, but you must save the Cube Registry to enable this change. To do this, open the Cube Registry page and select **Save**.

If you have not yet saved the Cube Registry, DeepSee writes a message to the DeepSee log reminding you to save the Cube Registry for the namespace.

2.2.2.6 Dynamic Object Changes

Changes in How %ToJSON Handles Illegal Numeric Values

To conform to the JSON standard, this release changes the way %ToJSON handles illegal numeric values. In this release, if the dynamic object contains the value $DOUBLE("INF"), $DOUBLE("-INF") or $DOUBLE("NAN"), the %ToJSON method signals the <ILLEGAL VALUE> error. In the previous release, these values would have contained INF, —INF, or NAN in the JSON output. If the %ToJSON() method encounters the value $DOUBLE(-0), the JSON output contains —0.0. In the previous release, the value $DOUBLE(-0) would correspond to the JSON numeric literal 0.

2.2.2.7 Ensemble Changes

Virtual Document Search Tables Replace | (Vertical Bar) with + (Plus Sign)

In this release, Ensemble replaces | (vertical bars) with + (plus signs) in Virtual Document search tables, such as EnsLib.HL7.SearchTable. When searching for data that contains a vertical bar, you must specify a plus sign in its place. The SearchTable class is implemented using an index that does not allow vertical bars in values. In previous releases, Ensemble did not make this substitution and SearchTable instances that contained a vertical bar could cause SQL errors.

If you have any custom search tables, you must recompile them to get this correction.

Ensemble SFTP Adapter Now Uses Connect Timeout

In previous releases, the SFTP adapter did not have a timeout and could wait indefinitely. In this release, it uses the Connect Timeout setting, which has a default of 5 seconds. If your production is encountering timeout errors, increase the value of the Connect Timeout setting.

Ensemble TCP Duplex Adapter No Longer Uses Private TCP Sockets by Default

The TCP duplex adapter now uses a mechanism other than private TCP adapters. This will be a transparent change to most custom code; however, if your custom code depends on the use of private TCP sockets, pass a parameter of 1 in a call to the adapter's OpenEventDevice() method. This causes the TCP duplex adapter to use a private TCP socket.

Change in how Ensemble Transformations Handle Missing Source Properties in XML Virtual Documents

In previous releases XML VDoc transformations handled missing source properties by creating a target property with an empty string value, but in this release the target property is omitted and will be missing to match the state of the source property. Both the previous and current behavior returned a missing tag error to the transformation, and both behaviors are ignored if the specify IGNOREMISSINGSOURCE.

If you want the transformation to have the previous behavior, you can insert a DTL action that replaces a missing property with a property with an empty string in the source. One way to do this is with a code action in the form:

```
$S(source.{propertypath}'="":source.{propertypath},1:"")
```

Change in How Ensemble Parses XSD with Single Repeating Child

In previous releases when Ensemble parsed an XSD whose root element had a single repeating child element, it collapsed the structure and you could only access the first child. In this release the structure is not collapsed. For example, if the XSD contains:

```
<schema ...>
  <element name="root">
     <complexType>
       <sequence>
         <element name="outer" minOccurs="1" maxOccurs="unbounded">
           <complexType>
             <sequence>
               <element name="inner" minOccurs="1" maxOccurs="unbounded" type="string"/>
             </sequence>
           </complexType>
         </element>
       </sequence>
     </complexType>
  </element>
</schema>
```

In this release, to reference the inner element, specify outer().inner(). In previous releases, the reference would have been inner(). If you have code that references an element in this way, you must update your code.

2.2.2.8 Global Mapping Changes

Report Errors for Subscripts after BEGIN

In a global mapping definition, the BEGIN keyword should not be followed by any subscripts. In previous releases, any following subscripts were ignored. In this release, a BEGIN followed by a subscript fails validation, which may cause errors for mappings. For example GLOBAL("TST",BEGIN):("TST",10000) is valid, but GLOBAL(BEGIN,"xyz"):(END) is not and fails validation.

2.2.2.9 Java and Gateway Changes

Jar File Name Changes and Removal of Duplicate Location

In this release, we are renaming the jar files as part of a transition to using the Apache Maven repository to distribute jar files. In this and future releases, the file names will include a version number. We recommend that you use Maven (or another build tool such as Ant) to manage file names and paths in your application. The following files have new names:

- cache-db-2.0.0.jar—in previous releases was cachedb.jar.
- cache-jdbc-2.0.0.jar—in previous releases was cachejdbc.jar.
- cache-gateway-2.0.0.jar—in previous releases was cachegateway.jar.
- cache-extreme-2.0.0.jar—in previous releases was cacheextreme.jar.

In previous releases, these jar files were provided in two locations in the install, but in this release they are installed in only a single location:

- *install-dir*/dev/lib/java/JDK18—the jar files continue to be installed in this directory.
- *install-dir*/lib—this directory does not contain jar files in this release.

Ensemble Java Gateway Defaults to JDK 1.8

In previous versions, the Ensemble Java Gateway defaulted to JDK 1.7, but in this release it defaults to JDK 1.8. If you need to continue using JDK 1.7, specify that version in the Java Gateway settings.

Java, DotNet Object, and XLST 2.0 Gateways Support Validated Connections

In this release, you can start the Java, DotNet Object, and XLST 2.0 gateways and require passphrases. If you are connecting from the same Ensemble instance as the gateway and use the %Connect API, %Connect automatically uses the gateway's

passphrase. If you have custom code that accesses a gateway and does not use %Connect, you should modify the code to provide the passphrase or change it to use %Connect. If the gateways do not require passphrases, then your code will run correctly without any changes.

For information on specifying the passphrase parameters when starting a gateway, see the class reference. For example, see **EnsLib.JavaGateway.Service.StartGateway()** or **%Net.Remote.Java.XSLTGateway.StartGateway()**.

If you are starting the gateway using a command line, you need to do the following to provide a passphrase to the gateway:

1. Get the passphrase list. For example, if the port is 53912, you can get the passphrase list with:

   ```
   Set tPassList=##class(%Net.Remote.Utility).GeneratePassphrase(53912)
   ```

2. Get the HEX passphrase from the list with:

   ```
   $LISTGET(tPassList,2)
   ```

3. Run the command line to start the gateway.
4. Register the passphrase with:

   ```
   Do ##class(%Net.Remote.Utility).RecordPassphrase(53912,tPassList)
   ```

Address Setting Changes for Java, XSLT2, and Java-Based Object Gateways Used in Ensemble Productions

In previous releases, in the Java, XSLT2, and Java-based Object Gateways, you could specify the address in any one of three ways:

- `127.0.0.1`
- `localhost`
- *local-machine-name*

In this release, specifying *local-machine-name* can prevent the gateway from being started if *local-machine-name* resolves to an address other than `127.0.0.1`. In that case, you must modify the gateway business service settings to use the `127.0.0.1` or `localhost` address.

Starting a Java, XSLT2, or Java-Based Object Gateway for Use by Another System

If you will be using a Java, XSLT2, or Java-Based Object Gateway that uses the Java Gateway from another system, you now must explicitly start the gateway and cannot rely on the default start mechanism. For example, you could use the following command line to start the Java gateway. (The command line is broken into multiple lines because it is very long, but should be entered as a single line.)

```
java.exe -Xrs -classpath
.;H:\intersystemsEnsMain\dev\java\lib\xml\saxon9.jar;H:\intersystemsEnsMain\dev\java\lib\JDK18\cache-gateway-2.0.0.jar;H:\intersystemsEnsMain\dev\java\lib\JDK18\cache-jdbc-2.0.0.jar
com.intersys.gateway.JavaGateway "61785"  "" "" "192.168.224.100" 2>&1
```

JAR Directory Set Automatically from the JVM Version

In previous versions, the JDKVersion was set by a property and other properties specified the location of the JAR files. In this version, we detect the Java version by examining the system default Java based on the $PATH value and use its version number to locate the correct Caché JAR file directory. Consequently, we have deleted some settings and moved related settings. If you have any code that directly accesses these properties or script that assumes a location for the JAR files, you will have to update it.

2.2.2.10 Language Bindings .NET Changes

Generate Exception for Mismatched Quotation Marks

In previous releases, the ADO.NET binding would allow a string to be started with a single quote and terminated with a double quote. In this release, this is considered an error and will fail. For example the statement

"update PL31359T set VC = VC || 'qwerty\" where VC is null"

will fail.

2.2.2.11 Migration

Must Re-run FM2Class Utility for files with computed fields or some word processing list collections

The structures generated the FM2Class for certain structures have changed in this release. If you have generated classes with the FM2Class utility to select data from computed fields or word-processing that are mapped as list collections, you have to re-run the FM2Class mapper to generate the correct code. If you only mapped word processing fields as child tables, the structures have not changed and you do not have to regenerate the classes.

2.2.2.12 Object Changes

OnRollBack Method is now called Even if Serialization Fails

If you implement %OnRollBack() method, it is now called during a transaction rollback on objects that were serialized during the transaction even if the serialization failed. In previous releases, the %OnRollBack() method was called only for objects which were successfully serialized. If your %OnRollBack() implementation assumes that the object was successfully serialized, you should modify it to handle the case where the serialization failed. In addition, if VERSIONPROPERTY was updated during the transaction, the rollback restores VERSIONPROPERTY to its pre-transaction value.

The %BuildIndices Method Rebuilds Bitmap Extent Index

In this release %BuildIndices rebuilds the bitmap extent index in some circumstances that previous releases would not have rebuilt. This ensures that the index data is current, but may use additional resources if the bitmap extent index is not needed.

Compiler Catches SQL Undefined Conditional Map

In this release the compiler catches classes with a conditional map and a null condition and produces a descriptive error message. In Caché 2015.1, 2015.2, 2016.1, and 2016.2, this compile would fail but there would not be a useful error messages. In Caché 2014.1 and earlier releases, the class compiled without an error but created invalid .int code.

2.2.2.13 ObjectScript Changes

Allow Proxy to Add By-Reference Arguments to List

If an intermediate routine receives arguments using the `args...` syntax and passes those arguments on to another routine using `args...` any new entries added to the args array will now be passed by reference. In previous releases the added entry was always passed by value so any subnodes were ignored and any change by the lower level routine did not affect the caller's arguments. If the intermediate routine is sensitive to changes made in the added arguments you will have to change the code.

Change in Length of Bit String after Rollback

In prior versions, when a $bit set was rolled back, an attempt was made to restore the old length of the bit string if possible. In some cases this would kill the global node. In this version, Caché does not attempt to restore the old length, as this could not be done consistently in all cases. Applications using $bit and related function to access the bit string global nodes do not depend on this behavior. The exception is using $bitcount function to count zero bits. It is unusual not recommended for an application to count zero bits of a bitstring, but such applications may see a change in behavior. For details, see Bit Strings in *Using Caché ObjectScript*

2.2.2.14 Security

LDAP Security Controlled on System-Wide Basis

LDAP cached credentials are now on or off on a system-wide basis. In previous releases, you could apply them on a per service or per application level.

2.2.2.15 SQL Changes

Correction to INSERT and UPDATE Compiled in Display Mode

In rare cases, INSERT and UPDATE can be compiled in DISPLAY mode. In previous releases, Caché was not correctly converting values for INSERT and UPDATE compiled in DISPLAY mode. This has been corrected. If you use #sqlcompile select=DISPLAY mode in your embedded SQL for INSERT or UPDATE and your input values are in Logical mode, you should update your application code to change the mode to logical or to supply display values as input.

JDBC Return Value Change for Auto Generated Key Value for Multi-Property IDKEY Index

JDBC can automatically generate a key value if any of the IDKEY values is system assigned. For example, if JDBC inserts into a child table with a childsub counter field, it can auto generate a key value. In previous releases, JDBC returned the auto-generated ChildID, the childsub counter field value. This was not sufficient to identify the row just inserted. In this release, JDBC returns a string consisting of the ParentID followed by "||" and then the ChildID. This is sufficient to identify the newly inserted row. However, if you have code that is processing the return value consisting of just the ChildID, you should modify your code to use the new return value.

Make Child Table Read-only If Some Indexes Are Not Projected

In this release if a child table has any indexes that are not projected, the table is read-only and you cannot perform an insert, update, or delete on the table. In previous releases, it was possible to perform these actions, but that action could corrupt an index. If a table projected from the MVSASSOCIATION includes an indexed property, it will be marked as read-only.

New SQL.JDBC Parser

We have replaced the SQL.JDBC parser with one that has significant performance improvements. We believe that it should produce the same results as the parser in previous releases, but there is the possibility that some unusual SQL statements will produce different results.

SQL JDBC UpdatableResultSet Reports Deletions More Consistently

In previous releases, Caché did not adhere to the JDBC specification on how our metadata reports updateable resultsets behavior when records are deleted. In this release, Caché handles these deletions more consistently and follows the behavior described in the specification.

SQL JDBC SysList and SysListProxy Deprecated

The SysList and SysListProxy classes are deprecated. They are included in this release, but we recommend that you replace them with the CacheList, CacheListBuilder and CacheListReader classes.

Do Not Recalculate Computed Field When Field Is Updated But Value is Unchanged

In previous releases an SqlComputeOnChange field computation would be triggered even if the value for the field updated was the same as the previous value. In this release, the computation is done only if one of the field values changes.

Corrective Action for Frozen Plans Created with Earlier Versions of Caché

If you created a plan with Caché 2016.2 and the plan contains any INSERT ... SELECT statements, you need to take actions to correct an error in the plan data.

If you upgrade from Caché 2016.2 to Caché 2017.1 or a later version, these plans will exhibit the following behavior:

- If the statement plan is Unfrozen, the plan will not be set to Frozen/Upgrade. It will remain Unfrozen. You should recompile the container code (routine, class) that contains the INSERT ... SELECT statement (or Purge cached queries).
- If the statement plan is Frozen, the plan will be put into an error state with the following message:

Frozen plan metadata for INSERT/SELECT statement frozen prior to version 2017.1 potentially incorrect. Must unfreeze, recompile (or Purge and re-prepare) and then re-freeze if wanted frozen.

Remove Option for Exporting Query Results in qButtons

In Caché 2017.1.3 and later, the ResultDataXML option for exporting query results in qButtons is not available. If your code uses this option, you must remove the reference to it.

2.2.2.16 System Changes

Correction to $BitLogic xor Operator Can Change Results

In previous releases undefined bits in the highest byte of the $BitLogic xor operator could produce incorrect results. This problem has been corrected.

Performance Improvements with Large Routines Increase Memory Usage

In this release, Caché has improved performance handling large routines with many public variables. This change increases the amount of memory Caché uses. In rare instances if you have a memory-constrained system, the increased memory requirements may cause other performance issues.

Changes Handling Named TCP Pipes on Windows System

In previous releases if the server opened a named pipe and attempted to write before the client opened the same pipe, the write operation caused an error. In this release, the write operation waits for the client to open the pipe.

Reduced JSON Nesting Allowed on Solaris X64

This change is in Caché 2017.1.3 and later. When the %FromJSON( ) method read a deeply nested JSON object on the Solaris x64 platform, recursive calls involving the JSON parsing routines ran out system stack space and caused a system crash with a memory access violation. To eliminate these process crashes, the %FromJSON nesting limit on the Solaris x64 platform has been reduced from 1024 levels to 400 levels. Any deeply nested JSON object that has more than 400 levels of nesting, will encounter an error on the Solaris x64 platform for exceeding the limit but the system crash will not occur. This error was encountered in testing nesting limits. In most cases, JSON objects are not nested to more than 400 levels. In rare conditions, it is possible to have this deep level of nesting. In these rare cases, code that worked in previous versions can encounter this nesting level limit error and fail.

2.2.2.17 Studio Changes

Change in Key Combination for Formatted Paste

In previous releases, you could do a formatted paste Ctrl/Alt/V using either the left or right Alt key. To avoid a conflict with another key combination on Czech keyboards, in this release you can only use the left Alt key.

2.2.2.18 Unit Test Changes

Routines Are Not Preserved Between Suites in a Test Run

In previous releases, the Unit Test Manager did not delete routines between suites in a test run. In this release, the manager deletes .MAC, .INT, .INC, .OBJ, and .BAS routines that it loaded. It does not delete .MVB or .MVI routines. If you have code that depends on routines being preserved, you should modify your code.

Unit Test Manager Supports UDL Files

In this release the unit test manager supports UDL files exported by Atelier, as well as XML files from Studio. UDL files have file types .cls, .mac, .int, and .inc. By default the Unit Test Manager will look for and load both UDL and XML files in the test directory. The new qualifiers /loadxml=0 and /loadudl=0 may be used to prevent the loading of either type.

2.2.2.19 Web Services Changes

Support Use of HTTPS Protocol for WS-Policy Bindings

In this release SOAP web services supports WS-Policy symmetric and asymmetric bindings. These bindings require an SSLConfiguration. If you specify a symmetric and asymmetric binding without an SSLConfiguration, you will encounter an error. In previous releases, the HTTPS protocol would have been downgraded to an HTTP protocol and would have not encountered an error from the missing SSLConfiguration.

2.2.2.20 XML Changes

SendRequestToGateway Method Signature Changes

In this release the %Net.Remote.Java.XSLTGateway.SendRequestToGateway() signature has changed. The method now requires an instance of Net.Remote.Java.XSLTGatewayRequest, which encapsulates all of the previous arguments, as well as some additional ones, in a single object. If you have custom code that calls SendRequestToGateway(), you must modify the code.

XML Processing Encounters Error from Buffer Overrun

In this release, the %XML.SAX and %XML.XLST parsers prevent a buffer overflow and throw an exception if one would occur. In previous releases this overflow was not detected and could cause serious problems. However, in some cases the buffer overflow would not cause problems and the operation would appear to execute correctly. Under these circumstances, an operation that appeared to work correctly in the previous version will encounter an exception and fail. If you encounter a length error exception reading a stream, it indicates an error in the stream Read method, which should be corrected.

Memory Changes Handling XML

This release of Caché contains changes in how XPATH is processed for XML. Under most circumstances, these changes reduce the memory required, but under some circumstances, they increase the memory required. In memory-constrained systems, this change may cause performance problems.

2.2.2.21 Zen Changes

White-space Setting Removed from <pre> Element

In previous releases, setting white-space to normal was always done for the <pre> element. This was a side-effect of a previous fix and caused problems in some Zen pages. In this release, this white-space setting is no longer applied to <pre> elements. If you have CSS definitions that rely on this `white-space: normal;` setting, you should modify the CSS and explicitly set the white-space to normal.

3

Caché 2016.2

This chapter provides the following information for Caché 2016.2:

3.1 New and Enhanced Features for Caché 2016.2

This section includes:

3.1.1 Major Developments

The following major, new features have been added to Caché for this release:

3.1.1.1 Support for Atelier, New Eclipse-Based IDE

This is the first Caché release that provides server support for Atelier, the new Eclipse-based IDE. The Atelier IDE brings together the powerful and popular Eclipse development environment and the InterSystems Caché database. Atelier allows you to develop Caché applications using a modern file-based IDE on a client system. Atelier handles uploading the application to the Caché server where it can be run and debugged.

The focus of future development will be on the new Eclipse based IDE. Studio will remain an option to install and developers can continue to develop code with it. However, it will be treated as a maintenance product and will not see new functionality added as we move forward with Atelier. Some minor bugs may not be addressed either depending on resources required versus the severity of the issue.

Atelier is available as a separate download in addition to Caché or Ensemble. You can choose to install either a stand-alone Rich Client Platform (RCP) application, or a plug-in that can be added to an existing Eclipse installation. Users of the RCP application can add additional Eclipse plug-ins. Atelier uses the Eclipse auto-update mechanism to help users get the latest changes. For information on downloading Atelier and for the Atelier documentation, see http://www.intersystems.com/atelier, the Atelier home page.

3.1.1.2 Support for OAuth 2.0 and OpenID Connect

Starting with this release, Caché supports the OAuth 2.0 framework and OpenID Connect Core. Caché can act as a client, as a resource server, and as an authorization server. See *Using OAuth 2.0 and OpenID Connect with Caché*. Also see this article https://community.intersystems.com/post/caché-open-authorization-framework-oauth-20-implementation-part-1 posted on InterSystems Developer Community.

3.1.1.3 SQL/JSON Support

The SQL syntax has been enriched to support parts of the SQL/JSON standard. This allows to construct JSON data as a result from an SQL query. The functions supported are:

- JSON_OBJECT—construct a JSON object for each encountered row.
- JSON_ARRAY—construct a JSON array for each encountered row.

3.1.1.4 SQL Improvements

This release has the following SQL performance enhancements and improvements:

- %Parallel—Improves performance of queries by breaking the query into chunks that can be executed in parallel on multiple processor systems. In this release the %Parallel infrastructure has the following improvements:
  - Lower overhead to breaking a query into chunks.
  - Caché dynamically breaks the query into a number of chunks based on the number of hardware threads that are available.
  - Increased number of query conditions that can be broken into chunks. For example, Caché can now break set looping constructs like IN (?, ?, ?) or %INLIST() into chunks.

  The result of these changes is that you can add %Parallel to most queries without extensive analysis. If Caché can improve the performance by breaking the task into chunks and using parallel processing, it will. If the query is not amenable to parallel processing or if the system only has a single hardware thread available, Caché recognizes this situation and does not incur the overhead of breaking the query into chunks.
- Improved performance when iterating over list collections.
- Frozen plans—allows you to freeze the plans while you make index and other changes.

3.1.2 Other Items of Note

In addition, many more lesser improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

Other minor areas of improvement include:

- New REST API to access Caché source code files is available in this release. This API was implemented to provide access to the Atelier IDE, and it allows you to create IDEs and similar application. See *Accessing Caché Source Code Files Using REST* for details.

- The DataCheck utility makes more efficient use of parallel processing in order to complete checks faster. The ^DATACHECK operator interface has an enhanced status page and start dialog for improved usability.
- Caché is enhanced to use faster memory synchronization mechanisms (memory barriers) in certain critical execution paths. Data ingestion tests show more than 10% improved performance on Linux x86-64. The improvements apply to the following platforms:
  - Apple Mac OS X for x86–64
  - IBM AIX® for Power System-64
  - Microsoft Windows for x86–32, x86–64
  - Linux for x86–32, x86–64
  - Solaris for x86–64, SPARC-64

  These improvements do not affect the Itanium and Alpha platforms.

3.2 Caché 2016.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2016.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2016.1 and 2016.2.

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

3.2.1 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 2016.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

3.2.1.1 Operational Changes

This section details changes that have an effect on the way the system operates.

SQL Parallel Changes May Require Increase in Generic Memory Heap

In this release Caché uses InterProcess Queues (IPQ) in the Generic Memory Heap (gmheap) to process SQL parallel queries. Since these queries did not use this resource in previous releases, you may need to increase the size of the Generic Memory Heap. To increase its size, select **System Administration**>**Configuration**>**Additional Settings**>**Advanced Memory** and increase the **gmheap** size and restart Caché.

Worker Job Changes May Require User Permissions Change

To improve efficiency, this release shares the worker jobs between all users. These worker jobs are started by the super server. This is different from the previous release where each worker job typically ran under the account of the user running Caché. Consequently, these jobs run under a different OS user compared to previous versions. If the OS user for the super server does not have the same file privileges as the current user, the worker jobs may not have the file permissions needed to access files. If this problem arises, you can solve it by giving additional permissions to the operating system user account, or, if the problem occurs in custom code, instruct the developer to pass “/multicompile=0” to the work queue manager. This parameter instructs the work queue manager to execute the worker job under the current user’s account.

^JCONVERT Journal Conversion Changes

In this release, ^JCONVERT no longer supports RAW format as an output format and the prompt for this format has been removed. In this release ^JCONVERT converts journal files only to the Variable/UTF8 format.

Although previous releases supported RAW format output, there were substantial limitations, such as not handling binary journal data. If you have used ^JCONVERT to produce RAW files, you should evaluate your conversion needs.

Note: If you are running ^JCONVERT from a script, you must modify the script as the first prompt has been eliminated.

Increase in Shared memory for Global Buffers

In previous releases, automatic configurations were limited to 256 megabytes shared memory for global buffers. In this release, an automatic configuration may use up to 1 gigabyte of shared memory for global buffers. If you are running many instances on a single system, this increase in memory may limit the number of instances you can use. Typically, these configurations are used for test or development environments. Note that “automatic” configurations should not be used for productions systems or for producing performance benchmarks.

3.2.1.2 Platform-Specific Items

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

UNIX®

In previous versions when you uncompressed the .tar.gz compressed archive file, it placed all the files in the same top-level folder containing the compressed archive file. In this release, it creates a new directory with the same name as the compressed archive file (with the .tar.gz extension omitted) and places all of the extracted files in that directory.

If you use a script that uncompresses the archive file and then executes the install, you will have to modify the script to handle the new directory.

Mac OS X

In previous versions when you uncompressed the .tar.gz compressed archive file, it placed all the files in the same top-level folder containing the compressed archive file. In this release, it creates a new directory with the same name as the compressed archive file (with the .tar.gz extension omitted) and places all of the extracted files in that directory.

If you use a script that uncompresses the archive file and then executes the install, you will have to modify the script to handle the new directory.

3.2.2 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.

3.2.2.1 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release:

- %Dictionary — CompiledSystemMethod, CompiledSystemMethodQuery, SystemMethodDefinition, SystemMethodDefinitionQuery
- %EM.Session — AssignToMirrorGroup, CreateMirror, EditAborted, EditApproved, EditCompleted, UpdateInstances
- %Library — AbstractObject, Array, Object
- %SYSTEM — WorkIPQ, WorkMgrSend

Removed Methods

The following methods were present in the previous version and have been removed in this release:

- %CSP.Documatic
  - RenderSystemMethodInfo
- %EM.Session
  - AssignToMirrorGroup
  - CreateMirror
  - EditAborted
  - EditApproved
  - EditCompleted
  - UpdateInstances

Removed Properties

The following properties were present in the previous version and have been removed in this release:

- %Dictionary.ClassDefinition
  - SystemMethods
- %Dictionary.CompiledClass
  - SystemMethods
- %Library.ClassDefinition
  - SystemMethods
- %SYSTEM.WorkMgrIPQ
  - queueIdx
  - tailIdx
- Security.SSLConfigs
  - CRLFile

Removed Parameters

The following parameters were present in the previous version and have been removed in this release:

No parameters were removed.

Removed Indexes

The following indexes were present in the previous version and have been removed in this release:

No indexes have been removed.

Modified Methods

The following methods have different signatures in this version of Caché:

- %CSP.UI.Portal.SQL.Home

  - old method: `doReferenceTo (newtable) As (none)`
  - new method: `doReferenceTo (newtable, type) As (none)`

- %CSP.UI.Portal.SQL.Utils
  - old method: `SQLExecuteQuery (pText:%String(MAXLEN=""), &pClassName:%String, pRuntimeMode:%Integer=2, pDialect:%String="Cache", pObjectSelectMode:%Boolean=0, %caller:%String="", &tGlobals) As %String`
  - new method: `SQLExecuteQuery (pText:%String(MAXLEN=""), &pClassName:%String, pRuntimeMode:%Integer=2, pDialect:%String="Cache", pObjectSelectMode:%Boolean=0, %caller:%String="", &tGlobals, &tLines, &tDiskRead) As %String`
- %CSP.UI.SQL.RunQueryPane
  - old method: `DrawResult (&tRS:%ResultSet, tGlobals:%String, tStart:%String) As (none)`
  - new method: `DrawResult (&tRS:%ResultSet, tGlobals:%String, tStart:%String, tLines:%String, tDiskRead:%String) As (none)`
  - old method: `ExecuteResultSet (pParam:%String, &pRS:%ResultSet) As %Status`
  - new method: `ExecuteResultSet (%pParam:%String, &%pRS:%ResultSet) As %Status`
- %DeepSee.Component.Widget.widget
  - old method: `printSVGContent (svgFrameId, parms) As (none)`
  - new method: `printSVGContent (svgFrameId, parms, svgContent) As (none)`
- %DeepSee.Component.deepseeSvgImageProvider
  - old method: `parseSize (sizeStr) As (none)`
  - new method: `parseSize (size) As (none)`
  - old method: `prepareFilterTable (filterNames, filterValues) As (none)`
  - new method: `prepareFilterTable (filterNames, filterValues, parms) As (none)`
- %DeepSee.ResultSet
  - old method: `%PrepareKey (pCubeName:%String, pQueryKey:%String) As %Status`
  - new method: `%PrepareKey (pCubeName:%String="", pQueryKey:%String="", pPrepareMDX:%Boolean=0) As %Status`
- %DeepSee.UI.FolderManager
  - old method: `ExportItems () As %String`
  - new method: `ExportItems (pUseContainerClass:%Boolean, pContainerClassName:%String="") As %String`
- %EM.Session
  - old method: `IsEditable (groupName:%String, serviceName:%String) As %Boolean`
  - new method: `IsEditable (groupName:%String, serviceName:%String) As %String`
- %EM.Session.IsEditable

  - old method: `Invoke (%Client, %Action, groupName:%String, serviceName:%String) As %Library.Boolean`
  - new method: `Invoke (%Client, %Action, groupName:%String, serviceName:%String) As %Library.String`

- %Iterator.Array
  - old method: `%OnNew (oref:%Library.AbstractObject) As %Status`
  - new method: `%OnNew (oref:%Library.DynamicAbstractObject) As %Status`

- %Library.GTWCatalog
  - old method: `SQLProceduresExecute (&QHandle:%Binary, dsn:%String, pname:%String, schname:%String, catalog:%String="") As %Status`
  - new method: `SQLProceduresExecute (&QHandle:%Binary, dsn:%String, pname:%String, schname:%String="", catalog:%String="") As %Status`

- %Net.Provision.Configure
  - old method: `AuthenticateWithKeyPair (username:%String, publickeyfile:%String, privatekeyfile:%String, passphrase:%String) As %Status`
  - new method: `AuthenticateWithKeyPair () As %Status`
  - old method: `AuthenticateWithUsername (username:%String, password:%String) As %Status`
  - new method: `AuthenticateWithUsername () As %Status`
  - old method: `Provision (cachekitfile:%String="") As %Status`
  - new method: `Provision () As %Status`

- %SYSTEM.OBJ
  - old method: `CompileList (&list:%String="", qspec:%String="", &errorlog:%String) As %Status`
  - new method: `CompileList (&list:%String="", qspec:%String="", &errorlog:%String, &updatedlist:%String) As %Status`

- %SYSTEM.WorkMgr
  - old method: `Clear (timeout:%Integer=5) As %Status`
  - new method: `Clear (timeout:%Integer=0) As %Status`
  - old method: `WaitForComplete (qspec:%String) As %Status`
  - new method: `WaitForComplete (qspec:%String, errorlog:%String) As %Status`

- %SYSTEM.WorkMgrIPQ
  - old method: `Decode (data:%String, qspec:%String, &AtEnd:%Boolean) As %Status`
  - new method: `Decode (qspec:%String, &AtEnd:%Boolean) As %Status`

- %Studio.Project
  - old method: `realCompile (qstruct, &%errorlog:%String, &itemlist:%String) As %Status`

  - new method: `realCompile (qstruct, &%errorlog:%String, &itemlist:%String, &updatedlist:%String) As %Status`

- %Studio.SASchemaClass
  - old method: `GetSchemaForClasses (&pClassList:%String, pNode) As %Status`
  - new method: `GetSchemaForClasses (&pClassList:%String) As %Status`
  - old method: `loopGlobal (pStream:%Stream.Object, pNode:%String) As %Status`
  - new method: `loopGlobal (pStream:%Stream.Object) As %Status`
- %ZEN.Auxiliary.altJSONProvider
  - old method: `%ArrayToAET (&pMetaData, &pData) As %AbstractObject`
  - new method: `%ArrayToAET (&pMetaData, &pData) As %DynamicAbstractObject`
  - old method: `%GetTypeInfo (pClass:%String, pProperty:%String) As %Object`
  - new method: `%GetTypeInfo (pClass:%String, pProperty:%String) As %DynamicObject`
  - old method: `%ObjectToAET (pObject:%RegisteredObject, &oVisited, pLevel:%Integer=0, pFormat:%String="acelo") As %AbstractObject`
  - new method: `%ObjectToAET (pObject:%RegisteredObject, &pVisited, pLevel:%Integer=0, pFormat:%String="acelo") As %DynamicAbstractObject`
  - old method: `%TestForNestedObjects (pObj:%Array) As %Boolean`
  - new method: `%TestForNestedObjects (pObj:%DynamicArray) As %Boolean`
- %ZEN.Auxiliary.jsonMDXProvider
  - old method: `%ConstructNewDynamicArray () As %Array`
  - new method: `%ConstructNewDynamicArray () As %DynamicArray`
  - old method: `%ConstructNewDynamicObject () As %Object`
  - new method: `%ConstructNewDynamicObject () As %DynamicObject`
- SYS.DataCheck.DestinationSYS.DataCheck.Destination
  - old method: `Start () As %Status`
  - new method: `Start (restartworkflow:%Boolean) As %Status`

3.2.2.2 Studio Changes

Deployed Methods are Marked as Read-Only

In previous releases, Studio ignored datatype classes that processed with the $system.OBJ.MakeClassDeployed() method. In this release, they are visible in Studio but are marked as Read-Only.

3.2.2.3 DeepSee

DeepSee Uses Locale Setting for Displaying Date-Time Values

In this release, if you specify that dates should be displayed using the current locale, DeepSee displays the date according to the locale. For example, in previous releases, a date would be displayed in the “mmm dd y” format but in this release will be displayed in the “mm/dd/y” format. You can revert to the previous behavior by specifying that the current locale

should not be used for dates or by explicitly specifying a format. To specify that the current locale should not be used for dates, select **System Administration** > **Configuration** > **National Language Settings** > **Locale Definitions** and select **No** for **Use locale date/time/number formats for** *locale-name*.

DeepSee Changes to Tracking Update Status in ^OBJ.DSTIME

To avoid a race condition, this release tracks the status of class updates in a different location. If your application does both of the following, you will need to update it to reflect these changes:

- Reads the log of updates to a class from ^OBJ.DSTIME.
- Uses DeepSee to build or synchronize cubes based on that class.

If your application does both of these, you should change the reference from ^OBJ.DSTIME to ^DeepSee.Update. The ^DeepSee.Update global contains the same log of the update status as was previously stored in ^OBJ.DSTIME. The %PurgeDSTIME method now clears the ^DeepSee.Update global. The ID entries for a class stored in the ^OBJ.DSTIME global are killed with each update to any cube based on that class.

Individual IDs are stored under an identical structure to the ^OBJ.DSTIME buffer:

^DeepSee.Update(class,counter,ID) = updateType

3.2.2.4 Ensemble

IsEnsembleNamespace Method Accepts Parameter

In this release, the %Library.EnsembleMgr IsEsembleNamespace() method allows you to specify the name of the namespace as a parameter. The method returns 1 if the namespace is enabled for Ensemble and 0, if not. If you do not specify a parameter, the method's behavior is unchanged—it returns the value for the current namespace. In previous releases this method did not accept a parameter, but, if you specified a parameter, the method would always return a 0.

Changed X12 Responses to Conform to Implementation Guides for Batch Transactions

In this release, the X12 business services sends a 999 response for X12 batch transactions only for the transaction types where the 999 response is specified in the Implementation Guides. Ensemble sends a 997 response for other X12 batch transactions. In Ensemble 2014.1 though Ensemble 2016.1, the X12 business services sent a 999 response for any incoming batch with a version greater than or equal to 5010. The behavior in Ensemble 2016.2 is the correct behavior, but if your code expects the 999 response, you can override the OnConstructReply() method in a custom X12 Business Service class in order to construct the 999 response document.

Specifically, the X12 Business Services construct 999 responses only for X12 batch transactions which have one of the following as their Industry Identifier Code (associated Transaction Set Identifier Codes indicated in parentheses):

```
005010X187 (269)
005010X279 (270/271)
005030X209 (274)
005040X254 (275)
005010X212 (276/277)
005010X214 (277CA)
005010X217 (278)
005010X218 (820)
005010X220 (834)
005010X221 (835)
005010X222 (837P)
005010X223 (837I)
005010X224 (837D)
```

Efficiency Improvements May Cause Production Configuration Page to Display Stale Data

The Ensemble production configuration periodically queries the production status and also tests whether the production needs updating. These checks lead to locks on core runtime data. In this release, the production configuration page reduces the amount of time it is holding locks. This improves the production efficiency but may cause stale production status or update information to be displayed. Even if the page is displaying stale update information in the hover tip, selecting update always uses the current value, not the stale value being displayed.

Prevent User from Editing Documents When Prohibited by Source Control

In this release, the Ensemble editors support source control and do not allow the user to modify a document when the source control system marks it as read-only. If the user opens a document when it is not checked out, it is displayed as read-only. To edit the document, the user must check out the document from source control and refresh the editor. If the document was checked out when the user opened it, but is later marked read-only by the source control system, the user will not be able to save the document and must use Save As to rename it. The following Ensemble editors now have this behavior:

- BPL Editor
- DTL Editor
- Rule Editor
- Record Mapper
- Complex Record Mapper
- Data Lookup Tables
- Record Mapper

Changes in how XML Schema Global Storage

In this release, we have changed the internal global storage used for XSD. Although it was not a documented feature, in previous releases it was possible to make direct access to the array nodes in the virtual XSD global storage. This access no longer works in this release. It is possible to access this data using the GetContentArray() API.

This change reduces the storage needed to represent the XSD. A side-effect of this change is that if you re-import large schemas, extensive journaling could be recorded as the obsolete schema data is removed.

3.2.2.5 JSON Dynamic Object

Changes to JSON Dynamic Object Classes

In this release we have made substantial changes to the JSON dynamic object classes that were introduced in Caché 2016.1. See the Developer Community article JSON changes in Caché 2016.2 (https://community.intersystems.com/node/413171) for a detailed description of these changes. This article also provides guidance on how to update any code that you have written that works with the Caché 2016.1 JSON dynamic object classes.

3.2.2.6 Utilities Changes

CacheHung.sh Utility Output Directory Change

In this release, the CacheHung utility removes the output file from the current directory after moving it to the *instance*`/mgr` directory. In previous releases, this utility copied the output file to the *instance*`/mgr` directory but left it in the current directory. If you have scripts that assumes that the output file is in the current directory, you should modify them.

Note: If the current process does not have write permissions to the *instance*`/mgr` directory, then the move fails and the output file is left in the current directory.

ReturnUnusedSpace Now Always Returns Valid %Status Values

Although SYS.Database.ReturnUnusedSpace returns %Status, in prior versions it also had documented -1 and -2 as possible return values. Those values were returned when certain conflicting database operations were in progress (expansion and backup for example), but other conflicting operations would be returned as a proper error status. The interface has been updated to return all failures as proper %Status returns, including failures due to an expansion or backup in the same directory. Code that calls ReturnUnusedSpace should be inspected to ensure that the %Status return is handled properly.

4

Caché 2016.1

This chapter provides the following information for Caché 2016.1:

4.1 New and Enhanced Features for Caché 2016.1

This section includes:

4.1.1 Important New Features

Caché 2016.1 has the following important new features:

4.1.1.1 Improved JSON Processing Performance

JSON provides a lightweight mechanism to easily communicate structured data. In this release, you can use the new %Object and %Array classes to enable high performance parsing and generation of JSON-compliant content. The improved integration of JSON with ObjectScript allows you to efficiently convert between JSON and ObjectScript. For example, curly braces in JSON generate %Object instances in ObjectScript and square brackets generate %Array objects. Curly braces and square brackets can be arbitrarily nested and used with ObjectScript expressions for values. In this release, the JSON processing performance has improved to the point where you can have the benefit of the simplicity of working with JSON and have very efficient code. For information, see *Using JSON in Caché*.

For future development, InterSystems recommends that you use the %Object and %Array classes in place of %ZEN.proxyObject.

If you have existing code that is using JSON through Zen, you can gain improved efficiency with minimal code changes by using the new %ZEN.Auxiliary.altJSONProvider and %ZEN.Auxiliary.altJSONSQLProvider classes instead of the previous JSON provider classes, %ZEN.Auxiliary.JSONProvider and %ZEN.Auxiliary.JSONSQLProvider, which are still accessible, although no longer specifically documented.

Note: In the %ZEN.Auxiliary.altJSONProvider and %ZEN.Auxiliary.altJSONSQLProvider classes, the **%ConvertJsonToObject()** method requires more memory than the corresponding method of the older classes. Specifically, the method in the new class requires twice as much memory as the JSON string used as input, because the method holds two copies of that data in memory.

The methods in the new %ZEN.Auxiliary.altJSONProvider and %ZEN.Auxiliary.altJSONSQLProvider classes always return standards-compliant JSON. Null data items project as JSON NULLs and strict JSON compliance is always enforced.

Normally, the newer classes simply ignore any flags that affect white space. The newer classes also ignore the u and q flags. In the interests of backwards compatibility, the new JSON provider has a fallback mode where, if the request fails, it will try again using the old jsonProvider logic. In that event, all of the old flags are used.

4.1.1.2 Improved SQL Performance

The performance of SQL commands is a critical component in how efficiently your Caché application handles large databases. We have been making significant improvements to Caché's SQL performance. This release continues these improvements with the following new enhancements:

- Performance optimization for xDBC server code.
- Server performance improvements for query execution. In this release, the SQL query processor generates more efficient code for many common query operations. One query with a significant performance improvement is querying a count of UNION ALL. In this release, Caché optimizes the calculation of the count separately for each member of the union.
- Improved ability to cancel an operation.

For details see "UNION" in *Caché SQL Reference*.

4.1.1.3 DeepSee REST Services and JavaScript Library

There are new REST services that provide access to the DeepSee engine. The services include data services for executing pivot tables and MDX queries and returning the results in JSON. There are also information services for getting lists of cubes, pivot tables, and cube information (dimensions, members, measures, and so on.)

For JavaScript developers, there is a new DeepSee JavaScript library that provides access to the DeepSee REST services.

For information on the REST services and the JavaScript library, see *Tools for Creating DeepSee Web Clients*.

4.1.2 Major Developments

The following major, new features have been added to Caché for this release:

- New DeepSee Features
- iFind Enhancements
- iKnow Enhancements
- Zen Mojo Support of Twitter Bootstrap
- .NET Entity Framework Support
- ADO .NET Query Support of Cancel
- Mirroring Improvements

- Security Enhancements
- Performance and Scalability Improvements
- Windows x64 Support for Apache Version 2.4
- Improved Utility Features

4.1.2.1 New DeepSee Features

In addition to the REST services and JavaScript library, the following new features have been added to DeepSee in this release:

- *%SynchronizeCube in Parallel* — %SynchronizeCube can now use multiple DeepSee agents to update the cube in parallel. Prior to this release, %SynchronizeCube used only one process to update the cube.

  For details, see the class reference for %DeepSee.Utils.

- *Shared calculated members* — Calculated members defined using Analyzer can now be made available to other pivot tables that use the same cube. There is also a new API to create calculated members. Prior to this release, calculated members defined in Analyzer were only available to the pivot table in which they were defined.

  For details, see the chapter "Defining Calculated Elements" in *Using the DeepSee Analyzer*.

- *Measure-specific listings*—When getting a detail listing from a cell in a pivot table, the rows displayed in a detail listing can now be influenced by the measure that cell represents. Prior to this release, the listing was influenced by the row, column, and filter, but not the measure. The definition of a measure now includes new optional attributes to influence the rows displayed in the detail listing.

  For details, see "Specifying Additional Filtering for Listings" and "Specifying Additional Filtering for Listings for a Calculated Measure" in *Defining DeepSee Models*.

- *Analyzer and Architect: Allow resizing of panes* — In Analyzer, the left pane that shows the dimension tree can be resized by dragging the border on the right side of the pane. Also, the vertical borders between the rows, columns, measures, and filters boxes can be dragged to resize these boxes. In Architect, the left pane that shows the source class can be resized.

- *New utility methods to get and execute MDX from a pivot table*—There are several new utility methods in %DeepSee.Utils:

  – **%GetMDXFromPivot()**: returns MDX text for a pivot table. Optionally executes the query.

  – **%ExecutePivot()**: executes the MDX for a pivot table. Optionally returns the associated %DeepSee.ResultSet.

  – **%GetResultSetFromPivot()**: returns the %DeepSee.ResultSet for a pivot table. Optionally executes the query.

  For details, see the class reference for %DeepSee.Utils.

- *Ability to define additional listing filtering for a plug-in* — Plug-ins provide the ability to define custom measures is DeepSee. Prior to this release, a listing from a plug-in always showed all the rows used to calculate the measure. This release provides new functionality that allows the plug-in to limit the listing to using a subset of the rows.

  For details, see the chapter "Defining Plugins" in the *Advanced DeepSee Modeling Guide*.

- *New %OnAfterBuildCube() method* — In addition to allowing custom logic before a cube is built, this release adds a new **%OnAfterBuildCube()** method to allow for custom logic after a cube is built.

  See "Customizing Other Cube Callback Methods" in the *Advanced DeepSee Modeling Guide*.

- *New $$$VARIABLES token* — When accessing a dashboard via a URL, you can now use the token $$$VARIABLES. This token will be expanded to the current pivot variables and their values.

  See the chapter "Accessing Dashboards from Your Application" in the *DeepSee Implementation Guide*.

- *Calendar filter and first day of the week* — When filtering on a date, the calendar filter now respects the system setting for the first day of the week. Prior to this change, the calendar always showed Sunday as the first day of the week.

4.1.2.2 iFind Enhancements

This release provides the following iFind enhancements:

- iFind now supports *fuzzy search* to look up records containing words that "almost" match the search string, accounting for small variations in writing (color vs colour) and misspellings (collor vs color). iFind will evaluate fuzzy matches through comparing their edit distance (number of added, removed or changed characters) to the search string with a threshold, which defaults to 2.
- iFind text search now supports new capabilities. Search results can now be ranked according to relevance with respect to the search terms. For Analytic indices, this ranking can take advantage of iKnow's unique entity level and associated metrics. Furthermore, iFind can now also return snippets of text containing the actual search matches highlighted. Finally, word and entity information can now also be stored per iFind index rather than shared for the entire namespace, which will benefit cases where indices of different size and settings appear in the same namespace.

For information, see the chapter "iFind Search Tool" in *Using iKnow*.

4.1.2.3 iKnow Enhancements

An **iKnow Architect** page has been added to the System Management Portal for managing iKnow domains. This page allows you to define the static aspects of an iKnow domain, such as metadata and configurations, as well as the data locations to load data from. Domains can also be loaded (built) and emptied from this interface, leveraging the underlying Domain Definition infrastructure. For details, see "Architect" in *Using iKnow*.

Starting with this release, iKnow also supports Swedish and Japanese.

4.1.2.4 Zen Mojo Support of Twitter Bootstrap

In this release, Zen Mojo supports Twitter Bootstrap, a framework that helps you develop web pages. See the chapter "Helper Plugin for Bootstrap" in *Using Zen Mojo Plugins*.

4.1.2.5 .NET Entity Framework Support

In this release, Caché includes support for the .NET Entity Framework. The Entity Framework is an object-relational mapper that enables .NET developers to work with relational data using domain-specific objects. Caché supports Entity Framework versions 5 and 6. For details on the Caché support of Entity Framework, see "Using the Caché Entity Framework Provider" in *Using .NET and the ADO.NET Managed Provider with Caché*. For more information on the .NET Entity Framework, see http://www.asp.net/entity-framework.

4.1.2.6 ADO .NET Query Support of Cancel

This releases adds the ability to cancel an ADO .NET query.

4.1.2.7 Mirroring Improvements

This release has the following improvements in mirroring:

- Optional compressed network communication to improve performance and reduce network load. For details, see "Journal data compression" in the *Caché High Availability Guide*.
- Support for larger TCP window size allows improved network throughput to distant mirror members (high-latency connections). For details, see "Network Latency Considerations " in the *Caché High Availability Guide*.

4.1.2.8 Security Enhancements

This release has the following security improvements:

- Includes Open SSL library version 1.0.1p.
- Improved reliability of auditing by requiring journaling of audit database. For details see "About System Audit Events" in the *Caché Security Administration Guide*.

4.1.2.9 Performance and Scalability Improvements

This release has the following performance improvements:

- Enhancements to improve scalability of database accesses by reducing contention for pointer blocks. The improvements are particularly relevant to applications that perform kills of large subtrees or that frequently induce physical disk reads.
- Improved performance of dejournaling (the process of applying journals to databases in mirroring, shadowing and JRNRESTO) by eliminating internal block contention with dejournal prefetch jobs.
- Removed limit of number of journal files allowed per day.
- Enhanced caching on ECP application servers. Application servers now cache big string data values. ECP also automatically prefetches some database blocks and caches them on application servers in advance.

4.1.2.10 Windows x64 Support for Apache Version 2.4

This release provides Windows x64 native support for the Apache Version 2.4 web server.

4.1.2.11 Improved Utility Features

This release includes the following improved utilities:

- New utility validation method to check routine buffer. See the **ValidateRoutineBuffers()** in the %SYSTEM.Util class.
- New methods in %SYS.ProcessQuery provide information about the process.
- Minor enhancements to Buttons report.
- To help users writing scripts using the output of ccontrol, a new option prevents wrapping long field names.
- Management portal provides additional filtering options for databases, namespaces, locks, roles, tasks, and ECP server pages.

4.1.3 Other Items of Note

In addition, many more lesser improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

Other minor areas of improvement include:

- Mirroring
  - Incoming journal transfer rate for a mirror member is now displayed in the mirror monitor.
  - Behavior is improved on reconnecting a disconnected member that needs to retrieve and apply many journal files. A more useful status, Synchronizing, is now reported instead of Transition. Journal transfer and dejournaling progress are visible from the mirror monitor. Journals purges are unchanged by the disconnect and reconnect.
  - Mirror monitor now displays when a primary is in the trouble state.

  - Database compaction can now move the mirror information block if needed, allowing it to return unused space in mirrored databases more reliably.

- Mirror-aware CSP—The CSP Gateway now has an option to automatically redirect to the current mirror primary without use of the Mirror Virtual IP or external alternatives. This enables web-based applications to be deployed more easily in certain hardware configurations where establishing a Virtual IP is not possible or not a complete solution. For mirror configurations that use the mirror Virtual IP or another technology to redirect all types of connections to the primary it suffices to configure the Gateway to connect to the Virtual IP without opting for it to be mirror aware. For more information see "Redirecting Application Connections Following Failover or Disaster Recovery" in the *Caché High Availability Guide* and "Configuring Server Access" in the *CSP Gateway Configuration Guide*.
- Performance and Scalability:
  - Improved performance of the garbage collector cleaning up database blocks from large KILLs.
- The DataCheck View Status page now displays information about a DataCheck configuration's resource consumption. The amount of resources used at various Throttle setting is updated to scale more smoothly at the low end, allowing more flexibility on systems for which, previously, a setting of 2 was too low and a setting of 3 was too high. For details see "Performance Considerations" in the "Data Consistency on Multiple Systems" chapter of the *Caché Data Integrity Guide*.
- Mail server monitoring utility allows you to monitor specified port.
- The File DriveList includes UNIX mounted drives.
- Trace information is now available for processes that terminate with a FRAMESTACK error. An informational message is written to cconsole.log to aid in detecting and diagnosing the problem. The message shows the line reference that encountered the error:

  ```
  (pid) 0 <FRAMESTACK> at +13^|"USER"|test
  ```
- When purging journal files, cconsole.log contains detailed information on any journal not purged, such as those used for open transactions.
- The performance of the bitcount function was improved.
- When using the Security.Roles.Copy() API to copy Role definitions to create a new Role, this call will now also include the SQL Privileges defined for this Role. Users do not need to call Security.Roles.CopySQLPrivileges() in addition anymore.
- On UNIX® systems Caché requires access to /dev/urandom, which is used at Caché startup to seed the secure random number generator. With this release the severity of not being able to access /dev/urandom has been increased from informational to warning in the cconsole.log.
- Prior to this release the MONLBL utility, which monitors routine performance, did not warn users that stopping the monitoring process will delete all collected data. Now the user receives a reminder and can decide how to proceed.
- To assist customers that use process private globals (PPG) metrics, this release introduces two new metrics:
  - gloref_ppg - total number of process private global references made by the process (analogous to gloref)
  - gloset_ppg - number of updates to process private globals made by the process (analogous to gloset)
- Caché will now also log failed TCP connections on Windows platforms, and generate an entry in SYSLOG which is already used by non-Windows system for this condition.
- Instance variables such as i%property can now be used as the control variable in a FOR command.
- Prior to this release, if you increased the number of users in a license, you had to restart the instance to allocate additional space to store license consumption. To better support customers with frequent increases in license units, an instance

no longer requires a restart if the number of users licensed (since the last restart) is increased. An exception is that if the number of licenses is increased by more than 227,000, then a restart is required.

- The application license facility has been enhanced to report on whether the number of instances operating under a shared license exceeds the number of instances permitted, through use of the **##class(%SYSTEM.License).ApplicationServerLogin()** class method which is described in the class documentation.

4.2 Caché 2016.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2016.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2015.2 and 2016.1.

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

4.2.1 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 2016.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

4.2.1.1 Version Interoperability

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

4.2.1.2 Management Portal Changes

Performance Monitor Headings Changes

In previous releases, the PERFMON performance monitor had misleading headings on the report for metrics on blocks and buffers for big data reads and writes. For metrics 9, 16, and 23, the headings for these metrics indicated that they were for routine reads and writes. In this release, the headings correctly indicate that these metrics are for big data reads and writes. The headings for these metrics are now:

- 9—BdataBlkRd (big data block reads)
- 16— BdataBlkWt (big data block writes)
- 23— BdataBlkBuf (big data block requests satisfied from a global buffer)

SSL/TLS Configuration Changes

In previous releases, the screens for creating or editing an SSL/TLS configuration included a **File containing Certificate Revocation List** field. This field has been removed.

4.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

ECP Application Servers Maintain Connections Independently of TCP KeepAlive Setting

In previous releases, the TCP keepalive setting impacted how long ECP kept open the connection between the ECP application servers and the ECP data servers. If the data server became unavailable, this setting impacted the time it took to redirect the connection. The TCP keepalive no longer has any effect on maintaining connections between the ECP application servers and the ECP data server. If you have been adjusting the TCP keepalive at the operating system level on this basis, for example to allow for timely redirection of application server connections following a mirror primary outage and failover, you need no longer do so.

New Scheme for Naming Journaling Files May Impact Directory Names and Scripts

In order to increase the maximum amount of data that can be journaled per day, this release changes the scheme for naming journal files. The scheme for naming the first 999 journal files remains the same. It is:

```
YYYYYMMDD.NNN
```

Where `YYYYYMMDD` is the date and `NNN` is a number between 001 and 999. In previous releases, Caché could not create more than 999 journal files. In this release, Caché increases the number of digits in `NNN` to accommodate the number. For example, the 1st, 10th, 100th, 1000th and 10000th journal file created on 1/1/2015 would, respectively, have the names 20150101.001, 20150101.010, 20150101.100, 20150101.1000 and 20150101.10000.

In order to handle the longer file names, Caché limits the combined length of directory and prefix (if any) to 208 characters, which is 7 fewer than allowed before. If your sites uses very long directories and/or prefixes, you should change them to shorter ones.

In addition, if you have scripts that assume that journal files are named with three digits after the date, you may have to modify the script. For example, both the numeric and alphabetic sorting order of the file names does not match the chronological order because 20150101.1000 is sorted before 20150101.999.

Revised DataCheck Utility Changes Optimal Throttle Setting

In this release the DataCheck utility has been updated. Consequently, you may want to alter the Throttle setting to achieve optimum performance. The transition between settings is smoother. You may get better performance increasing the Throttle setting without consuming too much additional resources. Experimentation is the best way to determine an appropriate Throttle setting; it can be adjusted at any time and takes immediate effect.

4.2.1.4 Platform-Specific Items

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

Windows

- Windows Installs and the CSP Gateway

  In this release the unattended install uses the same defaults as the regular install. For example, if you use the unattended install, it does not install the CSP Gateway by default. To install the CSP Gateway in an unattended install, specify the CSP Gateway component in the ADDLOCAL property of the command line. To include all features, specify ADDLOCAL=ALL. See "Running an Unattended Install" in the *Caché Installation Guide*.

  In a Windows install, it does not override CSP.ini if the IIS CSP Configuration is present. In previous releases, the install always overrides this file. Consequently, if CSP.ini was misconfigured, then the install will not correct the problem. To ensure that the install overrides CSP.ini, remove the IIS CSP Configuration before installing Caché.

  In previous releases, when the CSP Gateway was installed, it would automatically configure Apache, if it was present on the system. In this release, you must manually configure Apache for the CSP Gateway. For details, see the CSP Gateway Configuration Guide.

- Windows Management Instrumentation (WMI) Support Removed

  This release does not support Windows Management Instrumentation (WMI). Consequently, we do not install the iscprov.dll library.

- cdirectmgr Utility Omitted from Install

  The cdirectmgr utility is no longer built and installed. This Windows utility provided direct control over server operation from a Windows COM/OLE or C++ client program. If you have been using the cdirectmgr utility, you should perform the same function using the cube.

- Revised Warning Message For Abnormal Shutdowns

  The warning message written to the console log during startup following an abnormal shutdown on Windows has been enhanced to report that the emergency (fast) shutdown procedure completed when Windows was shut down without a clean Cache shutdown. If the emergency (fast) shutdown procedure completed, the new console log message at the next startup will be:

  06/18/12-03:30:40:421 (1856) 2 Previous system shutdown was abnormal, system forced down or crashed. Fast shutdown complete.

  If the emergency (fast) shutdown procedure did not complete, the usual console log message at the next startup will be:

  06/18/12-03:30:40:421 (1856) 2 Previous system shutdown was abnormal, system forced down or crashed.

UNIX®

In this release, the installer always restarts Apache when it updates the CSP Gateway binaries. In previous releases, the installer asked the user if Apache should be restarted. This change is only a compatibility issue for scripts or programs that automate the install process and are coded to expect the query dialog.

Mac OS X

- Directory Changes in Installation

  In this release, links for ccontrol and csession are in /usr/local/bin instead of /usr/bin. If your code has explicit reference to these links in /usr/bin, you must update your code to reflect the new location.

- Restriction on Use of Caché eXTreme Features with OS X 10.11

  Two features of Caché eXTreme, Java Globals API and in-memory XEP (eXTreme Event Persistence), cannot be used on OS X 10.11. Caché eXTreme provides external access through Java to Caché multidimensional data storage. You cannot use either the Caché eXTreme Java Globals API or the in-memory XEP on OS X 10.11 unless the OS X SIP (System Integrity Protection) feature is disabled.

4.2.2 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.

4.2.2.1 Routine Compiler Changes

Class Compiler Handles Long Class Names by Truncation

Although Studio prevents you from defining class names that exceed the allowed lengths, it is possible to generate code with names that are too long. In previous releases, this could result in a <NOROUTINE> error while compiling the class. In this release the class name is truncated to the maximum length.

Incremental Compile of Classes Feature Removed

In this release the /incremental compiler switch has been removed. In past releases, this switch instructed Caché to compile modified class methods only. Since the incremental compile provided only a minor reduction in compile time and actually

produced less efficient code, this feature has been removed. If you specify the /incremental switch in this release, the compiler ignores the switch.

Compiler Disallows Illegal Class Names

In this release the class compiler produces an error if a class name contains either "||" (two vertical bars) or "." (period). These symbols were not allowed in class names and typically, cause errors during execution, but were previously allowed by the class compiler.

4.2.2.2 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release

- %CSP.UI.Portal.iKnow — Configurations, Dictionary, Settings
- %CSP.UI.Portal.iKnow.Dialog — AddDomainConfig, DeleteDomain, finderDialog
- %DeepSee.DomainExpert — SQLUtils
- %DeepSee.DomainExpert.queries — IK
- %DeepSee.DomainExpert.utils — ConfigurationManager, CubeUtils, EntityUtils, HtmlUtils, IndexBuilder, LinkedLifeData, SentenceParser, SetUtils, UMLSUtils, Utils
- %DeepSee.DomainExpert.utils.classgen — CubeGen
- %DeepSee.DomainExpert.utils.umls — EntityMappingBuilder, EntityTableGenerator, LocalMappingBuilder
- %DeepSee.DomainExpert.wizard.Dialog — newModel
- %DeepSee.DomainExpert.wizards — HeaderKeyExplorer, ITablesLoadingWizard, TransformationEditor
- %JSON — ContentHandler, Parser
- %SYS — TaskSuper
- %SYSTEM.Security — System
- %UMLS — TFIDF
- %UMLS.Install — CBuilder, Scripts
- %UnitTest — DSQL, ESQL, ITablesRegression, JDBCSQL, ODBCSQL, Utility
- %iFind — Comp, CompToWord, Stem, StemToWord
- %iFind.Find — Abstract, Basic, Semantic
- %iFind.Index — AbstractAttribute, AbstractAttributePos, AbstractDominance, AbstractEntBits, AbstractEntSpread, AbstractEntSpreadRec, AbstractEntity, AbstractPath, AbstractPathVal, AbstractProximity, AbstractWordPos
- %iKnow.Objects.dd.ui — HistoryDialog, SearchHistory
- %iKnow.UI — ITablesStatusView, ImportWizard, LoadingWizard, SetAnalysis, SwirlPortal
- %iKnow.ont — Matcher, Matcher2, SQLStoredProc
- SYS.Monitor — SystemSubscriber

Removed Methods

The following methods were present in the previous version and have been removed in this release

- %CSP.Page
  - HyperEventBody

  - HyperEventFrame
- %CSP.Request
  - Delete
  - DeleteCgiEnv
  - DeleteCookie
  - DeleteMimeData
  - InsertCgiEnv
  - InsertCookie
  - InsertMimeData
  - Set
- %CSP.Stream
  - CharSetGet
  - CharSetSet
  - ContentTypeGet
  - ContentTypeSet
  - ExpiresGet
  - ExpiresSet
  - FileNameGet
  - FileNameSet
  - HeadersGet
  - HeadersSet
  - MimeSectionGet
  - MimeSectionSet
- %CSP.UI.Portal.TaskHistory
  - doDetails
- %CSP.UI.Portal.TaskSchedule
  - ExecuteRS
- %DeepSee.UI.MDXPDF
  - Test
- %iFind.Find.Basic
  - FindResultsLevel
  - FindResultsNodeFindResultsNode
  - FindResultsPattern
  - GetExtent

  - GetWordBitsBeg
  - GetWordBitsFin
  - GetWordBitsMid
  - ParseInput
  - ResolveCompCombArray
  - ResolveCompWords
  - ResolveCompoundToWords
  - ResolveCompoundToWordsAND
  - ResolveStemToWords
  - ResolveStemmedCombArray
  - ResolveStemmedWords
  - SearchBeg
  - SearchFin
  - SearchMid
  - SearchPos
  - SearchPosD
  - SearchPosInternal
  - SearchPosS
  - SearchWrd
  - SearchWrdD
  - SearchWrdS
  - WordBitsToRecIds
  - WordToRecIds
- %iFind.Find.Semantic
  - BuildRecordIds
  - BuildRecordIdsAnd
  - CheckEntBeg
  - CheckEntFin
  - CheckEntGrp
  - CheckEntGrpD
  - CheckEntGrpEnt
  - CheckEntGrpEntD
  - CheckEntGrpEntS
  - CheckEntGrpS
  - CheckEntMid
  - CheckEntWrd

  - CheckEntWrdD
  - CheckEntWrdS
  - CheckGrp
  - CheckGrpD
  - CheckGrpEnt
  - CheckGrpEntD
  - CheckGrpEntS
  - CheckGrpS
  - FindResultsLevel
  - GetEntsByWordPos
  - ParseInput
- %iFind.Index.Basic
  - GetIndexParams
- %iKnow.UI.IndexingResults
  - UpdateSentences
  - updateSentencesClient

Removed Properties

The following properties were present in the previous version and have been removed in this release

- %CSP.UI.Portal.Locks
  - isVMS
- %CSP.UI.Portal.LocksView
  - isVMS
- %Exception.CPPException
  - iStack
- %iKnow.UI.IndexingResults
  - sentFrom
  - sentTo

Removed Parameters

The following parameters were present in the previous version and have been removed in this release

- %Library.CacheLiteral
  - SELECTIVITY
- %SQL.Manager.ShowPlan
  - RESOURCE

- %UnitTest.SQLRegression
  - CORRELATIONLIST
  - DATACLASS
  - DATAFILE
  - DATATAG

Also, the property parameter *SELECTIVITY* is now deprecated and is no longer shown in Studio or in the documentation. If the parameter is present, it has the same effect as in previous releases. Customers are encouraged to remove this property parameter from existing code.

Removed Indexes

The following indexes were present in the previous version and have been removed in this release

- %iFind.Comp
  - ValueIdx
- %iFind.Stem
  - ValueIdx

Modified Methods

The following methods have different signatures in this version of Caché:

- %CSP.UI.Portal.Mirror.Monitor
  - old method: `DrawFailoverInfo () As %Status`
  - new method: `DrawFailoverInfo (RefreshCompress) As %Status`
  - old method: `updateView () As (none)`
  - new method: `updateView (InTimer, NoCompress) As (none)`
- %CSP.UI.Portal.Namespaces
  - old method: `filterChanged (value) As (none)`
  - new method: `filterChanged (value, tableID) As (none)`
- %CSP.UI.Portal.TaskSchedule
  - old method: `filterChanged (value) As (none)`
  - new method: `filterChanged (value, tableID) As (none)`
- %Debugger.System
  - old method: `DebugStub () As (none)`
  - new method: `DebugStub (pWriteOutput) As (none)`
- %DeepSee.Component.Widget.pivot
  - old method: `exportPDF () As (none)`
  - new method: `exportPDF (printMultiple, preserveTempFiles) As (none)`

- %DeepSee.ResultSet
  - old method: `%GetFiltersForCellRange (*pFilters:%String, pStartRow:%Integer, pStartCol:%Integer, pEndRow:%Integer, pEndCol:%Integer, *pMeasure:%String, pAllRows:%Boolean=0, pAllCols:%Boolean=0, pSortCol:%Integer=0, pSortDir:%String="ASC") As %Status`
  - new method: `%GetFiltersForCellRange (*pFilters:%String, pStartRow:%Integer, pStartCol:%Integer, pEndRow:%Integer, pEndCol:%Integer, *pMeasure:%String, pAllRows:%Boolean=0, pAllCols:%Boolean=0, pSortCol:%Integer=0, pSortDir:%String="ASC", &pPluginInfo:%String) As %Status`
- %DeepSee.UI.Analyzer
  - old method: `adjustSizes () As (none)`
  - new method: `adjustSizes (load) As (none)`
- %DeepSee.UserPortal.DashboardViewer
  - old method: `ClearDashboardAutosaveState (pAutosaveItems:%ZEN.proxyObject) As %Status`
  - new method: `ClearDashboardAutosaveState (pAutosaveItems:%ZEN.proxyObject, pDashboardName:%String, pAutosave:%String) As %Status`
- %DeepSee.Utils
  - old method: `%CompressIndices (pCubeName:%String, pVerbose:%Boolean=0) As (none)`
  - new method: `%CompressIndices (pCubeName:%String, pVerbose:%Boolean=0) As %Status`
- %Library.Persistent
  - old method: `%BuildIndices (pIndexList:%CacheString="", pAutoPurge:%Integer=0, pLockFlag:%Integer=0, pJournalFlag:%Integer=0, pStartID:%CacheString="", pEndID:%CacheString="") As %Status`
  - new method: `%BuildIndices (pIndexList:%CacheString="", pAutoPurge:%Integer=1, pLockFlag:%Integer=0, pJournalFlag:%Integer=1, pStartID:%CacheString="", pEndID:%CacheString="") As %Status`
  - old method: `%DeleteExtent (concurrency:%Integer=-1, &deletecount, &instancecount, pInitializeExtent:%Integer=1) As %Status`
  - new method: `%DeleteExtent (concurrency:%Integer=-1, &deletecount, &instancecount, pInitializeExtent:%Integer=1, *errorLog:%Status) As %Status`
- %Library.RoutineMgr
  - old method: `TS (name:%String, &compiletime:%TimeStamp) As %TimeStamp`
  - new method: `TS (name:%String, *compiletime:%TimeStamp, *uptodate:%Boolean) As %TimeStamp`
  - old method: `GetCurrentTimeStamp (&compiletime:%TimeStamp) As %TimeStamp`
  - new method: `GetCurrentTimeStamp (&compiletime:%TimeStamp, &uptodate:%Boolean) As %TimeStamp`
- %SYSTEM.DeepSee

  - old method: `SynchronizeCube (pCubeName:%String, pVerbose:%Boolean=0) As %Status`
  - new method: `SynchronizeCube (pCubeName:%String, pVerbose:%Boolean=0, *pFactsUpdated:%Integer, pReadCommitted:%Boolean=1, pCheckReferences:%Boolean=1, pAsync:%Boolean=0) As %Status`

- %SYSTEM.OBJ
  - old method: `CompileList (&list:%String="", qspec:%String="", &errorlog:%String) As %Status`
  - new method: `CompileList (&list:%String="", qspec:%String="", &errorlog:%String, &updatedlist:%String) As %Status`

- %SYSTEM.iKnow
  - old method: `GetDomainId (pDomainName:%String) As %Integer`
  - new method: `GetDomainId (pDomainName:%String="") As %Integer`

- %Stream.Object
  - old method: `FindAt (position:%Integer, target:%CacheString, &tmpstr:%CacheString, caseinsensitive:%Boolean=0) As %Integer`
  - new method: `FindAt (position:%Integer, target:%CacheString, &tmpstr:%CacheString="", caseinsensitive:%Boolean=0) As %Integer`

- %Studio.Project
  - old method: `realCompile (qstruct, &%errorlog:%String, &itemlist:%String) As %Status`
  - new method: `realCompile (qstruct, &%errorlog:%String, &itemlist:%String, &updatedlist:%String) As %Status`

- %XML.Node
  - old method: `FirstAttributeName () As (none)`
  - new method: `FirstAttributeName () As %String`
  - old method: `LastAttributeName () As (none)`
  - new method: `LastAttributeName () As %String`
  - old method: `NextAttributeName (attributeName:%String) As (none)`
  - new method: `NextAttributeName (attributeName:%String) As %String`
  - old method: `PreviousAttributeName (attributeName:%String) As (none)`
  - new method: `PreviousAttributeName (attributeName:%String) As %String`

- %XML.Writer
  - old method: `Document (document:%XML.Document) As %Status`
  - new method: `Document (documentArg:%XML.Document) As %Status`
  - old method: `Object (object:%XML.Adaptor, tag:%String, namespace:%String, local:%Boolean, className:%String, bare:%Boolean) As %Status`
  - new method: `Object (objectArg:%XML.Adaptor, tag:%String, namespace:%String, local:%Boolean, className:%String, bare:%Boolean) As %Status`

- %ZEN.Mojo.Wizard.MojoWizard
  - old method: `CreateZenMojoTemplate (tempName:%String, pkgName:%String, application:%String, domain:%String, pPageManager:%String) As %Status`
  - new method: `CreateZenMojoTemplate (tempName:%String, pkgName:%String, application:%String, domain:%String, pageName:%String, pPageManager:%String) As %Status`
- %ZEN.Report.Display.report
  - old method: `%DrawToHTML (&context:%String, &XSL:%GlobalCharacterStream, delay:%GlobalCharacterStream, &incell:%Boolean, embedXSL:%Boolean=0, UseInternalXSLT:%Boolean=0, SubReport:%String, MainReport:%String) As %Status`
  - new method: `%DrawToHTML (&context:%String, &XSL:%GlobalCharacterStream, delay:%GlobalCharacterStream, &incell:%Boolean, embedXSL:%Boolean=0, UseInternalXSLT:%Boolean=0, SubReport:%String, MainReport:%String, pXSLTMode) As %Status`
- %iFind.Utils
  - old method: `TestSearchString (pSearchString:%String, *pNormalizedString:%String) As %Status`
  - new method: `TestSearchString (pSearchString:%String, *pNormalizedString:%String, *pDidYouMean:%String) As %Status`
- %iKnow.Tables.Utils
  - old method: `SyncCustomizations (pDomainId:%Integer="", pPackageName:%String="", pAutoPurge:%Boolean=0, pStartSrcId:%Integer="", pEndSrcId:%Integer="", &pRanges) As %Status`
  - new method: `SyncCustomizations (pDomainId:%Integer="", pPackageName:%String="", pAutoPurge:%Boolean=1, pStartSrcId:%Integer="", pEndSrcId:%Integer="", &pRanges) As %Status`
  - old method: `SyncDictionaryMatchCustomizations (pDomainId:%Integer="", pPackageName:%String="", pAutoPurge:%Boolean=0, pStartSrcId:%Integer="", pEndSrcId:%Integer="", pFilter:%iKnow.Filters.Filter="") As %Status`
  - new method: `SyncDictionaryMatchCustomizations (pDomainId:%Integer="", pPackageName:%String="", pAutoPurge:%Boolean=1, pStartSrcId:%Integer="", pEndSrcId:%Integer="", pFilter:%iKnow.Filters.Filter="") As %Status`
- Ens.MessageHeader
  - old method: `NewRequestMessage (&pHeader:Ens.MessageHeader, pMessageBody:%Library.Persistent, &pSessionId:%String) As %Status`
  - new method: `NewRequestMessage (&pHeader:Ens.MessageHeader, pMessageBody:%Library.Persistent, &pSessionId:%String, &pSuperSession:%String) As %Status`
- Ens.VDoc.SearchTable
  - old method: `genGetCodeList (tGetExpression:%String) As %String`

  - new method: `genGetCodeList (tGetExpression:%String, tPreExpression:%String, tPostExpression:%String) As %String`

- Ens.VDoc.XMLSearchTable
  - old method: `genGetCodeList (tGetExpression:%String) As %String`
  - new method: `genGetCodeList (tGetExpression:%String, tPreExpr:%String, tPostExpr:%String) As %String`

4.2.2.3 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

4.2.2.4 Language Binding Changes

There are no compatibility issues in this area.

4.2.2.5 SQL Changes

UNION %PARALLEL is not allowed in INSERT, UPDATE, and DELETE Queries

In past releases, %PARALLEL was not allowed in non-union INSERT, UPDATE, and DELETE Queries, but did not cause an error when you used %PARALLEL in a union INSERT, UPDATE, or DELETE query. The %PARALLEL did not work reliably in these queries. In this release, %PARALLEL is not allowed in either a union or non-union INSERT, UPDATE, and DELETE query. If you had used this construct in a previous release, it would not have caused an error message, but would likely have caused problems. You should remove any %PARALLEL option that is present in an INSERT, UPDATE, or DELETE query.

Fix to Floor Function Changes Returned Value

In previous versions, the ODBC Floor() function did not properly truncate values when the return type was a $list type 6 with an integer value and an SQLType Numeric (2) with zero scale. For example, the function could return “1234.” with an incorrect trailing decimal point. In this release, the Floor() function correctly truncates the number returning “1234” without the decimal point. If you have coded your application expecting this incorrect result and compensating for it, you should remove this correction code. This error only occurred with SQLType Numeric (2) and a zero scale. It did not occur with other types or nonzero scales.

Collation Expressions Are Checked for Errors

In this release Caché checks collation expressions for errors. In previous releases, Caché ignored these errors. In most cases, the error in the collation expression caused an error when using SQL, but it is possible that some collation expression errors did not cause an SQL error. In these cases, you will encounter an error in this release in code that appeared to execute successfully in previous releases.

Aggregate Functions Now Handle Zero-Length String Correctly

Aggregate Functions Now Handle Zero-Length String Correctly In this release, the XMLAGG() and LIST() aggregate functions handle 0-length strings correctly. If code relies on the incorrect way previous releases handled 0-length strings, it should be modified.

JDBC Data Models Have Significant Internal Changes to Support Read Ahead

This release contains many performance improvements for SQL JDBC data models. We do not know of any incompatibilities caused by these changes, but it is possible that implementation detail changes may influence behavior in unusual circumstances. We recommend that you test code that exercises this feature.

4.2.2.6 Changes To Locale And I/O Translation Tables

Functions Converting XML and ODBC Times Locale Change

The %Library.Time methods that are converting from XML and ODBC time formats used the current locale in previous releases. This caused a problem in locales where the decimal separator is not a . (period character) because XML and ODBC always uses a period character for this separator. In this release, these functions always use a period for the decimal separator. If you are calling these methods to set or evaluate XML or ODBC messages, there is no problem. But, if your code uses these methods in another context and expects the current locale's decimal separator, you need to modify your code. The changed %Library.Time methods are: **LogicalToXSD(), XSDToLogical()**, **LogicalToOdbc()**, and **OdbcToLogical()**.

The %Library.Time methods were inconsistent about truncating fractional seconds in previous releases. In this release, by default the methods always retain fractional seconds. There is a new PRECISION parameter that controls the number of decimal places to keep. If PRECISION equals 0, the time value will be rounded to the nearest second.

4.2.2.7 iKnow Changes

Compiling an iFind Index Requires Software License

The use of iFind has always required a valid iKnow license. Starting with this release, Caché will verify whether there is an iKnow-enabled license for your instance when you compile a class with an iFind index and an error will be thrown if you are not properly licensed for using iKnow. There is no strict need to recompile classes with iFind indices after upgrading to 2015.3, though it is generally recommended because it can provide improved optimizations.

Certain iFind Indexes Require Recompiling and Rebuilding

iFind indexes that use stemming or decompounding (INDEXOPTION != 0) have a changed compiled form. Consequently, after upgrading to this release, you must recompile and rebuild any existing indexes that use stemming or decompounding.

We recommend that you use the new %iFind.OriginalWord and %iFind.WordTransformation tables instead of relying on the specific mappings for stems and compounds. While using the specific mappings remain a supported feature, using the new tables is the recommended practice.

4.2.2.8 CSP Changes

There are no compatibility issues in this area.

4.2.2.9 SMTP, XML, Web Services, SOAP, And REST Changes

Converting from JSON to Object Error Handling Changed

In previous releases, the $fromJSON() function would fail silently but not return a valid object. In this release, if the function encounters an error condition, it throws an error. You should ensure that the error does not interrupt your application by including the $fromJSON() call in a try-catch block.

For example, if you had the following code to check that the $fromJSON() function returned a valid object:

```
set obj = ##class(%AbstractObject).$fromJSON(source [,.stats])
if ('$isobject(obj)) {
  w "Compilation error occurred",! q
}
```

You should replace it with a try-catch block such as:

```
try {
  set obj = ##class(%AbstractObject).$fromJSON("[1,2,WE WILL FAIL HERE")
} catch ex {
  w "JSON Parsing error",!
  w "Name = "_ex.Name,!
  w "Code = "_ex.Code,!
  w "Location = "_ex.Location,!
}
```

Behavior in Entering SMTP Password Changed

In previous releases, Caché queried for the SMTP password only a single time. In this release Caché queries twice for the passsword and checks to see that the same password was entered to each prompt. This change eliminates the condition where Caché was attempting to use a null string as a password.

If you have written a script that responds to the password prompt, you must modify the script to handle the second password request.

4.2.2.10 DeepSee Changes

MDX Context is Provided without Encryption

In order to fix problems with some locales, this release does not persist the MDX context and does not encrypt it. If you are using custom action code in %OnDashboardAction that references the MDX context (pContext.mdx), you will have to modify your code and recompile. For example, if you have the following code to decrypt the MDX context:

```
Set tMDX = $$$cspDecode(%session.Key,pContext.mdx)
```

You should replace it with the following that directly references the MDX context without decrypting it:

```
Set tMDX = pContext.mdx
```

If you do not make this change the cspDecode call causes an error.

New Option for Members with Numeric Names

If your cubes contain any levels that are based on source expressions and that have members with numeric names, it is possible to receive incorrect results in a very specific scenario.

The problem scenario is when a query uses an MDX range expression — for example, `([UnitsPerTransaction].[H1].[UnitsSold].&[7]:&[10])` — *and* one of the end point members does not exist. In such a scenario, DeepSee attempts to choose a different end point member. If the level is based on a source expression, DeepSee does not have the necessary information to choose a replacement member correctly and thus would sometimes choose the wrong member. In this scenario, it is necessary to provide additional information to indicate that DeepSee should treat the members as numeric.

InterSystems recommends that you review the level definitions in all your cubes and then do the following:

1. If the level is based on a source property, no change is needed.
2. If the level has members with purely numeric names, modify the cube in Studio and add `castAsNumeric="true"` to that level definition.

   This option is different from the **Sort** option, which does not affect how DeepSee searches for a new end point member.
3. Recompile the cube class.

   It is not necessary to rebuild the cube.
4. If you have previously encountered the problem scenario and if you might have results cached for the problem queries, reset the result cache.

   To reset the result cache, use the **%Reset()** method of %DeepSee.Utils. Note that this method also has an immediate impact on any users and is generally intended only for use during development. Also note that it affects only the current namespace.

pContext.mdx No Longer Encrypted

In previous releases, within the **%OnDashboardAction()** method, the underlying MDX query was provided in encrypted form. Specifically, when the data source is a pivot table, *pContext*`.mdx` contains the MDX query. In previous releases, it was necessary to use code like the following to decrypt the query:

```
set myvariable = $$$cspDecode(%session.Key,pContext.mdx)
```

Now you can use code like the following instead:

```
set myvariable = pContext.mdx
```

Scorecards Based on Pivot Tables with Crossjoins May Need to be Modified

This releases fixes a problem with scorecard widgets that are based on pivot tables with crossjoins. If you have a scorecard widget that meets all the following criteria, then you will need to edit the widget and reselect the properties for the scorecard. The criteria are:

- It is a scorecard widget.
- The data source for the scorecard is a pivot table.
- The pivot table has one level in Columns and one measure in Rows.

You only need to edit the scorecard if all of the criteria are true. You do not need to edit a scorecard that has multiple measures or one that is based on a KPI.

Labels for Pivot Tables with a Single Measure

In prior releases, pivot tables were inconsistent in how they displayed measure headers if there was only one measure. A singe calculated measure always displayed a measure header, but a single non-calculated measure did not display a measure header. Pivot tables are now consistent in how they display measure headers. The default is to display a header for the measure if there is more than one measure (calculated or not calculated). The can be changed by using Measure Options.

If you have a pivot table with a single calculated measure and still want to see the measure header, use Measure Options and change Display Measure Headers to Always.

4.2.2.11 Zen Changes

Improved Handling of Long Labels and Titles May Change Output

This release improves how Zen handles long labels and titles and eliminates overlap between them. If the labels would take up more space than the chart itself, Zen displays the chart with the y axis titles only and suppresses the labels. In some cases, this improved handling of labels and titles changes the way charts are displayed.

Reports Now Require a Body Element

In this release the Zen Reports compiler requires that a report have a body element. In most cases, the error is useful and points to a report that needs to be fixed. In rare cases a valid report might not need a body element. These reports will not successfully compile. You should add a body element to these reports.

4.2.2.12 Zen Mojo Changes

There are no compatibility issues in this area.

4.2.2.13 System and Utilities Changes

Custom Callout Library Functions that Set TZ Need to be Modified

In order to make the time functions reentrant and secure, in certain environments you need to modify the code that sets the TZ environment variable. If you are running on Linux or Windows, you will need to add a call to tzset() after setting the

TZ environment variable in a custom callout library function. The call to tzset() forces the update of the internal variables tzname, timezone, and daylight.

Changes to Output from Integrity Check May Cause Problems if Code is Dependent on Log Format

Improvements made to the integrity check, caused changes in the log output. If you have code that is dependent on the exact text in the log message, you need to revise it.

Lock Command Deferred Unlock ("D" mode) Changes

In previous releases, the behavior of "D" mode unlock was not well defined in the case of nested locks for the same lock node. This has been improved so that, "D" mode unlock respects the state of prior unlocks even when the prior unlocks were nested within an "outer" lock that is unlocked in "D" mode.

4.2.2.14 Unit Test Changes

New Assert Failure Macro

In this release, the new $$$AssertFailure macro unconditionally fails a test and logs the specified message. If the /debug qualifier was specified, the test will break in the debugger at that point. Following the convention of the other assertions, it returns 0, indicating failure. This macro is intended to replace the convention of asserting false, such as in a try block after an exception is expected in a test.

This is only a compatibility issue if you have subclassed the UnitTest class and the new macro conflicts with an extension to the base class.

5

Caché 2015.2

This chapter provides the following information for Caché 2015.2:

5.1 New and Enhanced Features for Caché 2015.2

5.1.1 Important New Features

5.1.1.1 Support For CORS

This version introduces support for handling CORS (Cross-Origin Resource Sharing) requests via REST. Caché now supports accepting the OPTIONS header and adds a new callback method, **HandleCORSRequest(pUrl)** to %CSP.REST. With this, the user can set the required access control headers to achieve the desired CORS compliance. The default implementation does nothing, but example code is provided in the class documentation for %CSP.REST **HandleCORSRequest()**. There is also an additional parameter to control whether or not CORS is allowed for a RESTful application or even for a specific URL.

5.1.1.2 Additional Parallel Query Support

The processing of the %PARALLEL keyword has been enhanced to support, where possible, parallel execution of queries partitioned over non-integer fields and their associated indices. Previously, only integer fields were supported. SQL processing will also now intelligently partition the range of the field based on the actual distribution of data in the table or index for both integer and non-integer fields.

5.1.1.3 Bitslice Index Improvements

The SQL aggregates **SUM**, **AVERAGE**, and **COUNT** now exploit a bitslice index on the aggregated field in the presence of a WHERE clause. This can result in substantial performance gains when aggregating over large subsets of the rows in a table.

5.1.1.4 An Additional Two-Factor Authentication Option

Caché now provides an enhancement to the existing SMS-based two-factor authentication mechanism. This new approach uses the time-based, one-time, password algorithm (TOTP) defined by RFC 6238. An important advantage of this two-

factor authentication option is that the user does not need a network connection so it allows many more applications to take advantage of this additional security layer. Additional details are available here.

5.1.2 Major Developments

The following major, new features have been added to Caché for this release:

5.1.2.1 Rapid Application Development

DeepSee Improvements

This release provides additional improvements in DeepSee capabilities, namely:

Presentation Of Dashboards

New controls provided by the dashboard editor include:

- Title bar: color, opacity, hide/show
- Tool bar: color, opacity, hide/show
- Widget borders: hide/show
- Legend: color, opacity
- Dashboard background: image, color, opacity

Listing Groups

Prior to this release, listings had to be defined as part of a cube. If you wanted to allow someone to create new listings, you had to give them access to edit the cube definition, which would allow them to change anything in the cube definition.

In this release, you can allow someone to create new listings without giving them access to cube definitions. Listings can now be part of a new entity called a Listing Group. A Listing Group is a class that contains 1 or more listings. Listing Groups can be edited with Studio or the new Listing Group Manager that is available in the DeepSee Tools menu. Access to the Listing Group Manager is controlled via security resources. Details on Listing Groups are available in the DeepSee material.

Cube Versions

This release provides a new feature that allows you to upgrade a cube with minimal impact on current users.

Changes to a cube definition require the cube to be rebuilt. Prior to this release, the first step of rebuilding a cube was to always delete the existing cube. So while the cube was rebuilding, queries on the cube were prohibited. When the rebuild completed, you could again query the cube. Depending on the size of the cube and when the rebuild happened, the cube may have been unavailable longer than is acceptable to users.

This release provides a new optional feature for defining cube versions. With cube versions, the existing cube is not deleted at the start of the rebuild. While the new (pending) version of a cube is being built, the current (older) version of the cube is still available for queries. After the new version of the cube finishes building, you run a method that makes the pending cube the current cube and deletes the older cube.

Support For QR Codes

QR (Quick Response) codes have become a popular mechanism for presenting information in an easily-consumed format. They are used for URLs, business cards, setup information, and other short segments. With this release of Caché, application developers can now use the methods of the %SYS.QRCode class to generate their own QR codes. InterSystems itself uses QR codes in the new two-factor, time-based, one-time password (TOTP) implementation.

New iKnow Language Models For iKnow: Russian And Ukrainian

iKnow now includes language models for Russian and Ukrainian, adding to the languages already supported: English, Spanish, Portuguese, German, Dutch, and French. It now also supports stemming, allowing you to work with stems as a level of abstraction above entities in languages with a full case system like Russian and Ukrainian.

Support For Composite Search Strings In iFind

When using iFind for full-text search, you can now submit composite search strings, using parentheses and Boolean operators (AND, OR, and NOT) to express complex criteria in a single search string. For example, passing the string "(long strings) AND (memory OR disk*) AND NOT {error reference}" to the search_index() method in SQL will yield all records that contain the word sequence "long strings" as well as the word "memory" or a word starting with "disk", excluding the records that contain the entity "error reference".

Other Improvements In iKnow And iFind

In this release, iKnow and iFind also include many other improvements and simplifications for working with unstructured data, including the following new features:

- iFind now supports stemming and decompounding for all index levels.
- You can now define and query iFind Basic indices for any language that uses spaces as word separators if no iKnow language model is specified or available.
- Both iKnow and iFind leverage an improved algorithm for calculating the semantic dominance of entities in a record and will consume less disk space for storing these metrics.
- You can now customize attribute detection (negation, sentiment) by supplying marker terms the engine should include when identifying the presence and scope of these semantic attributes.

Support For Islamic Calendar Dates

In this release, Caché adds support for dates expressed according to the Islamic calendar (Hijri). Dates may be specified in accord with either the calculated or observational systems; both are supported. Users can determine and store the observational result to determine the length of a given month. In addition to providing this feature to application developers, DeepSee also takes advantage of this new capability.

5.1.2.2 Performance and Scalability

Database Caching Internal Optimizations

This release expands the scalability of database references on large multi-core systems by substantially reducing runtime contention in the management of the database cache.

Dynamic Allocation Of Global Vectors

This release substantially improves the performance of global references for processes that repeatedly access more than 32 different globals. The performance improvement comes from an increase in the number of global vectors that a process allocates. Each global vector contains information about a different global that the process accessed recently, and this information is used to optimize its next reference to that global. Previously, processes were limited to 32 global vectors. Now, processes allocate global vectors dynamically as needed, up to a maximum of 1000. For more detail, see the Upgrade Checklist.

ECP Concurrency Improvements

This release expands the scalability of large multi-core ECP data servers. Changes made to the server reduce the runtime contention associated with managing the database cache on behalf of ECP clients.

Use Power 8 Encryption Hardware

When running on the IBM AIX® platform, Caché checks for the presence of encryption hardware. If present, it performs encryption using the new Power 8 In-Core option to maximum performance when processing encrypted data. If the hardware is not present, Caché performs the encryption in software as it does on prior releases.

5.1.2.3 Reliability, Availability, Maintainability, Monitoring

Fast Verification Of Recent Database Blocks After Host Failure

This release introduces additional verification within Caché startup on Windows and UNIX® to check for physical database inconsistencies on recovery after a host failure. The verification runs for a limited amount of time so as not to impact availability; it does not change normal operating procedures. If inconsistencies are detected, it is typically evidence of a storage device problem, and startup waits for user intervention in order to protect the system from running in an uncertain state. See "WIJ Block Comparison" in the *Caché Data Integrity Guide* for more information.

Mirroring Improvements

This release of Caché has improved mirroring support and capability, namely:

- Quicker failover

  The time for automatic mirror failover to complete has been improved. The improvements are most substantial in the event where the primary host crashes or becomes network-isolated. When that happens, the failover time is reduced by approximately 15 seconds. In many environments, this represents the largest portion of the failover time and reduces the time from detection of an outage to the databases being available on the new primary to several seconds.

- Simpler authorization of new members when using SSL/TLS

  Async Mirror Members using SSL no longer require manually entering the Distinguished Name to authorize their membership. Instead, they are now added to the mirror in the same way that a second failover member is added: the administrator approves the new async member from the primary.

Non-Root Installations

Starting with this release, the installer for Caché and Ensemble can be run without requiring root or administrator privileges. In this case, the installation belongs to the installing user who has full access to and control of the instance. The registry entries for this instance also belong to this user account only; it is not shared with other users or the root user. Please refer to the Installation Guide for more detailed instructions and information.

5.1.3 Other Items Of Note

In addition, many more lesser improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

5.1.3.1 Support For Node.js Version v0.12

This version of Caché now supports Node.js version v0.12.

5.1.3.2 $SYSTEM.Version Enhancements

This release enhances the $SYSTEM.Version class to provide additional information on the InterSystems components running on the instance in addition to Caché or Ensemble. In the HealthShare case, applications can now see which HealthShare components are installed and what the version of each component is. At this time, Caché and Ensemble do not have components and do not provide this additional information.

5.1.3.3 Support TLS Version 1.1 And 1.2

The OpenSSL libraries that ship with Caché have been updated to support TLS version 1.1 and 1.2. This enables customers to control access to earlier versions, which is particularly important when used with other external standards such as the Payment Card Industry Data Security Standard (PCI DSS).

5.2 Caché 2015.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2015.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2015.1 and 2015.2.

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

5.2.1 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 2015.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

5.2.1.1 Version Interoperability

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

5.2.1.2 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 addition of pages to assist with database mirroring and the separation of roles.

5.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

New Startup Default And Journal Database Encryption Key Management

The Caché database encryption facility has two settings that define default key usage behavior:

1. the database encryption key to be used when creating new encrypted databases (including CACHETEMP), and
2. the key for new encrypted journal files.

The initial values for these settings for a Caché instance are determined the first time database encryption keys are activated, and are derived from material in the encryption key file used. They can be changed at any time using the management portal or ^EncryptionKey. The current values are preserved across Caché shutdowns and restarts.

Important: Scripts that modify database encryption startup properties by modifying the key encryption file will need to be replaced.

Two-Factor Time-Based One-Time Password Authentication

Beginning with this release, Caché supports Two-factor Time-based One-time Password authentication (TOTP). Please refer to Configuring for Two-factor Authentication for more information.

Users who have scripts which create or modify users using **^SECURITY** or the Security.Users class may need to modify the scripts to add the new fields needed by this capability.

Remote System ID Is Now ECP System ID

In the Management Portal and the routine, ^**JRNDUMP**, the field that was formerly labeled "Remote System ID" has been changed to be"ECP System ID". The value in that field is now the *ECP System ID* instead of the *Remote System ID*. The two values differ only in the top few bits, which are zeros in the ECP System ID but typically non-zero in the Remote System ID. Also *ECPSystemID* is now present as a new property of a %SYS.Journal.Record object and a new column in the List query of the class.

System Monitor Changes

System Monitor extends the Health Monitor sensor object to %SYS via 2 new classes: %SYS.Monitor.Sensor() and %SYS.Monitor.SensorItem(). These classes provide alerting not only on Sensors, but can also turn on alerting and set alert values for individual sensor items. For example, DiskPercentFull (critical = 99%, warn = 90%) is the default for all databases. But this can be modified or turned off on an individual database such as /mgr/cachelib.

Also, Notifications and SensorReadings are no longer stored in the multidimensional items *SensorReadings* and *Notifications*. Instead the interfaces provided must be used to access notifications and sensors: **%SYS.Monitor.AbstractComponent.GetNotification()** and **%SYS.Monitor.AbstractSubscriber.GetNextSensor()**, respectively.

Mirroring

- More Stringent Checks For Mirror Member Names

  The following characters (and 2–character combinations) are no longer valid for mirror member names. Their presence will cause a validation failure when the configuration class is saved and an error will be returned to the user to avoid future problems:

  ```
  #    ;    //    /*    =    ^    ~    "    <space>    <tab>
  ```

  Important: Systems that used one of the prohibited special characters in a mirror member name will need to change the member names before upgrading or the upgrade will fail because the .cpf file will fail validation.

- Newly Created Mirrors Use a Longer Quality of Service Timeout

  In this release, the default Quality of Service for newly created mirrors is now 8 seconds. The previous default of 2 seconds was too short to account for certain events at the host or network level that could cause a system to become temporarily unresponsive; this resulted in alerts and unwanted failovers. Existing mirrors that are upgraded will remain unchanged. On some hardware configurations, particularly in virtual environments, the previous default of 2 seconds was too short to account for the timing of events at the host or network level; this caused the system to be unresponsive resulting in alerts and unwanted failovers. Refer to Configuring the Quality of Service (QoS) Timeout Setting for more detail.

- Mirror Journal File Purge Policy

  There are two changes to the journal file purge policy that are designed to allow more flexibility while retaining journal files needed by the mirror.

  1. In prior versions, the backup failover member would not purge any journal file that the primary was retaining. Now, the number of days to retain journal files is honored independently for each member. In this way, members can retain differing amounts of journal history, as long as those journal files are not needed by other members of the mirror.
  2. Also in prior versions, all async mirror members, including disaster recovery (DR) members were allowed to purge journal files as soon as they had been applied to the databases. For DR members, this could cause problems when other members tried to connect to a DR member after it had been promoted. Now, DR members follow the same mirror journal purge rules as failover members.

System administrators should review their configuration to ensure that DR members have sufficient space to retain the configured amount of journal files. A DR member must be configured with enough journal space to act as primary in the event of a disaster. While this requirement is not new in this version, DR members with insufficient space would have gone unnoticed in previous versions until the member was promoted.

- Enforce Restriction For Changing Reporting To DR Async Member Type

  Caché now places restrictions on member type conversion between DR, Read-Only Reporting, and Read-Write Reporting async members. The DR can be changed to Reporting async member without any restriction, and there is also no restriction between Read-Only and Read-Write Reporting members. However, a Reporting type cannot be changed to DR type when there is any mirrored DB has FAILOVERDB attribute cleared. Caché also does not allow Reporting async member to become DR, if the ISCAgent is not running.

- Mirror Names May Not Differ Only In Case

  Caché no longer allows users to create mirrored databases with a name that differs from existing mirrored databases only in the case of the name. For example, if there is an existing mirrored database with a name of "Test", then Caché will prevent the creation of a database with a name of "TEST". Beginning with this release, mirror names will be converted to uppercase.

SHA256 Added As Default Signature Hash

The `DefaultSignatureHash` property has been added to the System.Security class to allow specifying the default hashing algorithm to be used. The default is SHA256. The XML Signature is enhanced to use the new System.Security property to determine the default hashing algorithm if none is explicitly specified. The Management Portal was enhanced also to allow `DefaultSignatureHash` to be displayed and modified on the "System-wide Security Parameters" page. And, the default for AlgorithmSuite in the SOAP Configuration wizard is changed to `Basic128Sha256`.

If the receiver of the signature which is created with the new default of SHA256 does not support SHA256, then the customer will need to either explicitly set the **DigestMethod** in his code or change `DefaultSignatureHash` to be "SHA1". The actual XML Signature is self-describing so that if the receiver supports "SHA256" (which all standard toolkits do), then code will continue to work and be more secure.

Error Text Differences

The text of error messages describing the failure of attempts to access a subscripted variable now provide more detailed information. For example, in previous releases the error would look something like this:

```
<SUBSCRIPT>TEST1+5^TEST1 *a(1,"hello","")
```

when the third subscript was empty. In this release, the error text will look like this:

```
<SUBSCRIPT>TEST1+11^TEST1 *a() Subscript 3 is ""
```

Other variants of the message are:

```
<SUBSCRIPT>TEST1+65^TEST1 *a() Subscript 5 > 511 chars
<SUBSCRIPT>TEST1+24^TEST1 *a() Encoded subscript 1 > 511 bytes
```

These new messages provide more information about which subscript in the array is failing and the reason for the failure.

Shadowing "DejrnCOS" Setting Eliminated

In prior versions, an undocumented setting allowed shadowing to use an alternate (older) implementation of dejournaling referred to as "DejrnCOS". In this release, the setting is removed and no longer available. If shadowing is started with this setting present, it is removed and converted to the standard implementation of dejournaling. Shadows converted in this way will be set to run with zero prefetcher jobs; this most closely matches the runtime characteristics of "DejrnCOS".

Use Of $CHAR(1) As Monthlist Delimiter Invalid

As a result of changes made to accommodate the Hijri (Islamic) calendar capability, the use of $CHAR(1) as the delimiter in the monthlist passed to **$ZDATE**, **$ZDATEH**, **$ZDATETIME**, and **$ZDATETIMEH** is now prohibited. Applications using $CHAR(1) to separate the month names must choose a new character as the delimiter.

Application Licensing Entries Updated

Prior to this version, the $System.License methods **TakeApplicationLicense()** and **ReturnApplicationLicense()** assumed that only CSP server processes would take an application license for a CSP session. This assumption is invalid. The class and the underlying methods have been corrected. A description of the argument list and behavior is in the online documentation of class %SYSTEM.License.

Error Trap Code Runs Using Current Roles

$ZTRAP *** (or **$ETRAP *) indicates that the code in the error trap should be run at the same level as the code that caused the error. In previous versions, when control was dispatched to the error trap code, *$ROLES* was reset to the state it was in when the process began. This means that if application roles were assigned, and the application set an error trap expecting it to have the application roles, that expectation was not met and the error trap might not function properly.

Beginning with this release, the error trap will be run with the roles that were in effect for the execution level at which the error trap was set.

Upgrades Overwrite Manually Updated Zen Mojo

Upgrading a Caché instance may downgrade the installed Zen Mojo version if it was updated manually since the last time the instance was installed or upgraded. Consider the following sequence of events:

- Caché 2015.1 ships with Zen Mojo version <N> preinstalled.
- Zen Mojo version <N+1> is released.
- Caché 2015.1.1 is released and it contains Zen Mojo version <N+1>.
- Zen Mojo version <N+2> is released. The user manually updates the instance of Caché 2015.1 with Zen Mojo version <N+2>.
- Later, that instance is updated to Caché 2015.1.1. The installer updates Zen Mojo back to version <N+1>.

In this case, the user must manually re-install Zen Mojo version <N+2>.

Relationship Storage Moved To Objects

The temporary structures that keep track of relationships among objects (for example, indexes for parent-child tables) are no longer held in process-private globals as they were in prior versions. With this release, they are now allocated as part of the parent and child objects. This increases speed of access at the cost of a small increase in the space taken up inside a partition. Systems whose memory is managed tightly may need to increase the memory allocation for the partition.

Increase In Global Vector Cache Size

Each process caches information about the globals that it uses in order to optimize performance. This cache is referred to as global vectors, and is stored in process memory; it is counted against the memory usage of the process. In previous versions, the number of global vectors per process was limited to 32. With this release, the limit has been raised to 1000. Moreover, it is now dynamically allocated, expanding as different globals are used. (A global vector is used for each global having a different global name, or stored in a different physical database.) This change substantially improves performance when a process repeatedly accesses more than 32 different globals.

Application processes that reference many different globals will now consume more physical memory than in previous versions. The memory allocated to each global vector entry is approximately 750 bytes. Processes that reference the largest number of unique global names can allocate as much as 1000 global vectors, or 750KB. If your application uses many different globals, and is configured with a highly restrictive setting for Maximum Per-Process Memory or $ZSTORAGE, you may need to increase that setting to avoid <STORE> errors. See the article on Caché Process Memory for additional information.

As part of this change, the vectors configuration parameter is now ignored and will be removed in a future version.

TLS v1.1 and v1.2 are Disabled by Default

In Caché 2015.2, you can explicitly enable TLS v1.0, v1.1, and/or v1.2 on the SSL/TLS configuration page. In Caché 2015.1, you could only enable TLS v1.0 on the SSL/TLS configuration page, but TLS v1.1 and v1.2 were enabled by default and could be used.

If you upgrade from Caché 2015.1 to Cache 2015.2 and have existing SSL/TLS configurations, TLS v1.1 and v1.2 will initially be disabled for those configurations. If you want these configurations to use TLS v1.1 and/or v1.2, you must go to the SSL/TLS Configuration page and enable TLS v1.1 and/or TLS v1.2.

To enable TLS v1.1 and/or TLS v1.2 for SSL/TLS configurations:

1. Select **System Administration** > **Security** > **SSL/TLS Configurations**.
2. Select the Edit link for each SSL/TTL Configuration requiring update.
3. Select the TLSv1.1 and/or TLSv1.2 check boxes and select **Save**.

5.2.1.4 Platform-Specific Items

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

Windows

- Installation Permission Change

  Caché installation on Windows adds an access control entry (ACE) to the installation directory that permits full access to all authenticated users. This ACE can be displayed with the **ICACLS** command. whose output looks like this:

```
C:\intersystems\latest NT AUTHORITY\Authenticated Users: (F)
NT AUTHORITY\Authenticated Users:(OI)(CI)(IO)(DE,Rc,WDAC,WO,GW,
    GE,GA,RD,WD,AD,REA,WEA,X,DC,RA,WA)
```

  Installation will now create a Caché instance group named Cache_Instance_<InstanceName> and add an ACE to the installation directory granting the instance group full access to the install directory. The installation process will also make the Caché service account a member of this group, granting it full control over the install directory.

  When a Caché service account is assigned or changed with

```
cinstall setserviceusername <InstanceName> <username> <password>
```

  the new service account will be made a member of the instance group, creating the group if necessary, and adding the ACE to the installation directory if not already present. It is therefore necessary to use `cinstall setserviceusername` to change the Caché service user account rather than the Windows control panel applet.

  It is now possible to impose stricter limits on the access of Windows users to the Caché installation directory by removing the ACE granting access to all authenticated users. The following command can be used to remove the ACE. *%CACHE_INSTALL_DIRECTORY%* represents the path where the Cache instance is installed, for example "C:\InterSystems\Cache".

```
icacls %CACHE_INSTALL_DIRECTORY% /remove "NT AUTHORITY\Authenticated Users"
```

  If the ACE granting "NT AUTHORITY\Authenticated Users" access to the instance directories is removed, users of the local terminal connection to Caché on Windows must be granted access to the installation directory by being made members of the Caché instance group or must be administrators to guarantee that they have proper access to the files in and under the instance's installation directory. The local terminal connection can be recognized by the appearance of `TRM:PID (InstanceName)` in the title bar.

Important: The modifications to the installation directory tree described here should not be applied to instances installed under earlier versions of Cache unless upgraded to a version with this change present.

- Installation To Global Assembly Cache

  As of this release, the installer will no longer place CacheProvider assemblies into the Global Assembly Cache (GAC) by default. To have assemblies installed into the GAC, one must set property `ISCINSTALLINTOGAC` to 1, either by passing a command line argument "ISCINSTALLINTOGAC=1" or by editing the MSI file in Orca or other similar tool.

- Changes To The Behavior Of $NOW On Multi-Core Windows Systems

  Prior to this release, Caché attempted to keep the value of **$NOW** consistent on systems with multiple cores. Under some circumstances, this could lead to more than one CPU attempting simultaneously to reset the time and result in **$NOW** generating a <FUNCTION> error.

  In this release, InterSystems has modified the code that adjusts the microsecond clock on Windows systems so that it will no longer make adjustments visible to all processes running on a Caché instance. Instead, each Caché Windows process will make its adjustments local to the individual process. A consequence of this is that the value returned from **$NOW(...)** on differing Caché processes can disagree with each other. While his may affect some applications that do detailed timing studies, Caché will no longer generate <FUNCTION> errors when calling **$NOW(...)**.

UNIX®

- Recommendation to Set TZ Environment Variable for Linux Platforms

  On Linux platforms, InterSystems recommends that you set the TZ environment variable, which indicates the time zone. If this variable is not set, there can be significant degradation of the performance of Caché time-related functions. For details on setting this variable, see the man page for tzset.

- ABRT and Caché (Red Hat Linux Installation)

  Starting with RHEL 6.0, Red Hat introduced a utility called ABRT (Automatic Bug Reporting Tool) which is intended to intercept program crashes and store the relevant information (including a core dump, if one exists) in a centralized location for further analysis. Under normal circumstances, it does not interact with Caché and one can simply ignore it. However, in the following two situations, it may be desirable to change the default configuration for the needs of the site administrator.

  – A crash in Caché kernel — In the event of a Caché malfunction that causes a process to crash, it is useful to contact InterSystems and, if possible, send the core dump file to be analyzed. Depending on Caché installation options, the user running the process at the time of the crash and permissions of the current directory, a core dump may be created or not. If a core file is not created but the user would like to have one to be analyzed, please see the recommendations below.

  – A crash in the cstat utility — The cstat utility (ccontrol stat) accesses a number of tables in shared memory that can be rapidly changing. Depending on the system load and other circumstances, an inconsistent memory snapshot may cause cstat to crash. This type or crash is harmless and cstat has code to prevent a core dump from being created. However, ABRT may still record the event and copy a core file to its centralized database. To preserve disk space, the site administrator might want to prevent these crashes from being recorded.

  This following provides some basic information about ABRT.

  – There are two major ABRT versions, 1.x and 2.x. To determine which you are running, use the 'abrt-cli' command:

    ```
    $ abrt-cli --version
    ```

  – When a crash happens, check the ABRT directory to see if the event was recorded. To do so, use the applicable version-specific command:

ABRT 1.x	`$ abrt-cli --list`
ABRT 2.x	`$ abrt-cli list`

– Each event (crash) gets its own directory under the base ABRT directory.

ABRT 1.x	/var/spool/abrt
ABRT 2.x	/var/tmp/abrt

– ABRT sends lots of messages to /var/log/messages, so it might be useful to inspect this file (usually the very end) when investigating what happened to a certain crash.

– The online ABRT documentation is available at https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/ch-abrt.html

– If ABRT is somehow interfering with Caché and it is not needed for any other reason, the easiest way to deal with the situation is by disabling it. To do so, use the following command:

```
# service abrtd stop
```

If running ABRT 2.x (RHEL 6.2 or later) you also need to disable this related service:

```
# service abrt-ccpp stop
```

Besides, it is necessary to change the "core pattern" that tells the system the format of the core file name. Normally one uses "core.<pid>" and this is obtained as follows:

```
# cat >/proc/sys/kernel/core_pattern
  core.%p
  <CTRL+D>
```

To permanently disable ABRT (persistent over reboots), also execute the following:

ABRT 1.x and ABRT 2.x	`# chkconfig abrtd off`
ABRT 2.x	`# chkconfig abrt-ccpp off`

– To instruct ABRT to save a Caché core dump, do all of the following instructions that apply:

- Edit the configuration file:

ABRT 1.x	/etc/abrt/abrt.conf
ABRT 2.x	/etc/abrt/abrt-action-save-package-data.conf

- If the Caché installation was done via cinstall script (most common), change the following parameter from 'no' to 'yes': ProcessUnpackaged = yes
- If the Caché installation was done via RPM, change the following parameter from 'yes' to 'no': OpenGPGCheck = no

– To instruct ABRT not to save a cstat core dump. If you enable ABRT to process unpackaged executables, core dumps will be created not only for the Caché executable but also for cstat. If you do not want the core dumps for cstat, do both of the following to exclude them:

- Edit the configuration file (see notes above)

- Update the BlackListedPaths parameter so that it includes the full path to cstat:

  ```
  BlackListedPaths = [retain existing list], <installation dir>/bin/cstat
  ```

– To get the status of the ABRT daemon:

```
$ service abrtd status
```

– To restart ABRT:

```
# service abrtd restart
```

- Asynchronous And Direct I/O For Database Writes On UNIX® Platforms

  Caché has been enhanced to use Asynchronous I/O for writes to database files (the files named CACHE.DAT) on AIX, HP-UX, Solaris, and Mac OS X. This is coupled with automatic use of direct I/O (or concurrent I/O as appropriate) instead of buffered I/O on a subset of these platforms. These changes are designed to optimize the disk I/O characteristics for database files, allowing higher throughput on busy systems at peak load. Of course, synchronization still occurs at key points to guarantee data integrity.

  The use of direct I/O (or concurrent I/O) instead of buffered I/O for database files may require attention in certain configurations. The following describes the change on each UNIX platform along with any considerations that apply:

  – *AIX*

    In addition to now utilizing asynchronous writes, Caché now uses concurrent I/O automatically for database files (concurrent I/O is the recommended form of direct I/O on AIX). Many AIX sites already configure databases to use concurrent I/O via the `cio` mount option. Sites that do not (thus were using buffered I/O in previous versions) should note the following:

    - Check that the sufficient database cache is configured. File system buffering may have allowed Caché to perform acceptably with an under-configured database cache in previous versions.
    - In unusual configurations, where an external command is used to read a database file while Caché is running, the external command may fail because the file is opened for concurrent I/O by Caché. An example is performing an External Backup using the **cp** command instead of a more sophisticated backup or snapshot utility. Mounting the file system with the `cio` option resolves this by forcing all programs to open the files with concurrent I/O.

  – *HP-UX*

    Caché continues to use buffered I/O for database files because HP-UX does not provide program-level control of concurrent I/O (the recommended form of direct I/O on HP-UX). Concurrent I/O can be enabled by mounting the OnlineJFS or VxFS filesystem with the `cio` mount option. Caché uses asynchronous I/O in this version to provide better write characteristics than previous versions, particularly when concurrent I/O is used (even though concurrent I/O is not required).

  – *Solaris*

    Caché now uses direct I/O automatically for database files located on UFS filesystems; this applies to NFS as well although NFS is not a recommended filesystem on Solaris. If using UFS (or NFS), check that sufficient database cache is configured. Filesystem buffering may have allowed Caché to perform acceptably with an under-configured database cache in previous versions. Direct I/O is not relevant to the ZFS filesystem.

    In all cases, Caché now uses asynchronous I/O to write to the database files.

  – *Linux*

Caché continues to use buffered I/O for database files and does not use asynchronous I/O. This is unchanged from previous versions.

– *Mac OS X*

Caché uses buffered I/O, but now uses asynchronous writes for database files. No further considerations apply.

- Caché 2015.2 Upgrade Fails on AIX without Additional Libraries

  You must install additional C/C++ runtime libraries on AIX before installing or upgrading to Caché 2015.2. If these files are not present, the installer will not complete the upgrade.

  Caché 2015.2 AIX is compiled with version 13. According to IBM Support documentation, programs compiled with version 13 and that use encryption features require 3 runtime filesets from runtime package, IBM_XL_CPP_RUNTIME_V13.1.0.0_AIX.tar.Z, on machines where Caché is installed:

  – xlC.aix61.rte 13.1

  – xlC.rte 13.1

  – xlC.msg.en_US.rte 13.1

  Complete details on the package are available at IBM XL C/C++ Runtime for AIX 13.1.

Mac OS X

Programs that use the callin feature and incorporate cache.o must be linked by a C++-aware linker, such as via the C++ compiler, or by manually adding "-stdlib=libstdc++" to the makefile.

5.2.2 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.

5.2.2.1 Routine Compiler Changes

Diagnose Invalid Default Arguments

Previously, if the default value for a formal argument was too long or was not a valid expression, the ObjectScript compiler may not have identified this situation. Instead, a <LIST> error would be thrown when the entry point was called.

Now, the compiler will give a compile error on the entry point. If the entry point is called, it will throw <PARAMETER> or <SYNTAX> depending on how it is called.

5.2.2.2 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release

- %CSP.UI.Portal.Dialog — MirrorAuthorizedAsync
- %CSP.UI.Portal.Mirror.Dialog — AddFailover
- %Collection.MV — ArrayOfObjCN, ListOfObjCN
- %SYS.Monitor — AbstractAsyncSensor
- %iFind — EntityI

- %iFind.Stemmer — AbstractStemmer
- %iKnow.Objects — Utils
- SYS.Monitor — CSPGateway, TestAsync
- SYS.Monitor.Health — AbstractCallback

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CSP.UI.Portal.EncryptionManage	method	SetDefault, doDefault
%CSP.UI.Portal.SQL.QButtons.IndexAnalyzer	method	changeOption, endGather, endResults, startGather
%DeepSee.CubeManager.Utils	method	RegisterCube, ScheduleUpdaterTasks, UnregisterCube, UnregisterGroup
%Library.CacheLiteral	method	Compute
%SQL.AbstractFind	method	%OnClose, %OnNew, GetResult, NextItem, NextItemInclusive, PreviousItem, PreviousItemInclusive, ResultContainsItem
%SYSTEM.iKnow	method	CreateDomainTables
%iFind.Comp	index	NewIndex1
%iFind.Find.Abstract	property	lower
%iFind.Find.Basic	method	CheckPos, FindResults, FindResultsDecompound, FindResultsStemmed, ResolveWordParts
%iFind.Find.Semantic	method	CheckBegEnt, CheckEntBegEnt, CheckEntFinEnt, CheckEntMidEnt, CheckEntWrdEnt, CheckEntWrdEntD, CheckEntWrdEntS, CheckFinEnt, CheckMidEnt, CheckWrdEnt, CheckWrdEntD, CheckWrdEntS, FindResults, FindResultsDecompound, FindResultsStemmed
%iFind.Index.AbstractDominance	index	di
%iFind.Index.AbstractProximity	index	xit
%iFind.Stem	index	NewIndex1

Class	Type	Name(s)
%iFind.Utils	method	GetIndicesClose, GetIndicesExecute, GetIndicesFetch, GetIndicesInClassClose, GetIndicesInClassExecute, GetIndicesInClassFetch, GetIndicesInNamespaceClose, GetIndicesInNamespaceExecute, GetIndicesInNamespaceFetch, RebuildFullIndices
—	query	GetIndices
%iFind.Word	index	NewIndex1
%iKnow.DomainDefinition	method	%ParseExpression
%iKnow.Matching.DictionaryQAPI	query	GetTermElements
%iKnow.Matching.DictionaryWSAPI	method	GetTermElements
%iKnow.Matching.MatchingAPI	method	GetHighlightedSentences
%iKnow.Objects.Source	property	FirstEntityOccurrenceId
%iKnow.Queries.EntityQAPI	query	GetTopGroups
%iKnow.Queries.EntityWSAPI	method	GetTopGroups
%iKnow.Queries.EntityWSAPI.GetLiteral	property	domainId, entOccId
%iKnow.Queries.EntityWSAPI.GetOccurrenceAttributes	property	pEntOccId
%iKnow.Queries.SentenceWSAPI.GetByCrcIds	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetByCrcMask	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetByCrcs	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetByEntities	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetByEntityIds	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetCountByCrcIds	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetCountByCrcMask	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetCountByCrcs	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetCountByEntities	property	sourceidlist
%iKnow.Queries.SentenceWSAPI.GetCountByEntityIds	property	sourceidlist
%iKnow.Queries.SourceQAPI	query	GetByEquivalentIds, GetByEquivalents
%iKnow.Queries.SourceWSAPI	method	GetByEquivalentIds, GetByEquivalents
%iKnow.Semantics.ProximityAPI	method	GetClustersBySource, GetProfileForEntity
SYS.Mirror	method	PurgeAsyncMemberJournalFiles
Security.Applications	property	TwoFactorEnabled

Class	Type	Name(s)
Security.Services	property	AutheEnabledCapabilities, Capabilities, TwoFactorEnabled
Security.System	property	TwoFactorEnabled

Method Return Changes

The following methods have different return values in this version of Caché:

- %Compiler.UDL.TextServices — GetTextAsFile
- %Library.RegisteredObject — %ConstructCloneInit
- %Library.RoutineMgr — getGlobalList
- %SYSTEM.Process — StoreErrorReason
- %UnitTest.Manager — LogStateEnd
- %iKnow.Filters.Filter — GetNextSrcId
- Config.CommonMultipleMethods — Create
- SYS.Database — Copy
- SYS.Shadowing.Shadow — GetLatency, LatencyGet

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CSP.REST	Http500
%CSP.UI.Portal.Mirror.Dialog.SSL	SaveData
%CSP.UI.Portal.SQL.TuneTable	SaveData
%CSP.UI.Portal.SSL	SaveData
%DeepSee.Component.Portlet.abstractPortlet	%OnGetPortletSettings
%DeepSee.Component.Widget.portlet	%GetWidgetPropertyInfo
%DeepSee.Component.pivotTable	%DrawDataTable, %DrawEmptyTable
%DeepSee.DomainExpert.utils.HtmlUtils	genHTML, pathHTML, sentenceHTML
%DeepSee.ResultSet	%GetOrdinalLabel
%DeepSee.UI.Dialog.CubeAdd	SaveData
%DeepSee.UserPortal.DashboardViewer	fetchOptionList, saveDashboard
%Library.EnsembleMgr	map2enslib, unmap2enslib
%Library.File	ComplexDelete, CopyFile, CreateDirectory, CreateDirectoryChain, CreateNewDir, Delete, Exists, RemoveDirectory, Rename, SetAttributes, SetReadOnly, SetWriteable
%Library.Persistent	%OpenId
%Library.RegisteredObject	%ConstructClone, %ConstructCloneInit

Class Name	Method Name(s)
%Library.SQLQuery	SendODBC
%Projection.AbstractProjection	CreateProjection, EndCompile, RemoveProjection
%SOAP.WebService	FileWSDL
%SQL.Manager.ShowPlan	ShowPlan
%SYS.PTools.SQLStats	LogHeader
%SYSTEM.Encryption	Base64Decode, Base64Encode
%SYSTEM.License	ReturnApplicationLicense, TakeApplicationLicense
%SYSTEM.SQL	SetSQLStats
%SYSTEM.Security	Check
%SYSTEM.TSQL	GetAnsiNulls, GetCaseInsCompare, GetQuotedIdentifier
%Stream.Object	%Exists
%Studio.SourceControl.ItemSet	IsValidOSProcessName
%UMLS.Install.Utils	InsertCUI, InsertSUI
%UnitTest.Manager	DebugRunTestCase, LogAssert, PrintErrorLine, PrintLine, RunTest, RunTestSuites
%XML.Reader	OpenURL
%ZEN.Mojo.Plugin.dojo1912DChartHelper	setupChart
%ZEN.Mojo.Plugin.dojo191DijitHelper	addProp
%ZEN.ObjectProjection	EndCompile
%ZEN.Report.Display.tableOutput	%DrawSort
%iFind.Entity	GetStrippedEntityId
%iFind.Find.Basic	ResolveCompWords, ResolveStemmedWords
%iFind.Find.Semantic	CheckEntBeg, CheckEntFin, CheckEntMid, CheckEntWrd, CheckEntWrdD, CheckEntWrdS
%iFind.Utils	RebuildFullIndicesInNamespace
%iFind.Word	GetStrippedWordId
%iKnow.DomainDefinition	%Build, %DropData, %UpdateDictionaries
%iKnow.NativeSupport.UserDictionary	AddAcronym, AddInputFilter
%iKnow.Queries.EntityAPI	GetLiteral, GetOccurrenceAttributes, GetSimilar, GetSimilarCounts, GetValue
%iKnow.Queries.EntityQAPI	GetLiteral
%iKnow.Queries.EntityWSAPI	GetLiteral, GetOccurrenceAttributes, GetSimilar, GetSimilarCounts
%iKnow.Queries.PathAPI	GetValue

Class Name	Method Name(s)
%iKnow.Queries.SentenceAPI	GetByCrcIds, GetByCrcMask, GetByCrcs, GetByEntities, GetByEntityIds, GetCountByCrcIds, GetCountByCrcMask, GetCountByCrcs, GetCountByEntities, GetCountByEntityIds
%iKnow.Queries.SentenceQAPI	GetCountByCrcIds, GetCountByCrcMask, GetCountByCrcs, GetCountByEntities, GetCountByEntityIds
%iKnow.Queries.SentenceWSAPI	GetByCrcIds, GetByCrcMask, GetByCrcs, GetByEntities, GetByEntityIds, GetCountByCrcIds, GetCountByCrcMask, GetCountByCrcs, GetCountByEntities, GetCountByEntityIds
%iKnow.Semantics.DominanceAPI	GetBySource, GetCountBySource, GetCountBySourceInternal, GetProfileByDomain, GetProfileBySource, GetProfileCountByDomain, GetProfileCountBySource
Ens.Projection.Rule	ResolveRuleAlias
Ens.Util.XML.SecuritySignature	ValidateSAML

5.2.2.3 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

5.2.2.4 Language Binding Changes

Read-Ahead Added To TCP/IP For ADO.Net

In previous releases, Caché processed xDBC messages synchronously for many messages. This meant that when the client was processing the server was idle; and when the server was processing the client was idle. This was evident when benchmarking xDBC with the client and server on the same machine. Even though there were two processes, the application would never consume more than 100% of the CPU for the sum of both processes.

In this version, after the client reads the header of a server message "A", it immediately requests the next server message to be sent ("B"). The client continues to process server message "A" as directed by the application, while the server prepares the next portion of the result set. If the application continues to process the result set, at some point it will reach the end of buffer "A" and will read the header of message "B". The process will then repeat with message "B" and the client will request "C" …

Although the processing of the data will still logically occur in the same order, this new behavior may uncover timing issues due to the read ahead nature.

Improve Throughput In TCP/IP Data Transfer

In previous versions, when transferring lists of data to or from an instance, Caché used extensive locking to properly manage shared values used to interate over the lists. These locks slowed performance because of the number of accesses sometimes made to data items. In this version, changes to the way data is accessed eliminate the need to lock reads (gets), and writes are protected by channeling them through a single thread. This removes most locking from putting and getting data from the server via TCP/IP. While this improves performance, new obscured timing issues may be uncovered in applications.

Differences In The Transmission Of Double Values

This release contains minor changes to the format of $DOUBLE values when transmitted to clients running on earlier releases. The changes suppress insignificant digits; values transmitted by the server as “0.93” in previous releases are now sent as “.93”. $DOUBLE values may also differ in the least significant digits from prior releases because of changes in rounding algorithms.

Applications that depend on the exact characters or values being identical to those previously transmitted may need to be changed.

5.2.2.5 SQL Changes

SQL Statistics Logging Changes

The error logging for SQLStats has changed. In previous releases, every error encountered while logging SQLStats would be stored in *^%sqlcq($Namespace, "PTools", "Error", $JOB, counter)*. Storing the error for every process in the global led to the global becoming quite large. The logging error trap has been changed; beginning with this release it only stores the last error for every job: *^%sqlcq($Namespace, "PTools", "Error", $JOB)*.

Also when you purge SQLStats for a namespace, as in

```
Do ##class(%SYS.PTools.SQLStats).Purge("SAMPLES")
```

the errors will also be purged. If you only purge statistics for a specified routine, the errors will not be purged.

Note: Code that loops over the SQLStats errors will need to be changed; the structure has one less subscript level now. Previously, it was

```
^%sqlcq($Namespace, "PTools", "Error", $JOB, counter)
```

and now it is

```
^%sqlcq($Namespace, "PTools", "Error", $JOB)
```

Invalid Dates And Times Now Return Zero From CAST

In prior releases, an attempt to CAST a invalid value to a date or time would result in a “ ”. Beginning with this release, a

```
CAST( x AS DATE )
```

or

```
CAST( x AS TIME )
```

when *x* is a %String, %TimeStamp, or %FileMan.TimeStamp value that is not a valid value will return a zero. Applications that depend on a return value of null must be changed.

Length Of The Empty String Corrected

The SQL empty string has a length of 0 and is now reported as such by **$LENGTH**. Applications expecting SQL $LENGTH('') to return 1 need to be adjusted to recognize that 0 is now be returned.

Change Default For Gathering Statistics

In prior releases, the default value of *$SYSTEM.SQL.SetSQLStatus* was 1 which causes generated SQL code to check if the query should log any additional data about its execution. This slows down execution speed. In this release, the default has been changed to 0 so queries will not generate unneeded checks if they should log any additional data.

If a situation arises where this logging is required, the user can modify the system wide value, or the process specific value and then either purge the query they are interested in or recompile the routine/class which contains the query and then obtain the logging information as before.

Changes To SQL Plan Optimizer

In previous releases, when attempting to choose the best query plan, the optimizer could face a choice between two plans that seemed equally fast. In this release, the calculation of "cost" for plans has been refined and choices that formerly seemed equivalent now differ. In this case, the choice the optimizer makes could, in fact, be worse for some types of queries depending on the metadata present about the tables involved.

SQLStats Generation Now On By Default For Level 2

This release restores the default behavior to generate SQLStats code in routines. The default is to only generate level 2 SQLStats call in the generated code. This is one line in the query OPEN and one in the query CLOSE. SQLStats still has a Level 3 option that gathers info at every Module level. This option is generally only used once a problem query has been identified and further research is need.

Also, in the past, switching from SQLStats = 2 to SQLStats = 3 did not require a recompile of the SQL. With this release, a recompile of SQL is needed when you change SQLStats to level 3.

5.2.2.6 Changes To Locale And I/O Translation Tables

$CHAR(152) Mapping In CP1251

As a convenience to users employing Cyrillic scripts, the undefined code point 152 in Windows Code Page 1251 (CP1251) now maps to itself when being translated to Unicode. Conversely, when U+0098 (152) is converted from Unicode to CP1251 the result is also 152. This change is convenient to Russian users, for example, because it allows the round trip of $C(152) when translating between Unicode and CP1251.

Changes To System Locales Prohibited

The **Modify()** method in the Config.NLS classes will now return an error if you try to edit a system locale or table. The method continues to work as before for custom locales and tables. The system locales are supplied by InterSystems and their contents is assumed; so they should not be edited by customers.

5.2.2.7 iKnow Changes

Non-Relevant Text Now Stored In EntOccId

In this release, non-relevant sentence parts will now be stored in the same data structures as entity occurrences. This enables more transparent SQL access to all constituent parts of a sentence and will more easily accommodate future extensions (new entity types, annotating non-entity sentence parts, ...).

Applications relying on the unwritten (and never documented!) rule that entity occurrence IDs only represented entities will now need to take into account the role of the corresponding entries to ensure they're looking at entities rather than non-relevants.

New Dominance Algorithm Implemented

This release introduces refinements to the definition of dominance and a new algorithm for calculating these entity relevance metrics in both iKnow and iFind. In addition, some of the underlying data structures were rationalized, which may mean up to 15% decrease in storage consumed by fully-generated iKnow domains. This change also introduces new queries for the DominanceAPI and ProximityAPI that support filtering.

New "local dominance" values for entities are calculated based entirely on information from within the document. These values range from 0 to 1 (then multiplied by 1M and rounded for internal efficiencies) and are therefore comparable between documents.

For iFind users, only the actual dominance values will change.

For iKnow users, there are a few more changes accompanying the new algorithm:

- The dominance profile is generated based on a new formula selecting a relevant fraction of the upper quartile (top 25%), always including the highest values.

- Dominance values for CRCs and paths are still generated (as a weighted average of the entities they consist of), but should primarily be used to compare relevance with other CRCs or paths respectively.
- The default values for queries retrieving dominance profiles are therefore updated to retrieve concept profiles rather than aggregate ones.
- The DominanceAPI has a new entry point in the form of a straightforward GetTop() method (supporting filtering). The main entry point for the ProximityAPI is now GetProfile() or GetProfileById().

Note: Some query parameter defaults have been changed, renewing the focus on entities (concepts). Also, dominance values now have a more predictable range and should be easier to handle/compare in iKnow-based applications.

Customized Sentiment And Negation Attributes In The User Dictionary

This release changes the user-visible part of the mechanism to customize sentiment and negation detection through a User Dictionary from the implementation in its trial exposure. By using the three new methods – AddNegationTerm(), AddPositiveSentimentTerm(), and AddNegativeSentimentTerm() – of the %iKnow.UserDictionary class, specific words can be annotated as indicating negation or sentiment. This causes the iKnow engine to treat them appropriately and expands them to the relevant parts of the sentence.

5.2.2.8 CSP Changes

Change Default CSP Error Page

In previous releases, if a CSP application did not have any error page specified, Caché would default to %CSP.Error.cls which displayed a lot of information about the *%request*, *%response*, *%session* objects in order to aid in debugging the problem. In this release, if a customer has not supplied their own error page, the default error page is now %CSP.ErrorLog.cls which just displays a simple message saying that an error has occurred and logs all the information to the ^%ETN error log. Caché does this to ensure we do not leak information when an application error happens.

%CSP.ErrorLog class is the superclass of %CSP.Error, and it has a method, **LogError**, which captures the information so **%ETN** can log this correctly.

5.2.2.9 XML, Web Services, SOAP, And REST Changes

Base64–Encoded XML Output No Longer Includes Line Breaks by Default

In previous releases, Caché by default included line breaks when generating output for base64–encoded XML, specifically for properties of type %Binary or %xsd.base64Binary in XML-enabled classes. This default posed problems when working with non-Caché web services and web clients. In this release, Caché no longer includes line breaks when generating output for such properties.

To include these line breaks for SOAP web service or web client binary output, set the Base64LineBreaks property = 1 or the *BASE64LINEBREAKS* parameter = 1. If both the Base64LineBreaks property and the *BASE64LINEBREAKS* parameter are set, the property overrides the parameter.

Default Output Order of Properties Changed in Case of Multiple Inheritance

In some cases, a given class might be based on multiple XML-enabled superclasses. In previous releases, the corresponding XML schema was created with inheritance from the primary superclass, followed by properties from the second superclass, and so on. The order of properties for export and import was inconsistent with that: Caché *started* with the properties of the right-most superclass. As of this release, by default, the properties for export/import are also generated in left-to-right order. To obtain the behavior of previous releases, specify the *XMLINHERITANCE* parameter as `"right"` in the subclass.

CLASSNAME=1 Required For Serial Classes With Subclasses

When developing applications via the SOAP or XML schema wizards, use the CLASSNAME=1 property parameter for properties which reference a serial class that has subclasses. If CLASSNAME=1 is not added as a property parameter on

the referencing property, attempting to save the object will generate an error. Without this property parameter, the generated code will no longer be able to identify subclasses of the referenced object.

Implement CORS Within REST

Beginning with this release, Caché now handles the HTTP OPTIONS verb even if the REST application requires authentication. Browers do not send authentication headers when making a pre-flight options request and thus previously there was no way of handling this when the application required a login. Now we return the `Allow:` header containing the list of allowed HTTP verbs and the ACCESS-CONTROL-ALLOW-ORIGIN header specifying the origin derived from the ORIGIN header of the request.

The %CSP.REST class now contains a new callback method,, **HandleCORSRequest(pUrl)**. An application can set the required access control headers to achieve the CORS compliance it requires. The default implementation does nothing, but example code is provided in the body of the method.

Note: Applications that have implemented their own CORS handling should be changed to use the new procedures.

5.2.2.10 DeepSee Changes

Pivot Tables Handle Calculated and Regular Measures Consistently

In previous releases, the Analyzer generated pivot tables of different forms for calculated measures than for regular measures, in one specific scenario. That difference has been corrected so that the Analyzer now treats both kinds of measures in the same way.

The specific scenario is a pivot table defined with a level or other item used in **Columns** *and* a single measure (or calculated measure) used in **Measures**. In this scenario, there are two possible pieces of information to include in column headers: the name of the item specified in **Columns** and the name of the measure. In previous releases:

- If the measure was not calculated, the pivot table did *not* include a header that shows the name of the measure.
- If the measure was a calculated measure, the pivot table *did* include a header that shows the name of the measure. This header was displayed below the header for the item specified in **Columns**.

In this release, by default the pivot table does not include a measure header in this scenario. The Analyzer provides a new option that you can use to control whether the measure header is shown; to access this option, click the Options icon in the **Measures** box. This option applies only to the scenario described here.

Restriction on Use of Measures on Multiple Axes

In previous releases, it was possible to create a pivot table that used measures on more than one axis, and the results were indeterminate, but DeepSee did not display an error. In this release, the MDX engine has been adjusted to prevent the existence of measures on more than one axis. The Analyzer, for example, no longer lets users drag and drop measures to multiple axes. If measures are defined on more than one axis, it will throw the error: "Measures cannot exist on multiple axes".

As part of this adjustment, many of the MDX functions have been formalized internally so that, for example, they return either a scalar value or a member. The *DeepSee MDX Reference* provides the returned type for each function.

If a function returns a number, it is effectively a measure and is thus subject to the restriction described here. As a result of that change, some pivot tables that used to work now display the error `ERROR #5001: Two measures cannot be crossjoined (2)`.

Note the following exceptions:

- The engine does not treat the functions %CELL and %CELLZERO as measures, because they are processed as a final step in query execution, and when used properly should not create ambiguity.
- The engine does not treat aggregating functions such as SUM as measures, unless these functions are used with their optional expression argument.

For example, the following queries are legal:

```
>>SELECT Measures.[Amount Sold] ON 0, AVG(Product.P1.[Product Name].MEMBERS) ON 1 FROM HOLEFOODS
                                    Revenue
AVG                                $5,199.61

>>SELECT AVG(Product.P1.[Product Name].MEMBERS,Measures.[Units Sold]) ON 1 FROM HOLEFOODS

AVG                                   997.82
```

In contrast:

```
>>SELECT Measures.[Amount Sold] ON 0, AVG(Product.P1.[Product
Name].MEMBERS,Measures.[Units Sold]) ON 1 FROM HOLEFOODS

ERROR #5001: Measures cannot exist on multiple axes
[%Prepare+174^%DeepSee.Query.query.1:SAMPLES]
```

Better Handling of Legend Padding on Dashboards

This release provides a different implementation for the padding of chart legends on dashboards. If you have existing dashboards that specified non-default sizes or padding for chart legends, you may need to readjust those dashboards. In particular, a chart legend that was previously visible could no longer be visible; if this occurs, use the Dashboard Editor to modify the chart legend size and padding.

Hiding Measure Captions Or Labels

The issue of how to deal with measures in pivot tables has been an ongoing one. Single measures were placed in the slicer instead of on one of the displayed axes (rows/columns). This caused problems with calculated measures since not all made sense in the slicer; so calculated measures were placed on a displayed axis. The difference then between simple and calculated measures was sometimes confusing for users. There was no way to hide the label of a calculated measure, since it would never be placed in the slicer (items in the slicer are never shown in a pivot table).

In this version, the display of measure headers/labels is now a configurable part of the pivot table. The measures dialog has been expanded to include a new setting, "Display measure headers", which has three options: `When More Than 1 Measures Are Used`; `Always`; and `Never`. This setting is saved with the pivot table definition and will be active whenever it's displayed. The default is `When More Than 1 Measures Are Used`.

5.2.2.11 Zen Changes

Client-Side Changes To %ZEN.Component.Group Layout Property Prohibited

To prevent execution of arbitrary class methods from Javascript, the layout property of %ZEN.Component.group and sub-classes (other than %ZEN.Component.menu) are now encrypted client-side and can only be changed on the server. Changes made on the client to the layout property of Zen groups (i.e., zen('myGroup').setProperty('layout','horizontal')) will result in <ILLEGAL VALUE> errors when the component properties are next processed by the server.

Generated Javascript Documents Are Now UTF-8

In prior releases, files generated by Zen were not explicitly encoded; they assumed to be in the locale of the instance on which they were generated. This behavior could create issues if the application assumed the files were in a different encoding (such as Unicode), or used them in a different locale.

Beginning with this release, Zen generates files explicitly encoded as UTF-8. Application that assumed a specific locale must be changed to accommodate the new encoding.

5.2.2.12 Zen Mojo Changes

jQM: Id Attribute Change

The *id* attribute for the layout object *$panel* is deprecated; it should be replaced with the attribute, *key* to be consistent with other layout objects. This means that code which presently reads

```
{type:'$panel', id:'leftPanel', displayMode:'push', children:[]}
```

should be rewritten as

```
{type:'$panel', key:'leftPanel', displayMode:'push', children:[]}
```

In this release, *$panel* will still work if *id* is specified, but InterSystems will remove support for it in some future version.

6

Caché 2015.1

This chapter provides the following information for Caché 2015.1:

- New and Enhanced Features for Caché 2015.1
- Caché 2015.1 Upgrade Checklist

6.1 New and Enhanced Features for Caché 2015.1

The following major, new features have been added to Caché for this release:

- Improved Database Scalability

Furthermore, this version of Caché has been improved and enhanced in the following areas:

- Rapid Application Development
  - Statelesss Service Requests
  - Support For XSLT 2.0
  - DeepSee Improvements
  - Soft Modal Dialogs
  - Zen Mojo
- Performance and Scalability
  - Semaphores
  - Suppport For NGINX Web Server
- Reliability, Availability, Maintainability, Monitoring
  - More Robust Mirroring
- Security
  - Multiple Database Encryption Keys
- Early Adopter Features
  - Predictive Analytics

– Text Categorization

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

6.1.1 Major New Features

6.1.1.1 Improved Database Scalability

This version of Caché has important changes specifically targeted at improving performance and scalability on large multi-core systems. Initial benchmarks show near-linear scalability from 16- up to 64-core systems. From a performance standpoint, InterSystems has simplified the database engine, and introduced more effective data usage algorithms, improving performance for a variety of application access patterns. Most applications will benefit from these changes, but those who desire to take advantage of scaling from between 16 - 64 cores will experience benefits that previously were out of reach.

6.1.2 Rapid Application Development

6.1.2.1 Stateless Service Requests

The new Stateless Service Request capability of the Java Gateway enables simple and efficient calls out to any running Java service that implements the new com.intersys.gateway.Service interface. In addition to performance gains, this gateway feature is not dependent on any saved state for the way Ensemble proxies are mapped to their corresponding Java counterparts. Arguments and results are represented as %Strings on the Ensemble side and byte arrays on the Java side. This allows ANY serialized value to be passed to and from the underlying engine; the actual wire format used for data transport is JSON.

6.1.2.2 Support For XSLT2

In the previous release, InterSystems provided support for XSLT Version 2 as an Experimental Technology Preview feature. This release now provides full support for XSLT Version 2 as a standard part of Caché. XSLT 2 provides a significant advance in capability over XSLT Version 1; among its new features are:

- A more convenient and expressive transformation language plus support for XPath 2.0 and the new XDM (XPath Data Model).
- Strong typing, support for XSL schema types, and the ability to define your own (schema) types. XPath 2.0 also includes a new sequence type not present in Version 1.
- A much more powerful functional language with improved string processing, date and time handling, node-set manipulation, and boolean operators.
- The ability to define and write functions in pure XSLT via the xsl:function instruction.

These, and many other improvements/new features significantly increase the productivity of any XSLT programmer. Strong typing allows many errors to be caught at compile time and to be corrected immediately.

XSLT Version 2 functionality is available via the classes in the %XML.XSLT2 package. For details, see "Performing XSLT Transformations" in *Using Caché XML Tools*.

Note: XSLT2 functionality is not available on the OpenVMS platform.

6.1.2.3 DeepSee Improvements

This release contains many corrections and improvement to DeepSee. Among the more important are:

- DeepSee Query Auditing

DeepSee has always kept a simple log of all MDX queries in ^DeepSee.QueryLog. But for more advanced query auditing, you can now define code that will be executed every time an MDX query is executed. This code has access to the query text and other information about the query. See "Auditing User Activity" in the *DeepSee Implementation Guide*.

- New Time Duration Format

  There is a new format for DeepSee measures that represent a span of time. If the measure represents a duration in seconds, then using the format "%time%" will display the value as hh:mm:ss. For example, 3800 seconds would be formatted as 01:03:20 (1 hour, 3 minutes, 20 seconds). See " Specifying a Format String" in *Defining DeepSee Models*.

- Expansion of %TIMERANGE Function

  The %TIMERANGE function (an InterSystems extension to MDX) now supports relationships. See the *DeepSee MDX Reference*.

6.1.2.4 Soft Modal Dialogs

Caché 2012.2 included a new feature called Soft Modals. Instead of popping up a new window for dialogs, the code instead simply draws a layer on top of the current page DOM and displays the dialog there. Once the dialog is satisfied the is cleared. This feature gives much better behavior especially on tablets and mobile devices where a pop up opens a new tabbed browser instance.

In version 2012.2, this was introduced but not enabled by default. With this release, this switch has been reversed and soft modals is now the default behavior. This can be overwritten by changing the global node at *^%ISC.ZEN.useSoftModals*, or by overriding the default behavior of the **%OnUseSoftModals()** to return false. The change can be done in a base class or on a page by page basis as needed by the customer.

6.1.2.5 Zen Mojo

Zen Mojo was first released in March of 2014 as a standalone kit that could be installed on 2013.1 and later versions of Cache and Ensemble. Beginning with this version, the latest supported version of Mojo will be shipped pre-installed as well. The standalone kit will continue to be made available and will continue to be updated independently of the major platform releases.

For information on Zen Mojo, see the books *Using Zen Mojo* and *Using Zen Mojo Plugins*.

6.1.3 Performance and Scalability

6.1.3.1 Semaphores

This release introduces semaphores to Caché applications. Semaphores provide a fast, efficient mechanism for signaling, control and synchronization among processes, especially between those running on an ECP system. For more about the use of semaphores, consult the class documentation, %SYSTEM.Semaphore, or the technical article on the subject.

6.1.3.2 Support For NGINX Web Server

This release includes support for new Web Server platform for CSP, Zen and Zen Mojo: the NGINX Web Server. NGINX is a high-performance, event-driven web server technology and it is available as another option along with IIS and Apache.

6.1.4 Reliability, Availability, Maintainability, Monitoring

6.1.4.1 More Robust Mirroring

In this release, the mirroring capability now employs a separate system called the **arbiter**. This significantly increases availability and simplifies configuration by supporting safe, built-in, automatic failover under failure scenarios where it was not previously possible: when the primary's host has failed completely or become isolated. When this occurs, the backup can take over because the arbiter confirms that it too has lost contact with the primary; the primary, if it is up, stops acting as primary, eliminating the possibility of both failover members acting as primary.

Furthermore, this release contains the following in a series of ongoing improvements to simplify mirror configuration, deployment, and management:

- The *Mirror Architecture and Planning Guide* streamlines access to all the details to consider in the planning phase of deployment
- More comprehensive information available in the Mirror Monitor
- Better configuration-time detection of problems in SSL setup

6.1.5 Security

6.1.5.1 Multiple Database Encryption Keys

Starting with this release, InterSystems customers may employ multiple encryption keys. This enhancement enables customers that require different databases to have different encryption keys. This capability will also allow database re-encryption in a future release.

6.1.6 Early Adopter Features

This category intended as a way of introducing and providing access to new software capabilities that InterSystems believes will be useful in enhancing the effectiveness of existing and future applications.

The capabilities listed here are ready for use by customers, but they are not yet complete in functionality and design. Customers who take advantage of these capabilities must understand:

- InterSystems makes no guarantee that this feature will achieve general availability in it present form;
- There is no backward compatibility guarantee between this version and future updates, if any;
- Customers may incorporate these capabilities in deployed applications, but must first check with InterSystems to determine best course of action;
- Customers who deploy these capabilities in their applications must commit to upgrading to the final released version.

InterSystems strongly encourages those who incorporate these items in their software to provide feedback on their experiences.

6.1.6.1 Predictive Analytics

This release of Caché introduces runtime support for predictive models defined using the PMML (Predictive Modelling Markup Language) standard, version 4.1. Data scientists or analysts creating predictive models through third-party tools (such as KNIME, R, SAS, SPSS, and others) for data mining and modelling can now import the PMML representation of these models into Caché. Upon compiling them, code will be generated automatically so that applications (or business processes) can invoke the compiled models directly from with Caché without need for a connection or reliance on the third-party tool. The PMML runtime support introduced by Caché enables customers to leverage their existing predictive modelling activities in a Caché-powered production environment.

More about PMML can be found here as well as in the Technical Article accompanying this release.

6.1.6.2 Text Categorization

Text Categorization allows computers to assign labels or invoke actions based simply on a text fragment. This release of Caché introduces a framework to build and execute Text Categorization models. Users can build categorization models based on a set of already categorized text fragments known as the training set. By selecting key features of the text as identified by iKnow and choosing a model type, they can automatically generate the model. These models can then be deployed to a production environment independently of the training set and subsequently invoked from ObjectScript or SQL from a business process or custom application.

User who wish to learn more about text categorization can begin with this key research paper.

6.2 Caché 2015.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2015.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2014.1 and 2015.1.

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

6.2.1 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 2015.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

6.2.1.1 Version Interoperability

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

6.2.1.2 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 addition of pages to assist with Caché mirroring and the separation of roles.

6.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

Change To <STORE> Error Handling

This release changes the way Caché handles low memory conditions to allow an application that experiences a <STORE> error due to exceeding the maximum memory for the process to recover more gracefully without requiring special code in the error handler. From versions 2012.1 through 2014.1, when a process generated a <STORE> error, $ZSTORAGE was automatically increased the maximum memory by 1MB to provide extra memory to handle the error. However, unless the application explicitly reset *$ZSTORAGE* after cleaning up from the first <STORE> error, the next <STORE> error would not have the benefit of the extra memory.

With this release, when an application frees memory after a <STORE> error, it returns to a mode where future <STORE> errors can take advantage of the same additional 1MB that was available to the first <STORE> error. The specifics of the new approach:

- *$ZSTORAGE* no longer changes when the process generates a <STORE> error
- *$STORAGE* may now become negative
- <STORE> errors are thrown based on the value of *$STORAGE* rather than internal memory counts which makes them more predictable

Application error handling code that was making assumptions about *$ZSTORAGE* automatically increasing (per the 2012.1-2014.1 behavior) needs to be reviewed. Furthermore, code that depends on *$STORAGE* being positive needs review. For details on the behavior see Caché Process Memory.

Setting The %-Routine Path Is Now A Privileged Operation

Beginning with this release, changing the process search path for percent routines, with either $ZU(39) or %SYSTEM.Process.SysRoutinePath(), is now a privileged operation. The process must have Write permission for the CACHESYS database. If the percent routine path is set to a non-default value, and ROUTINE^%R is used to operate in a namespace other than the current one, Caché will throw a <PROTECT> error after the operation is complete; the routine path will not be restored.

Any users who are setting a non-standard percent routine path in a non-privileged process will have to provide a secure method for setting the path or change their code to not require a non-standard path.

^cspRule Relocated

In this release, the location of ^cspRule is moved from the default global database to the routine database. Customers will need to recompile their CSP rules as part of the upgrade from earlier versions to 2015.1.

Default Memory Used For Routine Cache Increased

This release, when automatic configuration of memory is selected, the amount of memory allocated to the routine cache is increased by approximately 12MB. As a result of this change, configurations that do not specify this explicitly will consume more shared memory for routines.

Change To %SYS.SYSTEM.GetNodeName()

The method, **%SYS.System.GetNodeName(1)** is documented to return the operating system defined node name for the system it runs on. In previous releases, it would get the node name, and then convert it to uppercase. It now returns the node name as defined by the system, and does not change its case.

Terminal Control-C Input Handling Changes

In previous releases, some terminal input operations dismissed any pending control-C signal before reading from the terminal. Other terminal input operations directly acted on any control-C that was pending before reading from the terminal. Now all terminal read operations will dismiss any pending control-C before reading from the terminal.

If a user was previously used to doing type-ahead for interactive input, that user will have to type a second control-C after the read is commenced; the control-C in the type-ahead buffer is now ignored by an input operation. Any operation that clears the contents of the terminal type-ahead buffer (using, for example, WRITE *-1 or WRITE *-10) will now also dismiss any pending control-C at the same time the type-ahead buffer is emptied.

Note: There is no change in behavior if the user types a control-C directly in response to a terminal input operation.

Argumentless QUIT Command Error Reporting Change

If a function ends with an argumentless QUIT when it should instead return a value, the <COMMAND> error will now report the location of the QUIT (which may be implied at the end of a routine or procedure). `$ZERROR` and the error exception object will have a Data field that says that the function must return a value.

If a routine is detecting this condition by checking for <COMMAND> within itself, that check will fail because the routine name in `$ZERROR` will be different. The test can be changed to look for the text "*Function must return a value".

$INCREMENT Now Signals An Error On Failure

In previous releases, $INCREMENT could fail to signal an error when an expression such as

```
$INCREMENT(X, Y)
```

with a non-zero Y value failed to change the value of X. Such situations were possible, for example, when X was an integer value and Y was a small decimal fraction (less than around 5E-19*X); or when X contained a decimal value and Y was a small $DOUBLE binary fraction (less than around $DOUBLE(1E-16)*X).

Now, a failure to change the value of X when Y is non-zero will always result in a <MAXINCREMENT> error.

Note: A Y value of zero (no increment of X) is valid and no error will ever be signalled in such a case.

Mirroring Changes

- Mirror Database Names Must Pass Validation For Config.Database Dataset Names

  Mirror database names are now stored in uppercase although they can still be entered in either case for compatibility with existing databases from older systems. Existing mirror database names are not affected by this change. This means that it is no longer possible to create two mirrored databases with names that differ only in case. In addition, the only characters valid for mirror database names are the same characters used in dataset names. The leading character must be an alphabetic or an underscore and the rest must be a sequence of alphanumerics, hyphens and underscores.

- Remove Mirror Sync Rules Other Than syncrule_ack

  Starting in this release, mirroring no longer supports multiple acknowledgment modes: Sync, Acked, Committed. The advanced mirroring configuration option, `Acknowledgement Mode`, has been removed. In previous versions, choosing the non-default value of "Committed" could introduce unwanted side effects and does not provide any improvement in reliability or robustness.

- Agent Contact Replaced By Arbiter

  A new mirroring feature employs a separate system called the arbiter to significantly increase availability and simplify configuration by supporting safe, built-in, automatic failover under failure scenarios in which it was not previously possible: when the primary's host has either failed completely or become isolated. When this occurs, the backup can take over because the arbiter confirms that it too has lost contact with the primary; the primary, if it is up, stops acting as primary, eliminating the possibility of both failover members acting as primary.

  In prior versions, when one of these failure scenarios occurred, the backup had no way to distinguish between primary failure and network isolation, and thus could not ensure that the primary would no longer act as primary; by default, the backup could not take over automatically in this situation. Under special conditions, automatic failover could be enabled in this situation by adding a user-provided **IsOtherNodeDown** entry point to the **^ZMIRROR** routine and clearing the **Agent Contact Required for Failover** setting. However, implementing a completely reliable mechanism for confirming the primary down was difficult, if not impossible in most environments. As a result, most users of this method accepted some risk that both failover members could simultaneously act as primary in certain cases, however rare.

  Starting in 2015.1, this mechanism is eliminated and replaced by arbiter-based failover mechanics. The built-in arbiter feature generally provides better response to failure scenarios, in a completely safe manner, and without custom coding. In 2015.1, the Agent Contact Required for Failover setting is removed and the **IsOtherNodeDown** entry point of **^ZMIRROR** will no longer be used. Sites using **IsOtherNodeDown^ZMIRROR** must enable the arbiter in order for automatic failover on primary host failure to occur in 2015.1.

  Upgrading an existing mirror to 2015.1 and enabling the arbiter requires the following steps:

  1. Identify a system to act as arbiter, as described in Locating the Arbiter to Optimize Mirror Availability in the Caché High Availability Guide. (A single arbiter can be shared by multiple mirrors.)

2. Install and configure an arbiter as described in Installing the Arbiter in the same document. This involves minimal software installation and does not require that Caché be installed.
3. Upgrade the mirror members as described in Minimum Downtime Upgrade with Mirroring in the Caché Installation Guide.
4. Configure the mirror to use the arbiter as described in Editing or Removing a Failover Member in the Caché High Availability Guide.

- Corrections For Selecting Async Members When Managing Names And Promoting

  The code in ^MIRROR which shows a numbered list of mirror names as part of promoting and managing authorized IDs for async members has been modified. It now works properly when there are mirror members with numeric names. Previously, there could be confusion about whether the numeric name was a numbered choice in the list or the name of a member. When managing Authorized IDs, adding a member is now a separate operation triggered by entering `Add` (and a valid name) when prompted for a member name to manage. Otherwise, the response is first treated as a member name, and then as (potentially) an choice from the numeric list.

- Changes Made To Mirror Monitor Status Reporting

  In the Mirror Monitor, there are several reporting changes:

  – in the section titled "Mirror Member Status", there are new values reported for "Journal Transfer Latency" and for "Dejournal Latency".

  – the "Mirrored Databases" section now shows new values for "Status".

  – the "Latency" column of "Mirrored Database Status" has been renamed to "Next Record To Dejournal".

  – the renamed column contains the *Timestamp*, *File Name* and *Journal Offset* position in the mirrored database. The fields are separated by commas (,).

  Please refer to the High Availability Guide for detailed explanations of the status values.

- Remove Encrypted Communication Setting From ^MIRROR Utility

  In previous versions, individual mirror members could be configured to use SSL independently of other members by using the `Encrypt Communication` setting on each mirror member. In this release, this setting is removed from the mirror configuration screens.

  InterSystems instead recommends configuring the mirror to require SSL for all members by setting the `Use SSL` option. The `Encrypt Communication` setting is preserved on members during upgrade and can be managed via the Config.MapMirrors class, but systems using this option should change to the recommended `Use SSL` option as soon as practical.

- Prevent Logins To Failover Mirror Members At Startup

  On failover mirror members, the system wide security parameter, `Inactive Limit`, is set to zero each time the mirror member starts up. This parameter controls the interval during which unused accounts are marked as disabled. To prevent accounts from getting disabled on the backup mirror member, this is set to zero to disable the feature. If a disaster recovery member is promoted to a failover member, it will be set to zero at that time as well.

  Sites depending on this to clean up idle accounts must employ a different strategy in a mirror.

- Maximum Mirror Members Is Now 16

  With this release, the maximum number of mirror members within each mirror set is increased from 8 to 16.

  Important: In order to properly implement this change, all the members of a mirror set must upgrade to this version.

Journaling Changes

Before this release, the system was set to freeze on a journal error after 10 seconds of journal inactivity while there is pending journal I/O. In this version, the time limit is increased from 10 seconds to 30 seconds. At the 10-second mark, this warning message is generated in cconsole.log:

```
Journal Daemon has been inactive with I/O pending for x seconds
```

Customers expecting the previous, shorter timeout need to adjust the threshold.

Remove %Service_Object:U From Public Permission In A Normal Security Install

For security reasons, **%Service_Object:U** is no longer a public permission in a normal security installation. **%Service_Bindings** clients will need a role with **%Service_Object:U** to use **%Service_Object**. **%Developer** and **%Manager** have this permission, and others can be added as needed. Users of Studio on a normal security installation must have a role with **%Service_Object:U**.

Rename <NOCATCH> Error To <THROW>; Change Exception Object Handling

The <NOCATCH> error that happens when there is a THROW and no matching TRY/CATCH to catch it has been renamed to <THROW>. Furthermore, the object will now be saved in the special variable $THROWOBJ instead of being immediately closed. This will allow an error trap to access the object to determine how to proceed.

An uncaught exception object is now persistent until $THROWOBJ is cleared. If the exception object holds resources that need to be released, there could be an issue with the longer life of the object. The error trap should set $THROWOBJ to the empty string to close the exception object (and release resources) when it no longer needs it.

Disallow DO/GOTO With +Offset In CACHESYS % routines

It is now illegal to attempt to enter %-routines in the CACHESYS database using the +offset syntax of DO and GOTO. A <NOLINE> error will be thrown when this is detected.

SYS.Agent.VerifyConnection Now Returns A %Status Value

The return value for a failed VerifyConnection() call has changed. Instead of "0", the result will be a %Status value. This should only cause an incompatibility if application code is comparing the return value of this method directly to "0" instead of using a boolean operator. For example:

```
If ##class(SYS.Agent).VerifyConnection() = 0
```

will now not work if VerifyConnection fails. Successful verifications will return $$$OK; that is identical with the value 1, so unchanged.

CacheSSH Return Value Changes

The %Net.SSH.SFTP class has a method **Dir()** that will enumerate the contents of the remote directory. In previous releases, the default behavior was to return directory entry names beginning with ".", such as ".", "..", and ".rc". Now, following the convention of the SFTP interface, these files are excluded by default. The prior behavior can be restored by setting the new, optional 4th argument to **Dir()** to true.

Note: If the application supplies a specific pattern for matching names, that will take precedence over the default behavior.

Class Descriptor Cache More Carefully Managed

Previously, Caché set the maximum size of the shared class cache to a percentage of the number of the routines in the system. If this number was set too large, it could use all of the shared heap for class caching when a lot of unique classes were called. In this release, the shared class limit is set to 900.

This should not be a problem if a small and fixed set of classes are actually used in the system. If larger set of classes is used, the system may not cache all of the classes. This instance may affect performance. If so, the cached class limit can be adjusted manually; please contact the InterSystems Worldwide Customer Support (WRC).

ZSAVE In XECUTE Does Not Switch To New Routine

When run from a command prompt, the **ZSAVE** switches the current routine to the one named in the command. In prior releases, it was possible to execute the statement

```
XECUTE "ZSAVE SomeRoutine"
```

and switch to a new routine. However, when this is done inside an **XECUTE**, it frequently caused crashes because of uninitialized variables.

Beginning with this release, the command does not switch to the new routine.

Error Purge Task Uses Local Databases, Not Namespaces

The *^ERRORS* global is purged periodically by a system task. In prior releases, when the task ran, it purged the *^ERRORS* global for each namespace where the default database was local to the machine on which the task ran. This caused issues in the case when databases existed but there was no namespace defined for it. In this case, the ^ERRORS global would not get purged, and could grow large.

In this release, the task now loops through each database defined on the local system (as defined by the [Databases] section in the .cpffile. For each database which is defined, the following conditions are checked:

- The database is an ECP database.
- The database is mirrored, and the system running the task is not the primary.
- The database is read only.
- The database is dismounted.

If any of these condition are true, the purge is skipped for that database. For the last condition, if the database has not yet been mounted by the system, then Caché will attempt to mount it so the condition can be tested.

Shared Libraries Now Managed With Reference Count

In previous releases, a single call to **$ZF(-4, 2, <LibraryHandle>)** would unload the library immediately, even if more than one class had loaded the library with **$ZF(-4, 1, <PathToLibrary>)** or if the library had been loaded by a call to **$ZF(-6, <index>, <Function>, <arg1>, <arg2>, ...)**.

Now, a reference count of the number of times a library has been loaded with **$ZF(-4, 1)** is maintained. Each call to **$ZF(-4, 1, <PathToLibrary>)** increases the reference count. Each call to **$ZF(-4, 2, <LibraryHandle>)** decrements the reference count, and unloads the library if the reference count goes to zero.

The behavior of **$ZF(-4, 2)** with no library handle argument, which unloads all libraries, is unchanged. All libraries are unloaded immediately without regard to the reference count.

There is now a degree of independence between loading libraries by **$ZF(-4, 1, <PathToLibrary>)** and by **$ZF(-6, <index>, <function>, <args>)** where the library index was previously established by a call to **$ZF([-4, 5 or 7], <index>, <path>)**. Libraries loaded with **$ZF(-4, 1, <path>)** are not unloaded by a call to **$ZF(-4, 4, <index)** and libraries loaded implicitly with **$ZF(-6, <index>, <function>, <args>)** are not unloaded by **$ZF(-4, 2, <LibraryHandle>)**. However **$ZF(-4, 2)** with no `<LibraryHandle>` argument will unload all libraries regardless of how they were loaded.

%SYS.ProcessQuery – ClientNodeName, StartupClientNodeName, And Current Device Lengthened

In previous releases, the value for *ClientNodeName*, and *StartupClientNodeName* were truncated to 32 characters, and the value for *Current Device* was truncated to 24 characters. In this release, all three can now be to 64 characters in length before truncation occurs.

Clarification For Unified Trigger Mechanism

Unified trigger support for Objects and SQL was introduced in version 2014.1. When a field or property that is a stream is referenced in a trigger definition, the value of the reference is the OID of the stream.

- For a trigger designated as BEFORE INSERT or UPDATE, if a new value is specified by the INSERT, UPDATE or %Save operation, {StreamField*N} will be either the OID of the temporary stream object, or the new literal stream value.
- For a BEFORE UPDATE trigger, if a new stream value is not specified for the field or property, {StreamField*O} and {StreamField*N} will both be the OID of the current field or property stream object.

$SYSTEM.TSQL Validation Added

The "Set" methods in $SYSTEM.TSQL now validate the parameter values passed to them. The "Get" methods no longer take parameters because they were redundant.

^%SYSMONMGR Menu Change

Option #4, "Manage Sensor Readings" in ^%SYSMONMGR has been relabeled as "Manage Debug Data".

The Character Sequence "||" Is Not Allowed In Global Mappings

When defining global mappings, the character sequence "||" is not allowed anywhere in the mapping.

Java Virtual Machine (JVM) Version Requirements

As of this release, Caché functions that depend on the Java Virtual Machine require a JVM version of 1.7 or later.

Docserver Application Now Disabled

Beginning with this release, the application /csp/samples/docserver will be disabled by default on upgrades and new installations.

6.2.1.4 Platform-specific Items

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

Windows

- Make Instance Values Consistent With UNIX®

  Instance status values displayed by "ccontrol all" have been changed on Windows to be consistent with the values displayed on UNIX®. Specifically, the status text "ab" has been changed to " ". The space value indicates the instance is probably partly up without ^STU completing and logins are disabled. (The complete set of instance status values are displayed by "ccontrol help").

  Instance status values displayed by "ccontrol list" have also been changed on Windows to be consistent with the values displayed on UNIX® and OpenVMS. Specifically, the status abbreviated text has been expanded to full words. The status text is documented in the web page displayed by "ccontrol help".

- Change To Pattern-Matching For 8–Bit Locales

  In prior versions, the undefined characters in the Windows Code Pages (CP12xx) which lay in the range 128-159 were matched only by the pattern code, "1E" (match any character). Beginning with this release, they will also be matched by "1C" (match a control character). The purpose of this change is to allow programs to easily filter these characters in order to avoid garbled terminal output. By code page, the specific character affected are:

  – CP1250: 129, 131, 136, 144, 152
  – CP1251: 152
  – CP1252: 129, 141, 143, 144, 157
  – CP1253: 129, 136, 138, 140, 141, 142, 143, 144, 152, 154, 156, 157, 158, 159
  – CP1255: 129, 138, 140, 141, 142, 143, 144, 154, 156, 157, 158, 159
  – CP1257: 129, 131, 136, 138, 140, 144, 152, 154, 156, 159

CP1256 has no undefined characters in the range 128-159; and CP1250 already had its undefined characters marked as non-printing (but not as control characters). For information on pattern-matching, see the pattern-matching operators in Using Caché Objectscript guide.

- Caché Service Account Need Not Be In Windows Administrator Group

  The Caché service account, in which most Caché processes run on Windows, no longer needs to be a member of the Windows Administrators group. The account designated as the Caché service account during installation will be made a member of a "CacheServices" group. The CacheServices group grants its members a limited set of rights needed to start, stop and control the Caché instance. Membership in the group is also granted when the Caché service account is changed with the

  ```
  cinstall setserviceusername <InstanceName> <username> <password>
  ```

  This change should have no effect if you choose to run Caché as the local system account or as an administrative user account. If you elect to run Caché under a non-administrative user account, you may need to insure that the Caché service account (or the CacheServices group) has been granted sufficient (read/write/execute) access to all directories where databases, journal, and log files reside, and has appropriate access to the files themselves.

UNIX®

- Recommendation to Set TZ Environment Variable for Linux Platforms

  On Linux platforms, InterSystems recommends that you set the TZ environment variable, which indicates the time zone. If this variable is not set, there can be significant degradation of the performance of Caché time-related functions. For details on setting this variable, see the man page for tzset.

- Non-Root Installation No longer Permitted

  Beginning with this release, Caché installation must be done by the root userid.

Red Hat Enterprise Linux 7 for Power System-64

This operating system version on this platform does not currently support Perl or Python.

SUSE Linux 12 Linker Change Affects Light C++ Binding

A change in the linker on SUSE 12 systems prevents Light C++ binding applications from building. The SUSE 12 linker now requires application makefiles to explicitly specify dependent libraries of other libraries with which they link, even if those other libraries specified the dependent libraries when they were linked. In this case, libcachet.so depends on libpthread.so, so starting with SUSE 12 Linux, both libraries must be explicitly specified. Therefore, one of the following changes is needed to successfully build the application:

1. Add `–lpthread` to the library specifications in the ld or g++ command line in the makefile used to build the application. For example, if using a makefile based on one of the Master.mak files installed with Light C++ Binding sample applications, change the line:

   ```
   CACHETLIB = -L$(CACHETPATH) -lcachet
   ```

   to

   ```
   CACHETLIB = -L$(CACHETPATH) -lcachet –lpthread
   ```

2. Set the environment variable *MULTITHREADED* to 1 in the environment in which make is invoked. In the Bourne shell, the syntax is:

   ```
   export MULTITHREADED=1
   ```

Either option resolves the issue; they do not conflict with each other, so both options can be used together.

OpenVMS

- Change Naming Of JOBs In OpenVMS

  In OpenVMS, the JOB command allows you to specify a process name. If this process name already exists, the JOB command would throw a <SYSTEM> error, and as a result also produce a stack dump log (a CERRSAVE-pid.LOG file). Beginning with this version, the error has been changed to an <INVALID ARGUMENT> error; no log file is produced.

  This is an incompatible change for those applications that check for a specific return code.

- Java-Dependent Facilities Now Unavailable

  This release stipulates that the minimum required version for Java is 1.7. Since the last Java release for OpenVMS is version 1.6, all capabilities (such as Zen Reports) that depend on access to a Java Virtual Machine (JVM) are no longer available.

Mac OS X

InterSystems has followed the Apple switch from the use of GCC to Clang/LLVM; however, it was not clear which C++ Standard Library should be used. So Caché supports the use of two versions:

- libc++ – LLVM's newer, C++11 compliant version
- libstdc++ – The GNU version

Applications can use both versions in the same program, but cannot mix and match them; that is they cannot pass a std::string from one to a function in the other. Because Caché uses custom char types it explicitly uses the older GNU libstdc++

Any C++ users linking with ODBC, or any Objective-C users using ObjectiveCache, should see no differences; however C++ binding users may need to add "-stdlib=libstdc++" to their build flags if they use any of the C++ binding APIs that handle classes from the `std::` namespace (such as `std::string`).

6.2.2 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.

6.2.2.1 Routine Changes

Rename %SYS.GSIZE to %GSIZE

Applications that called %SYS.GSIZE will need to go back to calling the original name, %GSIZE, making it again part of CACHELIB. Users that do not have privileges for a database should not be able to see what globals are present, or their sizes.

6.2.2.2 Routine Compiler Changes

Changes To Handling Of Comments Inside Statements

Consider the following two statements:

```
          _________A________      _________B________

Line 1    If (i = 1) {}           If (i = 1) {}
Line 2    ElseIf (i = 2) {}       ElseIf (i = 2) {}
Line 3    // comment              ;;  comment
Line 4    Else Set I = 5          Else Set I = 5
```

The “A” statement sequence is illegal and will not compile because it contains a Caché IF...ELSEIF statement followed by a legacy ELSE statement; the comment on Line 3 of A is ignored, so combining Caché IF syntax with a legacy, line-oriented IF syntax is illegal.

Prior to this release, the “B” sequence would compile because the “;;” comment on line 3 terminated the syntax analysis of the Caché IF statement which allowed the legacy ELSE syntax to be accepted as legal.

Now “;;” comments are accepted following a close bracket and the “B” statement sequence is analyzed just like the “A” sequence. This will now generate a syntax error if it is compiled.

6.2.2.3 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release

- %CSP.UI.Portal — UsersParameters
- %CSP.UI.Portal.Dialog — DatabaseWizardEMS, ServiceConnections
- %CSP.UI.System — AuditPane, AuditSearchPane, ManageAuditPane
- %DeepSee.DomainExpert.components — MetadataValues
- %Net.Remote.Java — JDBCGateway
- %SYS — Monitor
- %iKnow.Classification — ClassifierProjection, LinearBuilder, NaiveBayesBuilder
- %iKnow.Classification.Definition — Term, Transformation
- %iKnow.Compiler — LexrepIdeographicStateOutputFunction, MetadataTable, PreprocessOutputFunction
- %iKnow.Objects.tfidf — MetadataAnalyzer
- %iKnow.Queries — EquivAPI
- %iKnow.UI — ITablesLoadingWizard
- %iKnow.UI.Dialog — Classifier
- %iKnow.ont — SourceMatches, TableGenerator, TableManager
- SYS — Volume
- Security — Common

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CSP.Documatic.CubeInfo	method	DrawOverLvl, DrawOverProp
%CSP.Portal.Template	method	ZStripW
%CSP.Portal.standardDialog	method	ZStripW
—	property	EMSGroupName, IsEMSGroup
%CSP.UI.Component.RoleMemberTab	property	SelectedItems, emsGroupName
%CSP.UI.Component.SQLPrivileges	property	emsGroupName

Class	Type	Name(s)
%CSP.UI.Component.SQLTables	property	emsGroupName
%CSP.UI.Component.abstractRoleTab	property	emsGroupName
%CSP.UI.Portal.Applications.Web	property	PermittedClassTitle
%CSP.UI.Portal.Database	property	isMirror
%CSP.UI.Portal.DatabaseTemplate	method	LoadErrorMessage
—	property	OldReadOnly, RestartMessage
%CSP.UI.Portal.Dialog.EncAddAdmin	method	ZStripW
%CSP.UI.Portal.Dialog.RemoteDatabase	method	ServerIsManaged, changeServerEMS
—	property	MsgNotManaged
%CSP.UI.Portal.Dialog.Service	method	showConnections
%CSP.UI.Portal.EncryptionDatabase	method	doSetDefault
—	property	lblSetDefault
%CSP.UI.Portal.Mirror.Dialog.AsyncEdit	method	DrawEncrypt, SaveData, doSave
%CSP.UI.Portal.Mirror.EditFailover	method	DrawMemberInfoTable
—	property	OtherMemberECP, OtherMemberName, OtherMemberPrivateAddress, ThisMemberECP, ThisMemberName, ThisMemberPrivateAddress
%CSP.UI.Portal.Mirror.JoinAsync	method	changeSSL
%CSP.UI.Portal.NLS	method	saveLocale
%CSP.UI.Portal.SQL.Home	property	FILTER
%CSP.UI.Portal.Shadow	method	refreshTable
%CSP.UI.Portal.Template	property	EMSGroupName, IsEMSGroup
%CSP.UI.Portal.ViewLog	property	FILENAME
%CSP.UI.Portal.iKnow.Dialog.AddDomainConfig	method	browseSelect, changeLang
%CSP.UI.System.MappingsAPI	method	DrawMapJS, PrepareInsert
%CSP.UI.System.Mirror	property	MemberStatusList, MirroredDatabaseList
%DeepSee.DomainExpert.queries.IK	method	%DefAtuiClassName
%DeepSee.PMML.Builder.AbstractBuilder	method	GeneratePMMLAsString, GeneratePMMLClass

Class	Type	Name(s)
%DeepSee.PMML.UI.ModelTester	method	BuildRowDetailsQuery, CheckSQL, DrawCategorizedText, DropTestResults, ExportToCubeDefinition, GetConfusionMatrix, GetTestResults, OnGetSQLAccuracy, OnGetSQLCatValDetails, OnGetSQLErrors, OnGetSQLScatter, changeModel, export, launchExport, onCatValElementClick, onTestModel, setConfusionMatrixValue, showText, test, updateCatValDetails
—	property	currentActualValue, currentPredictedValue, currentRowId, customSQL, definitionClass, highlightedSentencesOnly, mode, modelType, testId, verParamPrecision, verParamZeroThreshold
%DeepSee.Report.UI.BuildLIDR	method	reallySaveDCR
%DeepSee.Report.UI.abstractIconBar	property	titleList, valueList, widgetList
%DeepSee.Report.UI.alignIconBar	property	titleList, valueList, widgetList
%DeepSee.Report.UI.arrangeIconBar	property	titleList, valueList, widgetList
%DeepSee.Report.UI.editIconBar	property	titleList, valueList, widgetList
%DeepSee.Report.UI.layerIconBar	property	titleList, valueList, widgetList
%Installer.InstallerWizard	method	ZStripW
%Library.CacheSQLStorage	method	%BuildIndices
%Library.CacheStorage	method	%BuildIndices
%Library.Persistent	method	%CheckUnique
%Net.Remote.Gateway	method	%PopDeviceStack, %PushDeviceStack
—	property	DeviceStack
%Net.Remote.Java.XSLTGateway	method	ConnectGateway
%Net.Remote.ObjectGateway	method	GatewayState
%SYS.Monitor	method	RefreshSensors, SaveSensors
%SYS.Task.PurgeZENReports	property	FileLifeTime
%SYSTEM.Process	method	KillAllPrivateGlobals
%SYSTEM.WorkMgr	method	SignalAll, StopWorkers
%UMLS.Utils	method	getLanguges
%ZEN.Portal.assistedText	method	ServerCallOnUpdateDataServer
%ZEN.Template.AddInWizard.Template	method	ZStripW

Class	Type	Name(s)
%ZEN.Template.WebServiceWizard	method	ZStripW
%iKnow.Classification.Builder	method	%AddTerm, %GetCategoryPosition, %GetTermPosition
—	property	Categories, Terms
%iKnow.Classification.Definition.ClassificationMethod	property	localTermMetric
%iKnow.Classification.Definition.Term	property	value
%iKnow.Classification.IKnowBuilder	method	%GetCategoryFilters
%iKnow.Classification.LinearBuilder	property	CategoryGlobalTermWeights, CategoryLocalTermMetric, CategoryLocalTermWeights, CategoryNormalization
%iKnow.DeepSee.UI.Analysis.Entities	method	AnalyzeString, GetChartData, GetLabelX, RemoveEntity, UpdateCharts, getChartDataClient, getLabelXClient, onChangeMetricClient
—	property	termIndex
%iKnow.ont.Matcher	method	GetATUI
Config.ComPorts	parameter	EMSLISTQUERY
Config.CommonMapMethods	parameter	EMSLISTQUERY
Config.CommonMethods	parameter	EMSCHANGEBIT, PROPERTIESMAYBEINCPF
Config.CommonMultipleMethods	parameter	EMSLISTQUERY
Config.Databases	parameter	EMSLISTQUERY
Config.Debug	parameter	EMSLISTQUERY
Config.DeviceSubTypes	parameter	EMSLISTQUERY
Config.Devices	parameter	EMSLISTQUERY
Config.ECPServers	parameter	EMSLISTQUERY
Config.LicenseServers	parameter	EMSLISTQUERY
Config.MapGlobals	parameter	EMSLISTQUERY
Config.MapPackages	parameter	EMSLISTQUERY
Config.MapRoutines	parameter	EMSLISTQUERY
Config.MapShadows	parameter	EMSLISTQUERY
Config.Mirrors	property	ACKRequirement, AgentContactRequiredForTakeover, TroubleTimeout
Config.Namespaces	parameter	EMSLISTQUERY
Config.Shadows	parameter	EMSLISTQUERY

Class	Type	Name(s)
Config.SqlSysDatatypes	parameter	EMSLISTQUERY
Config.SqlUserDatatypes	parameter	EMSLISTQUERY
Ens.Enterprise.Portal.MonitorStatus	method	DrawMsgBankLinks, showDetails
Ens.Util.Documentation	method	BuildVMSFileURL
SYS.Monitor.Health.SensorObject	property	MaxValue
Security.Applications	property	HyperEvent

Method Return Changes

The following methods have different return values in this version of Caché:

- %iKnow.Utils.MaintenanceAPI — BlacklistContainsElement
- %iKnow.Utils.MaintenanceQAPI — BlacklistContainsElement
- %iKnow.Utils.MaintenanceWSAPI — BlacklistContainsElement
- SYS.Agent — VerifyConnection

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CSP.UI.Component.ApplicationRoles	AssignRole, RemoveAllRoles, RemoveRole
%CSP.UI.Component.RoleMemberTab	AssignRoles, RemoveAllRoles, RemoveRole, doRemoveAllRoles, doRemoveRole
%CSP.UI.Component.SQLPrivileges	AssignPrivs, RemoveAllPrivs, RemovePriv
%CSP.UI.Component.SelectBoxUtils	DrawArrows, DrawAvailableList, DrawSelectList
%CSP.UI.Component.UserRoles	AssignRoles, RemoveAllRoles, RemoveRole
%CSP.UI.Portal.Applications.PrivRoutine	AddRoutine, RemoveRoutine
%CSP.UI.Portal.Applications.Utils	Delete
%CSP.UI.Portal.Applications.Web	BuildAutheArray, CopyApp
%CSP.UI.Portal.Audit.EventsTemplate	ChangeStatus
%CSP.UI.Portal.Audit.UserEvents	Delete
%CSP.UI.Portal.Config.Advanced	SaveData
%CSP.UI.Portal.Config.AdvancedList	DeleteData
%CSP.UI.Portal.Config.ValueEditor	SaveData
%CSP.UI.Portal.Database	changeJournal, changeReadOnly
%CSP.UI.Portal.Dialog.DatabaseDelete	DisableEnsNamespace
%CSP.UI.Portal.Dialog.DatabaseWizard	validateSize
%CSP.UI.Portal.Dialog.EncAddAdmin	SaveData
%CSP.UI.Portal.Dialog.LicenseActivate	Activate, DrawCurrent, PrepareActivate

Class Name	Method Name(s)
%CSP.UI.Portal.Dialog.RemoteDatabase	CheckDBName
%CSP.UI.Portal.Dialog.SQLView	SaveData
%CSP.UI.Portal.Dialog.ShadowDBMapping	MapShadowExists
%CSP.UI.Portal.Domains	Delete
%CSP.UI.Portal.ECPDataServers	DeleteServer
%CSP.UI.Portal.EncryptionDatabase	SetDefaultKey
%CSP.UI.Portal.EncryptionManage	doAdd
%CSP.UI.Portal.License.Utils	DrawFile, DrawLicense, GetLicenseInfo
%CSP.UI.Portal.LicenseServers	DeleteData
%CSP.UI.Portal.NLS	CopyNow, DeleteNow, GetLocaleDesc, GetLocaleDescription, ReloadDefault, SaveNow
%CSP.UI.Portal.Namespace	doNew
%CSP.UI.Portal.PhoneProviders	Delete
%CSP.UI.Portal.Resources	Delete
%CSP.UI.Portal.Roles	Delete
%CSP.UI.Portal.SQL.Home	SaveFilter, updateTreeItems
%CSP.UI.Portal.Shadow	DeleteMap
%CSP.UI.Portal.Shadows	DeleteData
%CSP.UI.Portal.Users	Delete
%CSP.UI.Portal.iKnow.Configurations	doClose, doDelete, doSave
%CSP.UI.Portal.iKnow.Dialog.AddDomainConfig	SaveConfig
%CSP.UI.System.ExpResultPage	CopyMapsFrom
%DeepSee.Component.pivotTable	GetCurrentQueryText
%DeepSee.PMML.Model.NaiveBayes	GetLikelihoods
%DeepSee.Report.UI.QueryBasedDSS	updateGroupings
%DeepSee.Report.UI.schemaEditPanel	addDBItemFromDrag, autopopulateDBItem, makeNode
%DeepSee.TermList	%ImportCSV, %WriteCSVRecord
%DeepSee.extensions.modelling.Processor	applyModel, applyOperation
%Dictionary.ClassDefinition	%LockId
%Dictionary.CompiledClass	%LockId
%Dictionary.CompiledConstraint	%LockId
%Dictionary.CompiledConstraintMethod	%LockId
%Dictionary.CompiledForeignKey	%LockId

Class Name	Method Name(s)
%Dictionary.CompiledIndex	%LockId
%Dictionary.CompiledIndexMethod	%LockId
%Dictionary.CompiledIndexProperty	%LockId
%Dictionary.CompiledInstanceVar	%LockId
%Dictionary.CompiledMethod	%LockId
%Dictionary.CompiledParameter	%LockId
%Dictionary.CompiledProjection	%LockId
%Dictionary.CompiledProperty	%LockId
%Dictionary.CompiledPropertyMethod	%LockId
%Dictionary.CompiledPropertyUDLText	%LockId
%Dictionary.CompiledQuery	%LockId
%Dictionary.CompiledQueryMethod	%LockId
%Dictionary.CompiledStorage	%LockId
%Dictionary.CompiledStorageData	%LockId
%Dictionary.CompiledStorageDataValue	%LockId
%Dictionary.CompiledStorageIndex	%LockId
%Dictionary.CompiledStorageProperty	%LockId
%Dictionary.CompiledStorageSQLMap	%LockId
%Dictionary.CompiledStorageSQLMapData	%LockId
%Dictionary.CompiledStorageSQLMapRowIdSpec	%LockId
%Dictionary.CompiledStorageSQLMapSub	%LockId
%Dictionary.CompiledStorageSQLMapSubAccessvar	%LockId
%Dictionary.CompiledStorageSQLMapSubInvalidcondition	%LockId
%Dictionary.CompiledTrigger	%LockId
%Dictionary.CompiledUDLText	%LockId
%Dictionary.CompiledXData	%LockId
%Dictionary.ForeignKeyDefinition	%LockId
%Dictionary.IndexDefinition	%LockId
%Dictionary.MethodDefinition	%LockId
%Dictionary.ParameterDefinition	%LockId
%Dictionary.ProjectionDefinition	%LockId
%Dictionary.PropertyDefinition	%LockId
%Dictionary.PropertyUDLTextDefinition	%LockId
%Dictionary.QueryDefinition	%LockId

Class Name	Method Name(s)
%Dictionary.StorageDataDefinition	%LockId
%Dictionary.StorageDataValueDefinition	%LockId
%Dictionary.StorageDefinition	%LockId
%Dictionary.StorageIndexDefinition	%LockId
%Dictionary.StoragePropertyDefinition	%LockId
%Dictionary.StorageSQLMapDataDefinition	%LockId
%Dictionary.StorageSQLMapDefinition	%LockId
%Dictionary.StorageSQLMapRowIdSpecDefinition	%LockId
%Dictionary.StorageSQLMapSubAccessvarDefinition	%LockId
%Dictionary.StorageSQLMapSubDefinition	%LockId
%Dictionary.StorageSQLMapSubInvalidconditionDefinition	%LockId
%Dictionary.TriggerDefinition	%LockId
%Dictionary.UDLTextDefinition	%LockId
%Dictionary.XDataDefinition	%LockId
%Library.CacheSQLStorage	%LockId
%Library.EnsembleMgr	EnableNamespace, createMappings
%Library.FunctionalIndex	SegmentFinalize, SegmentInitialize, SegmentInsert
%Library.GlobalEdit	GetGlobalSizeBySubscript
%Library.GlobalStreamAdaptor	Read
%Library.ListOfDataTypes	DisplayToLogical, LogicalToDisplay
%Library.Persistent	%BuildIndices, %LockId
%Library.RoutineMgr	getPackageList
%Net.Remote.Proxy	%Constructor
%SOAP.WST.BinarySecret	Create
%SQL.CustomResultSet	%OnNew
%SYSTEM.Encryption	MD5HashStream, SHA1HashStream, SHAHashStream
%SYSTEM.SQL	SetSQLStats, SetSQLStatsJob
%SYSTEM.Util	GetSessionId
%Stream.TmpCharacter	Read
%Studio.General	Execute
%Studio.Project	InstallFromGbl
%Studio.SourceControl.Change	DisplayUncommitted
%UMLS.Utils	GetChildrenAsRS

Class Name	Method Name(s)
%ZEN.Auxiliary.jsonSQLProvider	%WriteJSONStreamFromSQL
%ZEN.Component.dateText	showDateSelector
%ZEN.Component.tablePane	executeQuery
%ZEN.Report.Display.call	%DrawToHTML
%ZEN.Report.Display.callsvg	%DrawToHTML
%ZEN.Report.Display.category	%DrawToHTML
%ZEN.Report.Display.class	%DrawToHTML
%ZEN.Report.Display.cssinclude	%DrawToHTML
%ZEN.Report.Display.fo	%DrawToHTML
%ZEN.Report.Display.group	%DrawToHTML
%ZEN.Report.Display.html	%DrawToHTML
%ZEN.Report.Display.img	%DrawToHTML
%ZEN.Report.Display.init	%DrawToHTML
%ZEN.Report.Display.inline	%DrawToHTML
%ZEN.Report.Display.line	%DrawToHTML
%ZEN.Report.Display.list	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.masterreference	%DrawToHTML
%ZEN.Report.Display.node	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.p	%DrawToHTML
%ZEN.Report.Display.pagebreak	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.pageendsidebar	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.pagefooter	%DrawToHTML
%ZEN.Report.Display.pageheader	%DrawToHTML
%ZEN.Report.Display.pagemaster	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.pagestartsidebar	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.report	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.section	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.table	%DisplayTableByRowsFO
%ZEN.Report.Display.write	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.xslinclude	%DrawToHTML
%ZEN.Report.Display.xslt	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.RenderServer	GetState
%ZEN.Report.SplitAndMerge	%DisplayPDF
%ZEN.SVGComponent.treeMapChart	plotItems

Class Name	Method Name(s)
%iFind.Find.Basic	ResolveCompCombArray, ResolveStemmedCombArray
%iKnow.Classification.Builder	%GenerateClassifier, %LoadFromDefinition, %OnGenerateClassifier, %PopulateTerms
%iKnow.Classification.IKnowBuilder	%TestClassifier
%iKnow.Filters.ContainsEntityFilter	%OnNew
%iKnow.Filters.ExternalIdFilter	%OnNew
%iKnow.Filters.GroupFilter	%OnNew
%iKnow.Filters.RandomFilter	%OnNew
%iKnow.Filters.SourceIdFilter	%OnNew
%iKnow.Matching.DictionaryAPI	CreateDictionaryItem, CreateDictionaryItemAndTerm
%iKnow.Matching.DictionaryQAPI	CreateDictionaryItem, CreateDictionaryItemAndTerm
%iKnow.Queries.SourceAPI	GetSummaryForText
%iKnow.Queries.SourceWSAPI	GetSummaryForText
%iKnow.UI.AbstractSourceViewer	ProcessInput
Ens.DataTransformDTL	Transform
Ens.Deployment.Utils	GetItemContentsByItemNumber
Ens.Director	IsProductionRunning

Behavior Change For ##class(%Library.%File).ParentDirectory()

This release modifies ##class(%Library.%File).ParentDirectory() so that, if you supply it a filename rather than a directory name, it will return "". Previously it returned the parent directory of the caller which was confusing.

%Net.HttpRequest Now Honors "charset" Values For All Types

The %Net.HttpRequest class will now honor the `CharSet` value in an HTTP response even if the Content-Type is not "text/". This allows the type "application/json" to be decoded correctly even if it specified the "utf-8" charset. Also according to RFC2616, there is no restriction to applying the charset only for text content.

Note: This is a slight change in behavior and is correct according to the standard. You can revert to old behavior by setting the `ReadRawMode` property to 1.

Allow Tasks To Continue After An Error

In this release, there is a new property, `SuspendOnError` in Tasks. Previously, all Tasks were suspended if they returned an error status. Now, by default, Tasks will be rescheduled and continue to run after an error. To revert to the old behavior, users can mark `SuspendOnError` as "Yes" in either the management portal or ^TASKMGR.

New Return Value From $SYSTEM.Lock.ReturnCode()

This change adds a new return code for the method, **$SYSTEM.Lock.ReturnCode()**. It can now return a value of 4 indicating that the lock has already been escalated. This new value indicates the lock node has been escalated in an earlier lock command, whereas a value of 2 indicates the lock escalation was triggered by the lock command.

6.2.2.4 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

Recursive Property Types Explicitly Illegal In Serial Classes

The type of a property in a serial class cannot be recursive, meaning the type class cannot be the container class or any primary super or sub class of the container class. For example, it is invalid to define a property in Sample.Address whose type is Sample.Address.

This change should not cause significant issues since this model would not previously have worked at runtime. Caché never supported SQL projection of recursive serial types. This change does report a new error at compile time while earlier versions allowed such classes to successfully compile.

Correct Handling Of Optimized Method Calls In DataType Classes

The release corrects an error in resolving a method call of the form "DO ..otherMethod()" when the target method is defined in both the local class and a declared datatype class.

Consider a data type class defined roughly as

```
Class User.DataType Extends %String
{
    Parameter PARAM = "HELLO";

    ClassMethod someMethod()
    {
        Write ..#PARAM,!
        Do ..otherMethod()
    }

    ClassMethod otherMethod()
    {
        Write ..#PARAM,!
    }
}
```

It has a parameter defined with a default value of "HELLO",and a classmethod (**someMethod**) that calls **otherMethod** which just writes out this parameter value.

Now consider a normal class the uses this datatype class:

```
Class User.User
Extends %RegisteredObject
{
    Property Prop As DataType(PARAM = "MYVAL");
}
```

In previous releases, the sequence of statements

```
SET A = ##class(User).%New()
DO A.PropsomeMethod()
```

would display

```
MYVAL
HELLO
```

This is wrong because it should write out MYVAL twice. However, the normalization of the "Do ..otherMethod()" was calling into the User.DataType class itself rather than calling the 'PropsomeMethod' member method declared locally.

In this release, the compilation of this call now correctly invokes the local member method so running the compiled code will now output MYVAL twice as it should. This also allows a classmethod member method to call a different private classmethod because it would be in the same class context, whereas today this generates an error.

While it is unlikely someone wants the existing behavior, there is a chance someone expects the method in the member class to be called rather than the local method. To accomplish this, replace the "Do ..otherMethod()" with "Do ##class(User.DataType).otherMethod()" and it will behave as before.

Simplify Work Queue Handling

In this release, the class compiler no longer signals its worker jobs using the **SignalAll** function because, in very busy systems, not all worker jobs may be involved in the current compilation. This release also removes the **StopWorkers** call for the same reason.

6.2.2.5 Language Binding Changes

Objective C Binding For Caché Uses New Runtime

A newer runtime for Objective-C necessitates the following changes in Caché –

ISCConnection

The new way to construct an ISCConnection is:

```
ISCConnection* conn = [ISCConnection connectionTo:@"127.0.0.1[1972]:SAMPLES" user:@"sys"
password:@"system"];
```

and for a secure connection:

```
ISCConnection* conn = [ISCConnection connectionTo:@"127.0.0.1[1972]:SAMPLES" principal:@"principal"
level:1];
```

ISCDatabase

The new way to construct an ISCDatabase is:

```
ISCDatabase* database = [ISCDatabase databaseWithConnection:conn];
```

where *conn* is a previously established connection.

ISCConnectionPool

The new method for setting up a connection pool is:

```
ISCConnectionPool* pool = [ISCConnectionPool pool];
```

CacheProvider Omits Code Generation For System Classes

In previous releases, the generated proxy classes for .NET contained many APIs that are not really meant to be used from the client (for example, ODBCToLogical and LogicalToODBC conversions). This change adds a way to remove these APIs from the generated code. The dotnet_generator now has an option `-skip-system-apis (true|false)`. If set to true (the default), only the user's APIs are generated which results in generated code that takes one–third the space as before.

Support For Typed Collections Of Object Properties

This release allows groups of Caché Objectscript objects to be projected as collections of types. For example, a class property defined as a %Library.ListOfObjects of type T is projected as CacheListOfObjects<T>. Similary, a %Library.ArrayOfObjects of T is projected as CacheArrayOfObjects<T>. This change removes the need for most casts related to projection of collections.

This change also deprecates the recursive version of ListOfObjects.OpenAllObjects() because this call may force the client to instantiate the generic (not template) versions of CacheListOfObects or CacheArrayOfObjects for server objects where the generated code now requires CacheListOfObjects<T> or CacheArrayOfObjects<T>.

Note: One limitation of this change is that if a collection is instantiated in a context where the type of its elements is unknown (for example, if the collection is returned from a method), the proxy class will be CacheListOfObects or CacheArrayOfObjects (without a template). If the same object is used later in a context where the element type is known and expected, it will cause an exception because by then the "wrong" proxy object is already in use and cannot be automatically replaced. Such cases should be very rare.

CacheProvider Adds Overloaded Methods For Open() And OpenId()

Code such as

```
Prop = obj.Property
...
Prop.Close()
```

which intended to unswizzle a property will now cause subsequent calls of obj.Property to return the null value even if the value on the server is not null. This is because obj owns the property and the property is automatically closed when obj is closed. A more correct way to accomplish this is to use the call to the new UnswizzleProperty(<PropertyName>) method.

Make CacheADOConnection Public For Direct ADO .NET Use

CacheConnection inherits from the CacheADOConnection class that provides all the ADO.Net functionality. CacheConnection allows an application to use both ADO and the DotNet Object binding via the same connection. While this is useful for many applications, it adds overhead in terms of extra messages to the server and the creation of internal objects inside the provider.

The XEP and Entity Framework connections each only relies on CacheADOConnection and the functionality it provides. DbConnection is the generic way used for DbFactory connections in ADO.Net, and is fully supported within CacheADOConnection. CacheConnection object adds no additional value when these generic ADO APIs are used, only overhead.

This change ensures that CacheConnection continues to function as it always has, but now ADO applications can just use CacheADOConnection directly if they do not use the object binding side of CacheProvider. In addition, CacheFactory will return CacheADOConnection objects with less overhead. ADO functionality will continue to work with either CacheConnection or CacheADOConnection, so users do not need to change their applications.

There are some public APIs of CacheConection that will not be present in CacheADOConnection. All of the required ADO.Net APIs will be contained within CacheADOConnection, but application writers may have taken to mixing APIs supplied by the object side of the CacheProvider. In that case, they may just continue using the CacheConnection class has they have been.

6.2.2.6 SQL Changes

Replace OIDs For $LISTs In Streams With $LIST Contents

Beginning with this version, it is now possible to store $LIST values in stream fields via SQL. This change also corrects a problem where the insertion of binary data into a binary stream field might fail if the binary data resembled a $LIST value (that is, $LISTVALID(value) was True).

In previous versions, if a $LIST value was inserted or updated as a stream value, Caché assumed the $LIST value was the OID of a another stream that would be copied into the target stream. This is because, by default, select queries that include stream fields in non-external tables return the OID of the stream object. An application that wanted to insert/update the value of a stream field with a value it selected from another stream field, would just open the given stream OID and copy the stream value.

Now, an SQL INSERT or UPDATE of a stream field with a value that is a $LIST value will no longer treat the $LIST value as the OID of a source stream for the value of the new stream. You must open the source stream and insert/update the field using the value of the OREF of the source stream. This does present a backwards incompatibility if applications are inserting or updating stream fields with the OID of a stream source. The compatibility issue is necessary in order to insert binary data that mimics a stream value or store $LIST data into a stream.

Require Parentheses Around Tables In FOR SOME Predicate

The syntax for the little used FOR SOME predicate has been changed to avoid ambiguity in the Caché SQL syntax. Previously, the syntax for the predicate was:

```
FOR SOME <tables> (<search condition> ...)
```

and now it will be necessary to express it as

```
FOR SOME (<tables>) (<search condition> ...)
```

Any application using the older FOR SOME <tables> syntax will need to add parentheses.

Note: This has no effect on the commonly used FOR SOME %ELEMENT syntax..

New SQL System Privileges

With this version, there are four new general SQL privileges that can be assigned to users or roles on a per namespace level:

- %NOCHECK — allows the user to run INSERT or UPDATE or DELETE commands with %NOCHECK.
- %NOTRIGGER — allows the user to run INSERT or UPDATE or DELETE commands with %NOTRIGGER.
- %NOINDEX — allows the user to run INSERT or UPDATE or DELETE commands with %NOINDEX.
- %NOLOCK — allows the user to run INSERT or UPDATE or DELETE commands with %NOLOCK.

These privileges only apply to the usage of the respective keywords in the restriction clause of the INSERT, UPDATE, or DELETE statements. It does not apply to the use of `%NOINDEX` as a preface to a predicate condition. Like all other SQL privileges, this is only enforced through ODBC, JDBC, %SQL.Statement, and %Library.ResultSet. You must have the appropriate privilege to use the restriction clause when preparing the statement.

Since TRUNCATE TABLE <tablename> performs a delete of the rows from the table with %NOTRIGGER behavior, TRUNCATE TABLE <tablename> now also requires the %NOTRIGGER privilege in order to run.

Important: This change may cause backward compatibility issues with current applications, but since these restriction clauses can lead to logically corrupt data, InterSystems has decided to move forward with this feature to better protect the integrity of customer data.

Correct Queries With Multiple JOINs Of <Column>=<Constant> Conditions

This release contains a correction for queries such as

```
SELECT t . test_code
FROM test . person p , test . test_result t
WHERE p . hospital_id = ?
AND    p . person_id = ?
AND    t . hospital_id = p . hospital_id
AND    t . person_id = p . person_id
```

where there is

- a join between two tables with 2 or more join conditions;
- and, for each such join, there is also <column>=<constant> condition on one of the columns of the join.

In previous releases, the SQL parser generated incorrect query plans for such requests. That error has ben corrected in this release.

Correction To Query Processing Using Timestamps

Prior to this release, an application that executed a query where:

- it included a filter in the WHERE clause like “TimeStampField > CURRENT_TIMESTAMP”

- TimeStampField was not a typical %Library.TimeStamp value
- the class instance of TimeStampField implemented **LogicalToTimeStamp** and **TimeStampToLogical** methods to convert from its logical format to the %TimeStamp logical format

may have received incorrect results from the query. This error has been corrected in this release.

Changes To Query Plans

In this release, the query processor will produce more accurate plans due to improved selectivity accounting in cases where some conditions imply other conditions. A typical instance of this is a query such as:

```
SELECT *
FROM T1, T2
WHERE T1.k = T2.k AND T1.k > 5
```

In this case, the optimizer derives the fact that T2.k > 5 is also true. It also notes that if conditions are tested in the order (T1.k > 5, T1.k = T2.k, T2.k >5), the last condition is redundant and does not add extra selectivity. Similarly, if the order of condition testing is: (T2.k > 5, T1.k = T2.k, T1.k > 5), a similar situation applies. However, if the order of condition testing is (T1.k > 5, T2.k > 5, T1.k=T2.k) then the last condition does need testing, but it adds less selectivity than it would in the other orders of testing. Prior to this release, there was no way for selectivity to be handled properly for this last case.

The result of this change is that plan changes will be somewhat common; most will be better, but some may be worse.

SQL Reserved Words Updated

The following words have been reserved in Caché SQL in prior releases, but not reported as such by **$SYSTEM.SQL.IsReservedWord()**:

```
%CLASSNAME, %ID, %KEY, %MINUS, %MVR, %ODBCIN, %PLUS,
%RUNTIMEIN, %RUNTIMEOUT, %SQLSTRING, %SQLUPPER,
%TABLENAME, %TRUNCATE, %VALUE, %VID
```

This release adds them to the list; and removes CATALOG as a reserved word.

DELETE %NOINDEX Changes

In prior releases, the SQL directive DELETE %NOINDEX has always been a no-op. Beginning with this release it will no longer delete the index entries defined for the deleted rows.

Changes To %SYS.PTools.UtilResults

Previously, the Index Analyzer page in the Management Portal would recalculate the different options every time you clicked on the radio button. Now the table can hold results for all four options and the data is only cleared when new SQL Statements are gathered. The first time you click on an option, the system generates the data, but thereafter just redisplays it.

If an application is querying the data, it can now limit the results returned by setting the value of the new property, `OptionName`. The four allowed values are:

- "IU" — Index usage
- "TS" — Table scans
- "TI" — Table indices
- "JI" — Join indices

Improved Checking When Connections Specified

When linked tables are defined as part of a class, the SQL processor now checks to make sure that all the values necessary to make the connection to the table are supplied as well. If this is not the case, then SQL error –162 ("SQL Connection is not defined") will be produced.

6.2.2.7 Changes To Locale And I/O Translation Tables

Change In Default Collation For Spanish Locales

`Spanish4` is now the default collation in the Spanish locales (`espw`, `esp8`, `esi8`, `esw8`) for local arrays and new databases. This new collation orders upper and lower-case letters separately; the characters follow this order:

a, á, à, b, c, ç, d, e, é, è, f, g, h, i, í, j, k, l

m, n, ñ, o, ó, ò, p, q, r, s, t, u, ú, ü, v, w, x, y, z

Note: The previous Spanish collation tables are still available, and the collation for previously existing databases remains unchanged.

New Maltese Collations Created

Maltese1

A Maltese Unicode locale (`mltw`) and collation (`Maltese1`) are now available in this release. The collation has the following characteristics:

- Cases are ordered separately, all uppercase precedes all lowercase
- The main ordering is (in lowercase, uppercase is correspndingly similar):

  ```
  a b &cdot; c d e f &gdot; g g&hstrok; h &hstrok; i ie j k l m n o p q r s t u v w x  z
  ```

- The letters with a dot above (c, g and z) are equivalent to the non-dotted versions but all else being the same they collate before the same letter without the dot
- “għ” collates as a single character between “g” and “h”
- “ie” collates as a single character between “i” and “j”
- “ħ” is equivalent to “h” but collates after

Although not present in the Maltese alphabet, latin letters “c” and “y” collate in their respective positions as in English; all other accented letters (such as “á”, “é”, “ã”, and so on) not in Maltese collate after the alphabet (same as in Caché standard).

Maltese2

A new collation, `Maltese2`, is also available in the Maltese Unicode locale. It has the same encoding as `Maltese1` but doesn't include the double-letter characters “g ” and “ie”.

New I/O Translation Tables For Unicode CP437, CP850, and CP852

This release contains new I/O translation tables between Unicode and locales `CP437`, `CP850` and `CP852`. These tables were added to the Unicode locales whose 8-bit versions are based on `Latin1` or `Latin2`. More specifically, the following Unicode locales have the new tables:

- csyw – Czech, Czech Republic
- danw – Danish, Denmark
- deuw – German, Germany
- engw – English, UK
- enuw – English, US
- espw: Spanish, Spain
- finw – Finnish, Finland
- fraw – French, France

- hunw – Hungarian, Hungary
- itaw – Italian, Italy
- nldw – Dutch, Netherlands
- plkw – Polish, Poland
- ptbw – Portuguese, Brazil

Upgraded Collations For Some Locales

The default collation in the following 8-bit locales has been upgraded to a new version that fixes minor problems in the older version (such as failing to account for some character sequences collate as a single character – "ll" in Spanish, and "ss" in German):

- csw8: Czech2/CP1250 Czech3
- huw8: Hungarian1/CP1250 Hungarian2
- esw8: Spanish2/CP1252 Spanish3
- esi8: Spanish2/Latin9 Spanish3

The German4 Locale Is Removed

The `German4` locale has been removed from the system in this release since it proved to be functionally equivalent to `German3`.

Magnetic Tape Translation Table Changed For Russian Locale

The default translation for the magnetic tape device has been changed from `RAW` to `UTF8` in the Russian Unicode locale (`rusw`). This is allows the labels of backups to tape to contain Cyrillic characters without further modifications.

Translation Table For Latin9 Unicode Updated

The tables that translate between `Latin9` and one of the Unicode encodings (`UnicodeBig`, `UnicodeLittle`, and `UTF8`) incorrectly mapped **$CHAR(128)** to U+20AC (the Euro sign). This mapping is used in many of the Windows CP tables, but not in Latin9, in which the Euro sign is **$CHAR(164)**. In these `Latin9` translation tables **$CHAR(128)** now passes unchanged.

New JSON Translation Tables Added

This release adds two new NLS translations: JSON and JSONML. They are similar to JS and JSML respectively, but have the following differences on the output side:

- Characters with codes 0 to 7, 11 and 14 to 31 are escaped as \uXXXX.
- Characters with codes 39 and 47 pass unchanged.

Update Handling Of Unicode LINE SEPARATOR And PARAGRAPH SEPARATOR

On Unicode systems, the characters LINE SEPARATOR ($CHAR(8232) = U+2028) and PARAGRAPH SEPARATOR ($CHAR(8233) = U+2029) are now escaped as "\u2028" and "\u2029", respectively, when passed through one of the JavaScript translations (`JS`, `JSON`, `JSML`, and `JSONML`) when used for example in **$ZCONVERT**.

6.2.2.8 iKnow Changes

Correct Handling Of Coded Dictionary Terms

If a Dictionary Term is composed of multiple elements, and these include both format and non-format elements (through use of the "coded" description of the format for adding composite terms), the dictionary term was not correctly saved in the internal data structures when the dictionary was optimized. This led to matching against the format and non-format parts separately, which could in turn lead to separate matches for the same term on the same CRC or path.

Customers who have used the advanced "coded" description feature to insert mixed terms (consisting of both format-based and non-format-based elements) must recreate these dictionary terms for this correction to take effect.

Changes To (Previously Experimental) PMML Modeling

Model-building capabilities were added as an experimental feature in 2014.1. Users who built applications to take advantage of this will find that some helper methods for generating PMML models have moved to the PMML model class from the Builder class. In addition,

- Generation of PMML "along the way" by different operators now happens where appropriate, resulting in a PMML file representing all operators executed
- A "Call" operator to execute arbitrary code as part of a modelling sequence is now present
- A "TypeColumn" property has been added in the Attributes configuration element to enable per-attribute specification of the datatype
- When instantiating a new row as part of a Sequence operation, a PMML model will no longer explicitly set all attributes to 0, but rely on InitialExpressions to populate initial values (0 for %Integer and %Double columns in the TableGenerator operation)
- A new "InsertOnly" property is present in the Sequence configuration element, allowing to append to a table already containing data. This means separate Sequence operations can be used to populate different columns of the same data set, as when part of the data comes from an iKnow domain and other parts can be retrieved straight from other sources (metadata)

6.2.2.9 Web Services And SOAP Changes

Explicitly Set Default Port Value For %Net.HttpRequest In %XML.Sax.Parser

In this release, %Net.HttpRequest explicitly sets default port value for %XML.SAX.Parser. Prior releases by default used the port for the last request using that %Net.HttpRequest object.

6.2.2.10 XML Changes

Set Character Stream Import LineTerminator Implicitly To $CHAR(10)

This release set the LineTerminator of character streams imported from XML document to $CHAR(10) implicitly, if XMLSTREAMMODE="block" (the default). This makes the import compatible with any traditional newline sequence (**$CHAR(10)**, **$CHAR(13)**, **$CHAR(13,10)**).

6.2.2.11 MultiValue Changes

ULTIMATE Emulation Changed Default Setting For STR.ONEISONE

When compiling a MultiValue BASIC program, the $OPTIONS STR.ONEISONE flag now defaults to OFF except for emulations of the legacy IN2 or ULTIMATE systems. Previously STR.ONEISONE only defaulted to ON for the legacy IN2 system. Only the emulation setting at compile-time affects this default; changing the CEMU setting at run time will not override the compile-time emulation setting.

The STR.ONEISONE flag is an alias for the $OPTIONS flags IN2.SUBSTR and T. These three $OPTIONS flags change the behavior of a substring reference of the form StringVar[N]. With any one of these flags, a reference to StringVar[N] is a reference to the single character at position N inside the string contained by variable StringVar. Without any of these flags, a reference to StringVar[N] is a reference to all the string characters starting at position N and extending to end of the contents of the string variable StringVar.

Any customer who uses the CEMU ULTIMATE compile-time setting (or $OPTIONS ULTIMATE in a MV BASIC source routine) and who also depends on the STR.ONEISONE flag being off must now explicitly include a

```
$OPTIONS -STR.ONEISONE
```

statement in their MV BASIC source programs which are affected by this change.

Remove Line Breaks From Base64 Encoded XML As The Default

Beginning with this release, the default is to omit line breaks from base64 encoded XML output in all properties of type %Binary or %xsd.base64Binary. To make line breaks the default,

- In XMLExport, append "base64linebreaks" to the format argument.
- In %XML.Writer, set the *BaseLineBreaks* property to 1 (the default is 0).
- For SOAP web service or web client binary output, set the *Base64LineBreaks* property to 1 or the *BASE64LINEBREAKS* parameter to 1. If both the parameters are set, the property overrides the parameter.

6.2.2.12 BASIC And MVBASIC Changes

Immediately Release MultiValue Process Locks

MultiValue Process Locks (created and released by the MVBASIC LOCK and UNLOCK commands) are supposed to be independent of transactions. However, in previous releases, an UNLOCK command during a transaction would not free the Process Lock until the end of the outermost transaction. Now, a MV BASIC UNLOCK command will release its corresponding Process Lock immediately, regardless of whether there is an active transaction or not.

Any application that depends on the fact that an UNLOCK command would not free the Process Lock until the end of the outermost active transaction will experience a change in behavior. The workaround to restore a semblance of the prior behavior is to replace the Process Lock with a File Lock taken out on a dummy file with a file name related to the name of the Process Lock. Process Locks are now released immediately, even when inside a transaction, but the FILEUNLOCK command inside a transaction will delay the release of the File Lock until the outermost transaction terminates.

WRITEV With Field Number 0 Appends To Beginning Of Record

In prior releases, the behavior of a WRITEV statement specifying an update to attribute number 0 of a record was undefined. Beginning in this release, The MVBasic run-time has been modified so that a WRITEV to attribute number 0 of a dynamic array record appends the new data as a new attribute at the beginning of the record. This new data becomes the leading attribute of the dynamic array and all previous attributes will be moved up one position. This is exactly analogous as assigning a string value to attribute 0 of a dynamic array variable.

6.2.2.13 xDBC Changes

ODBC 2.5 APIs Removed

The following APIs are no longer exported by the InterSystems ODBC 3.5 driver to address an error where both version 2.5 and version 3.5 APIs were being invoked by the Microsoft Driver Manager:

- SQLAllocConnect
- SQLAllocEnv
- SQLAllocStmt
- SQLErrorW
- SQLFreeConnect
- SQLFreeEnv
- SQLSetParam
- SQLTransact
- SQLGetConnectOptionW
- SQLGetStmtOption

- SQLSetStmtOption

Change Of Default C Type In Version 3.5 Driver For Date Time And Timestamp

The ODBC C type SQL_C_DEFAULT (99) can be used in calls to SQLBindColumn and the ODBC driver determines the proper C type based on the SQL_TYPE specified. The ODBC 3.5 standard changed the SQL_TYPES for Date, Time and Timestamp from 9, 10, and 11 to 91, 92, and 93, respectively. When SQL_C_DEFAULT(99) is specified as the C type, Caché internally calls SetDefaultType(sql_type) which returns the correct C type.

Previously, for the ODBC 3.5 version of the InterSystems driver, Caché did not detect the SQL Types for date,time and timestamp with values of 91, 92, and 93, and returned the C type as SQL_C_CHAR. This caused an error at runtime. Now SetDefaultType(sql_type) is changed to detect SQL_TYPES for Date, Time and Timestamp from 91, 92, and 93, respectively and to return the corresponding C types: SQL_C_TYPE_TIME, SQL_C_TYPE_DATE, SQL_C_TYPE_TIMESTAMP (91, 92, 93).

6.2.2.14 DeepSee Changes

Meaning Of NOW Relative To Time Offset Redefined

Time levels in DeepSee can be configured with a date offset using the "timeOffset" attribute. This feature is explained in the documentation. Prior to this fix, the NOW special member would behave strangely.

Consider the case where years are defined with an offset of “-6m”. This means that

- 2014 = July 2013 to June 2014
- 2015 = July 2014 to June 2015

Prior to this clarification, the meaning of "NOW" would change based on the offset. So for the same example, DeepSee would find out that NOW is July 2014, then apply the -6m offset to get January 2014, and then determine which year with a -6m offset January 2014 belongs to, which is 2014. To summarize, selecting NOW will now return the time period that the current date belongs to.

With this correction, if the current date is July 2014, the year level returns 2015 because “NOW” lies in the range of the year 2015 once a negative six month offset is applied to the start of 2015.

Enforced Distinction Between Calculated Members and Calculated Measures

MDX does not make a distinction between calculated measures and calculated members that are not measures. It is practical, however, to make this distinction. In this release, DeepSee distinguishes these items when they appear in crossjoins. Now, of the two sets used in a crossjoin, no more than one of them can contain measures (or items that behave like measures). One set (or both sets) can contain members (or items that behave like members). In this context:

- A calculated member is considered a measure if it uses AGGREGATE, SUM, MAX, MIN, AVG, or any other function that returns a number.
- A calculated member is considered a member if it uses %OR.

In previous releases, the following query was valid:

```
WITH  MEMBER [ColorD].[DEMO] AS
'AGGREGATE({[COLORD].[H1].[FAVORITE COLOR].&[Blue],
[COLORD].[H1].[FAVORITE COLOR].&[Red],
[COLORD].[H1].[FAVORITE COLOR].&[Yellow]})'
SELECT NONEMPTYCROSSJOIN([COLORD].[DEMO],[Measures].[Test Score]) ON 1 FROM [Patients]
```

In this release, the same query results in the error:

```
ERROR #5001: Two measures cannot be crossjoined
```

As a consequence of this change, you should review any calculated members and consider how they are intended to be used. If a calculated member is intended for use as a member, use %OR. If it is intended for use as a measure, use

AGGREGATE or another numeric function. Also, review any queries that use NONEMPTYCROSSJOIN or CROSSJOIN to ensure that you are not inadvertently crossjoining two measures.

Query Change When Excluding Multiple Members of List Level

In this release, DeepSee generates a different query than before, in the case when the user excludes multiple members of a list level. The new form of the query is more expected and is consistent with the query that DeepSee generates when a user excludes a single member of a list level. This change is not a change to the DeepSee engine, but solely a change to how DeepSee generates a query when a user interacts with a filter dropdown (in either the Analyzer or in a dashboard).

For example, consider the Diagnosis level in SAMPLES, which is a list level. In previous releases, if a user selected the `asthma` and `CHD` members of this level and clicked the **Exclude** option, in previous releases, DeepSee added the following filter to the query:

```
%FILTER
EXCEPT([DIAGD].[H1].[DIAGNOSES].Members,{[DIAGD].[H1].[DIAGNOSES].&[asthma],[DIAGD].[H1].[DIAGNOSES].&[CHD]})
```

This filter removes records for patients that have *only* the asthma diagnosis and patients that have *only* the CHD diagnosis. If the pivot table displays the Diagnosis level as rows, the result is as follows:

Diagnosis	Patient Count
None	831
asthma	7
CHD	4
diabetes	48
osteoporosis	24

The patients that have only asthma are not included anywhere in this pivot table, nor are the patients that have only CHD. In this pivot table, the `asthma` row represents the patients that have the asthma diagnoses *in addition to* some other diagnosis. Similarly, the `CHD` row represents the patients that have the CHD diagnoses *in addition to* some other diagnosis. This is a valid MDX query, but is not the query that the users expected to generate.

In the current release, when a user selects the `asthma` and `CHD` members of this level and clicks the **Exclude** option, DeepSee now adds the following filter to the query:

```
%FILTER ([DIAGD].[H1].[DIAGNOSES].&[asthma].%NOT,[DIAGD].[H1].[DIAGNOSES].&[CHD].%NOT)
```

This filter removes records for patients with the asthma diagnosis and patients with the CHD diagnosis. If the pivot table is currently displaying the Diagnosis level as rows, the result is as follows:

Diagnosis	Patient Count
None	831
diabetes	37
osteoporosis	18

In past releases, DeepSee used the %NOT function when a user excluded a *single* member of a list level, and DeepSee still does this. For example, when a user selects `asthma` and clicks the **Exclude** option, DeepSee adds the following filter to the query:

```
%FILTER [DIAGD].[H1].[DIAGNOSES].&[asthma].%NOT
```

This filter removes records for patients with the asthma diagnosis.

Export Of Zero-Padded Numbers Changed To Be Strings

Beginning in this release, DeepSee allows a format attribute of property in KPI to be defined as a "%string%" and the Excel export of such a property will be in accord with the format. That is, a property definition such as

```
<property name="EnfUnit" displayName="EnfUnit" columnNo="3" format="%string%"/>
```

will now export a number of the form 0008 to Excel as a string and not a number.

Cube Registry Now Requires A Name

Beginning with this version, there is now a dialog that opens on the first load of the Cube Registry. This will prompt the user to enter a class name that will be used for registry storage, along with options for the top-level, registry-wide settings. If a registry had been created in an earlier release, you can

- Reopen the one that existed by entering the default class name, DeepSee.CubeManager.CubeRegistryDefinition, as the new active registry class; or,
- Define a new registry with the name of your choice, and copy the contents of the Registry XData block of the previous DeepSee.CubeManager.CubeRegistryDefinition.

Separate Save And Task Scheduling Functions

This change removes the scheduling code from the call to **%DeepSee.CubeManager.Utils:WriteToRegistry()** and places it in **%DeepSee.CubeManager.Utils:ScheduleUpdaterTasks()**. The return of the latter will now allow for confirming success status of the task scheduling. With this channge, **WriteToRegistry()** is only responsible for validating and saving a cube registry while **ScheduleUpdaterTasks()** is responsible for calculating the scheduling requirements and communicating the resulting parameters to the relevant tasks.

6.2.2.15 CSP Changes

Improve The Parsing Of Individual Component Headers For Multi-Part Request Payloads

In previous versions, it is possible that the Caché streams holding individual components might have been named using the generic CSP Gateway-assigned name of "CONTENT" rather than the name assigned in the hosting form.

This release improves the parsing of individual component headers for multi-part request payloads. More data is now packed into the header lines, and further processing is done on the individual header lines so that the payload (and the associated type information) is accurately transmitted to Caché.

6.2.2.16 Zen Reports Changes

Table Width Default Changed

Prior to this release, if a table did not have a width specified, Zen Reports supplied a default value of "auto". Now it supplies a new default value, "100%". Those reports that relied in the old default must now set it explicitly.

Process Templates In Composite Items

Prior to this release, when templates were included in a composite they were ignored. Now they will be defined. This can lead to conflicts with templates defined in the report.

The work-around when there is a conflict is to remove the template from either the composite or report so it is only defined in one place. Since composites did not support templates before, the easiest fix when such conflicts arise on old report-composite combinations is to delete them from the composite.

The idea is that now XData in a custom class or composite can include XSLT template or variable definitions and these can be referenced in the report. Note that the generation point of the XData is not where the class or composite is included. This is because templates must be defined at the top level of XSLT source.

xyChart Does Not Include Point In Dataset If Y-Value Does Not Exist

In previous releases, if there was a missing y-Value Caché would assign a value of 0 to y, draw the marker point, and lines would go through the marker point if connecting lines were defined in the xyChart. Now, if the y-value is " " or not defined, the (x,y) pair in an xyChart is skipped; there will be no marking point.

Support Configurable Memory For All Batch/Command Files And Java Commands

Beginning with this release, new batch files for producing reports introduce a requirement that one accept the default maximum memory size (`-Xmx 512m`) or specify an environment variable. Before this, the server scripts contained such argument which means the memory allocated depended on the amount of physical memory on the machine. A better practice is to enforce the specification of a memory size.

In addition, the following environmental variables have been defined:

- EXCELMEMSIZE, EXCELSERVERMEMSIZE
- FOPMEMSIZE, RENDERSERVERMEMSIZE
- PRINTSERVERMEMSIZE
- SAXMEMSIZE
- PDFMERGEMEMSIZE

These default to 512m. For servers, the default used to be determined by the amount of physical memory on the machine; this was a bad practice and now servers default to 512m. To get larger memory size, use the environment variables.

Note: This change is not implemented on OpenVMS.

7

Caché 2014.1

This chapter provides the following information for Caché 2014.1:

7.1 New and Enhanced Features for Caché 2014.1

The following major, new features have been added to Caché for this release:

Furthermore, this version of Caché has been improved and enhanced in the following areas:

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

7.1.1 Major New Features

7.1.1.1 Support For REST

This version introduces support for handling REST requests, routing them to a given namespace, and processing them. This support is at the level of the CSP Gateway.

This feature is predicated on providing a Web application that routes all requests to a given handler class that, based on a URL mapping facility, then dispatches the calls to the correct class and method implementation. With this feature, Caché can now handle any REST interface; for data transport, Caché supports XML or JSON.

Note: This feature is not available on OpenVMS systems.

See Creating REST Services in Caché.

7.1.2 Rapid Application Development

7.1.2.1 64-bit Version of Activate

Caché applications on 64-bit systems can now take advantage of the new 64-bit version of Activate on all supported platforms and use 64-bit ActiveX objects.

Important: While the client technology that communicates over a protocol can be 32-bit or 64-bit, the server technology must consist entirely of only 32–bit or only 64-bit modules.

7.1.2.2 Support for Websocket Protocol RFC6455

In this release, InterSystems introduces support for WebSocket connections. Application developers may access to this technology via the class, %CSP.WebSocket which provides a protocol to build richer web applications, and a more lively user experience. WebSockets form a TCP connection between the server and the client over port number 80, and the protocol is supported by Google Chrome, Internet Explorer, Firefox, Safari and Opera.

7.1.2.3 Cube Manager for DeepSee

DeepSee previously included methods to build and synchronize cubes. With this release, DeepSee now includes the Cube Manager which provides a user interface for scheduling when to build and synchronize cubes. The Cube Manager is available from the DeepSee Admin menu, and allows you to define groups of cubes and the update frequency for these cubes.

The DeepSee Tools menu also includes the Model Browser which displays a diagram showing the relationships between cubes. This high-level diagram helps you understand how cubes are related.

7.1.2.4 Domain Definition Infrastructure

This new feature is of interest to all users that want to integrate the creation of iKnow Domains in a regular class-based implementation. It allows an application to specify an iKnow domain definition as part of a user class that inherits from %iKnow.DomainDefinition. When the class is compiled, the compiler will create an iKnow domain corresponding to the settings specified in the Domain XData XML. All "static" settings defined for the domain will be set at compile-time through calls to the underlying APIs.

The application can also define sources of text data to be loaded into the domain. This data becomes part of a generated **%Build()** method in a new class named [classname].Domain. When invoked, the method loads all data from the defined locations into the domain. Matching and metrics elements can be associated with the domain; these will be applied on all sources added to the domain.

7.1.2.5 Framework for Text Categorization

This new feature is a framework for categorizing text indexed by iKnow. Classification takes place in two different steps: building a classification model, and running the classification model against text to be categorized.

A classification model is defined through an XML structure (XData block) in a subclass of %iKnow.Classification.Classifier. A classification model can be run on random input text or on an existing iKnow source by using its **%CategorizeText()** or **%Categorize()** methods. This will return an ordered array with the categories associated with the text sorted according to their "score".

7.1.2.6 Globals C API

The C version of the Globals API provides an extremely fast, flexible persistence model that can be used to implement a wide variety of storage paradigms. It provides the same functionality as the Java, .NET, and Node.js Globals APIs, for developers working in C, C++, or any language that supports the C calling conventions. It can be used directly by application developers, and can also be used to implement versions of the Globals API in other languages such as Ruby.

7.1.2.7 UDP Socket Support

In this version, Caché introduces a new class, %NET.UDP which enables application developers to use UDP packets for communication. UDP is a connectionless way to communicate, and is often used for broadcasting or multicasting. For additional information and examples, please refer to the class documentation for %NET.UDP.

7.1.2.8 Unified Trigger Support for Objects and SQL

In this version, Caché adds support for triggers in Objects that equate exactly to SQL Triggers. In addition to the various *%OnXXXX* callback methods (which continue to be supported), you can now define a trigger that will fire for INSERT/UPDATE/DELETE events whether they are done via Object or SQL access. This change provides a consistent behavior regardless of whether data is manipulated as class instances or SQL tables. See the Caché Objectscript Reference for further details.

7.1.2.9 SQL INSERT Or UPDATE

In this release, the SQL parser now accepts the "Insert OR Update" syntax. The statement executes just like an INSERT; but when Unique constraints are encountered, it assumes you want to overwrite that record instead of inserting a new row. This syntax is very useful in the case where you want to see if a record exists in the database before deciding to do an insert or update. Now you can manage this in one execution.

Note: If SQL matches on a Unique check but the UPDATE would change the IDKEY value, the UPDATE will fail as this is an illegal operation.

7.1.2.10 64-bit Versions CacheActiveX and ConnectionGUI

Caché nows supports 64-bit versions of CacheActiveX and ConnectionGUI. Their connection info continues to be taken from the 32-bit registry.

7.1.2.11 ZEN Reports Generate HTML5 in HTML Mode

Beginning with this release, the HTML output of Zen Reports has changed. Caché will now have the HTML generator create a valid HTML DOCTYPE in the output, and the generated code generated will match the standard as well. This should result in optimal rendering for modern browsers that support the HTML5 standard.

7.1.3 Performance and Scalability

7.1.3.1 MDX Performance Diagnostic

The new utility class %DeepSee.Diagnostics.MDXUtils collects performance diagnostic information for MDX queries. The utility records, for a given MDX query, such information as cube statistics, query path, query statistics, and system resource usage. If an MDX query appears to have a performance problem, this utility provides information that can help diagnose the problem.

7.1.3.2 Reduce Memory Allocation in the iKnow Engine

This release optimizes the way in which dynamic memory is allocated and deallocated in the iKnow Engine. It improves the throughput speed of the iKnow Engine by 15% to 25%, depending on the platform. The change is active on all platforms except Solaris where the performance remains as it was in previous releases.

7.1.3.3 8-way Interleaved AES-NI CBC Decryption

Processor vendors continue to improve the hardware acceleration support for AES encryption. In this release, Caché now detects if the processor supports eight-way pipelines. By taking advantage of this enhancement, the performance of AES-NI further increases.

7.1.3.4 Push Subquery Conditions into UNION Legs

The Caché release includes an optimization for SQL performance involving sub queries. In prior versions, when a UNION query included a conditional expression on a sub query, the entire result set would be executed and then reprocessed to

apply the conditional check. With this optimization, the conditional check is applied on the first pass through the data greatly enhancing the performance of the operation.

No code change is required other than purging cached queries to take advantage of this feature.

7.1.3.5 Support Separate Selectivity for Frequent Outlier Values

In this version, TuneTable includes an additional feature. Certain outlier values, where conditions on these values sometimes result in decreased performance, have now been coded to produce inflated selectivity values. The intent of the change is that these values will result in queries that avoid using indices containing these outliers. This change should result in greater likelihood of queries using the best possible index.

For an application to benefit from this change, TuneTable must be used, and any existing cached queries purged, so the optimizer will take advantage of the new selectivity values.

7.1.3.6 ROLLBACK Performance

ROLLBACK performance has been improved in this release by roughly 20%-40% over previous versions. This improvement was measured while rolling back single (large) transactions, as well as bulk rollback (as would occur during startup recovery).

7.1.3.7 Database Defragmentation And Compaction

This release reintroduces two database management capabilities:

1. Compact a database.

   This feature moves free space distributed throughout the database to its end. You can then return the free space to the underlying file system by truncating the database (the truncation feature was already available in previous versions)

2. Defragment globals in a database.

   This feature rearranges global blocks within the database so that all of the blocks containing data for a given global are in consecutive sequence.

While these capabilities are not needed for day-to-day operation, and most systems may never need them, they can make certain special operations easier. In prior versions, when the type of database reorganization described above was needed, it required downtime while using the ^GBLOCKCOPY utility to migrate the data to a new database.

Details on these and related capabilities are described in the System Administrator documentation.

Note: Both of these capabilities had appeared in certain previous releases and both had been removed in the more recent releases. These features should not be used in any version prior to 2014.1. These capabilities have been revised and are available for use again now in 2014.1.

7.1.4 Reliability, Availability, Maintainability, Monitoring

7.1.4.1 License Registration in Cloud Environment

When an InterSystems license key is activated on a Caché, Ensemble, or HealthShare instance in the Public Cloud (for example, Amazon EC2), InterSystems may collect system information (# of processors, OS type, license key ID, etc.), and send this information to a secure server in the InterSystems network. The scope of the information collected is limited to that needed for an audit of InterSystems license usage in the Public Cloud.

7.1.5 Security

7.1.5.1 Implement a Secure Shell for Debugging

Administrators may now provide users terminal access with a secure debug shell to better control access to sensitive data. This new shell protects against malicious role escalation and the injection of code to run with higher privileges.

7.1.5.2 Delegated Authentication Now Supports Two-Factor Authentication

With the release, application developers can now use the built-in mechanism for two-factor authentication from their custom authentication mechanism(s).

7.1.6 Technology Preview

This category is new with this release. It is intended as a way of introducing and providing access to new software capabilities that InterSystems believes will be useful in enhancing the effectiveness of existing and future applications.

The capabilities listed here are ready for use by customers, but they are not yet complete in functionality and design. Customers who take advantage of these capabilities must understand:

- InterSystems makes no backward compatibility guarantee with future updates;
- Customers may incorporate these capabilities in deployed applications, but must first check with InterSystems to determine best course of action;
- Customers who deploy these capabilities in their applications must commit to upgrading to the final released version.

InterSystems strongly encourages those who incorporate these items in their software to provide feedback on their experiences.

7.1.6.1 iKnow Negation

InterSystems introduced this feature as experimental in Caché 2013.1 with focus on the English language. As of this release, negation is no longer experimental and iKnow Negation functionality is available for all iKnow supported languages (Dutch, English, French, German, Spanish, Portuguese).

The uniqueness of iKnow negation detection rests on the fact that not only linguistic markers of negation in a sentence (such as “not”, “no” and “didn't”) are detected, but that iKnow also marks their semantic scope and span. In the sentence, “John has no signs of headache but has a fever.”, the linguistic negation marker is “no”. The iKnow negation detection will not only detect the marker but also its extent. iKnow will establish that “John has no signs of headache” is the negative part of the sentence. An important aspect of this is that the span of a negation is limited to the length of the semantic pathway of which the negation is a part.

7.1.6.2 Studio Refactoring

This release InterSystems includes support for the refactoring of code in the Studio. Initially, the available options will be:

- Change Class Name
- Change Property Name
- Change Method Name

Additional options will be incorporated in future releases.

In this release, refactoring is only available when Studio is connected to a Windows server.

7.1.6.3 iKnow Text Indexing and Searching

A new, experimental feature in this release is the iFind index. iFind is a feature that supports full text searching with all the functionality expected by the advanced user coupled with the benefits from the iKnow semantic analysis.

iFind is a specific index that can be defined on all %String and %CharacterStream properties of a class. If the class already contains data, the index can be built by running the **%BuildIndices()** method of that class. The iFind index is automatically maintained every time the **%Save()** method of the class is invoked, just as for other indices. Once the iFind index is correctly defined and built, it can be used in SQL queries by invoking the Search_Index() function. The name of the iFind index to use and the search string are the two main arguments to specify for Search_Index.

Currently, there are two restrictions on iFind indices: A) they can only be defined on bitmap-friendly classes (negative ID value are disallowed), and B) out-of-the-box deployment is only possible for classes using Default storage.

The iFind index has 3 different flavors:

1. Basic — The basic index is intended for cases where the %String or %CharacterStream data only have to be searched without any need for semantic understanding or analysis of the data. This kind of iFind index supports the following pure search operations: simple string; wild-card string; word position; regular expression; stemmed word; and de-compounded word.
2. Semantic — This index is intended to be used in the cases where a user wants to take advantage of the concept and relation detection capabilities of iKnow in %String or %CharacterStream data. In addition to the operations supported by iFind.Basic index, an iFind.Semantic index allows some more advanced "semantic" search operations such as entity-based search or entity-expansion search.
3. Analytic — An analytic index is intended for cases where the user wants to take advantage of the full analytical capabilities of iKnow. In addition to the operations supported by the basic and semantic indexes, an iFind.Analytic index allows some more advanced "analytical" search operations extending to the full range of elements generated by iKnow on top of the concept and relation detection.

   An iFind.Analytic index allows you to select records or record elements (such as concepts, relations, pathways) using semantic dominance, semantic proximity, pathway membership, and combinations of these. In addition, analytic index preparation automatically generates a series of class projections allowing direct access via a simple SQL query to the dominance, proximity, pathway, entity information per record generated by iKnow. A typical query supported by iFind.Analytic selects the top 100 dominant iKnow concepts in all sources containing the word "InterSystems".

7.1.6.4 iKnow Customized Summaries

This feature was introduced in experimental form in Caché 2013.1 for those who desire to tune the content of iKnow generated summaries to their specific needs. This feature is now stabilized, it can now be used for all languages supported by iKnow (Dutch, English, French, German, Spanish, Portuguese) and the feature is ready for generalized usage.

Main parameters for tuning the summaries remain as they were:

- force sentences to be included in the custom summary, e. g. when summarizing a newspaper article it can be beneficial to always include the first two sentences.
- force inclusion or exclusions of sentences by listing words and/or word groups. If a sentence contains one of the elements in the list it will be included in the summary (or excluded from the summary).
- overweighting or underweighting certain words and/or sentence parts. In this case the weight of sentences containing one of the elements in the list will be adapted in accordance with the weight instructions specified by the user. These adaptations will influence the sentence rankings used in the summarization and include or exclude the elements specified by the user if the context of the text allows to do so.

The listed options can be set by means of an extra parameter in the SourceAPI.GetSummary method.

7.1.6.5 New Zen Support For Mobile Development

The version of Caché introduces a new helper page to aid mobile support in existing Zen applications. This new base page provides support for mobile specific notions such as multi touch events, geometry changes, rotation, and so on with appropriate callback support to enable users to handle these events programmatically. Some new widgets designed specifically for mobile devices like scribbleArea have been added to the Zen Component Library as well with matching samples.

7.1.6.6 Support For XSLT2

This version of Caché now provides support for XSLT Version 2, a significant advance in capability over XSLT Version 1. Among the new features Version 2 provides are:

- A more convenient and expressive transformation language plus support for XPath 2.0 and the new XDM (XPath Data Model).
- Strong typing, support for XSL schema types, and the ability to define your own (schema) types. XPath 2.0 also includes a new sequence type not present in Version 1.
- A much more powerful functional language with improved string processing, date and time handling, node-set manipulation, and boolean operators.
- The ability to define and write functions in pure XSLT via the xsl:function instruction.

These, and many other improvements/new features significantly increase the productivity of any XSLT programmer. Strong typing allows many errors to be caught at compile time and to be corrected immediately.

XSLT Version 2 functionality is available via the classes in the %XML.XSLT2 package.

Note: XSLT2 functionality is not available on the OpenVMS platform.

7.2 Caché 2014.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2014.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2013.1 and 2014.1.

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

7.2.1 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 2014.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

7.2.1.1 Version Interoperability

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

7.2.1.2 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 addition of pages to assist with database mirroring and the separation of roles.

7.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

Extent Support Removed

In version 2010.1, InterSystems announced that support for database extents was deprecated. In this version, support for extents has been removed altogether.

Those customers who still have systems employing database extents and who wish to upgrade to this release, must consolidate their existing databases down to a single cache.dat file. There are two options for this:

1. Before upgrading to this release, run the GBLOCKCOPY routine to copy the data from the database and its extents into a single new database.

   In this case, the database can be converted and used to replace the existing database. It can be configured and tested prior to the upgrade. This is especially helpful if the database is to be mounted at system startup.

2. After upgrade, manually consolidate the database using the cdbmerge utility program.

   In this instance, Caché will refuse to mount the database, even if it configured for mount-at-startup. Only after conversion will it be in a format suitable for use.

Option 1 allows for advance preparation and testing prior to the upgrading to this version. Option 2 is provided in case older databases need to be accessed after the system is upgraded.

Support For 2KB Databases Removed

Over the last several releases, support for 2KB databases has been incrementally restricted. InterSystems urged users with 2KB database to convert them to 8KB databases.

Beginning with this release, 2KB databases can no longer be mounted at all, and are therefore no longer supported. If you have 2KB databases that need to be converted, you may do so using Caché version 2011.1 or earlier; the conversion can be performed prior to upgrading from these versions, or using a separately installed Caché instance.

Cannot Move Security Exports To Earlier Releases

User security exports from 2014.1 are incompatible with 2013.1 due to the introduction of the "AccountNeverExpires" and "PasswordNeverExpires" fields. If an import to 2013.1 is attempted, the user records are skipped. Deleting these fields in the security export will allow the import of the XML into 2013. The user data will have to be manually transferred.

Change In Memory Usage For XML Processing

In this release, the XML reader now uses local variables for storing its information; previously, it used process-private globals. This results in both a performance improvement and an additional demand on memory. Depending on the sizes of XML documents read, the maximum process private memory may need to be increased to avoid fatal errors.

The current suggested per-process memory is 256MB (it may be set to less than this is upgrading from much earlier releases). See the information accompanying the configuration file parameter, bbsiz, the $ZSTORAGE special variable, and the technical article on Caché Process Virtual Memory for further information.

%SYS.System.WriteToConsoleLog() Defaults To UTF-8

String messages sent to cconsole.log via **%SYS.System.WriteToConsoleLog()** are now encoded in UTF-8. This applies to all systems, 8-bit and Unicode. Customers who have applications that parse cconsole.log need to modify those application to account for the multi-byte possibilities.

Messages containing only ASCII characters ($CODE < 128) are not affected. Characters with $CODE >= 128 (even those from Latin1, such as "é") are translated to a sequence of 2 or more bytes. The file, cconsole.log, is still byte-oriented, but in order to correctly display UTF-8 encoded characters, a suitable text editor or terminal emulator should be used.

%SYS.System.WriteToConsoleLog() is the method replacement for `$ZU(9, "", message)`.

Error In Saving SQL Persistent Data Containing Collections Of Objects

Beginning with version 2011.1, if an SQL-enabled class had a property which was a collection (list or array) of objects, the collection was saved in an incorrect format. It must be corrected before the object can be opened successfully.

To permit access the collection in this version of Caché,

- Access the stored data directly (for example, via a global reference), and remove the outermost $LISTBUILD(...) from the data representation, or
- Edit the class definition to modify the collection property and specify "CLASSNAME=1" as part of its definition. This tells Caché to use the oid format as was done prior to version 2011.1.

Added Scripting Restrictions

This version of Caché imposes additional requirements on user scripts. First, you can no longer specify an implied namespace on the command line when running a routine in application mode using the -U switch. If you wish to specify a namespace, it must be defined in the CPF file.

Second, Caché now checks the startup code so that, if the user is trying to execute a routine in application mode, the user must have USE access to the service they are trying to run under. The service is one of: %Service_Console, %Service_Terminal, %Service_Telnet, or %Service_ComPort.

Note: The roles %Developer and %Manager have all these resources defined as part of the role. In a locked down installation, the %Operator role does not have these services defined.

If the user has created custom roles which do not include these services, and the services are not public, the user must make these services Public, or add them to the individual roles.

Debugging Requires Write Access

A user who wishes to debug routines or methods originating in the CACHESYS database must have write access to the database. Otherwise, stepping, breakpoints, and watchpoints will be disabled while execution is in the routine.

Behavioral Changes For ccontrol stop

Beginning in this release, a command of "ccontrol stop –nofailover" issued to a non-primary mirror member will be ignored. In prior releases, it would cancel any pending shutdown of that member.

Terminal IO Escape Processing Change

The low-level processing for terminal input examines one character at a time. This processing has been changed so that it will no longer terminate input if a character designated as an input-terminator characters occurs as part of a terminal escape sequence.

Disallow //page.csp As Valid Application

In the CSP application matching code the URL format:

```
//servername/subdirectory/page.csp
```

allows one to specify a virtual server using the "//servername" format. In pror releases, this also matched a URL such as "//page.csp". This is incorrect and has been fixed in this version. The "/page.csp" is valid and can match the "/" application (if one is defined) but "//page.csp" will no longer produce a valid match.

Correct Error In Global Mapping

A re-work of the global mapping routines in 2011.1 introduced an error which allowed a user to enter global mapping ranges in the management portal that overlapped. For example, the following mappings try to map part of the ^X global out of the USER namespace into the SAMPLES database; but they are illegal because they overlap

```
^X(1):(10) --->SAMPLES
^X(5):(20) --->SAMPLES
```

When the illegal mapping was activated, the first mapping was ignored, and the second mapping used. In the example, only ^X(5):(20) would be mapped, and ^X(1):(5) would remain in the default database for the namespace, USER. In this case, Caché would write an error message to the cconsole.log file:

```
09/16/13-10:23:22:195 (9664) 2 Discarding subscript mapping (1):(10) mapping overlaps
                              with a prior mapping global ^X namespace USER
```

Those customers upgrading from a version after 2011.1 and who map globals or routines should check the cconsole.log for the given message. You can also verify that there are no global mapping overlaps by executing:

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

on the CPF file of the system they are going to upgrade. They should check using this version of the system when they perform the validation. Alternatively, you can use the CPF validator in the WRC. If there are overlapping mappings in the CPF file, they will be displayed.

The actual mapping being activated and used by the system can be show by using this routine entry point in the %SYS namespace on your existing system:

```
Do SHOW^%NSP("USER")
```

Important: If you have detected that you have one or more overlapping mappings, you must remove the mappings which overlap (and are discarded) before you upgrade. If you do not do this, the upgrade will fail, and you will see the overlap messages in the cconsole.log file.

If this happens, you can simply edit the cache.cpf file, remove the duplicate mappings, and then re-install. After the installation finishes, you can adjust any data locations or mappings desired.

Added Prompts In ^SECURITY

When using ^SECURITY to create a new application in the security tables, two new prompts have been added:

- CSP/ZEN Enabled? No =>
- Inbound Web Services Enabled? No =>

These prompts already existed in the management portal, but were missing from the routine. If there are any customer scripts which call ^SECURITY and feed it input, the customer will have to update the script.

Change Default Character Set For 8–Bit Installations

Unicode installations of Caché use UTF-8 encoding for their default page responses while 8–bit Caché installations previously defaulted to using the current locale encoding for CSP pages, e.g. iso-8859-1. As of this release, the default for 8–bit Caché installations will also be UTF-8.

The reason for changing this is due to how CSP form POST and URL encoding works in browsers: all URLs are expected to be UTF-8 encoded and then URL encoded. The server cannot determine if the data comes from a CSP form POST or from a URL link so it cannot correctly convert the data back into characters.

Change Default for CSP Parameter: GZIP Compression

In this version, the default setting for the “GZIP Compression” parameter is now set to “Enabled” for all Application Paths.

Enforce Naming Of All File Types Or Paths Served By CSP

By default, the current CSP Gateway, in the absence of any other configuration directives, will automatically recognize and process files of the following types: “csp”, “cls”, “zen”, “cxw”.

This change enforces a configuration policy through which all files processed by CSP must either be listed in a “CSPFile-Types” directive, or the hosting path enabled using a “CSP On” statement. In previous versions, the above file types were recognized for all web server paths and, as a result, could result in incorrect requests being routed to the CSP engine.

Note: This applies to Apache v2.0 and later.

Add Specific Permission Requirements For CSP Pages

Using the newly added security parameter, SECURITYRESOURCE, you will need to hold specific permissions to view or execute some system classes, namely:

Class Name	Permissions
%Studio.SourceControl.UI	**%Development:USE**
%CSP.StudioTemplateInsert	**%Development:USE**
%CSP.StudioTemplateSuper	**%Development:USE**
%Monitor.Adaptor	**<empty>; supplied in subclasses**
%Monitor.System	**%Admin_Manage:USE, %Admin_Operate:USE, %Developer:USE**
%BI_AnalyzerGrid	**%BI_LegacyAccess:USE**
%BI_CSPsuper	**%BI_LegacyAccess:USE**
%BI_Chart	**%BI_LegacyAccess:USE**
%BI_ChartCus	**%BI_LegacyAccess:USE**
%BI_ChartLgnd	**%BI_LegacyAccess:USE**
%BI_DashboardGrid	**%BI_LegacyAccess:USE**
%BI_EchoClass	**%BI_LegacyAccess:USE**
%BI_ExcelExport	**%BI_LegacyAccess:USE**
%BI_SMSmsg	**%BI_LegacyAccess:USE**
%BI_SVGCtxtMenu	**%BI_LegacyAccess:USE**
%BI_TestSVG	**%BI_LegacyAccess:USE**
%BI_VennSVG	**%BI_LegacyAccess:USE**
%BI_WebChartList	**%BI_LegacyAccess:USE**
%BI_WebCtxtMenu	**%BI_LegacyAccess:USE**
%BI_WebDashboardE	**%BI_LegacyAccess:USE**
%BI_WebEditQueryCom	**%BI_LegacyAccess:USE**
%BI_WebList	**%BI_LegacyAccess:USE**
%BI_WebReporter2	**%BI_LegacyAccess:USE**
%BI_WebRoleMtns	**%BI_LegacyAccess:USE**

Class Name	Permissions
%BI_WebSCMod	**%BI_LegacyAccess:USE**
%BI_WebTranslation	**%BI_LegacyAccess:USE**
%BI_WordExport	**%BI_LegacyAccess:USE**
%BI_webPreference	**%BI_LegacyAccess:USE**

7.2.1.4 Platform-specific Items

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

8–Bit Caché Systems Default To UTF8

Previously, all Unicode Caché installations had the default CSP page use UTF-8 encoding, but 8-bit Caché installations used a default of the current locale encoding, for example, ISO-8859-1. In this release, 8-bit Caché installs also use UTF-8.

The reason for changing this is how form POST and URLl encoding works in browsers; all URLs are expected to be UTF-8 encoded and then URLurl encoded. But form POST will encode using the charset of the page before URL encoding the data.

This means the server cannot determine if the data comes from a form POST or from a URL link (at least not consistently due to things like form POST that uses method="get"), so it cannot correctly convert the data back into characters.

Important: Sites where this is not an issue can restore the previous behavior by issuing the command

```
Set ^%SYS("CSP", "8BitLocaleCharset") = 1
```

Windows

- Changes To Callin Kernel

  If you are upgrading to this version, and have an application that depends on your own linking of the callin kernel, you must change the file, shdir.c, recompile it, and then relink callin in order for that application to execute. Specifically, you must add the statement

  ```
  const char exeinfo[]="User";
  ```

  before recompiling. Failure to do so will result in an unresolved symbol, `_exeinfo`, at link time.

- Launcher Now Automatically Starts At End Of Unattended Install

  In this release, the “launcher”, csystray.exe (also known as the “cube”) will be started automatically at the end of an unattended installation process. In preceding releases, it had to be started manually.

  The old behavior can be restored by setting the newly-added property, `ISCSTARTLAUNCHER`, to "0" in the unattended install.

UNIX®

- CSP Gateway Change

  The CSP Gateway installation for Apache on UNIX® now uses the “CSPFileTypes” directive instead of “AddHandler csp-handler-sa” in all places AddHandler was used before. This affects regular UNIX®, CSP Gateway standalone, RPM and DMG installs.

- cstat Options Change

  Beginning with this release, only the owner of the instance may run cstat specifying -u[1, 2, 4, 8].

SUSE Linux 12 Linker Change Affects Light C++ Binding

A change in the linker on SUSE 12 systems prevents Light C++ binding applications from building. The SUSE 12 linker now requires application makefiles to explicitly specify dependent libraries of other libraries with which they link, even if those other libraries specified the dependent libraries when they were linked. In this case, libcachet.so depends on libpthread.so, so starting with SUSE 12 Linux, both libraries must be explicitly specified. Therefore, one of the following changes is needed to successfully build the application:

1. Add `–lpthread` to the library specifications in the ld or g++ command line in the makefile used to build the application. For example, if using a makefile based on one of the Master.mak files installed with Light C++ Binding sample applications, change the line:

   ```
   CACHETLIB = -L$(CACHETPATH) -lcachet
   ```

   to

   ```
   CACHETLIB = -L$(CACHETPATH) -lcachet –lpthread
   ```

2. Set the environment variable *MULTITHREADED* to 1 in the environment in which make is invoked. In the Bourne shell, the syntax is:

   ```
   export MULTITHREADED=1
   ```

Either option resolves the issue; they do not conflict with each other, so both options can be used together.

OpenVMS

In this release, the support for REST Services announced in the Release Notes is unavailable on OpenVMS. Attempts to use it will result in <UNIMPLEMENTED> errors.

7.2.2 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.

7.2.2.1 Routine Changes

^MIRROR Interface And Behavioral Changes

Option 2 (“Mirror Management”) of the top-level selections has been enhanced. The new menu choices are:

1. Add mirrored database
2. Remove mirrored database
3. Activate or Catchup mirrored database
4. Change No Failover State
5. Try to make this the primary
6. Connect to Mirror
7. Disconnect from Mirror
8. Modify Database Size Field(s)
9. Force this node to become the primary
10. Promote Async DR member to Failover member

11. Demote Backup member to Async DR member
12. Mark an inactive database as caught up
13. n/a
14. Pause dejournaling for a database

Items #1, #2, #3, and #14 now include a multi-selection feature and filtered displays. Items #8 and #12 are enhanced with filtered display only.

The multi-selection feature allows you to select multiple databases from the given list. With a filtered display, only databases that are appropriate for the specified task are displayed, when the user presses "?". For example, when the user chooses to "add mirrored database", only journaled and non-mirrored databases are displayed.

^PROFILE Permission Requirements Added

Beginning with this release, any user who desires to run ^PROFILE must be a member of the **%All** role.

7.2.2.2 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release

- %CSP — WebLink
- %CSP.UI — DocLocalize
- %CSP.UI.Portal — AdvancedSettingsTemplate, Applications, ApplicationsClient, ApplicationsPrivRoutine, ColumnPriv, ISQL, ManageDBActions, RemoteDatabase, Service, ShadowPopup, ZenReportServer, ZenReportServers
- %CSP.UI.Portal.Config — Network, ZenReport
- %CSP.UI.Portal.Dialog — MgrDBActions, ZenReportServerAction
- %CSP.UI.SQL — FormPane, USER, UserPrivPane
- %CSP.UI.System — ApplicationPane, ApplicationTablePane, BroadcastPane, DatabasePane, DbRoutinesPane, JobInfoForm, LicenseKeyForm, LicensePane, LockForm, NamespaceForm, RemoteDatabasePane, SchemaListPane, SourceControlPane, SystemPane, TaskEditForm, TaskPane
- %DeepSee.ComputedDimension — iKnow
- %DeepSee.UI.Dialog — DashboardEditControl
- %Identity — API, Exception, Link1ResultSet
- %Identity.Data — Audithist, Classified, Comment, JoinIndex, LinkDefaults, LinkParameter, Linkage, MPI, SpecialAddr, Unclassified
- %Identity.EntityTypes — CategoricalName, CompoundName, CompoundNameAustralia, CompoundNameFrance, CompoundNameGermany, CompoundNameSpain, EntityAttribute, EntityAttributeFreq, Femalename, FemalenameAustralia, FemalenameFrance, FemalenameGermany, FemalenameSpain, Gender, GenderAustralia, GenderFrance, GenderGermany, GenderSpain, Givenname, GivennameAustralia, GivennameFrance, GivennameGermany, GivennameSpain, Malename, MalenameAustralia, MalenameFrance, MalenameGermany, MalenameSpain, MedicalTerm, Nickname, NicknameAustralia, NicknameFrance, NicknameGermany, NicknameSpain, StatesGermany, StatesUSA, StreetAddress, StreetAddressAustralia, StreetAddressFrance, StreetAddressGermany, StreetAddressSpain, Surname, SurnameAustralia, SurnameFrance, SurnameGermany, SurnameSpain, Text, TitlesUSA
- %Identity.Util — CopyFrom
- %SQL — CustomQuery
- %SQL.DICT — QueryTypeCustom

- %SYS.EMS — Install, InstallInfo, Utils
- %iFind — Index
- %iFind.Index — Find
- %iKnow.Objects.dd — IndexBuilder, PTask
- %iKnow.Queries.SentenceWSAPI — GetHighlighted
- Config — ExtendedHost
- EMS — Users
- Ens.Util — IO, PortalModelBase

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CSP.UI.Portal.Config.Advanced	method	DrawAfterCreatePage, DrawLocatorExtra, doBrowse, doRemoteBrowse, onPopupAction
—	parameter	PAGENAME
—	property	LocatorParent
%CSP.UI.Portal.Config.AdvancedTable	method	DrawAfterCreatePage, UpdateDetails, burstUpdateHandler, clearMessage, updateMsg
—	property	ConfigTypes, ConfigValues, SelectedType, SingleSubject
%CSP.UI.Portal.Config.Cluster	property	Category
%CSP.UI.Portal.Config.Compatibility	property	Category
%CSP.UI.Portal.Config.Device	method	DrawAfterCreatePage, GetPropertyList, cancelItem, saveItem
%CSP.UI.Portal.Config.Devices	method	DrawAfterCreatePage, GetPropertyList, SaveData, UpdateDetails
—	property	SingleSubject
%CSP.UI.Portal.Config.Memory	property	Category
%CSP.UI.Portal.Config.SQLDataType	method	UpdateDetails, saveItem
%CSP.UI.Portal.Config.SQLDataTypes	method	DrawAfterCreatePage
%CSP.UI.Portal.Config.Startup	property	Category
%CSP.UI.Portal.Config.ValueEditor	method	saveItem
—	property	pValue
%CSP.UI.Portal.DatabaseTemplate	method	DisplayError, doClose
%CSP.UI.Portal.Databases	method	Dismount

Class	Type	Name(s)
%CSP.UI.Portal.Dialog.Compile	method	updateoptFlags
%CSP.UI.Portal.Dialog.EncAddAdmin	method	isValid
%CSP.UI.Portal.Dialog.Resource	property	HasSecurity
%CSP.UI.Portal.Dialog.Service	property	CSPLink, SERVICENAME, ServicesLink
%CSP.UI.Portal.Dialog.ServiceConnections	property	SERVICENAME
%CSP.UI.Portal.EncryptionCreate	method	isValid
%CSP.UI.Portal.EncryptionDatabase	method	isValid
%CSP.UI.Portal.EncryptionManage	property	lblClose
%CSP.UI.Portal.EncryptionManaged	method	isValid
%CSP.UI.Portal.GlobalList	property	EditGlobalURL
%CSP.UI.Portal.JDBCGatewayServer	property	LogFile, Port
%CSP.UI.Portal.Journal	method	cancelItem, doRemoteBrowse, saveItem, valueChanged
—	property	valueModified
%CSP.UI.Portal.License.Utils	method	EvalResult
%CSP.UI.Portal.Mappings	method	CreateRS, ExecuteRS, ResetRows, SetTempRow, cancelMapping, editMappingD, editMappingE, saveMapping
—	property	ID1, ID2, changesMade, msgLeavingPage
%CSP.UI.Portal.MemoryStartup	property	AutoRtnMem, OldAutoMode, OldRtnMem
%CSP.UI.Portal.Mirror.Dialog.SSL	method	saveSetting
—	property	msgPassword
%CSP.UI.Portal.Mirror.Utils	method	doClose, onPopupAction, onloadHandler, resetMsg, showMsg
%CSP.UI.Portal.NLS	method	CopyDescription, ExportDescription
—	property	EditURL, msgConfirmDelete, msgConfirmInstall, msgExportFileName, msgSelectLocale, msgSelectTable, msgSelectTableName
%CSP.UI.Portal.NLSEdit	method	AddIOTable, DrawHelpText
%CSP.UI.Portal.ObjectGateway	method	cancelSetting, saveSetting
%CSP.UI.Portal.ObjectGatewayStart	method	doDone
%CSP.UI.Portal.ObjectGatewayStop	method	doDone

Class	Type	Name(s)
%CSP.UI.Portal.ObjectSettings	method	doClose, validate
%CSP.UI.Portal.PKI	method	doClose, showErrMsg
%CSP.UI.Portal.ProcessDetails	property	DashboardPage
%CSP.UI.Portal.RemoteDatabases	method	doNew
%CSP.UI.Portal.SQL.Home	method	DrawStatementCell
%CSP.UI.Portal.SQL.QButtons.IndexAnalyzer	method	doGather
%CSP.UI.Portal.SSL	method	cancelSetting, saveSetting
—	property	msgPage2, msgPage3, msgPassword, msgType0, msgType1
%CSP.UI.Portal.Shadow	method	cancelItem, doRemoteBrowse, editMapping, saveItem
%CSP.UI.Portal.Shadows	property	HideColumns
%CSP.UI.Portal.TaskInfo	property	DashboardPage, currTab
%CSP.UI.Portal.ViewLog	property	pageURL
%CSP.UI.Portal.X509Credential	method	cancelSetting, saveSetting
%CSP.UI.Portal.iKnow.Dialog.AddDomainConfig	property	MaxJobsAllowed
%CSP.UI.Portal.iKnow.Settings	property	MaxJobsAllowed
%CSP.UI.System.ExpResultPage	method	MirrorDB
%CSP.UI.System.ExpResultPane	method	DrawMirrorDB
%CSP.UI.System.SecurityAdvisorPane	property	ISREADONLY
%CSP.Util.AutoPage	method	GetEMSDetailPane, GetEMSServerDetailPane, GetEMSTitlePane, GetReadOnlyEMSDetailPane, GetReadOnlyEMSServerDetailPane
%DeepSee.UI.Architect	method	HasOwner, LastSavedTime
—	property	SavedTime, isLocked, msgLock, msgLock1
%DeepSee.UI.Dialog.PivotEditRule	property	defRange
%DeepSee.UI.standardPage	property	isModified
%DeepSee.UserPortal.DashboardViewer	property	isModified
%FileMan.MappedField	property	FILE
%FileMan.MappedFile	property	Classname, FIELDS, FKeys, Indices, MapTimestamp, Maps, SOCClasses, Tablename, Triggers

Class	Type	Name(s)
%SQL.Manager.Catalog	method	GetCachedQueryInfo, GetCalcTableExtentSize, GetCurrentTableExtentSize, GetTableInfo, SetFieldSelectivity, SetTableExtentSize
—	property	CachedQueryInfo, CachedQueryTable, CachedQueryTree, Constraints, FieldCalcSelectivity, FieldCurrentSelectivity, Fields, Indices, NamespacesWithXdbcErrors, ProcedureInfo, Procedures, ProceduresTree, QueryHistory, RWList, SQLCODEList, Schemas, SchemasOnly, Tables, TablesOnly, TablesTree, Triggers, ViewFields, ViewInfo, ViewInfo2, ViewsOnly, ViewsTree, XdbcErrors
%WebStress.Machines.Generators	property	SecurityCheck
%ZEN.Component.abstractPage	method	fireOnLoadEvent
%ZEN.Report.RenderServer	method	RenderServerExists
—	property	ByRenderServer
%ZEN.Report.reportPage	method	GenerateStream
%ZEN.SVGComponent.chart	method	addCommas
%iKnow.Configuration	property	MaxConceptLength
%iKnow.Queries.SentenceQAPI	method	GetHighlighted
%iKnow.Queries.SentenceWSAPI	method	GetHighlighted
%iKnow.UI.AbstractPortal	method	GetUrlForDomain, updateDomainClient
%iKnow.ont.TableManager	method	findSUI
Config.Cluster	parameter	EMSCHANGEBIT
Config.ComPorts	parameter	EMSCHANGEBIT
Config.ConfigFile	parameter	EMSCHANGEBIT
Config.Conversions	parameter	EMSCHANGEBIT
Config.Databases	parameter	EMSCHANGEBIT
Config.Debug	parameter	EMSCHANGEBIT
Config.DeviceSubTypes	parameter	EMSCHANGEBIT
Config.Devices	parameter	EMSCHANGEBIT
Config.ECP	parameter	EMSCHANGEBIT
Config.ECPServers	parameter	EMSCHANGEBIT
Config.IO	parameter	EMSCHANGEBIT

Class	Type	Name(s)
Config.Journal	parameter	EMSCHANGEBIT
Config.LicenseServers	parameter	EMSCHANGEBIT
Config.MagTapes	parameter	EMSCHANGEBIT
Config.MapGlobals	parameter	EMSCHANGEBIT
Config.MapMirrors	parameter	EMSCHANGEBIT
Config.MapPackages	parameter	EMSCHANGEBIT
Config.MapRoutines	parameter	EMSCHANGEBIT
Config.MapShadows	parameter	EMSCHANGEBIT
Config.MirrorAsyncMemberAuthorizedIDs	parameter	EMSCHANGEBIT
Config.MirrorAsyncMemberSources	parameter	EMSCHANGEBIT
Config.MirrorMember	parameter	EMSCHANGEBIT
Config.Mirrors	parameter	EMSCHANGEBIT
Config.Miscellaneous	parameter	EMSCHANGEBIT
Config.Monitor	parameter	EMSCHANGEBIT
Config.NLS.Locales	parameter	EMSCHANGEBIT
Config.NLS.SubTables	parameter	EMSCHANGEBIT
Config.NLS.Tables	parameter	EMSCHANGEBIT
Config.Namespaces	parameter	EMSCHANGEBIT
Config.SQL	parameter	EMSCHANGEBIT
Config.Shadows	parameter	EMSCHANGEBIT
Config.SqlSysDatatypes	parameter	EMSCHANGEBIT
Config.SqlUserDatatypes	parameter	EMSCHANGEBIT
Config.Startup	parameter	EMSCHANGEBIT
Config.Telnet	parameter	EMSCHANGEBIT
Config.config	parameter	EMSCHANGEBIT
Ens.Enterprise.Portal.MsgBankViewer	parameter	SessionTraceColumn
Journal.Restore	method	Monitor
SYS.Database	method	FileCompactUpdateBSB, IntegrityClose, IntegrityExecute, IntegrityFetch
SYS.Mirror	method	GetFailoverMemberStatusEMS
Security.SSLConfigs	property	ListEMS

Method Return Changes

The following methods have different return values in this version of Caché:

- %CSP.UI.Portal.Config.ValueEditor — DrawAfterCreatePage, SaveData
- %CSP.UI.Portal.Dialog.Resource — SaveData
- %CSP.UI.Portal.Dialog.Service — SaveData
- %CSP.UI.Portal.Journal — SaveData
- %CSP.UI.Portal.MemoryStartup — SaveData
- %CSP.UI.Portal.Mirror.JoinAsync — validateConnect
- %CSP.UI.Portal.Mirror.JoinFailover — validateConnect
- %CSP.UI.Portal.NLS — GetLocaleDesc, ReloadDefault
- %CSP.UI.Portal.SQL.QButtons.IndexAnalyzer — GatherStatements, PrepareAnalysis
- %CSP.UI.Portal.Shadow — SaveData
- %CSP.UI.Portal.X509Credential — UpdateDetails
- %DeepSee.UI.Dialog.CubeCompile — ondialogFinish
- %DeepSee.UI.Dialog.FieldList — GetSQLNames
- %DeepSee.Utils — %GetBaseCube

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CPT.JS.BuildCPT	%OnNew
%CSP.Page	EscapeURL, UnescapeURL
%CSP.UI.Portal.Config.ValueEditor	DrawAfterCreatePage, SaveData
%CSP.UI.Portal.DatabaseFreespaceCompact	GetFreeSpace, validate
%CSP.UI.Portal.Databases	filterChanged
%CSP.UI.Portal.Dialog.Compile	updateFlags
%CSP.UI.Portal.Dialog.DBActions	Mount
%CSP.UI.Portal.Dialog.DBMirrorAdd	SaveDataMulti
%CSP.UI.Portal.Dialog.LicenseActivate	Activate, DrawCurrent, PrepareActivate
%CSP.UI.Portal.Dialog.Resource	SaveData
%CSP.UI.Portal.Dialog.SQLView	SaveData
%CSP.UI.Portal.Dialog.Service	SaveData
%CSP.UI.Portal.Dialog.ServiceConnections	AddIP
%CSP.UI.Portal.EMS	InitializeForm
%CSP.UI.Portal.EncryptionCreate	SaveData
%CSP.UI.Portal.EncryptionDatabase	ActivateKey, DeactivateKey, SaveStartup, doDeactivate
%CSP.UI.Portal.JDBCGatewayServer	SaveJDBCSettings

Class Name	Method Name(s)
%CSP.UI.Portal.Journal	SaveData, doBrowse
%CSP.UI.Portal.License.Utils	DrawLicense
%CSP.UI.Portal.Mappings	CancelChanges, DeleteItem, Init, SaveChanges, editMapping
%CSP.UI.Portal.Mirror.JoinAsync	validateConnect
%CSP.UI.Portal.Mirror.JoinFailover	validateConnect
%CSP.UI.Portal.NLS	CopyNow, DeleteNow, GetLocaleDesc, GetLocaleDescription, ReloadDefault, changedDefault, changedLocale, tablenameChanged, tabletypeChanged
%CSP.UI.Portal.NLSEdit	ArrayToString, DrawIntTables, RemoveIOTables, SaveBasics, SaveFormats, SaveIODefaults, SaveIOTables, SaveIntTables, SaveInternalDefaults, SaveStrings
%CSP.UI.Portal.Resources	Delete
%CSP.UI.Portal.SQL.Home	GetCQs, SaveFilter, prepareProp
%CSP.UI.Portal.SQL.QButtons.IndexAnalyzer	PrepareAnalysis
%CSP.UI.Portal.SQL.QButtons.RuntimeStats	PrepareShowPlan
%CSP.UI.Portal.SQL.Utils	SQLExecuteQuery
%CSP.UI.Portal.Shadow	DeleteMap, SaveData
%CSP.UI.Portal.Shadows	DeleteData, deleteItem
%CSP.UI.Portal.Template	showMsg, validateRequired
%CSP.UI.Portal.X509Credential	SaveData
%CSP.UI.Portal.iKnow.Dialog.AddDomainConfig	SaveDomain
%CSP.UI.System.ExpResultPage	CopyMapsFrom
%Collection.ListOfDT	DisplayToLogical, LogicalToDisplay
%Collection.MV.ListOfDT	DisplayToLogical, LogicalToDisplay
%DeepSee.Component.pivotTable	%CreateResultSet, GetItemSpec, allClick, allClickPivot, cbClick, cbClickPivot, getFilterForCells, getLabel, startQueryTimer
%DeepSee.Component.searchBox	dateFromHorolog
%DeepSee.Dashboard.Utils	%GetMemberDimensionType
%DeepSee.Query.query	%LookupCalculatedMember
%DeepSee.Report.UI.reportPreviewer	GenerateDataSet, GeneratePresentationReport
%DeepSee.ResultSet	%PrepareMDX
%DeepSee.TaskMaster	%DequeueTask, %ExecuteTask, %QueueTask
%DeepSee.UI.Dialog.CubeBuild	BuildCube

Class Name	Method Name(s)
%DeepSee.UI.Dialog.FieldList	GetSQLNames
%DeepSee.UserLibrary.Utils	%Import
%DeepSee.UserPortal.DashboardViewer	dashboardEventHandler
%DeepSee.UserPortal.standardPage	updateCompanyCell
%DeepSee.Utils	%ProcessFact
%Library.CacheCollection	CollectionToDisplay, DisplayToCollection
%Library.IProcedureContext	%Next
%Library.Persistent	%OnBeforeBuildIndices, %OnBeforePurgeIndices
%Library.RoutineMgr	getUserDocument
%SYS.Journal.File	GetPrev
%SYSTEM.iKnow	CreateDomainTables
%Studio.SourceControl.ISC	BaselineExport, BaselineExportItem
%Studio.SourceControl.Interface	SourceControlClassGet, SourceControlClassSet, SourceControlCreate
%WebStress.Record	Run
%XML.Node	ReplaceElement
%XML.SAX.Parser	ParseStream
%ZEN.Auxiliary.jsonProvider	submitContent
%ZEN.Component.tablePane	showDateSelector
%ZEN.Report.Display.COSChart.cchart	getSelectedStyle, getXLabelText, getYLabelText
%ZEN.Report.group	DumpCalls, GetXMLFromCall
%ZEN.SVGComponent.chart	getSelectedStyle, getXLabelText, getYLabelText, setProperty
%ZEN.Utils	%GetIncludeDirectory, %GetPhysicalIncludeDirectory, %GetUserIncludeDirectory
%iKnow.DeepSee.BaseKPI	BuildCurrentFilter, BuildCurrentFilterStatic
%iKnow.Matching.DictionaryAPI	GetItemCount
%iKnow.Metrics.MetricAPI	GetValue
%iKnow.Metrics.MetricQAPI	GetValue
%iKnow.Metrics.MetricWSAPI	GetValue
%iKnow.Queries.EntityQAPI	GetOccurrenceCountByDomain, IsAttributed
%iKnow.Queries.MetadataQAPI	GetMetaSpreadByEntities, GetMetaSpreadByEntityIds, GetMetaSpreadInternal
%iKnow.UI.LoadingWizard	CreateConfig

Class Name	Method Name(s)
%iKnow.ont.Matcher	BestMatch, getSynonyms
%iKnow.ont.Matcher2	findPartialMatch
Ens.Host	OnGetConnections
Ens.Rule.Upgrade	ConvertOne
Ens.Rule.Utils	GetPropertyTree
Security.Users	Create

%Library.Persistent Defines New Trigger Names

The class, %Library.Persistent, now defines three new trigger names: `%OBJJRNIU`, `%OBJJRND`, and `%OBJGUID`.

Previously, these triggers were generated during class compilation and were similarly named, but the compiler took steps to avoid name collisions. Now, the new trigger names are statically defined; if they conflict with user-defined names, the names in the applications must be changed.

Single-Property Key Index Correction

A change introduced in version 2011.1 caused the value stored in a key index to contain only the ID value; the value should have contained the ID value and the value of *%%CLASSNAME*. In this release, the error is now fixed, but all affected classes must be recompiled and their indexes rebuilt.

The list of indexes impacted by this bug can be produced by running this query:

```
SELECT
   ci.parent, ci.name
FROM
   %Dictionary.CompiledIndex AS ci
JOIN
   %Dictionary.CompiledIndex AS oi
   ON
      oi.parent=ci.origin
      AND
       oi.name=ci.name
   WHERE
      oi.idkey <> 1
      AND
      ci.type = 'key'
      AND
      NOT
         (oi.parent->final = 1 AND oi.parent->super %STARTSWITH '%Library.Persistent')
      AND
         (
            SELECT count(*)
            FROM %Dictionary.CompiledIndexProperty
            WHERE parent = oi.%ID
         )
         = 1
```

This query will produce one row for each index in each of the classes where it is inherited, starting with the first class that extends %Library.Persistent. It is only necessary to rebuild the index in the first class in a class hierarchy as that will automatically force rebuilding in subclasses.

Add Certificate And DNS Name Checking To %Net.HttpRequest

In this version, when connecting to a SSL/TLS secured web server, the %Net.HttpRequest class will default to checking that the certificate server name matches the DNS name. This is part of RFC 2818, and is used to prevent "man-in-the-middle" attacks. This change can, however, result in a situation where the connection attempt reports errors connecting to a server that previously had been accessed successfully.

Note: This checking can be disabled by setting the *SSLCheckServerIdentity* property to zero.

%SYS.PTools.SQLStats.Export() Changes

In this release, the format of the file written by this class has changed, specifically,

- The file is now tab-delimited instead of comma-delimited. This avoids formatting problems with SQL statements that have commas, and can contain double quotes (delimited Identifiers).
- The blank line first line of the file has been removed.
- The first line contains only the file name and version.
- The data exported comes only from current namespace.

Update %SYS.Task To Permit Activity By Async Mirror Members

In prior releases, the MirrorStatus property in %SYS.Task did not allow async mirror members to run Tasks. In this release, the meaning of the 3 status options has been changed to:

1. Primary
2. Non-Primary
3. Any

This allows async (and backup) members to run tasks under option 2 or 3.

Change To Interpretation of REQUIRED & IDENTITY

In previous versions, an attempt to store a class with a property whose definition included both REQUIRED and IDENTITY would not succeed if the user did not define a value for the property. This is no longer true beginning with this release.

Now, Caché will assign a value automatically if no value is provided. The assignment happens during INSERT and the IDENTITY property cannot be changed after that time. This automatic value assignment satisfies the REQUIRED constraint so the compiler will no longer check the REQUIRED (NOT NULL) constraint for the IDENTITY property.

It is unusual for both IDENTITY and REQUIRED to be true for the same property but some SQL migrations include a NOT NULL constraint on the IDENTITY property. In this case, the new behavior now allows objects whose IDENTITY property has not yet been assigned to pass validation successfully; the old behavior would cause **%ValidateObject** to fail, even though a valid value would be assigned when the object is saved.

Journal.Restore.Monitor() Removed

The **Monitor** method of the Journal.Restore class should no longer be used. Instead, applications should call **MONITOR^JRNRESTF()** instead.

7.2.2.3 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

REQUIRED Constraint Now Fully Enforced

REQUIRED constraints on serial objects are now enforced. Any property whose type class is a %Library.SerialObject and is defined as REQUIRED will fail to save if that property is null. Prior to this change REQUIRED constraints on such properties were not enforced.

This change can also cause objects that once saved to no longer save without error if the referenced serial object is modified or swizzled.

New Class Parameter Added

Beginning with this release, Caché will add a new class property when a class is compiled. The parameter is named %RandomSig and consists of a 20-byte random value recomputed on every class compile.

Always Rerun Parameter Generator Methods In Subclasses

You can define a parameter to be an expression evaluated at compile time of the class using syntax like this:

```
Parameter CompileTime = { $HOROLOG };
```

This will insert the *$HOROLOG* value into the parameter when the class is compiled. However, if you create a subclass and compile this, Caché was just inheriting the superclasses parameter value rather than rerunning the expression generator code and inserting the *$HOROLOG* value of the subclass compile. Method generators are always rerun on subclasses and now expression generators are as well.

Maximum Limit On Index Name

Beginning in this release, the maximum length of an index name is 170 minus the length of the fully-qualified class name.

7.2.2.4 Language Binding Changes

Add Support For Named Parameters In .NET CacheProvider

This release adds support for the use of named parameters in SQL statements SELECT, INSERT, UPDATE and DELETE. Named parameters are now identified by having "@" as a prefix for the parameter name. The "@" is used in both the statement and to create the named parameter.

Named parameters do not need to be added to the CacheParameterCollection in any particular order. When a **Prepare()**, **ExecuteReader()**, **ExecuteNonQuery()**, the @Parameternames in the statement are identified. They must be defined by the time the statement is executed. For example,

```
Set SELECT = "SELECT *"
Set FROM = "FROM Sample.Person"
Set WHERE = "WHERE  id<@highnumber and id>@lownumber and id=@number"
CacheCommand cmd = new CacheCommand((SELECT _ FROM _ WHERE), conn);

 cmd.Parameters.Add(new CacheParameter("@LowNumber", 2));
 cmd.Parameters.Add(new CacheParameter("@Number", 6));
 cmd.Parameters.Add(new CacheParameter("@HighNumber", 9));

 CacheDataReader reader = cmd.ExecuteReader();
```

Note: This change should have no effect on applications unless they were using an "@" parameter name in their queries; these are now reserved for named parameter use.

CacheActiveX Uses Server Locale For BSTR Conversions

In prior releases, conversions involving BSTRs (Basic, or Binary STRings) used the locale of the thread doing the conversion. Now it uses the locale of the server.

TCP/IP Connections Now Uses Connection Timeout

In prior versions, the initial attempt to establish a TCP/IP connection used a timeout of 30 seconds, regardless of the value of `ConnectionTimeout`. This release changes that behavior to use the value specified by `ConnectionTimeout`.

7.2.2.5 SQL Changes

Change Handling Of SQLCODE And %msg In Dynamic SQL

The local variable, *SQLCODE*, is used to report the status of executing an SQL statement. When a dynamic statement is prepared, this local variable was inadvertently being set by %Prepare. The correct way for %Prepare is to report its result is to return a %Status value. In this release, *SQLCODE* is not changed by %Prepare and the only indication of success or failure is the returned %Status value.

Note: In addition to %Prepare, %SQL.Statement supports **prepare()**. The **prepare()** method can be called directly by the user. It accepts the same arguments as %Prepare but instead of reporting a *%Status* value, prepare() will throw an exception if an error is encountered. The exception object can report the status as an SQLCODE value.

Changes To SQL Privilege Handling For Mapped Tables (Classes)

When an SQL table is created, the owner is automatically granted all permissions on the table. Previously, however, if that table was mapped to multiple namespaces those permissions were not propagated to the other namespaces. Similarly, the deletion of a table did not remove the permissions in the other namespaces where the table was mapped.

This release corrects both of those issues: the owner has all SQL permissions in each namespace to which the table is mapped; and, those permissions are removed in each namespace when the table is deleted.

Change To Calling SQL Function In SQL Statement

User-defined SQL functions called from SQL statements are now executed from within try/catch constructs. If the function encounters an error, it will reported as the new SQL error code: -149, "SQL Function encountered an error". In this instance, the variable *%msg* will contain information regarding the error encountered in the SQL Function.

Changes To Selectivity Processing To Handle Outliers

Tunetable now looks for the case where a field value occurs much more frequently than other field values. In such cases, a separate selectivity will be computed for this value, along with a selectivity for "typical" values.

The default selectivity is always used for construction of the query plan unless consideration of the outlier value is requested. If the outlier value is not NULL, (for example, a Massachusetts company with many records containing 'MA'), then in some cases it will be necessary to suppress literal substitution to inform the query processor that the outlier value should be considered. To do this, put the outlier value in double parentheses (('MA')) when used in the Where clause. This syntax is necessary for Dynamic Queries, and for queries written outside of Caché that are sent in using ODBC/JDBC. It is not necessary for class queries, embedded SQL, or queries in a view.

Note: There may be some significant changes in selectivity, producing changes in plans. Despite this net benefit, some applications may need to override the default plans or use hints to get desired plans.

Corrections To ROUND With Scaling Factor Of Zero

In prior versions, SQL columns computing the ROUND function with a SCALE of 0 would not produce the correct SCALE value when the ROUND column came from a subquery or view and the query was run in DISPLAY mode. For example,

```
SELECT a FROM (SELECT {fn ROUND(Age, 0)} a FROM Sample.Person)
```

would return a value with a scale of 2 instead of 0 when selected in DISPLAY mode.

Change To LISTBUILD Function

The SQL **$LISTBUILD** function has been changed so it is now compatible with the Objectscript **$LISTBUILD** function, especially where null arguments are involved. For example, the SQL phrase

```
SELECT $LISTBUILD('a',,,,) ...
```

where the last element was empty previously produced a list of 4 elements. Now it correctly results ina 5-element list. Similarly, the phrase

```
SELECT $LISTBUILD(1,) ...
```

previously produced a 2-element list equivalent to the Objectscript function

```
$LISTBUILD(1, "")
```

while now it correctly results in a list with a NULL second element.

CAUTION: Applications that depend on the specific formats of lists with missing or null elements may have to be changed.

Dynamic SQL Now Updates %SQLCODE

The result of successfully executing a CALL statement is a special dynamic statement result object referred to as the "context object". The context object has two implicit properties: one is the result set sequence (RSS); the other is the implied cursor.

If the called procedure did not return any result sets, then the RSS is empty; invoking %Next() will simply return 0. If there is at least one result set in the RSS, the first call to %Next() will cause the first result set in the RSS to be bound to the implied cursor and the first row from that result set will be retrieved. If no rows are found then %Next() simply returns 0 and %SQLCODE will be 100.

Since accessing the implied procedure cursor sets the value of %SQLCODE and %Message it is important that the caller check %SQLCODE for an error condition after executing the call statement and before accessing the RSS. If an error is reported by the CALL then it is up to the user to decide whether or not to continue processing any result sets that might be in the RSS.

%TSQL.Manager Deprecated

Beginning with this version, %TSQL.Manager is deprecated, and will be removed in some future release. Files containing TSQL statements can be run using **DDLImport()** method of the %SYSTEM.SQL class or the SQL Shell.

7.2.2.6 iKnow Changes

Move System-Level Defaults To Namespace Level

From this release onward, the scope of system-level defaults move to the namespace level. "System-wide" parameters defined in prior releases will be re-registered in all namespaces containing iKnow data when upgraded to this release. Namespaces without iKnow data at the time of upgrade and namespaces created afterwards now need parameter defaults set manually (see %iKnow.Domain.SetParameter and %iKnow.Domain.GetParameter).

Important: An upgrade overwrites the SAMPLES namespace which by default does not contain iKnow domains. Therefore, former system-level defaults will not be registered as namespace-level defaults in SAMPLES.

Standardize Case-Sensitivity Of Element Names

This release makes most of the element names used by iKnow case-insensitive. In previous releases, the names used for Domains, Configurations, and User Dictionaries were insensitive to case. Now the names for Matching Dictionaries, Metrics, Blacklists, and Metadata fields are also.

This leaves only "external" references – Dictionary Item URIs and external ID components (group name and local reference) – as the only "names" still case-sensitive. This is because these names may be driven by external naming conventions.

Compatibility with prior versions and upgrading from these should not result in any issues, unless multiple elements within the same domain had names that only differed in capitalization.

7.2.2.7 Web Services And SOAP Changes

Web Service Test Page Requires %Development Resource

Beginning in this release, the web service test page requires **`%Development:Use`** resource to run.

XML Schema Reader And WSDL Reader Support 189-Character Properties

In this release, the maximum length of property name supported by Caché is extended to 180 characters; preiously, it had been 32. This means that running the wizard for an XSD with type and/or element names longer than 32 characters will result in different property names than the earlier run.

Note: Note that property names used as arguments are still restricted to 31 characters because of the Caché limit of 31 characters for variable names.

Changes In Handling Aggregates in DeepSee and Zen Reporting

In previous releases, when the runtime mode was “display” (the default), the XML data returned from a query would be formatted according to the locale. However, when a Zen or DeepSee Report employed aggregates, the calculation of these aggregates used operators that assumed the data was in US-English format. This produced unexpected results. For example, 1,23 and 1,31 when summed in a European locale would produce 2.

Now the calculation of aggregates is locale-sensitive. Assume the locale is a European one; when the run time mode is display, the "numbers" will first be translated to "US-English" format, so they will work in the arithmetic calculations that use Caché arithmetic operators. When the calculations complete, the results will be translated back to the European format.

Note: Only aggregates, their display and calculation are affected by this change. It is highly unlikely that applications exist which depend on the previous behavior. Nonetheless, it is an incompatible change.

7.2.2.8 MultiValue Changes

Improved Management Of CHAINed Stack

In prior versions, procs that called MVBasic programs that themselves CHAINed procs would leave the prior proc context on the stack. This was an incompatibility with legacy systems. If the call tree was deep enough, this would eventually cause an abort with a stack full. This version detects that condition and removes the prior proc context.

The goal is to allow only one level of proc execution at each execute level to replicate the behavior on legacy systems. However, there are situations when procs are mixed with paragraphs were this is not practical and the layer will not be removed.

Customer who have written their applications to take advantage of the previous behavior will see failures in this version.

MV Spooler Change For SP-FQDELETE And SP-START

If you run SP-FQDELETE or SP-START on a printer form queue, and that form queue already has a lock on it (implying a despool process is active), then the new message will be similar to

```
[882] Error. Unable to take a lock on form queue 'QNAME'.
There appears to be a lock already set by process id 14032.
You need to stop the form queue before you can perform this operation.
```

So if you are testing for the success or failure of SP-START or SP-FQDELETE by doing EXECUTE ... CAPTURING, then the screen output that is captured might differ from what was presented in prior releases.

7.2.2.9 BASIC And MVBASIC Changes

$DEFINE And EQUATE No Longer Apply To Global Names

The preprocessor can not distinguish between a global name and an exponentiation operator with no white space before the exponent. Thus, beginning in this release, no substitution will be done for the exponent local variable in this example:

```
EQU FOO TO EXP
CRT NUM^FOO
```

To get substitution, there must be white space after the operator, such as in

```
EQU FOO TO EXP
CRT NUM^  FOO
```

Note: An alternaitve is to use the “**” operator for exponentiation instead of the “^” operator.

SELECT File With FSELECT Copies File

Anyone using UniData emulation or the FSELECT option who also depends on the

```
SELECT <filevar>
```

statement *NOT* copying the <filevar> contents when the Select List is created will see a change in behavior.

Previously, when the MVBasic compiler translates a "SELECT <filevar>" statement, the newly created Select List is a direct reference to the file variable and does not contain a copy of the file contents. Any write to <filevar> will immediately change the contents of the previously selected Select List. This does not happen for a "SSELECT <filevar>" where the newly created, sorted Select List contains a copy of the indices from the file referenced by <filevar>.

In this new release, a "SELECT <filevar>" statement with $OPTIONS FSELECT turned on will create a Select List containing a copy of indices in the referenced file. Change to the <filevar> file contents will not affect a previously selected Select List of that file providing $OPTIONS FSELECT was enabled when the SELECT statement was compiled. The UniData emulation mode enables FSELECT by default, but all other emulations default to FSELECT being disabled.

To restore the previous behavior, execute a "$OPTIONS -FSELECT" statement in the routine containing the SELECT statement.

Check For Duplicate GLOBAL Symbols

Previously, the MVBasic compiler allowed a name to be defined more than once in the same DIM statement. Now such statement will generate a compile-time error message and the redundant symbol definition must be removed from the DIM statement.

Note: The compiler does allow a name to be defined more than once in separate DIM statements providing the name as never been defined in a global scope (such as a COMMON name.)

Additional Error Return: <ROUTINE TOO COMPLEX>

It is possible that complexity of the source program exceeds the space in preallocated stacks internal to the compiler. Previously, these compilations gave a <STACK> error. In order to better explain these compilation failures the compiler will now give the <ROUTINE TOO COMPLEX> error and it will also report the source line being processed when the excessive complexity was detected.

Anyone who might be detecting the *ERASTACK* value in the error parameter returned from $COMPILE may now have to instead look for the *ERTOOCOMPLEX* value in order to detect a too complex compilation. If $COMPILE is called without an error parameter then *ERASTACK* is still signaled.

7.2.2.10 xDBC Changes

Null Columns Now Report Length And Precision As 1 Instead of 0

Caché now reports a precision and length of 1 for a NULL literal column to xDBC for the column metadata. Previously, Caché reported the length/precision as 0, but that caused issues with at least one client application.

Automatic Conversion Of Long Strings To Streams Removed

It previous versions, applications could ask that literal strings longer than 16K to be automatically converted to character streams. While this supported INSERTs and UPDATEs, queries never supported streams in this way for literal values.

Now that Caché supports long strings in ODBC, the previous functionality is no longer needed. Applications that need to support huge literal string values in SQL statements should make sure that Caché has long strings turned on.

7.2.2.11 DeepSee Changes

Change to MDX Functions When Referring to Date-Part-Based Levels

This release changes the behavior of PREVMEMBER, NEXTMEMBER, and other functions when they are used with members of levels that are based on date *parts*. As background, there are two general kinds of time levels:

- *Timeline-based time levels.* This kind of time level divides the timeline into adjacent blocks of time. Any given member (such as `January 2011`) refers to a single block of time.

- *Date-part-based time levels.* This kind of time level considers only *part* of the date value and ignores the timeline. Any given member (such as `January`) refers to multiple blocks of time from different parts of the timeline.

For date-part-based levels, the level has a fixed number of members (`January` to `December`, for example). When used with a member at the start or end of this set, a function such as PREVMEMBER now returns null. For example:

```
SELECT [BirthD].[January].PREVMEMBER ON 1 FROM patients

                                        *
```

This result is correct, because the `January` member refers to records related to January in *all* years, and it is not meaningful to access records "earlier" than that.

In previous releases, the engine would have returned the records for `December`.

Change to PREVMEMBER and NEXTMEMBER Functions When Referring to Null Members

This release changes the behavior of PREVMEMBER and NEXTMEMBER so that these functions now return null when used with the null member of a level.

In previous releases, in both cases, the engine would have returned the records for the first non-null member of the same parent.

Editing Quality Measure Requires IE9

Beginning with this release, if the user chooses to edit the Quality Measures of DeepSee Tools using Internet Explorer, a version at or above IE9 is required.

Cube Manager Package Name Changed

This release changes the package name of the DeepSee cube manager from DeepSee.CubeMgr to DeepSee.CubeManager. Existing applications that reference classes in this package will need to be changed.

Visual Report Security Checks

In this version, specifying data sources now requires both the proper license and **`%Development:USE`** privilege. In addition, formating a DeepSee Visual Report requires both the proper license and **`%Deepsee_ReportBuilder:USE`** privilege.

Import Restriction On Legacy Zen Reports

In this release, you can only import data schema from a legacy Zen Report if that reportdefinition section <group> definition tags. Report definitions lacking <group> tags (such as those that use <call> or COS callbacks to gather their data) do not provide the information needed for DeepSee Visual Report to interface properly to the report, and are not suitable sources for this particular data collection builder screen.

Zen Report Variable Substitution Disallowed

Zen reports follows the convention of question marks used as placeholders in SQL WHERE clauses for parameters to be mapped later. DeepSee Visual Report uses `#(expression)#` syntax to inject the parameters directly into the clause (this is done to simplify the visual editor and hide the syntax structure of the underlying XData).

Starting in this release, DeepSee Visual Report both blocks the entry of the question mark as a keystroke as it is being entered and pops up an alert box suggesting the use of the #( )# syntax.

7.2.2.12 CSP Changes

CSP Session Support For REST

By default, the REST support introduced in this version takes out a license slot per request. This means that if you have a low license count (or no license), you are limited by the CSP grace period of 5 seconds. If you run out of license units, you will get a 503 HTTP response. You can avoid this by using CSP sessions, even though this (strictly speaking) violates the spirit of REST.

You enable CSP sessions by setting the UseSession parameter to 1 in your class definition and recompiling. The server will generate the CSPSESSION cookie and expects it to be returned in subsequent requests. This means you have to manage the CSPSESSION cookie manually if not using a browser. Tools such as CURL provide that facility with command line parameters but its also manageable in code.

Important: The name of the parameter which affects CSP session use has changed from EndSession to UseSession and its sense has changed as well; specify 1 to use sessions, and 0 to not.

New SECURITYRESOURCE Parameter Added

In this release, the %CSP.Page class defines a new parameter called SECURITYRESOURCE. This is a comma-delimited list of system resources and associated permissions. A user must hold the specified permissions on all of the specified resources in order to view this page or invoke any of its server-side methods from the client. The format of each item in the list is:

```
Resource[:Permission]
```

Permission is optional, and defaults to USE if not supplied. If it is supplied, it should be one of USE, READ or WRITE.

You can also specify OR grouping using the "|"character, so "R1,R2|R3,R3|R4" means you must have resource R1, and one of R2 or R3, and one of R3 or R4. So if you have R1,R3 it will pass, if you have R1,R4 it will not as it does not meet the R2|R3 condition. (The "|" (or) condition takes precedence over the "," (and) condition.

Do Not Store Cookies With Long Names

If the browser sends over an invalid cookie value where the key name is longer than 510 characters, Caché will skip attempting to set this into the %request.Cookies array because the array cannot handle subscript lengths this long.

Variables Used In OnSessionStart CSP Session Event Callback

In a prior release, the CSP event callback logic was changed and all session event callbacks protected any variables the user created in the callback were cleaned up to avoid exposing them in the main page logic. This change of behavior on OnStartSession callback affected several existing applications.

In this release, the previous behavior can be enabled by

```
Do $SYSTEM.CSP.SetConfig("ProtectOnStartSessionCallback", 0)
```

which will turn off the exclusive NEW in the OnStartSession event callback so public variables set here can be seen in the CSP page code.

7.2.2.13 Zen Changes

Changes To tablePane Component Functionality

If your application uses Zen components that generate queries (anything that inherits from querySource, such as tablePane) *and* you have not set up SQL privileges on your system, then it is possible for a client application to generate a query from the wrong table. This is now prevented by setting ZENENCRYPT=1 for the *tableName* property of the querySource class. This causes the table name to be encrypted when shipped to the client and prevents the client from modifying it.

Applications that rely on setting the *tableName* property in JavaScript will no longer work and will throw an <ENCRYPT> or similar error. The showQuery property, a diagnostic aid for displaying the current query, in tablePane and dataCombo is also now encrypted.

Additional Zen Component Security Changes

To further secure the queries automatically generated by Zen components, such as tablePane and dataCombo, Zen now prevents client logic from modifying generated queries in undesirable ways.

- In the %ZEN.Auxiliary.column class, used by tablePane, the colExpression property is now encrypted. This means that this property can only be set on the server.

- In the %ZEN.Auxiliary.QueryInfo class, used by many components to generate SQL statements, the columns and *columnName* properties are now required to be valid SQL property names, not expressions. This means that in the cases where an SQL statement is automatically created, the column names must not be expressions or subqueries. For cases where an expression is needed, the application should already be using colExpression.
- In the %ZEN.Component.querySource class, the *queryClass* property is now encrypted to prevent the client from modifying this.

Empty Cell Values No Longer Show Default Value

Cells that have no value no longer default to using the ChoiceColumn value; empty fields are now left blank. Their display is controlled by the value of the Zen dataCombo attribute, "showEmpty".

7.2.2.14 Zen Reports Changes

Class Attributes Inherited By Child Table Elements

In this release, table elements such as <tr>, <th> and <td> now inherit the class attribute values from their containing element. In previous release, the attribute values were ignored.

8

Caché 2013.1

This chapter provides the following information for Caché 2013.1:

8.1 New and Enhanced Features for Caché 2013.1

The following new features have been added to Caché for this release:

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the "Caché 2013.1 Upgrade Checklist."

8.1.1 Rapid Application Development

8.1.1.1 Support for %FIND in WHERE Conditions

With this release we implement a new SQL Condition with the following syntax:

```
<f> %FIND <x>  [ SIZE <const>]
```

<x> represents a set of values S used as a filter for <f>, or more precisely, the condition is satisfied if <f> is a member of S.

8.1.1.2 XEP Enhancements

This version of Caché contains the following improvements and extensions to XEP:

- It has expanded support for inheritance, references, interfaces, and synchronous indexing.
- It supports simpler options for persistence modes.
- XEP now has two types: XEP TCP/IP and XEP JNI.
- A new @Id annotation for both Java & .NET allows a field to be a database ID.
- New mechanisms are available for calling Cache Object Script class methods.
- XEP now has support for calling ObjectScript functions, procedures with multiple arguments, and return types.
- Performance improvements have been made to array-based data inserts
- There is a new parallel storage API to maximize multicore machine capabilities, and further increase performance of inserts.
- InterSystems enhanced JDBC-based queries to replace previous custom query functionality.
- XEP now has enhancements that reduce the amount of JNI usage across platforms.

8.1.1.3 JDK 1.7

This release now includes the Java 7 libraries as well as those for Java 6. In addition, Caché now supports JDBC 4.1.

8.1.1.4 GLOBALS Enhancements

The version of GLOBALS $query is now supported for both Java and .NET APIs. The Globals ValueList now support decimal values. Globals methods can now call Cache Object Script function, procedures with multiple arguments and return types; and Globals methods can now call Cache Object Script class methods. Support for Visual Basic is now available along with a sample that demonstrates its use.

8.1.1.5 RETURN Statement

The RETURN statement is now available for application developers to optimize code execution and readability. Its definition is analogous to the QUIT statement, but it returns from an enclosing subroutine to its caller.

8.1.1.6 Support Arguments By Reference With Args...

The args... syntax for receiving and passing a variable number of arguments can now be used when passing an argument by reference.

8.1.1.7 Websocket Support

The CSP Gateway now includes support for the HTML 5 specification for Web Socket connections between the web server and an HTML 5 compliant browser. This is available for Apache 2.2 and above, and will be available for IIS 8.0 which is part of Windows Server 2012.

There is a base class that must be extended by the user to handle / create WebSockets called %CSP.WebSocket

8.1.1.8 Improved FrameStack Handling

<FRAMESTACK> errors can now be handled in error traps. Caché will no longer remove items from the execution stack before calling an error trap.

8.1.1.9 Third-Party Library Upgrades

In this release, ANTLR has been upgraded to version 3.4; Xerces has been updated to version 3.1.1 on UNIX®.

8.1.1.10 Studio Assist For MVB

Studio assist now recognizes the following syntax elements:

- Caché Basic: "ClassName".ME.
- MVBasic: "ClassName"->@ME->

Caché Basic: "ClassName".ME. MV BASIC: "ClassName"->@ME->

8.1.1.11 XML Support For List of Streams

A list of streams will be projected correctly to XML. A list of streams may be specified by either using 'list Of' syntax or creating class which is a sublcass of %ListOfObjects with ELEMENTTYPE specifying a streamclass.

8.1.1.12 JSON Improvements

InterSystems made a major enhancements to our public JSON APIs. We now have non-internal function calls that can generate JSON strings from objects, arrays, or SQL statements. These can be subsequently be used to drive a third party client interface that supports JSON as a data transport mechanism. In addition to converting internal structures to JSON, we also support taking JSON sent by the client and converting it into a COS array that can then be consumed within an application.

8.1.1.13 DeepSee Improvements

DeepSee includes a new editor for creating and editing dashboards. The new dashboard editor takes advantage of improved ZEN charts and other new ZEN components. To edit a dashboard, you click on the bar along the left side of the dashboard. The new dashboard editor provides many more options for controlling the display of charts and tables.

The DeepSee User Portal also includes a new "Covers" view option which displays items on the home page as images. The new covers editor allows you to configure the covers (images, text, colors, etc.). Also, the types of items listed on the User Portal home page now include reports and URL links (in addition to dashboards and pivots).

8.1.2 Performance And Scalability

8.1.2.1 XML Performance

Performance for XML has been improved by improving the handling of XML between Caché and Xerces. Customers may see improvements of up to 50% for %XML.Reader imports, and up to 25% with SOAP messages.

8.1.2.2 HTTP 1.1 Keep-Alive

This release of Caché adds support for Keep-Alive with HTTP 1.1. It enables clients to keep the TCP/IP socket to the web server active so that multiple requests can be made via the same socket rather than constantly closing the socket and opening a new one for each request.

8.1.2.3 Improved De-Journaling Performance

InterSystems has improved de-journaling performance in this version by further optimizing the pre-fetching algorithms used. De-journaling - used by Shadowing, Mirroring, and journal roll-forward during startup recovery - uses a pre-fetching mechanism to speed up the actual roll forward of sets and kills to the databases. These pre-fetchers are responsible for "reading ahead" blocks into the buffer pool so that the actual updater process can move through very quickly and apply updates, and not be slowed down by waiting for disk reads to finish. This version includes a supplementary algorithm that allows the pre-fetchers to read in additional supporting (link) blocks, further speeding up the overall de-journaling performance.

8.1.2.4 Improved ROLLBACK Performance

In this release, ROLLBACK performance is roughly 10%-30% faster than in previous versions. This improvement applies to rolling back single (large) transactions, as well as bulk rollback (as would occur during startup recovery).

8.1.2.5 AES Encryption On Intel Hardware

Intel support AES-NI with their latest processors, which greatly improves encryption and decryption for AES. Early benchmarks have shown significant performance improvements over software execution of AES, which greatly minimizes the performance overhead for encrypted databases with high a high frequency of physical I/O.

8.1.2.6 Improved Async I/O Performance

This version includes improvements to asynchronous I/O that benefits sequential workloads. The new algorithm detects multiple sequential updates and performs larger coalesced writes, and has been measured to speed up sequential workload-related writes by a factor of 1.6 to 1.8.

8.1.2.7 Improved Encryption Performance

Previously, a performance feature called "parallel WIJ encryption" was enabled on the AIX platform. This feature has now been enabled on HP-UX, Linux, and Solaris, and can increase the speed of database encryption during write cycles by up to a factor of 4.

8.1.2.8 MultiCompile On By Default

Caché now uses MultiCompile for class compilation as a default, which will decrease compile time on multi-core processors.

8.1.2.9 Optimize Multi-Key Joins

This release removes some older case-specific functions used to optimize multi-column joins in favor of a more generic approach that results in much greater performance across all types of multi-column join statements.

8.1.2.10 Export/Import TuneTable Statistics

With this release, Caché SQL has added support for exported and importing TuneTable Statistics for a given namespace. This allows one to export TuneTable Data, Selectivity, and ExtentSize details from a given system and reload it without having to ship a class definition and compile it.

8.1.2.11 Improve Concurrent Update Performance

This version contains an optimization for concurrent updates on very large-scale systems. With this optimization, benchmarks have demonstrated 20% to 30% improvement in overall database access scalability.

8.1.2.12 Add Indices To A Class With Minimal Downtime

This version introduces a new feature, %Library.IndexBuilder. This class and its methods permits one or more indexes to be added to an existing class. It is specially designed to address the circumstances where:

- the normal **%BuildIndices** method will take a long time because of the amount of data involved, and
- the application is running on a system where downtime must be kept to a minimum.

8.1.2.13 Lock Escalation Improvements

This release introduces a new locking mode, "E", whose goals are to automate lock management and provide better management of the lock table. In this mode, when the number of existing locks held by a transaction crosses a threshold, and Caché determines it is safe to do so, the lock will be "consolidated" as a higher level lock. The lower level locks will then be released thus freeing space in the lock table.

8.1.3 Reliability, Availability, Maintainability, Monitoring

8.1.3.1 Support Advanced Format (4K) Disk Drives

This version includes support for native 4KB disk sectors as implemented by Advanced Format (AF) disk drives.

8.1.3.2 TIFF Output In Zen Reports

Zen Reports now has the capability to create content in TIFF format in addition to our previously released formats such as HTML, XML, PDF, Excel, and so on.

8.1.3.3 Application Specific Licensing

Some Caché customers wish to license their own applications as a whole or in functional units. Among the challenges they facein doing so is the proper accounting for license units in the face of varying application workflow and abnormal terminations. In addition, some customers desire to link their licenses to the underlying system license provided by InterSystems.

With Application Specific Licensing, InterSystems provides mechanisms to support the customer need to create, deploy and use application licenses. InterSystems itself uses this form of licensing for its products HealthShare and TrakCare.

8.1.3.4 Very Large Global Buffer Size

This version includes support for a very large global buffer pool. Previously, the limit was around 107 GB; global buffers can now be configured up to 16 TB.

8.1.3.5 Large Shared Memory Heap

This version includes support for a very large shared memory heap. Previously, the limit was around 4GB; the shared memory heap can now be configured up to 1 TB.

8.1.3.6 Eliminate Configuration-Related Restarts

The following Caché configuration options can now be updated dynamically, and do not require the system to be restarted for the new setting to take effect:

- MaxServers: Maximum number of connections to ECP servers from this (client) machine.
- VMSConsoleTerminal: Name of the console terminal device for logging messages on an OpenVMS system.
- netjob: Enable/disable remote job requests.

- wijdir: Write Image Journal file directory.
- console: Console log file name.
- LineRecallBuffer: Total size (in bytes) of all input strings to be stored in the command line/read line buffer.
- LineRecallEntries: Maximum number of entries held in the command line/read line recall buffer, subject to the space limitation in the LineRecallBuffer setting.
- bbsiz: The maximum memory allocation to permit for a process (in kilobytes).

8.1.3.7 Autostart Pre-Fetch Daemons As Needed

The block pre-fetching mechanism ($PREFETCHON / $PREFETCHOFF) requires system-wide pre-fetch daemons to be initialized. This version of Caché automatically ensures that the pre-fetch daemons are started as needed, and are stopped when no longer needed, thereby eliminating the need to manually start and stop these daemons.

8.1.3.8 Improved SQL Performance Tools

With this release Caché contain additional utilities that provide useful functions for analyzing SQL behavior on production data: how many times an Index is used; identify Queries that result in table scans, or that have dependencies on building temporary structures; and identify queries based on Joins and rank how well supported they are by existing indices.

8.1.3.9 Simpler Disaster Recovery Failover And Failback

In previous releases, Caché Database Mirroring has provided an Asynchronus Mirror configuration for Disaster Recovery purposes. In this release, InterSystems has made it easier to switch over to an Asynchronous DR Mirror node in the event of a disaster (or for testing purposes); and also made it easier to switch back to the main production failover mirror at a later date.

8.1.4 Security

8.1.4.1 WS Reliable Messaging

With this release Web Services Reliable Messaging can be used to reliably (identify, track and manage) deliver messages between a source and a destination.

8.1.5 Technology Preview

This category is new with this release. It is intended as a way of introducing and providing access to new software capabilities that InterSystems believes will be useful in enhancing the effectiveness of existing and future applications.

The capabilities listed here are ready for use by customers, but they are not yet complete in functionality and design. Customers who take advantage of these capabilities must understand:

- InterSystems makes no backward compatibility guarantee with future updates;
- Customers may incorporate these capabilities in deployed applications, but must first check with InterSystems to determine best course of action;
- Customers who deploy these capabilities in their applications must commit to upgrading to the final released version.

InterSystems stongly encourages those who incorporate these items in their software to provide feedback on their experiences.

8.1.5.1 Rapid Application Development

iKnow - Customized Summaries

This release of Caché contains the capability to generate custom summaries for sources. Custom summaries are developed for those who desire to tune the content of iKnow generated summaries to their specific needs.

Main parameters for tuning the summaries are the following:

- The user can indicate the number of the sentences he wants always to be included in the custom summary, e. g. when summarizing a newspaper article it can be beneficial to always include the first two sentences.
- The user can enumerate a list of words and/or sentence parts he wants to use as absolute triggers, this means that sentences containing one of the elements in the list will always be included in the summary or always be excluded from the summary.
- The user can give more or less importance to certain words and/or sentence parts. In this case the weight of sentences containing one of the elements in the list will be adapted in accordance with the weight instructions specified by the user. These adaptations will influence the sentence rankings used in the summarization and include or exclude the elements specified by the user if the context of the text allows to do so.

The options for custom summaries can be set by means of an extra parameter in the SourceAPI.GetSummary method.

iKnow Support For Negation

This release of Caché introduces iKnow negation detection. The uniqueness of iKnow negation detection resides in the fact that not only linguistic markers of negation in a sentence such as “not”, “no” and “didn't” are detected, but that iKnow also marks their semantic scope. In the example:

John has no signs of headache but has a fever.

the linguistic negation marker is “no”. iKnow negation detection will not only detect the marker, but also detect the semantic span of the negation. This means it will establish that “John has no signs of headache” is the negative part of the sentence. The span of the negation is indicated by negation attribute markers that can be accessed and used by the following method calls:

- EntityAPI.GetOccurrenceAttributes(): to see whether an occurrence of an entity contains a negation marker or not.
- SentenceAPI.GetHighlighted(): to highlight the negative parts in sentences
- SentenceAPI.GetAttributes(): to get the positions of negation markers and their scope markers at the level of the sentence
- PathAPI.GetAttributes(): to identify the negative parts of pathway
- SourceAPI.GetAttributes(): to get all negations for a source.

Refactoring

Beginning in this release InterSystems has included the first support for the refactoring of code in the Studio. Initially, the available options will be:

- Change Class Name
- Change Property Name
- Change Method Name

Additional options will be incorporated in future releases.

In this release, refactoring is only available on Windows systems.

8.2 Caché 2013.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2013.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2012.2 and 2013.1.

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

8.2.1 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 2013.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

8.2.1.1 Version Interoperability

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

8.2.1.2 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 addition of pages to assist with database mirroring and the separation of roles.

8.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

Check CACHETEMP/CACHE Encryption Status On Upgrade

The installer will check the encryption status of CACHE and CACHETEMP on upgrade and will ask for encryption key file location, username and password if it finds either database is encrypted. This information will be validated before file copying begins and the values will be passed to ^INSTALL.

Note: This behavior is not avauilable for RPM and DMG kinds of installs. In these cases, database encryption should be turned off prior to upgrade with RPM/DMG installers and re-enabled afterward.

Default Answer For ^JRNSTOP Changed To "No"

The default answer for ^JRNSTOP has been changed from "Yes" to "No":

```
%SYS>d ^JRNSTOP
Stop journaling now? No =>
```

This change may affect scripts that assume a default of "Yes".

Ask User For Confirmation Before Starting Journal Restore

A new prompt is now part of the journal restore utility (^JRNRESTO). The user must confirm the restore request before journal restore actually starts. With the change, the screen output looks like this:

```
...
  DEFAULT: Continue despite database-related problems (e.g., a target
  database is not journaled, cannot be mounted, etc.), skipping affected
  updates

  ALTERNATE: Abort if an update would have to be skipped due to a
  database-related problem (e.g., a target database is not journaled,
  cannot be mounted, etc.)

  DEFAULT: Abort if an update would have to be skipped due to a
  journal-related problem (e.g., journal corruption, some cases of missing
  journal files, etc.)

  ALTERNATE: Continue despite journal-related problems (e.g., journal
  corruption, some missing journal files, etc.), skipping affected updates

Would you like to change the default actions? No => No

Start the restore? Yes => Yes
```

This change will affect scripts that are not prepared to respond to the prompt.

Allow Mirror Dejournaling On Reporting Async Members To Be Stopped

All mirror members have a new menu item in ^MIRROR – Pause dejournaling for a database. On a backup member or an async member this allows you to notify the mirror dejournaling system to skip dejournaling to this database. This can be used if maintenance needs to be performed on a database for some reason (for example, it needs to be dismounted so it can be encrypted or decrypted).

On a disaster recovery async member or a backup a mirrored database can simply be dismounted and, if the dejournaling system runs into an error such as a <PROTECT> error, the database will be marked to be skipped and dejournaling to other databases will continue. It is a bit cleaner though to first pause the database because this drains any pending data in the dejournaling system before blocking any future records from being processed.

On a reporting member, however, dismounting a mirrored database can result in the dejournaling system shutting down for all databases in that mirror if there are subsequent records in the journal for that database. This is done because, on reporting members, sites may want to keep all databases in a given mirror in sync with each other so if there is an error in any one, dejournaling should stop there.

If you want dejournaling to continue it can then be restarted using a new ^MIRROR menu item for reporting async members, Manage mirror dejournaling on reporting member, and it will continue without the dismounted database. Alternatively, the database can be paused before it is dismounted and then dejournaling for other databases in the mirror won't shut down if there is data for the paused database in the journal stream.

System Monitor Now Extended To User Databases

Beginning with this release users may create Sensor, Subscriber, and Notification classes to extend System Monitor in non-SYS namespaces. This change requires that all SYS.Monitor.Abstract classes be changed to %SYS.Monitor.Abstract. In particular, users of SYS.Monitor.AbstractSensor must change the declaration to be %SYS.Monitor.AbstractSensor.

Use VIRTUALALLOC() and MMAP() For Larger Stack And Error Frames

The frame stack and error stack are no longer allocated as byte arrays in the partition. They are now allocated using VirtualAlloc() on Windows or mmap() for UNIX® and OpenVMS. Using this mechanism, Caché can allocate a larger frame stack with little impact on the system when the memory is not actually used. It can afford to set a larger buffer between the end of the space allowed to be used and the actual end of memory.

Error processing will now recognize that there is still unused buffer space available. So instead of popping existing frame entries to make room for executing an error trap (which risks losing important things like an application error trap), it will now reset the frame size to the actual end of the stack memory, which makes the previously unused stack space available to the error trap.

There are new $SYSTEM methods that can be used for testing the effects of this change:

- $SYSTEM.Process.FrameStackSize()

  gives the frame stack size for the current process.

- $SYSTEM.Process.FrameStackExpanded()

  returns 0 or 1 to indicate if the frame stack has been expanded after a <FRAMESTACK> error to allow for an error trap to be run. If this value is 1, then another <FRAMESTACK> error will cause the process to terminate without running an error trap. It will automatically return to 0 after enough stack entries are popped to return to normal operation.

- $SYSTEM.Util.GetFrameStackSize()

  returns the configured stack size. This is the size that will be used for any new processes that are created.

- $SYSTEM.Util.SetFrameStackSize(newsize)

  sets the configured stack size to a new value and returns the previous value. This new value will be used as the frame stack size for all newly created processes. If *newsize* is 0, the system default size is set.

Note: In concert with this new allocation policy, the size of the execution stack, which limits the levels of DO, XECUTE, and so on that are allowed, has been doubled. This does not, in practice, double the memory load on the system unless the full memory is actually used. However, systems with many processes may experience a heavier burden on system memory and/or paging space.

It is possible that for systems where the memory usage was close to the allowed limit, this change may cause memory exhaustion.

New Return Values From System And Memory Status Check

Anew register stored in shared memory and maintained by the System Monitor tracks system health. The values stored here are:

- 0 – green: all is well
- 1 – yellow: warnings are present
- 2 – red: alerts have been posted

All System Monitor subscribers may post notifications. Yellow is set when level 2 alerts are posted. Red is set when 5 alerts have occurred in 10 minutes. The health goes back to green if there are no further alerts for 1 hour.

Users can obtain the status via the following commands:

```
ccontrol list
ccontrol qlist
cache -cV
```

or programmatically via **$SYSTEM.Monitor.State()** which returns the values defined. See also the **SetState**, **Alert** , **Clear**, and **ClearAlerts** methods.

Third-Party Library Changes

In this release, ANTLR has been upgraded to version 3.4; Xerces has been updated to version 3.1.1 on UNIX®; CacheSSH now uses libssh2-1.4.2.

Make DNS Name Parser RFC1123 Compliant

The DNS name parser grammar that handles TCP endpoint specifications is now properly compliant with RFC1123. Previously, the parser did not allow any portion of a dotted DNS name to begin with a digit. According to RFC1123, it is only the final dotted portion of the DNS name that has this restriction in order to distinguish it from an IPv4 address in dotted quad form.

Recompile Generated Code During Install

During an install, Caché normally recompiles all user Z*, z*, %Z*, and %z* routine and classes. However, prior to this release, it skipped any routines marked as "Generated". Generated routines are routines which are derived or created by compiling other routines or classes.

Beginning with this release, a routine can be marked as generated when there is no source present. This can happen in two cases: (1) the original source is deleted; and (2) a %RO of .INT code generated by a .MAC file is marked in the %RO file as being generated (%RI, it retains the Generated flag), but has no source.

Caché now detects this case during install, and if the generated code is present, with no corresponding source for it, Caché will recompile the generated code.

$VIEW Adds Peak Memory Usage To Output

Beginning withthis release, **$VIEW(-1)** adds the peak memory usage, in Kb, as the 17th value returned. For example:

```
Set VSS= $VIEW(-1, $JOB)
Write "Peak memory usage for this process in KB is "_$PIECE(VSS, "^", 17), !
```

Start Shadow Servers As The User Who Started Caché

The system starts Shadow servers with the same user characteristics regardless when they are started -- whether at Caché startup or after Caché is up and running. Previously, when they were started in the later scenario (Caché is up and running), they could take on the characteristics of a different user. When that happened, process limits and file ownership might differ depending on the circumstances. Now they are always the same

Changes To $ZCONVERT

If a Unicode string ending in a high surrogate character is passed to the 4-argument form of $ZCONVERT() for UTF-8 output translation, for example,

```
Set S = $ZCONVERT(string, "O", "UTF8" , x)
```

this function now saves the last character in the local variable (*x* in this example). The value in *x* will be prepended to the input string in the next iteration.

If the string in the next iteration starts with a low surrogate character, the prepended *x* and the leading character are treated as a surrogate pair and translated accordingly. Otherwise, the previous high surrogate character is translated as a 3-byte sequence, and the initial character is treated individually.

The same rules apply to the JSML (Javascript/XML) translation.

Licenses Prompt During Upgrade Changed

Beginning with this release, the installation process will not ask users to enter a license key during the install. Instead, it will ask for the location of the key file. If the location is not known at upgrade time, the file can also be copied to the manager's directory or entered via the Management Portal after the installation finishes.

Output From Ccontrol Changed

When a mirror member is in the process of transitioning to become the primary, it temporarily blocks users from signing in for part of the transition. An application which is parsing any output from attempting to enter Caché may need to be updated to be sensitive to a new message which says:

```
Sign-on inhibited: Transitioning to primary mirror member
```

ECP Recovery Of Open Transactions Across Upgrade

ECP recovery delays handling of the ECP open transactions until the networking subsystem is initialized. During an upgrade, if the system was shut down during the startup recovery, and if there were open transactions that spanned multiple journal files, the second attempt to rollback those open transactions may fail after the system upgrade.

Note: To avoid this circumstance, make sure the system is shutdown gracefully before upgrading.

USE Permission Required For DeepSee Portal

As of this release, to use any DeepSee page, a user must have USE permission for the **`%DeepSee_Portal`** resource, in addition to other privileges required for the page. Unless you are using minimal security (which is not recommended for production environments), this change may require you to adjust your role or user definitions.

In particular, if your application includes embedded dashboards, you must provide any applicable users with USE permission for **%DeepSee_Portal** as well, so that they can use the dashboard viewer. In previous releases, users of embedded dashboards did not require this privilege.

Conversion Of Control Characters Changed

Begining with this release, **$ZCONVERT(string, "O", "JS")** and **$ZCONVERT(string, "O", "JSML")** will now translate the $CHAR(1) through $CHAR(7), and "/", to their hexadecimal equivalents: \x01 through \x07 and \x2F equivalently. Previously these characters passed unchanged.

If your application relies on **$ZCONVERT** for Javascript or Javascript Markup, you must change your application to accommodate this change – most often by using the hexadecimal equivalent of these characters.

Resource Required For Viewing %Monitor Data

In order to view data collected by the %Monitor facility via CSP, the user will need to have **%Admin_Manage**, **%Admin_Operate**, and **%Developer** on the relevant CSP pages and any subclass pages.

Change To Memory Usage And Allocation

In this release, the XML reader now uses local variables for storing its information; previously, it used process-private variables. This results in both a performance improvement and an additional demand on memory. Depending on the sizes of XML files read and the processing mix on the system, the memory allocation may need to be increased to avoid fatal errors. this can be done:

- permanently by setting the bbsiz parameter in the CPF file.
- dynamically by having the application adjust the value of the $ZSTORAGE special variable.

8.2.1.4 Platform-specific Items

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

Windows

- Pdfprint Now Requires The PrintServer On Windows

  On Windows platforms, pdfprint now requires the built-in PrintServer. The PrintServer is a Java program which uses the Adobe Acrobat reader to print.

  The user needs to define a PrintServer.properties file in the PrintServer directory which defines the location of the Adobe Reader, for example, on Windows 7:

  ```
  adobe=C:/Program Files (x86)/Adobe/Reader 10.0/Reader/AcroRd32.exe
  ```

  A ZEN Report now contains new properties, parameters, and ZEN URLs to pdfprint:

  – Property ZENURL PARAMETER

  – PrintServer $PRINTSERVER PRINTSERVER

  – PrintTimeOut $PRINTTIMEOUT PRINTTIMEOUT

  There is also a PS property, $PS ZENURL and PS parameter to define a printer for GenerateReport. The following URL shows how we put it all together to print a report using Acrobat Reader on Windows. (Line breaks are inserted to improve readability.)

  ```
  http://localhost:57790/csp/samples/ZENApp.MyReport.cls
       ?MONTH=1
      &$MODE=pdfprint
      &$PRINTSERVER=4321
      &$PRINTTIMEOUT=30
      &$PS=Brother%20HL-5250DN%20series
  ```

Note: PS must be the name of the printer in the Printer/General Tabs setting in Devices and Printers/<printer>/Properties.

- Cterm Option For VT Left-Cursor Command

  The Terminal emulator did not behave like a VT-nnn terminal under the following circumstances:

  – Auto wrap is configured.

  – A character has just been written to the last display column so the terminal is just about to wrap.

  – A sequence of N backspaces, or the cursor left N position escape sequence (CSI n D) issent..

  Under these circumstances, a VT terminal will move the cursor N positions to the left of the last column, while the Terminal would move the cursor N positions from a position just to the right of the last column of the display. This put the cursor in the Terminal one character to the right of where it would be on a VT terminal. The Terminal now behaves like the VT terminal under these circumstances if the following registry entry is defined:

  – On 32–bit Windows:

    HKLM\SOFTWARE\InterSystems\Terminal\UseStrangeVTBackspace="1"

  – On 64 bit Windows:

    HKLM\SOFTWARE\Wow6432Node\InterSystems\Terminal\UseStrangeVTBackspace="1"

  Important: Activating this behavior may create problems editing wrapped lines in programmer mode after command line recall. InterSystems does not recommended that customers enable the behavior unless they require this behavior in their applications.

- Changes To Address Missing .NET Assemblies

  While it is risky potentially for an application to not have all the components listed in its manifest installed properly, it appears to be a common situation in cases where it is certain the missing code will not be traversed or used. Beginning with this version, Caché will now continue loading all the assemblies it can, listing any that cannot be loaded in the log.

  Users who have problems running code because of deployment issues should use the log to resolve dependency issues.

Windows And UNIX® / Linux

The Caché internal web server has been updated to version 2.4.2 on all Windows, UNIX® and Linux platforms. The installer will automatically update httpd.conf as needed. This behavior applies to Windows install, UNIX® install, standalone CSP Gateway installation script and RPM installation for Linux.

Linux RedHat 32-Bit And 64-Bit

None.

MacOS 64

None.

PowerPC And HP-Itanium

During an upgrade, Caché now modifies the CPF file and increases the amount of memory allocated to the lock table by twice its current amount. It does this by doubling the existing *locksiz* parameter in the `[config]` section. This change also adds the same amount of memory to the *gmheap* parameter which is where the lock table memory is allocated from.

During an initial install, the default *locksiz* is now increased by twice the default, and the *gmheap* is also increased by the same amount.

Customers who are configuring new systems should increase the size of the lock table they use by twice the former amount, and add the same amount of size to the *gmheap* parameter.

OpenVMS Itanium

In this version, the file CACHE.EXE has been separated into two files: CACHE.EXE and CKERNEL.EXE.

If you use the Caché command line, you should be unaffected by this change. Running CACHE.EXE should have the same behavior as before, providing you keep CACHE.EXE and the corresponding CKERNEL.EXE in the same directory.

If you have modified CZF.C or any other source file so that CZF.EXE now does callins (the default CZF.EXE does not do any callins), then changes are necessary. The CZF.EXE image is activated by $ZF("RoutineName",...) calls. If you have built any other image that does callins, changes in how that image is built are also necessary. Previously, these images were LINKed against the CACHE.EXE shareable image. They must now be linked against the CKERNEL.EXE shareable image.

The CLINK.OPT linker options file that is installed into the [InstanceDir.source.cache] directory has been modified so that it refers to CKERNEL.EXE instead of referring to CACHE.EXE. When doing an OpenVMS LINK using CLINK.OPT/OPTIONS, you must either define the CKERNEL logical name to point at the location of the [InstanceDir.bin]CKERNEL.EXE file, or you must have a copy of CKERNEL.EXE in what is the default directory during the execution of the LINK.

If you do not use the CLINK.OPT options file when linking an image that does a Caché kernel callin, you must modify any command file or LINK command that references CACHE.EXE. Most references to CACHE.EXE must be changed to references to CKERNEL.EXE so that your build procedures can use the universal symbols defined by CKERNEL.EXE.

If you have created an executable image that calls in to the Caché kernel, that image must do a LINK as described above. However, when you execute that image, it is also necessary to locate the corresponding copy of CKERNEL.EXE. You do this by defining the logical name CKERNEL before running your image. (You previously had to define the logical name CACHE so that your executable program could find the CACHE.EXE shareable library image.) Defining the CKERNEL local name will look something like:

```
$ DEFINE CKERNEL Device:[InstanceDir.BIN]CKERNEL.EXE
```

If the image you built is INSTALLed as a known image with the addition of the /PRIVILEGED qualifier, the logical name CKERNEL must be created with DEFINE/EXECUTIVE_MODE or with ASSIGN/EXECUTIVE_MODE. Privileged executables that activate a shared library image require the shared library image to be a known image that is located by means of an executive mode (or kernel mode) logical name.

If you have an image that dynamically activates CACHE.EXE by calling **LIB$FIND_IMAGE_SYMBOL**, you must rewrite your call on **LIB$FIND_IMAGE_SYMBOL** so that it dynamically activates CKERNEL.EXE.

8.2.2 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.

8.2.2.1 Global Name Reservations

InterSystems reserves to itself all global names in the CACHESYS database except those that start with:

- "%z" or "%Z"
- "z" or "Z"

In all other databases, InterSystems reserves all global names starting with "ISC." and "%ISC.".

8.2.2.2 Routine Compiler Changes

None.

8.2.2.3 Routine Changes

VarArgs May Now Contain References As Well As Values

Beginning in this version, the args... syntax for receiving and passing a variable number of arguments can be used when passing an argument by reference. For example,

```
   set x=2,y(3)=4
   do sub1(.x,.y,.z)
   ; now $DATA(x)=0, y(1,2)=3, z=5
 quit
sub1(args...)
 set args(2,1,2)=3
 do sub2(args...)
 quit
sub2(a,b,c)
 kill a
 ; b(1,2)=3 and b(3)=4
 set c=5
 quit
```

Note: If current programs are passing arguments by reference with this notation, their behavior could change because the actions of the subroutine will now affect the original variables. Before this change, they were actually passed by value so the subroutine could not affect their value.

Z*.inc And z*.inc Include Files Renamed

In this release all Z*.inc include files are renamed to ISC*.inc; and all z*.inc include files are renamed to ISCz*.inc files. This is to avoid having Caché system include file names conflict with user include file names.

8.2.2.4 Class Changes

Class Deletions

The following classes were present in the previous version and have been removed in this release

- %BI — Excel, ExcelTry
- %CSP.UI.Portal — PrintQuery, PrintTable
- %CSP.UI.Portal.Config — Journal, RemoteDatabase, RemoteDatabases
- %CSP.UI.Portal.SQL — ExecuteQuery
- %CSP.UI.SQL — CQInfoPane, ViewInfoPane
- %CSP.UI.System — AppGroupTablePane
- %DeepSee — FactView, MetaView
- %DeepSee.Report.UI — Builder, BuilderCore
- %DeepSee.UI — SVGWidgetCatalog, WidgetPreview
- %DeepSee.UI.Dialog — WidgetBuilder
- %Monitor.System — Dashboard, Dashboard2
- %Monitor.System.Sample — Dashboard, Dashboard2
- %Net.Remote.Java — JavaGatewayService
- %WebStress — HttpRequest
- %WebStress.Utils — Recorder

- %WebStress.Utils.Recorder — Port, Recorded, Script, Status, Summary
- %iKnow.Filters — SimpleMatchFilter
- SYS.Monitor — AbstractSensor

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%BI.CDC	method	OnPage
%BI.CodeTableCSP	method	OnPage
%BI.DMG	method	OnPage
%BI.DatePicker	method	OnPage
%BI.EdmundPageCSP	method	OnPage
%BI.ExportImport	method	OnPage
%BI.ExportList	method	OnPage
%BI.Generator	method	OnPage
%BI.GridPivoting	method	OnPage
%BI.Help	method	OnPage
%BI.ImportList	method	OnPage
%BI.KnowledgeCSP	method	OnPage
%BI.LanguageSVGCSP	method	OnPage
%BI.LanguageSub	method	OnPage
%BI.ListFolderData	method	OnPage
%BI.PerfAlert2	method	OnPage
%BI.RulesSetup	method	OnPage
%BI.Scorecard	method	OnPage
%BI.SystemVariable	method	OnPage
%BI.TimeSeries	method	OnPage
%BI.Trees	method	OnPage
%BI.UserObjectListWeb	method	OnPage
%BI.ValueRange	method	OnPage
%BI.WebADTM	method	OnPage
%BI.WebAnalyzer	method	OnPage
%BI.WebAnalyzer2	method	OnPage
%BI.WebAnalyzerSetup	method	OnPage
%BI.WebCMeditor	method	OnPage

Class	Type	Name(s)
%BI.WebCdnEditor	method	OnPage
%BI.WebClsCom	method	OnPage
%BI.WebColorScheme	method	OnPage
%BI.WebComBank	method	OnPage
%BI.WebComponentKPI	method	OnPage
%BI.WebComponentTree	method	OnPage
%BI.WebCurrency	method	OnPage
%BI.WebDList	method	OnPage
%BI.WebDashboard	method	OnPage
%BI.WebDataCapture	method	OnPage
%BI.WebDocBrwsr	method	OnPage
%BI.WebDtAnalysis	method	OnPage
%BI.WebETL	method	OnPage
%BI.WebEmail	method	OnPage
%BI.WebExcelTemplate	method	OnPage
%BI.WebFile	method	OnPage
%BI.WebFold	method	OnPage
%BI.WebGrid	method	OnPage
%BI.WebKPIClass	method	OnPage
%BI.WebKnowledge	method	OnPage
%BI.WebLesenReg	method	OnPage
%BI.WebListbackup	method	OnPage
%BI.WebLnkRpt	method	OnPage
%BI.WebMain	method	OnPage
%BI.WebMain2	method	OnPage
%BI.WebMainTable	method	OnPage
%BI.WebManEnt	method	OnPage
%BI.WebMapTable	method	OnPage
%BI.WebMessaging	method	OnPage
%BI.WebMessagingExt	method	OnPage
%BI.WebMultiSites	method	OnPage
%BI.WebOCR	method	OnPage
%BI.WebPivTable	method	OnPage
%BI.WebPrinter	method	OnPage

Class	Type	Name(s)
%BI.WebPrn	method	OnPage
%BI.WebSCActions	method	OnPage
%BI.WebSMS	method	OnPage
%BI.WebSQLComGen	method	OnPage
%BI.WebSVG	method	OnPage
%BI.WebSVGTest	method	OnPage
%BI.WebSchema	method	OnPage
%BI.WebSchemaMain	method	OnPage
%BI.WebSecCorrection	method	OnPage
%BI.WebShortcut	method	OnPage
%BI.WebSqlWiz	method	OnPage
%BI.WebStoredProc	method	OnPage
%BI.WebSysFn	method	OnPage
%BI.WebTAClassDialogBox	method	OnPage
%BI.WebTableList	method	OnPage
%BI.WebTaskMgmt	method	OnPage
%BI.WebTransformation	method	OnPage
%BI.WebTranslationD	method	OnPage
%BI.WebUsersMtn	method	OnPage
%BI.WebUtil	method	OnPage
%BI.WebWorkFlow	method	OnPage
%BI.WebWorkFlowE	method	OnPage
%CSP.Login	method	DrawLogout
%CSP.UI.DocLocalize	method	DatabaseFreespaceCleanup, DatabaseFreespaceCompact, DatabaseFreespaceList, Journal, RemoteDatabase, RemoteDatabases
%CSP.UI.Portal.ClassList	method	allowSelectRow
—	property	AllowSelectRow
%CSP.UI.Portal.DatabaseFreespace	method	clearSpace
%CSP.UI.Portal.DatabaseFreespaceCleanup	method	DrawTitle1
—	property	msgValidate
%CSP.UI.Portal.DatabaseFreespaceCompact	method	DrawTitle1
—	property	msgValidate
%CSP.UI.Portal.Dialog.ExportResource	property	ActionDone

Class	Type	Name(s)
%CSP.UI.Portal.ProcessDetails	method	doOwnerDrawTabs, drawOneTab, drawTabs, startRefresh, tabClicked
—	property	TABAREAHEIGHT, TABHEIGHT, TABMAXWIDTH, TABOVERLAP, TABRADIUS, TABTEXTMARGIN, TABTOPREDUCTION, TEXTVOFFSET
%CSP.UI.Portal.RoutineCompare	method	getHTML
%CSP.UI.Portal.RoutineList	method	allowSelectRow
—	property	AllowSelectRow
%CSP.UI.System.SecurityAdvisorPane	property	readOnly
%DeepSee.Report.UI.ExecuteReport	method	showFileSelectionWindow
%DeepSee.Report.UI.ExtractDSS	property	nameSpace
%DeepSee.Report.UI.reportModelServer	method	ServerCommand
%DeepSee.Report.UI.schemaEditPanel	method	IsSQLReservedWord, processFieldName
%DeepSee.UI.Dialog.DashboardSave	method	DrawIcons, selectLayout, updateGridControls
—	property	dashboardDescription, dashboardGridCols, dashboardGridRows, dashboardKeywords, dashboardLayout, dashboardLocked, dashboardModify, dashboardResize
%DeepSee.UI.TermListManager	method	columnClick

Class	Type	Name(s)
%DeepSee.UI.WorksheetBuilder	method	CreateDataSet, GetLookupArray, GetWSData, LookupReferences, SaveToServer, SetRecord, applyChange, changeCell, changeColumn, changeGrid, changeRow, clearStyles, deleteItem, editCellHandler, editFormatString, getCellValue, getCurrPageCount, getCurrPageSize, getLogicalValue, gotoGridPage, gridChange, gridEditKeyDown, gridEditKeyUp, gridRender, initEngine, loadDataConnector, loadWorksheet, moveColumnLeft, moveColumnRight, moveRowDown, moveRowUp, nextGridPage, numberToColumn, prevGridPage, printWorksheet, recalc, renderNavBar, resolveFormulaRefs, selectCellHandler, selectColumn, selectGrid, selectRow, setActionTarget, setRecordValue, textEditorKeyup, textOKClicked, toggleSettings, updatePropertyPanel, updateStyleControls
—	property	currPage, dataConnector, gridMode, gridSelected, pageSize, selectedColumn, selectedRow, worksheetDescription, worksheetKeywords, worksheetLocked, worksheetOwner, worksheetPublic, worksheetResource
%DeepSee.UserPortal.DashboardViewer	method	SaveDashboardToFolder, addWidget, editWidget
%DeepSee.UserPortal.Home	method	DrawFoldersList
%Library.EnsembleMgr	method	AddDelegated, CreateFoundationResources, EnableFoundationNamespace, GetHealthShareNamespaceType, InstallFoundationSecurity, IsFoundationInstalled, IsHealthShareInstalled, IsViewerInstalled, ValidateHealthShare, map2hslib, moveHSSiteDefaultFiles, unmaphslib, upgradeUTCIndices
%Library.RoutineMgr	method	IsServerOnly

Class	Type	Name(s)
%Library.SQLCatalog	method	GetCachedQueryInfo, MakePat, SQLCODEListClose, SQLCODEListExecute, SQLCODEListFetch, SQLCachedQueryInfoClose, SQLCachedQueryInfoExecute, SQLCachedQueryInfoFetch, SQLCachedQueryTableClose, SQLCachedQueryTableExecute, SQLCachedQueryTableFetch, SQLChildTablesClose, SQLChildTablesExecute, SQLChildTablesFetch, SQLConstraintsClose, SQLConstraintsExecute, SQLConstraintsFetch, SQLFieldsClose, SQLFieldsExecute, SQLFieldsFetch, SQLForeignKeysClose, SQLForeignKeysExecute, SQLForeignKeysFetch, SQLIndicesClose, SQLIndicesExecute, SQLIndicesFetch, SQLParentTableClose, SQLParentTableExecute, SQLParentTableFetch, SQLProcedureInfoClose, SQLProcedureInfoExecute, SQLProcedureInfoFetch, SQLProceduresClose, SQLProceduresExecute, SQLProceduresFetch, SQLRelationshipsClose, SQLRelationshipsExecute, SQLRelationshipsFetch, SQLReservedWordsClose, SQLReservedWordsExecute, SQLReservedWordsFetch, SQLTablesClose, SQLTablesExecute, SQLTablesFetch, SQLTriggersClose, SQLTriggersExecute, SQLTriggersFetch, SQLViewFieldsClose, SQLViewFieldsExecute, SQLViewFieldsFetch, SQLViewInfoClose, SQLViewInfoExecute, SQLViewInfoFetch

Class	Type	Name(s)
%Library.SQLCatalogPriv	method	SQLRolePrivilegesClose, SQLRolePrivilegesExecute, SQLRolePrivilegesFetch, SQLRoleUserClose, SQLRoleUserExecute, SQLRoleUserFetch, SQLRolesClose, SQLRolesExecute, SQLRolesFetch, SQLUserExistsClose, SQLUserExistsExecute, SQLUserExistsFetch, SQLUserPrivsClose, SQLUserPrivsExecute, SQLUserPrivsFetch, SQLUserRoleClose, SQLUserRoleExecute, SQLUserRoleFetch, SQLUserSysPrivsClose, SQLUserSysPrivsExecute, SQLUserSysPrivsFetch, SQLUsersClose, SQLUsersExecute, SQLUsersFetch
%Library.SysLog	method	ConfigurePage, DecodeData, OnPage, RenderBanner, ServeStyleSheet, ShowDataPage, ShowLogPage
%MV.Adaptor	parameter	MVAUTOLOCKED

Class	Type	Name(s)
%SQL.Manager.Catalog	method	CachedQueryInfoClose, CachedQueryInfoExecute, CachedQueryInfoFetch, CachedQueryTableClose, CachedQueryTableExecute, CachedQueryTableFetch, ConstraintsClose, ConstraintsExecute, ConstraintsFetch, FieldCalcSelectivityClose, FieldCalcSelectivityExecute, FieldCalcSelectivityFetch, FieldCurrentSelectivityClose, FieldCurrentSelectivityExecute, FieldCurrentSelectivityFetch, FieldsClose, FieldsExecute, FieldsFetch, IndicesClose, IndicesExecute, IndicesFetch, MakePat, NamespacesWithXdbcErrorsClose, NamespacesWithXdbcErrorsExecute, NamespacesWithXdbcErrorsFetch, ProcedureInfoClose, ProcedureInfoExecute, ProcedureInfoFetch, ProceduresClose, ProceduresExecute, ProceduresFetch, PropertyInfo, RWListClose, RWListExecute, RWListFetch, SQLCODEListClose, SQLCODEListExecute, SQLCODEListFetch, SchemasClose, SchemasExecute, SchemasFetch, SchemasOnlyClose, SchemasOnlyExecute, SchemasOnlyFetch, TablesClose, TablesExecute, TablesFetch, TablesOnlyClose, TablesOnlyExecute, TablesOnlyFetch, TriggersClose, TriggersExecute, TriggersFetch, ViewFieldsClose, ViewFieldsExecute, ViewFieldsFetch, ViewInfo2Close, ViewInfo2Execute, ViewInfo2Fetch, ViewInfoClose, ViewInfoExecute, ViewInfoFetch, ViewsOnlyClose, ViewsOnlyExecute, ViewsOnlyFetch, XdbcErrorsClose, XdbcErrorsExecute, XdbcErrorsFetch
%SYS.ProcessQuery	method	PPGClose, PPGExecute, PPGFetch
%SYS.Task.IntegrityCheck	method	DirectoryIsValid
%SYSTEM.Event	method	DefinedIP, ListIP, SignalIP

Class	Type	Name(s)
%UnitTest.Report	method	DeleteSuite, GetTestColor, GetTestState, NoNamespace, ShowAsserts, ShowCases, ShowIndices, ShowMethods, ShowSuites, writeStyle
%XML.Node	method	AppendChild, InsertChild, ReplaceNode
%ZEN.Portal.assistedText	method	ghostGotFocus
%ZEN.Report.Display.COSChart.cchart	method	calculateYAxisWidth
%iKnow.DeepSee.Dimensions.Dictionaries	method	GetId
%iKnow.DeepSee.UI.Analysis.AbstractAnalysis	method	GetDomainId, GetFilter
%iKnow.DeepSee.UI.Analysis.Content	method	SetListing
%iKnow.DeepSee.UI.Analysis.Entities	method	GetDetailChartData, GetDetailLabelX, GetSelectedItemName, ReBuildEntityList, SetSelectedCell, getDetailChartDataClient, getDetailLabelXClient
%iKnow.Domain	property	SortField
%iKnow.Matching.DictionaryAPI	parameter	GetDictionaryElementsRT
%iKnow.Matching.MatchingWSAPI.InvalidateMatchingResults	property	keepEntUniMatches
%iKnow.Queries.MetadataWSAPI.AddField	property	buildBitstring
%iKnow.Queries.MetadataWSAPI.UpdateField	property	buildBitstring
%iKnow.Queries.MetadataWSAPI.UpdateFieldById	property	buildBitstring
%iKnow.Queries.SourceAPI	method	GetByEquivalentIds, GetByEquivalents
%iKnow.UI.AbstractPortal	method	GetDomain, GetTerm
%iKnow.UI.IndexingResults	method	GetSentFrom, GetSentTo
Ens.Enterprise.Portal.EnterpriseSearch	parameter	AssistantClass
Ens.Enterprise.Portal.MsgFilter.Filter	method	DeleteFromTemp, LoadFromTemp, SaveToTemp
Ens.Enterprise.Portal.SystemList	method	getSSLConfigList, getSoapCredentials
Ens.Rule.Model.ruleDefinition	method	test

Method Return Changes

The following methods have different return values in this version of Caché:

- %DeepSee.Report.UI.reportModelServer — ImportXML
- %DeepSee.Report.UI.reportPreviewer — DeleteTempFile
- %SQL.Import.Mgr — Import
- %SYSTEM.iKnow — IndexDirectory, IndexFile

- %XML.Node — InsertCharacter, InsertElement

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CSP.UI.Portal.Config.ZenReport	SaveData
%CSP.UI.Portal.DatabaseFreespaceCleanup	GetSize
%CSP.UI.Portal.DatabaseFreespaceCompact	GetFreeSpace
%CSP.UI.Portal.RoutineList	doFullView
%CSP.UI.System.GlobalDrillPane	SaveChange
%DeepSee.AbstractKPI	%GetFilterLogicalValue
%DeepSee.Component.pivotController	showMessage
%DeepSee.Component.pivotTable	getDataSourceCaption
%DeepSee.Dashboard.Utils	%GetMembersForFilter
%DeepSee.QualityMeasure.Utils	%GetQualityMeasureClass
%DeepSee.Report.UI.CreateDCR	scanForFieldNames
%DeepSee.Report.UI.reportPreviewer	GeneratePresentationReport
%DeepSee.Report.UI.schemaEditPanel	autopopulateDBItem, makeNode
%DeepSee.UI.WorksheetBuilder	gridKeyDown
%DeepSee.UserLibrary.FolderItem	%CheckResource
%DeepSee.UserPortal.DashboardViewer	navGetContentForLevel, navSelectItem, widgetSelected
%DeepSee.UserPortal.Home	refreshFolders, setFolderCategory
%DeepSee.UserPortal.standardPage	navSelectItem
%DeepSee.extensions.iKnow.ClassifierBuilder	genCrcTable
%Library.AbstractStream	Read
%Library.CacheCollection	CollectionToDisplay, DisplayToCollection
%Library.GTWCatalog	SQLTablesExecute, SQLTablesJExecute
%Library.GlobalEdit	CollationSet, CompactGlobal, Open
%Library.GlobalStreamAdaptor	Read, ReadLine
%Library.RegisteredObject	%OnValidateObject
%Library.RoutineMgr	ImportItemListExecute, getBackupList, getPackageList
%Net.Remote.Service	RunStartCmd, StartGateway, StartGatewayObject
%Net.Remote.Utility	RunCommandViaCPIPE, RunCommandViaZF
%SOAP.Addressing.Properties	GetDefaultRequestProperties

Class Name	Method Name(s)
%SQL.Util.Procedures	CSV, CSVTOCLASS
%SYS.PTools.SQLStats	GetStats, GlobalSave
%SYSTEM.OBJ	Load
%SYSTEM.iKnow	IndexDirectory, IndexFile
%Stream.Object	Read
%Stream.TmpCharacter	Read
%Studio.Extension.Base	OnAfterAllClassCompile, OnAfterClassCompile
%Studio.SourceControl.ISC	BaselineExport, BaselineExportItem
%WebStress.Record	Run
%XML.ImportHandler	SerializeBase64Node, SerializeNode
%XML.Node	AppendCharacter, AppendElement, InsertCharacter, InsertElement
%XML.Security.Signature	ComputeSha1Digest
%ZEN.Auxiliary.jsonProvider	%WriteJSONFromArray, %WriteJSONFromObject, getDataSourceCaption
%ZEN.Controller	InvokeClassMethod, InvokeInstanceMethod
%ZEN.Dialog.fileSelect	RebuildLookin
%ZEN.Portal.assistedText	doAction
%ZEN.Report.Display.COSChart.cchart	createSVGTextNode, getYAxisTitle, renderYAxisTitle, renderYLabels
%ZEN.Report.PrintServer	%ServeTransformAndPrint
%ZEN.SVGComponent.chart	calculateYAxisWidth, createSVGTextNode, getYAxisTitle, renderYAxisTitle, renderYLabels
%ZEN.SVGComponent.radialNavigator	drawNode
%ZEN.SVGComponent.tabBar	drawOneTab
%iKnow.Configuration	%OnNew
%iKnow.DeepSee.UI.Analysis.Entities	GetChartData, GetLabelX, getChartDataClient, getLabelXClient
%iKnow.Domain	%OnNew
%iKnow.Filters.Filter	GetFilteredCcFrequency, GetFilteredCcSpread
%iKnow.Matching.MatchingAPI	GetTopMatchesByDictionaryItemId, InvalidateMatchingResults
%iKnow.Matching.MatchingQAPI	InvalidateMatchingResults
%iKnow.Matching.MatchingWSAPI	GetTopMatchesByDictionaryItemId, InvalidateMatchingResults
%iKnow.Queries.MetadataAPI	AddField, UpdateField, UpdateFieldById

Class Name	Method Name(s)
%iKnow.Queries.MetadataI	AddField
%iKnow.Queries.MetadataQAPI	AddField, UpdateField, UpdateFieldById
%iKnow.Queries.MetadataWSAPI	AddField, UpdateField, UpdateFieldById
%iKnow.Source.Lister	SetConfig
%iKnow.Source.Processor	%OnNew
%iKnow.Utils.MaintenanceAPI	AddUserDictionaryEntry
%iKnow.Utils.MaintenanceQAPI	AddUserDictionaryEntry
%iKnow.Utils.MaintenanceWSAPI	AddUserDictionaryEntry
Backup.General	ExternalFreeze
Config.NLS.Locales	ImportDir
Config.NLS.SubTables	ImportDir
Config.NLS.Tables	ImportDir
Ens.Director	StartProduction
Ens.Enterprise.Portal.MsgFilter.EnterpriseAssistant	EnumerateExecute
Ens.Rule.Generator	generateActions, generateOneAction, generateOneRuleSet
Ens.Rule.Model.expression	constructCOS, convertToCOS, test
SYS.Database	Defragment
SYS.Monitor.Health.Chart	%OnNew

JavaGatewayService Class Deprecated

In this release the %Net.Remote.Java.JavaGatewayService class has been deprecated. This was an internal class; it was not documented for external use.

If you have developed custom code that relies on this class, you should update the code to instead use the %Net.Remote.Service class that incorporates the functionality of the deprecated class.

Object Synchronization

If an object is synchronized to a namespace which does not contain the class, the Caché will now catch the <CLASS DOES NOT EXIST> error and log it rather than abandoning the synchronization attempt. This gives the application the ability to more gracefully recover from the error.

If an application expected a failure in this case, it must be modified to check the error reported and handle the error condition.

Change Collation Handling In %Library.GlobalEdit.Create()

Either the external collation name (e.g. "Spanish2") or its internal number (e.g. 32) can now be used as the 3rd argument to **%Library.GlobalEdit.Create()**. If the specified collation is invalid or is not loaded, an error is generated. Previously, this method would accept only the numerical value, and silently use the database default collation if it were invalid.

Fix To Symbol Binding For Delimited Identifiers

A defect that caused an identity property to not be set when saving a new object has been fixed. If a class defines a property whose name is a delimited identifier as the IDENTITY property, the property was not properly set following an insert.

HTTP Changes

In this release, by default, if you reuse an instance of %Net.HttpRequest to send multiple requests, Caché sends all the messages over a single HTTP socket (using a HTTP 1.1 keep-alive connection). Specifically, Caché keeps the TCP/IP socket open so that Caché does not need to close and reopen it. For information on disabling this keep-alive behavior, see "Managing Keep-alive Behavior" in the chapter "Sending HTTP Requests and Reading Responses" in *Using Caché Internet Utilities*.

Change%Net.Remote.DotNet.Test To Accommodate New Directory Structure

Beginning with this version, Caché has two directories for .NET Gateway: one for Version 2.0, and the other for Version 4.0. The test method **ListTypeLibs()** of the class %Net.Remote.DotNet.Test now references the directory for Version 2.0 by default. Users may change the version by executing:

```
Do ##class(%Net.Remote.DotNet.Test).ListTypeLibs(port,,"4.0")
```

where *port* is a valid port number not in use at the time of the call.

Resource Requirements For Studio Source Control

With this release, the **%Developer** resource is needed to use the following Studio Source Control classes: %Studio.SourceControl.UI, %CSP.StudioTemplateInsert, and %CSP.StudioTemplateSuper.

8.2.2.5 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

IDs Used For Bitmap Index Must be Integers

The class compiler will now report an error if the class uses %CacheSQLStorage and attempts to define a bitmap index map and the ID of the table is not an integer type.

If your tables use SQL Storage, and there is an IDKEY index on the table, and the IDKEY is not defined on a single field that has a datatype with an SqlCategory of INTEGER, the following error will be reported:

```
Compiling class User.MyTable
ERROR #5485: Bitmap indices are only supported when the IDKEY is based on a single positive integer
attribute
 ERROR #5030: An error occurred while compiling class User.MyTable
```

Prior to this change, if a bitmap index map was defined, no check was performed for SQL Storage to make sure the ID was compatible.

Note: Any existing applications that have tried to use this probably have bad index data if the ID contains anything but positive integers. In this circumstance, the index should be deleted and rebuilt.

Multicompile Is On By Default

Beginning with this release, the ability to use multiple processes to speed compilation, or loading XML classes from a directory, ("/multicompile") is enabled by default. To change the default to off for a system you can execute:

```
  Do $SYSTEM.OBJ.SetQualifiers("/multicompile=0", 1)
```

Projections Now Utilize Multiple Jobs

When you run a class compile with /multicompile turned on, Caché will now run the class projection calls to CreateProjection using the worker jobs in the same way it compiles classes. This improves the performance of this phase of the compile if class projection code is doing a significant amount of work.

Most projection code will continue to function as it has prior to this change; however, the two areas that should be noted are:

1. Projections may be run in multiple, simultaneous jobs so if a projection is using *$JOB* to store information to co-ordinate with other projection code it should change to use *%ISCName* which will be defined and be the same amongst all multicompile jobs.
2. It is important to lock data when you need exclusive access to prevent interference. With multiple projections running at the same time, there is a chance another projection may be manipulating the same data structure in another process. This was a potential issue prior to this change when you compiled the two classes at the same time in different jobs, but with multicompile it makes the chance of this occurring higher.

Notice Of Serial Property Change

In prior versions, the REQUIRED constraint was never effectively enforced. It will be in the next version.

Any existing application with properties whose type class is %Library.SerialObject and are defined as REQUIRED will now see that constraint enforced. This can cause objects that once saved to no longer save without error if the referenced serial object is modified or swizzled.

Parameter Properties Format Changed

The stored format of parameter values for compiled programs has been changed to be a $LIST. If your application parses the stored value of a parameter using SQL, it will need to be changed. The new method **GetParameterValue** has been added with the %Dictionary to assist in this. Also, the query %Dictionary.LegacyQuery will return the values in the previous format.

8.2.2.6 Language Binding Changes

XEP EventPersister Changes

The OPTION_REFERENCES and OPTION_INHERITANCE (and related flags) have been removed. After removal of these two options, no options are left and the EventPersister.setOption and getOption methods have also been removed.

Since these last two options are really generation/import time flags, the following two importSchema flavors have been added in their place:

```
String[] importSchema(String classOrJarFileName, int options)
String[] importSchema(String[] classes, int options)
```

where the options are:

- IMPORT_OPTION_NONE = 0

  No import options - defaulting to no references and flattened hierarchy

- IMPORT_OPTION_REFERENCES_ENABLED = 1

  References are enabled as part of the import

- IMPORT_OPTION_FULL_INHERITANCE = 2

  Do not flatten inheritance

Changes To Enum Handling

This release changes the projection of Java and DotNet enums. Names, and not ordinal positions, are now stored on the Caché side. This change should be completely transparent for new applications.

Old applications, that already have some enums stored, will have to delete and then re-import the data to avoid errors.

8.2.2.7 SQL Changes

The SQL Filer No Longer Calls IsValid For Reference Properties

The SQL Filer will no longer call the <property>IsValid method for a reference to a persistent class (for example, the Sample.Person.Spouse property). It is rare that a <property>IsValid method would even exist (it must be defined by the developer of the class). The Object **%Save** method does not call <property>IsValid methods for properties of type persistent or serial either, so now the Object and SQL filers are in sync in this behavior.

If an application depends on this behavior, the reference will have to be validated in another way, perhaps through a trigger.

Correction To StorageToLogical And LogicalToStorage

This release corrects an existing problem storage and retrieval of list or array collections where StorageToLogical and/or LogicalToStorage methods were defined for the property or datatype of the property. Previously, Caché was not properly applying the StorageToLogical or LogicalToStorage to each element of the collection. Now it does with the effect that the returned values for list or array collections may differ from previous releases.

Correct Translation Of Delimited Identifier To Numeric Literal

A Dynamic SQL bug that caused a delimited identifier to be translated into a numeric literal has been fixed. Previously, when a delimited identifier that is also a valid numeric literal was encountered by the statement preparser, it would be translated into a numeric literal. For example, in the query:

```
SELECT Name FROM Sample.Person WHERE  %ID = "1"
```

the literal “1” was interpreted as the integer value 1. The correct translation is that the delimited value "1" is retained and the statement will now report an error when prepared.

Correction To TRUNCATE SCALE In DISPLAY Mode

Previously, “{fn TRUNCATE(value, number)}” where number is greater than 0 and not 2 would display the wrong SCALE for the results when the column was selected in DISPLAY mode; the output of the result in DISPLAY mode would always use a SCALE of 2.

Now the output is properly scaled.

When DDL Compiles A Class, Compile Sub-Classes Too

When an SQL DDL statement is executed and a class definition needs to be compiled, the class now compiled using the “b” flag. This will compile any sub-classes for this class. This behavior is needed for many types of DDL statements, for example, where an index was added to a class with a CREATE INDEX statement, and the class has a subclass. Without this, the building of the index will not work for any rows in the subclass extent until the subclass has been compiled.

Note: It is thought to be rare that customers run DDL against tables where the class has subclasses.

SQLTablePrivileges And SQLColumnPrivileges Changes

In prior versions, the ODBC SQLTablePrivileges and SQLColumnPrivileges catalog queries, along with the JDBC getTablePrivileges and getColumnPrivileges catalog queries, would only return privileges where the current user was the grantee of the privilege or a role held by the user was a grantee of the privilege. Now a row is also returned if the current user is the grantor of the privilege.

DROP TABLE From Management Portal Subject To Checking

In this release, you will no longer be able to drop a table, view, or procedure from the Management Portal if DDLAllowed=False is defined in the class that projected the SQL object.

Note: This does not apply for linked tables. If a class is defined as a linked table, the Drop action will drop the linked table on the local system even if the linked table class is defined as DDLAllowed=0; but it will not drop the table this link references on the server he actual table resides on.

Compiled Names Of Queries Have Changed

In previous versions, if embedded SQL defined a query within a routine or method, and the routine or method was called recursively, the generated name of the cursor would cause the earlier value to be overwritten by the later statement execution. Unpredictable results ensued.

Begining in this release, the SQL Query Processor has changed how cursor variable names are generated and each compile of the statement will generate a different variable name. If application code is relying on SQL to generate certain variable names for an SQL cursor, this code will likely need to be updated to not rely on the variables names generated by SQL. SQL Cursor names must still be unique within a single routine or class method.

Statement-Level AFTER Triggers No Longer Recurse

In this version, the following changes have been made to how statement-level triggers AFTER behave:

1. In previous versions, an AFTER statement level trigger was only executed if SQLCODE for the statement was 0. Now the trigger will be executed if SQLCODE for the statement is >=0. This means that if the statement does not find any rows to INSERT, UPDATE, or DELETE (SQLCODE = 100), the trigger is now executed.

   This behavior is required for proper behavior of language=TSQL triggers.

2. Caché now prevents AFTER statement-level triggers from executing recursively.There are a couple of ways triggers can recursively call themselves. Direct recursion is when a table T1 has a trigger which performs the same operation on table T1. Indirect recursion can also occur if table T1 has a trigger that performs an insert into table T2 and table T2 has a trigger that performs an insert into table T1. Another example of indirect recursion is when table T1 has a trigger that calls a routine/procedure and that routine/procedure performs an insert into T1. With this change, when recursion occurs the trigger will not be executed if it detects it has been called previously in the execution stack. No error will be returned when either type of recursion is encountered, the trigger will simply not execute a second time.

   There are a couple of ways triggers can recursively call themselves.

   - Direct recursion is when a table T1 has a trigger which performs the same operation on table T1.
   - Indirect recursion can occur if table T1 has a trigger that performs an insert into table T2 and table T2 has a trigger that performs an insert into table T1. Another example of indirect recursion is when table T1 has a trigger that calls a routine/procedure and that routine/procedure performs an insert into T1.

   With this change, when recursion occurs the trigger will not be executed if it detects it has been called previously in the execution stack. No error will be returned when either type of recursion is encountered, the trigger will simply not execute a second time.

Important: Recursion protection is only performed for AFTER statement-level triggers. If recursion is attempted in a BEFORE statement-level trigger, a runtime <FRAMESTACK> error may occur if the trigger code does not handle the recursion itself.

Note: A statement level trigger with language Caché that attempts to define direct recursion via embedded SQL will fail to compile with a <FRAMESTACK> error. If your application requires recursion for a Caché trigger, indirect recursion must be used. This can be as simple as using Dynamic SQL or Deferred SQL instead of embedded SQL in the trigger code.

%sqlcontext No Longer Automatically Created

In Release 2009.1, an embedded SQL CALL statement began to NEW the *%sqlcontext* variable during its execution and remove it upon return. Beginning with this release, this no longer happens except when the statement contains a USING clause.

8.2.2.8 iKnow Changes

Proximity Value Format Change

Beginning with this version, the signature of the proximity data value has changed. Previously, it was defined as a decimal value, but this caused problems in conversion when using certain C++ libraries. Now, the decimal proximity value is represented as an integer with a value between 0 and 256. This allows for an appropriate range without suffering loss of precision in using the library.

Access To iKnow UI Classes

Beginning with this release, access to the user interface pages of iKnow (%iKnow.UI.*) as well as the class, %iKnow.Queries.AbstractWSAPI, and any generated subclasses will need **%Developer**.

8.2.2.9 XML Changes

%XML.TextReader Use Of Temporary Storage

The development of %XML.TextReader predated the appearance of process-private globals; it used a hard-coded reference to ^CacheTemp for its temporary storage. beginning with this release %XML.TextReadernow uses the process-private global, ^||CacheTemp, and adds an option to change the global name via an additional parameter on the **ParseXXX()** class methods.

In the rare case where an application has subclassed %XML.TextReader and added its own code, the application will have to fix up the hard-coded global references.

8.2.2.10 Web Services And SOAP Changes

New Keep-alive Behavior

In this release, by default, if you reuse a Caché web client instance to send multiple request messages, Caché sends all the messages in a single HTTP transmission (using a HTTP 1.1 keep-alive connection). Specifically, Caché keeps the TCP/IP socket open so that Caché does not need to close and reopen it. For information on disabling this keep-alive behavior, see "Disabling Keep-alive for a Web Client" in the chapter "Fine-tuning a Caché Web Client" in *Creating Web Services and Web Clients in Caché*'.

Change to SOAP Body Available to ProcessBody()

If you have customized a web service by implementing **ProcessBody()**, you might need to modify that method. In previous releases, **ProcessBody()** provided access to the SOAP body as is, without all the XML namespace prefix definitions. That is, when you called the **Read()** method of the requestBody argument of **ProcessBody()**, Caché returned the body without all namespace prefix definitions. In this release, the returned body includes any prefix definitions which may have been done in the Body or Envelope elements.

XMLNAME Parameter Included in Generated Methods

In some cases, the SOAP wizard now includes the *XMLNAME* parameter in the web methods in the web service and web client that it generates. This occurs if the WSDL defines the return type of a web method in terms of an element whose name does not follow the form *MethodName*`Result`. For example, in many WSDLs, the return type of a web method is defined like this:

```
<s:element name="TestResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="TestResult" type="s:string"/>
        </s:sequence>
    </s:complexType>
</s:element>
```

In some cases, the return type is defined like this instead:

```
<s:element name="TestResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="MyOutput" type="s:string"/>
        </s:sequence>
    </s:complexType>
</s:element>
```

In such cases, the SOAP wizard now includes the *XMLNAME* parameter in the generated web methods as follows:

```
Method Test(a As %String) As %String(XMLNAME="MyOutput") [ ... ]
{
...
}
```

Support REQUIRED Keyword For Web Services

The SOAP Wizard is now modified to now add REQUIRED=1 for required arguments. This has no affect on the behavior of the service or client; it affects the WSDL which is created from a service.

The REQUIRED parameter is allowed for web method arguments to influence the generated WSDL. REQUIRED=0 (the default) results in minOccurs=0 in the WSDL. This change also adds support for the REQUIRED parameter in the return type. The default for the return type was REQUIRED=1, however, now REQUIRED=0 is valid and affects the WSDL.

The SOAP Wizard is also modified to now add REQUIRED=1 for required arguments.

XML Prefix Definitions Now In Request Stream Passed To ProcessBody

Previously, when calling ProcessBody call of web service, the body was copied as is. Beginning with this release, the body now includes any prefix definitions which may have been done in the Body or Envelope elements. Any code that depends on the exact form of the request including missing prefix definitions (for example, to add these definitions) must be modified to accommodate the new information.

Correct Processing Of Signed Requests When Using MTOM

When signing and validating signature for MTOM SOAP messages, the attached streams must be substituted for the MTOM reference before before computing the digest of the message body. Previously, the body including the references was incorrectly signed without substituting for the MTOM Include elements.

Note: Caché will no longer interoperate correctly with web services that incorrectly do the signing before the substitution.

WebStress And UnitTest Classes Restricted To Developers

In this release, use of the %WebStress.Portal and %UnitTest.Portal classes requires the user to have the **%Developer** resource. Customers may also have to manually open up access to these classes by setting the ^SYS("Security","CSP") global as documented in Application Access To %CSP Pages Now Controlled to allow access to these classes via CSP.

8.2.2.11 MultiValue Changes

Log File Switching

When the mv.log file grows larger than 5MB (or the size limit of cconsole.log) it will be renamed and logging will start with a new file. This action is done by the %SYS.Task.PurgeErrorsAndLogs task.

Emulations @(–nn) Correction

For a number of MultiValue emulations, the @(-nn) implementations were wrong. These are all derived from the original Universe emulation and, since Universe doesn't differentiate between @(-nn) codes in its emulations, neither should Caché. For example, Caché doesn't provide a PRIME emulation, it provides a "PRIME as emulated on UNIVERSE" emulation.

The emulations affected are PICK, PRIME, PIOPEN and IN2. Applications will need source code changes if they use any of the following @(-nn) codes:

- @(-11) to @(-12)
- @(-17) to @(-53)
- @(-56) to @(-64)

Changes To Handling Accounts In Root Directory

Beginning with this release:

- CREATE.ACCOUNT will refuse to create accounts in "/" or "<DriveLetter>:\".
- DELETE.ACCOUNT will now prompt for confirmation before clearing and deleting a directory unless the "N" option is used.

Any automated procedure that deletes accounts unattended should be modified to include the "N" option in the command line. If this cannot be done then a lowercase "n" on attribute 5 of the DELETE.ACCOUNT verb will revert to the old behavior as a temporary measure.

Changes To TANDEM Command

The **TANDEM** command includes several new changes in this release.

1. Input handling

   In previous releases, if the tandem slave was sleeping at an **INPUT** statement, then the tandem master could not interrupt the **INPUT** statement. This could often render the use of **TANDEM** less useful.

   With this change, once a tandem slave allows a future session (see below), then all **INPUT**s will have a maximum of 60 second timeout allowing a tandem master to interrupt with a maximum of a 60 second wait. Although the tandem slave process is continually doing timed out inputs, the application will not see this and will behave as normal.

2. Future TANDEM Sessions

   A slave terminal can allow a future TANDEM session by executing one of the following commands: **TANDEM ON**; **TANDEM SYSPROG**; or **TANDEM (N)**.

   All these 3 variants allow an optional numeric parameter to determine the timeout to apply during **INPUT** statement processing. The default is 60 seconds, as noted earlier. For example, to change the INPUT timeout to 10 seconds, you can do one of the following: **TANDEM ON 10**, **TANDEM SYSPROG 10** or **TANDEM 10 (N)**.

   By choosing a lower timeout value, it means that when a tandem master is initiated, if the slave is processing an INPUT statement then the master has to wait less time for an initial response. The downside of a lower timeout value is a marginal increase in CPU overhead during INPUT processing.

   Note that once the master has initiated a sequence with the slave, this timeout value no longer applies -it is purely for the initial connection.

3. "(A)" Option

   The (A) option allows you to use <CTRL-A>as a sequence identifier instead of the ESCAPE key. For example, normally when you start a master sequence you do this **SYSPROG:TANDEM 6** TANDEM to port 6 in VIEW ONLY mode. To exit TANDEM, enter ESC + "X"

   With the (A) option, the ESCAPE is replaced with <CTRL-A> so to TANDEM enter <CTRL-A> "X".

4. Input History Buffer

   A Tandem slave now keeps the last 20 INPUT statements in a circular buffer. The master can now see this history using the ESCAPE + "H" sequence (or <CTRL>A + "H"if the "(A)" option is in use). This will allow an operator to get some idea of what the slave process has recently been doing.

5. Terminal independence in MultiValue is obtained using calls to **CommandInit()** and **CommandNext()**. These allow an application to easily interpret characters from keyboards. For example if you press an arrow key on some keyboards,

instead of returning 3 characters (CHAR(27):"[A") it would return an identifier saying "CURSOR UP". These functions did notwork very well when using TANDEM and the program would instead see 3 characters. This has been fixed now. The MultiValue editor JED uses these commands, and so now you can JED edit an item using TANDEM.

6. Double Echoed Characters

   Due to an error, when a tandem master was in FEED mode (that is, sending keystrokes to the slave terminal), then these keystrokes would double-echo on the master terminal. This has now been fixed.

SP-ASSIGN Command Line Syntax Changes

This release makes slight changes to the syntax of the SP-ASSIGN command make it more compatible and remove anomolies:

- The method of specifying the channel and associated form queue name can no longer have spaces. For example,

  ```
  SP-ASSIGN 3 = STANDARD
  ```

  must now be represented as

  ```
  SP-ASSIGN 3=STANDARD
  ```

- Any numeric on its own in the statement will define the number of copies. For example,

  ```
  SP-ASSIGN 3 = STANDARD
  ```

  is now interpreted as assigning channel 0 (as it is not defined) to STANDARD and requesting 3 copies of the output.

- A standard form queue name format is supported across all emulations.

  ```
  SP-ASSIGN 5=F3HS
  ```

  now means for channel 5, assign form queue 3 with options H (Hold) and S (Skip).

- The options on the command line can now be passed in parenthesis, for example,

  ```
  SP-ASSIGN 1=CANON (HS)
  ```

  is the same as

  ```
  SP-ASSIGN 1=CANON HS
  ```

Rebuilding Indices From Prior Releases

Multivalue users who upgrade their applications from a release prior to 2012.1 to 2012.1 or later may experience a slowdown in CMQL queries that had previously made use of an index. Previously, CMQL assumed the collation for left-justified strings was SPACE. It is now SQLSTRING(150). This will not match an existing index that was created with the previous default of SPACE collation.

To correct this issue, use studio to edit the class associated with the multivalue file and change the COLLATION parameter to SQLSTRING(150) on all properties that are used as index keys. Then rebuild the affected indexes.

Changes To Transaction Lock Handling

In previous releases, committing a transaction would release all locks held on the records involved. Beginning with this release, that behavior has changed. It now follows these rules:

1. Any operation that creates or frees a lock during a nest of transactions will have any release of the lock delayed until execution of the COMMIT or ROLLBACK that exits the outermost transaction in the nest. Furthermore, any lock that is promoted from SHARED lock type to EXCLUSIVE lock type during a nest of transactions will have any demotion of that lock delayed until the exit from the outermost transaction in the nest.

   This behavior will be followed without exception, even in preference to subsequent rules.

2. Numbered or named locks that are created and freed by the MVBasic LOCK and UNLOCK statements are not incremented. You can do any number of "LOCK 2" statements, but a single "UNLOCK 2" statement will free lock number 2.

   None of the file locks or the record locks are incremented.

3. If a FILE or RECORD is locked with either the SHARED or EXCLUSIVE lock type on entry to an outermost transaction, then on exit from the outermost transaction (with either a COMMIT or a ROLLBACK) that FILE or RECORD will still be locked with the same lock type.Locks promoted from SHARED to EXCLUSIVE inside the transaction nest will be demoted when the nest exits. Locks newly created inside the transaction nest will be freed when the nest exits.

   Nonetheless, the following exceptions apply to this rule:

   a. If a FILE was CLOSEd during the transaction, then all FILE and RECORD locks on the closed file are released on exit from the outermost transaction.

   b. Executing either "FILEUNLOCK fvar" or "RELEASE fvar" (but not "RELEASE fvar, recid") will free the file lock AND all record locks on the file referenced by fvar. If this occurs during a transaction, the locks will be released when the outermost transaction exits.

Note: Delaying unlocking until the outermost transaction is a feature of the Caché implementation. Most legacy MultiValue implementations only delay the release or demotion of a newly created or promoted lock to the exit of the nested transaction level where the creation or promotion took place.

MV Spooler Changes To Named Queue Form Numbers

The Caché MultiValue spooler attempts to serve emulations where the spooler form queue is usually numeric as well as platforms where the spooler form queue is a name. For example we support "F23" (interpreted as form queue number 22) and "CANONLAB" (interpreted as a form queue name CANONLAB and internally represented by a form queue number assigned automatically and starting at 2 — STANDARD is always form queue 1).

The result of this practice is that there are cases when this automatically generated form queue number clashes with form queues that have explicit numbers.

Starting with this release, all named form queues created will default to a form queue number from 1000 upwards. That is,

- `SYSPROG: SP-NEWTAB (S)`

  creates a new form queue table.

- `SYSPROG: SP-CREATE CANONLAB LPTR /dev/lp3`

  will create a named form queue whose number is allocated starting at 1000 (in this case it will be 1000) .

- `SYSPROG: SP-CREATE F20 LPTR /dev/lp20`

  will create a numbered form queue whose number is 20 but is also allocated a form queue name of F20.

If you wish to change the starting point from the default 1000, then you can add this as an option to the SP-NEWTAB command line with the (Qnnnn) option. For example, to start allocation of named form queues at number 400, do this

```
SYSPROG: SP-NEWTAB (Q400)
```

Note: Any MultiValue application that creates named form queues and expects these form queues to have exact form queue numbers will fail; the default form queue number for new named form queues is now 1000 instead of 2. To resolve this, the developer must either remove that association from the application or use "SP-NEWTAB (S1)" to give results compatible with prior to this change.

Terminating A Process Now Runs Normal Wrapup Tasks

In this version, terminating a MultiValue process from the Management Portal now forces the process to perform the normal MultiValue wrapup tasks before halting.

Studio Will Not Overwrite Catalog Pointers In VOC That Point To A Different File

The MultiValue compile and catalog operation in Studio will no longer overwrite the catalog pointer if the existing catalog pointer refers to a program from a different file.

Correct MVR collation Generation For Long Embedded Numbers

In prior versions, MultiValue right-justified collation (MVR) generated incorrect keys for mixed alphanumeric values containing more than eight digits in a row and where the digit string followed a non-numeric character. This release correctly generates the keys.

Important: All indexes based on right-justified fields with collation MVR that may contain mixed alpha numeric fields with an embedded sequence of more than eight digits should be rebuilt. This does not affect fields that are purely numeric or numeric with a non numeric suffix.

DATA Statement Changes

Prior to this release, the input stack would only contain the line from the last **DATA** statement. Multiple **DATA** statements will now stack multiple lines in PARAGRAPHS

Directory Copy Uses Binary Mode

The MuliValue **COPY** command will now copy items in binary mode when both the source and destination are directories. Previously the items were processed to convert between AMs and newlines. The current behavior now copies directories unmodified to the destination.

8.2.2.12 BASIC And MVBASIC Changes

Only "–>" Allowed For Object Component Access

The MV BASIC compiler would sometimes accept the dot, ".", character in places where an right-arrow token, "–>", should have been used for access to the property or method of an object instance. This was true even though the MVBASIC language definition permits dot, ".", to be used only as an identifier character.

Beginning with this release, the MVBASIC compiler always interprets the "." as part of a name. Those developers who incorrectly copied COS or Cache BASIC examples of object references using "." into their MVBASIC programs will have to recode those examples to use "–>" instead.

Correct Processing Of "@XXX–>Component"

In prior versions, macro substitution was improperly done on names following a system variable and "–>". In this release, the component name is properly associated with the system variable. This means that in the sequence

```
EQU UserId Lit "X->UserId"
X = "MV.TestClass"->%New()
X->UserId = 2
@ME->UserId="99"
```

the last two lines would previously have been interpreted as

```
X->UserId = 2
@ME->X->UserId="99"
```

but now are processed as

```
X->UserId = 2
@ME->UserId="99"
```

Change Handling Of Substitution With "–>"

Previously, names which were part of a "A–>B–>C..." sequence were not expanded if they were defined using EQU. This was only true if no whitespace separated "–>" from a name (before or after). Now only names which follow "–>" are not expanded; the sequence, "–>", coming after a name no longer inhibits EQU definition expansion. Whitespace between "–>" and a following name does not affect the inhibition of EQU definition expansion.

This means that the lines

```
EQU UserId Lit "X->UserId"
CRT UserId
CRT UserId->UserId
CRT UserId->UserId->UserId
CRT UserId -> UserId -> UserId
X->UserId = 2
```

used to result in

```
CRT X->UserId
CRT UserId->UserId
CRT UserId->UserId->UserId
CRT X->UserId -> X->UserId -> X->UserId
X->UserId = 2
```

but starting with this version they become

```
CRT X->UserId
CRT X->UserId->UserId
CRT X->UserId->UserId->UserId
CRT X->UserId -> UserId -> UserId
X->UserId = 2
```

Identifiers Can Start With $

Previously, MVBASIC identifiers starting with a dollar sign, "$" could only use letters as the remaining characters of the identifier. This restriction has now been lifted and any legal identifier character can be used after the dollar sign that starts such an identifier.

However, InterSystems reserves MVBASIC identifiers of the form $SYSTEM.XxX, where the letters "SYSTEM" in the identifier prefix may appear in any combination of upper and lower case, but the identifier characters following the "$System." prefix are case-sensitive. An MVBASIC identifier of the form "$System.XxX" is treated as the name of the system package, and the *$System.XxX identifier* may be followed by a right-arrow token, '->', then a method name and then a method argument list in order to call the appropriate method in the system package.

Note: Package names using the $System.Xxx identifier syntax cannot have a blank before or after the dot characters that are part of the identifier. Previously,it was possible to put white space following the package name prefix. Programs that previously contained such white space must now be modified to remove the white space that follows any dot characters used in package names.

Change Triggers On WRITEV

Previously, when a WRITEV statement was executed on a file with WRITE, INSERT or UPDATE triggers, the record passed to the trigger routine consisted of just the attribute value to be modified rather than the entire record to be modified. This has been fixed and now these triggers get the entire record when a WRITEV modifies an attribute in that record.

Trigger routines executed because of a WRITEV statement that depended on being the passed record containing only the attribute to be modified must be rewritten. The passed record contains the entire record that will be modified and the attribute number so it can extract and/or modify the particular attribute value that will be modified.

Important: Trigger routines that do additional I/O on the file containing the trigger must now be prepared for recursive calls on the trigger routines and must protect against infinite recursion.

Change Triggers On READ/CLEAR/DELETE/OPEN

Previously in MVBASIC, when a READV statement was executed on a file with a POSTREAD trigger just the attribute being read was passed to the trigger routine rather than the entire record containing that attribute. This has been fixed and now POSTREAD trigger routines always get the entire contents of the file record being read.

Also previously, any active trigger routine on a file disabled calling any additional triggers on that file while that trigger routine remained active. This prevented trigger routines from being called recursively. Now none of the triggers will disable additional trigger calls during the execution of trigger routine.

Note: The trigger routine is not passed the attribute number as an argument so the attribute number must be passed by other means if the trigger routine needs to extract and/or modify the particular attribute value that will be read by the READV statement.

Important: Trigger routines that do additional I/O on the file containing the trigger must now be prepared for recursive calls on the trigger routines and must protect against infinite recursion.

SETTING Clause Processing Changes

Before this release, an MVBASIC I/O statement which included a variable in a SETTING clause was setting that variable to the I/O error status after executing any THEN, ELSE or ON ERROR clauses.

This had two bad effects. First, the variable in the SETTING clause could not be tested in the THEN, ELSE or ON ERROR clauses. Second, any I/O operation executed by the THEN, ELSE or ON ERROR clauses of the current I/O statement would override the status that would be stored into the variable specified by the SETTING clause of the current I/O statement.

The MVBASIC compiler now generates code to store the I/O status into the variable specified by a SETTING clause before executing any THEN, ELSE or ON ERROR clauses.

I/O Statements Now Clear $MVSYSRETCODE On Success

Previously, most I/O statements did not zero @SYSTEM.RETURN.CODE when starting a new I/O operation. This meant that any I/O statement with a SETTING clause that executed without any I/O error would leave behind the previous error code that was still left over in the system variable @SYSTEM.RETURN.CODE. Now, @SYSTEM.RETURN.CODE is reset to zero just before executing any I/O statement.

More Emulations Have HEADING Statement

In this version, for the emulations jBASE, Reality, Unidata, Udpick, Information, and Piopen, when a second or subsequent HEADING statement for terminal output is executed, it takes effect immediately.

For the other emulations, the second or subsequent HEADING statement will not have any immediate effects until the terminal executes a page break; this is when the HEADING statement is used.

Duplicate Names Not Allowed In DIM Statement

Previously, the MV BASIC compiler allowed a name to be defined more than once in the same DIM statement. For example,

```
DIM X(), X();
```

was permitted. Under some circumstances this results in a corrupted symbol table that could crash the process doing the compilation.

Now such statement will generate a compile-time error message and the redundant symbol definition must be removed from the DIM statement. The compiler does allow a name to be redefined multiple times in separate DIM statements providing the name as not been defined as a COMMON name.

8.2.2.13 xDBC Changes

Automatic Conversion Of Long Strings To Streams Removed

It previous versions, applications could ask that literal strings longer than 16K to be automatically converted to character streams. While this supported INSERTs and UPDATEs, queries never supported streams in this way for literal values.

Now that Caché supports long strings in ODBC, the previous functionality is no longer needed. Applications that need to support huge literal string values in SQL statements should make sure that Caché has long strings turned on.

Change To Null Parameter Handling

Beginning in this release, Caché conforms to the Microsoft documentation regarding null parameter values, namely, when you send a null parameter value to the server, you must specify DBNull, not null. The null value in the system is an empty object that has no value. DBNull is used to represent null values.

Prior to this, Caché had been treating null and DBNull the same,, but the mapping for parameter values are as follows in terms of the $list values sent to Caché. The only change is for a null to specify a default value.

.NET	$LIST	SQL
null	01	default value
DBNull	02 01	null value
“ ”	03 01 00	empty string

This changes makes the behavior of Caché consistent with the ODBC native provider, and all other ADO providers in terms of null Parameters representing default values.

Accomodate New SQL Server Mappings

This version changes the JBDC mappings for `varbinary` and `nvarchar` (BLOB and CLOB, respectively) to use streams in accord with changes in Microsoft SQL Server version 2008.

SQLBindParam API Removed From ODBC

SQLBindParam is an older 2.x API that Caché added with Version 3.5 release. However, it conflicts with SQLBindParameter which was present then as well and its presence ended up causing problems with iODBC driver manager. It has been removed from this versionof Caché to allow iODBC to function correctly by calling SQLBindParameter with the proper arguments.

Support Binary In %SQL.Gatway.ODCResultSet

In previous releases, binary values returned by queries were converted to the corresponding ASCII values (visible in linked stored procedures). Beginning with this release, binary values are returned as is.

8.2.2.14 DeepSee Changes

Changes To Filter Handling In Pivots

In prior versions,

- Multiple valued filters – If you added 2 filters from the same hierarchy and level (for example, DateOfSale.Actual.YearSold.Members), both filters would display on the filter bar, but only one value was used in the query.
- Single valued filter – Multiple single valued filter from the same hierarchy and level (DateOfSale.Actual.YearSold.2009 and DateOfSale.Actual.YearSold.2010) were logically ORd together.

Now, in both cases, they are logically ANDed together. This is an improvement in several ways:

1. It makes all filters consistent – all filters in the filter bar are now ANDed together with no exceptions.

2. For list-type levels, it lets you filter on combinations (such as allergy on mites AND dessert toppings).
3. It removes a lot of "Why do I get multiple counts when I add more filters" questions.

DeepSee Key Values Now Checked For Illegal Characters

KEY values for DeepSee members may not contain the characters "|", "&", "~", or ":". In previous releaqses this was not explicitly checked. Beginning with this release, attempting to build a cube with illegal KEY values will result in a validation error.

Improved Expression Checking

An expression or scalar function on a SET, such as,

```
ISNULL(Country.Members,0)
```

does not make sense. In previous version, executing the expression would give a nonsense answer. Now, it will return an error.

Add %DeepSee.ReportBuilder Resource For Access Control

In this release, a new resource has been added which grants access to the DeepSee Visual Reporting Report Builder. Users and applications that wish to access this functionality must now have **`%DeepSee_ReportBuilder`** and a valid license for the feature, or they will be denied access.

8.2.2.15 CSP Changes

Define Logout Handling

Beginning with this release, the recommended way to logout of a CSP session is to link to the application home page passing a URL that contains the string, "CacheLogout=end". This will end the current session – release any license acquired, delete existing session data, remove the security context of the session – before it attempts to run the home page.

If this CSP application requires authentication, there will be no session and no authenticated user. In this case, Caché will not run the home page logic but will display the login page instead. When the user submits a valid login, this will start this new session and then display the home page.

SECURITYRESOURCE Parameter Added To %CSP.Page

In this release, the class %CSP.Page now has an additional class parameter, *SECURITYRESOURCE* which controls access to the page. The value of the parameter is a comma-delimited list of system resources and associated permissions. A user must hold the specified permissions on the specified resources in order to view this page or invoke any of its server-side methods from the client.

The format of each item in the list should be as follows:

```
Resource[:Permission]
```

Resource is a registered system resource. Permission is optional, and defaults to `USE` if not supplied. If it is supplied, it must be one of `USE`, `READ`, or `WRITE`.

You can also specify OR grouping using the "|" character, so

```
ResA,ResB|ResC,ResC|ResD
```

means you must have resource ResA, and one of ResB or ResC, and one of ResC or ResD. So if you have ResA and ResC, you will have access. If you have ResA1 and ResD, you will not since the ResB or ResC condition is not met.

Note: The "|" binds more tightly than ",".

Application Access To %CSP Pages Now Controlled

Beginning in this release, application access to arbitrary %CSP pages (and their subclasses) can now be better controlled. By default, the new rules allow a user application to access:

- All non-% classes
- Pages of the /csp/sys/ application and all its subapplications.
- Pages of the isc/studio/templates/ and /isc/studio/usertemplates/ applications.

and also explicitly to the following classes:

- %CSP — Broker, StreamServer, Login, PasswordChange, PageLookup
- %ZEN — SVGComponent.svgPage, Dialog.*
- %Z* and %z* — all are allowed except for the %ZEN.* pattern other than the specific exceptions allowed for %ZEN already mentioned

This checking is performed in addition to any "Permitted Classes" checking defined for this CSP application. So a class reference must pass both sets of tests in order to be allowed.

An administrator can permit access to additional classes by configuring the global, ^SYS("Security", "CSP", <category>) *in the %SYS namespace* by using the `AllowClass`, `AllowPrefix`, and `AllowPercent` keywords. The following sections describe these keywords.

Important: Checking is done by applying the default rules first, then the categories in the order listed. This permits making an entire package accessible, except for one class in the package whose access is restricted.

Category: AllowClass

If your application relies on invoking a particular class, the `AllowClass` keyword creates an exception for that class and makes it available.

Important: If your application relies on invoking any class other than those listed above, it could potentially be unsafe to use. InterSystems recommends that you determine if calling this class is required, and have performed a risk assessment for your deployment, so that you understand the implications of making the class available.

To create an exception for a particular class, set the ^SYS global subscript to a value of 1 as follows:

```
Set ^SYS("Security", "CSP", "AllowClass", "<web-app-name>", "<package.class>") = 1
```

where entire body of the command – except <web-app-name> and <package.class> – is used in the exact form as it appears here. For the variable content of the command:

- <web-app-name> is the web application's name as it appears on the Management Portal's Web Applications page in the Name column.
- <package.class> is the name of the class being allowed.

Setting the data value of the global to 0 specifies explicitly that Caché does not permit access to the class.

For example, to allow a specific classname (in this case, %User.Page) to be called from a /csp/webapps/ application, the command is:

```
Set ^SYS("Security", "CSP", "AllowClass", "/csp/webapps/", "%User.Page") = 1
```

To allow, for example, the %User.Page class in all CSP applications, the command is:

```
Set ^SYS("Security", "CSP", "AllowClass", 0, "%User.Page") = 1
```

Category: AllowPrefix

If your application relies on invoking one or more classes or packages that begin with the same set of characters, the `AllowPrefix` keyword creates an exception for those packages or classes and makes them available.

Important: If your application relies on invoking any class other than those listed above, it could potentially be unsafe to use. InterSystems recommends that you determine if calling this class is required, and have performed a risk assessment for your deployment, so that you understand the implications of making the class available.

To create an exception using the `AllowPrefix` keyword, set the ^SYS global subscript as follows:

```
Set ^SYS("Security", "CSP", "AllowPrefix", "<web-app-name>", "<prefix>") = 1
```

where the entire body of the command – except <web-app-name> and <prefix> – is used in the exact form as it appears here. For the variable content of the command:

- <web-app-name> is the web application's name as it appears on the Management Portal's Web Applications page in the Name column.
- <prefix> is the beginning of the package or class names being allowed.

Setting the data value of the global to 0 specifies explicitly that Caché does not permit access to the relevant packages or classes.

For example, to allow the entire %MyApp package to be called from a /csp/webapps/ application, the command is:

```
Set ^SYS("Security", "CSP", "AllowPrefix", "/csp/webapps/", "%MyApp.") = 1
```

In this case, adding the dot prevents access to any packages whose names begin with and are longer than "%MyApp".

To allow, for example, all packages that begin with "%My", the command is:

```
Set ^SYS("Security", "CSP", "AllowPrefix", 0, "%My") = 1
```

As a further example, you can allow %MyPkg.*, but disallow %MyPkg.Class1 in the /csp/samples/ application by setting:

```
Set ^SYS("Security", "CSP", "AllowClass", "/csp/webapp/", "%MyPkg.Class1") = 0
Set ^SYS("Security", "CSP", "AllowPrefix", "/csp/webapp/", "%MyPkg.") = 1
```

Category: AllowPercent

If your application relies on invoking the packages that begin with the % character generally, the AllowPercent keyword creates an exception for those packages and makes them available.

Important: If your application relies on invoking any class other than those listed above, it could potentially be unsafe to use. InterSystems recommends that you determine if calling this class is required, and have performed a risk assessment for your deployment, so that you understand the implications of making the class available.

To disable %-page checking totally in all CSP applications:

```
Set ^SYS("Security", "CSP", "AllowPercent") = 1
```

Note: Setting the data value of the global to 0 specifies explicitly that Caché does not permit access to the percent packages.

Special Case: DeepSee

For a web application to use DeepSee, it needs access to all the classes in the %DeepSee package. By default, only the "/csp/samples" web app is set to allow Deep See access. If you want to use Deep See in any other web app, you must explicitly allow it.

For example, to enable the /csp/webapp web application to use DeepSee, the syntax is:

```
Do EnableDeepSee^%SYS.cspServer("/csp/webapp/")
```

which is the same as

```
Set ^SYS("Security", "CSP", "AllowPrefix", "/csp/webapp/",  "%DeepSee.") = 1
Set ^SYS("Security", "CSP", "AllowClass", "/csp/webapp/",  "%CSP.UI.Portal.About") = 1
```

To allow all web applications to use DeepSee:

```
Do EnableDeepSee^%SYS.cspServer(0)
```

which is equivalent to:

```
Set ^SYS("Security", "CSP", "AllowPrefix", "",  "%DeepSee.") = 1
```

If you have enabled Deep See access for all web apps and you want to disable it for a specific web app, you should set the global to a 0 value as shown in this syntax:

```
Set ^SYS("Security", "CSP", "AllowPrefix", <web-app-name>, "%DeepSee.") = 0
```

Note: This is only for DeepSee versions released with Caché version 2011.1 or later – that is, only for what was previously known as "DeepSee II". If you are using a version of DeepSee released prior to Caché 2011.1 even on a version of Caché that is more recent, there is no issue.

Special Case: iKnow

For a web application to use the sample UI pages for consulting or managing iKnow data, access also needs to be granted explicitly as follows:

```
Do EnableIKnow^%SYS.cspServer("<web-app-name>")
```

To disallow an application from using iKnow, the syntax is:

```
Set ^SYS("Security", "CSP", "AllowPrefix", <web-app-name>, "%iKnow.") = 0
```

Granting or disallowing access for all web applications works analogously the way it is done for DeepSee.

Translation For JS and JSML Changed

The **EscapeURL(url)** function of %CSP.Page first encodes a URL using the current device charset; then it escapes this using the standard encoding for URLs.

If the current device is set to the `JS` or `JSML` translate table, this function converts "/" characters into '%5Cx2F' with the result that the URL no longer functions correctly.

8.2.2.16 Zen Changes

Improve Parsing Of #(...)# Expressions

Zen allows for expressions in values of the form `#(<object>.<propname>)#`. In this release, the Zen compiler will only allow the names of properties to be used in place of `<propname>`.

ZenMethod Keyword Now Enforced

In the very early versions of ZEN, there was no `ZenMethod` keyword; Zen overloaded the `WebMethod` keyword for this purpose. Beginning with this release, support for the `WebMethod` keyword is now removed. Applications that use `WebMethod` must be changed to use `ZenMethod` instead.

ZenMethods Can Only Be Called From The Current Page

If a Zen class defines a ZenMethod, this method can now only be called from the client/browser.

You can define a security resource (using a class parameter) for a given Zen page. This prevents any methods on this page from being invoked unless the user has **Use** permission on the resource. Hence, any Zen methods defined on that page can only be called from methods on that page.

Note: Zen methods of Zen components can be called from any page. Thisis due to the dynamic nature of Zen – pages can create instances of any compnent as needed at runtime. This means that care must be taken when creating new Zen components containing server-side methods.

Fix Parsing Of Escaped Data In Zen jsonProvider

The Zen jsonProvider component was correctly escaping data on output, but was not unescaping the same data on input. This change updates the parser to correctly handle the following escape sequences defined in RFC 4627:

```
\/   \r   \n   \f   \b   \t
```

Furthermore, the escape sequences for Unicode values are also converted into the raw Unicode characters on input. The jsonProvider parser will now automatically convert escape sequences of the form \uXXXX to the character representation of the specified Unicode code point.

WARNING! This may cause problems for 8-bit Caché systems where incoming data containing code points beyond \u00FF, as the conversion process will now produce wide characters.

Charts

This version of Zen significantly updates the chart functionality in the following areas:

- Use of real coordinate space – new render logic computes the scaling factor in the x and y dimensions and saves these in local properties so that other rendering methods can see them.
- Label positioning – You can now place x-axis labels on the top or bottom.

  You can now place y-axis labels on the right OR left.
- Time-based x-axis – It is possible to use a time line as the x axis for certain chart types (namely line, bar, and combo). This draws a set of calendar markings for the x axis between a start time and an end time.
- Subtitles – Chart may now have a subtitle.
- New zoom buttons and styles

CAUTION: Applications that subclass charts will need to be changed to use the new superclasses. Applications that do not use the default setttings may need to modify their styles for existing charts to achieve the desired visual effects.

8.2.2.17 Zen Reports Changes

Zen Report Must Use %SQL.Statement For Result Sets

As of this release, %ZEN.Report.ResultSet has been deleted. Zen report data generators and collectors should use %SQL.Statement instead for proper results. To define the method as an SQL stored procedure, use the method keyword, “sqlproc”.

Change To Multiple Line-Feed Handling

In prior releases, a sequence of multiple line-feeds sent to the output would ignore any after the first. To have multiple line-feeds appear in the PDF output, one must do the following:

- On the table set preserveLineFeed=“true”, or
- If the items are not in a table, put them in a block with linefeedTreatment=“preserve” as an attribute setting.

On the item set the following attribute as follows: breakOnLineFeed="true"

Automatically Provide Unique Sheet-Names For Excel

Zen Reports no longer takes the sheet name from the group name. Instead, it now automatically provides unique sheet-names for Excel.

When generating multisheet output for Excel, each sheet must have a unique name. This may be done by specifying an expression in "excelSheetName". By default (that is, when excelSheetName is not specified), each sheet in a multi-sheet report be given a unique sheet name using the same default names Excel uses for sheets: "Sheet1", "Sheet2", etc.

If any application is depending on our taking the sheet name from the group name they can use excelSheetName to set the sheet name equal to the group name.

Changes To ongetXXX Method Handling

Beginning with this release, the **ongetXXX** methods now pass a chart object as the final argument of the method. This allows the callback methods to access properties of the chart such as id.

A ZEN Report chart on a chart callback method that worked in prior releases will now get a parameter error. There are two ways to work around this situation:

You can work-around this parameter error two ways:

- You specify passChartObject="false" on the chart. This tells the code not to pass the chart object.
- You can modify the callback method to accept a final parameter which will be the chart object when ZEN Reports calls back the call back.

%ZEN.Report.ResultSet Removed

In this release, %ZEN.Report.ResultSet has been removed. Applications using %ZEN.Report.ResultSet must define their own ResultSet class to emulate its functionality, for example,

```
Class MyResultSet Extends %Library.ResultSet {

    Method %Get(name As %String) As %String [ ProcedureBlock = 1 ] {
        if (name '= "") && ($Data(i%Data(name))) {
            set rReturnValue = $get(i%Data(name))
            }
    else  {
        s rReturnValue=""
        }
    quit rReturnValue
    }
}
```

Two-Dimenional Arrays In CallbackTables Are Now Zero-Based

Beginning with this release, callback tables will now be zero-based to make them consistent with callback charts. For example,

```
<table callBackMethod="NamesAndAddresses">
    <item fieldnum="1" suppressDuplicates="true">
        <caption value="Name"/>
     </item>
     <item fieldnum="2">
         <caption value="Address"/>
     </item>
 </table>
```

will be interpreted as

```
Method NamesAndAddresses(ByRef var As %String)
{
    // record of names
    // record 1
    Set var(0,0)="Santa Claus"
    Set var(0,1)="North Pole"

    // record 2
```

```
    Set var(1,0)="Zeus"
    Set var(1,1)="Olympus"

    // record 3
    Set var(2,0)="Robin Hood"
    Set var(2,1)="Sherwood Forest"

    // record 4
    Set var(3,0)="Robin Hood"
    Set var(3,1)="Nottingham"

    Quit
}
```

9

Caché 2012.2

This chapter provides the following information for Caché 2012.2:

- New and Enhanced Features for Caché 2012.2
- Caché 2012.2 Upgrade Checklist

9.1 New and Enhanced Features for Caché 2012.2

The following major, new features have been added to Caché for this release:

- Using Unstructured Data In Analytics
- DeepSee Visual Reporting

Furthermore, this version of Caché has been improved and enhanced in the following areas:

- Rapid Application Development
  - iKnow Enhancements
  - HTML5 Is The Default
  - Improved Arabic Rendering In Reports
  - Regular Expressions
  - Support For Node.js
  - Unlimited Local Arrays
  - .NET eXtreme
  - Java eXtreme Over TCP
  - .NET 4.0 Components For Object Provider And .NET Gateway
  - Long Strings Are The Default
- Performance and Scalability
  - ECP Rollback
  - Use AES Hardware Encryption When Available

- Reliability, Availability, Maintainability, Monitoring
  - Simpler Disaster Recovery Failover And Failback
  - Configurable UNIX® Installation
- Security
  - Public Key Infrastructure

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in "Caché 2012.2 Upgrade Checklist."

9.1.1 Major New Features

9.1.1.1 Using Unstructured Data In Analytics

DeepSee support for unstructured data

DeepSee now supports unstructured data. DeepSee cubes can use free-text as the source for dimensions and measures. Using the iKnow technology, this text is analyzed, and the results of this analysis (the "concepts" from the text) are available as members of the dimensions.

Cubes can contain a mixture of dimensions based on structured and unstructured data. Like all DeepSee cubes, these cubes can be queried via DeepSee Analyzer, dashboards, and programmatically via MDX queries.

Term Lists

DeepSee includes new Term List functionality. A Term List is a set of key/value pairs that can be used for lookups and translations. Term Lists can be used during cube building, and also in MDX via the %TERMLIST function. For defining Term Lists, there is a Term List Manager option on the DeepSee menu in the Management Portal.

Business Rules

Ensemble business rules can be used at cube build-time. A sourceExpression can use the new %Rule method to invoke a business rule. The business rule is passed an instance of the current source object, and the result of the business rule is returned.

9.1.1.2 DeepSee Visual Reporting

This release introduces a new capability that enables Caché, Ensemble, and HealthShare users to interactively define reports via a drag-and-drop browser interface. This new technology, DeepSee Visual Reporting, consists of three parts:

- Report Data Definition

  DeepSee provides a browser-based UI for creating report data definitions. This is analogous to creating a query that gathers data for the report and specifying how that data will be formatted when it is displayed. The user creating the definition needs an understanding of the underlying source data and its interrelationships.

- Report Format Definition

  Once the data definition is complete, you can display the data in any number of formats. You define these layouts also via a browser-based interface that provides the ability to preview the output using the actual data chosen.

- Report Execution

  Once the user completes that definition of the data and output formats, the report can be run at any time by the author or those with whom the report is shared.

Note: For 2012.2, the Report Data Definition and Report Format Definition components of DeepSee Visual Reporting are available for evaluation, but are not supported for production. These components will be fully supported in a future release.

Reports created with DeepSee Visual Reporting are supported in production in 2012.2.

9.1.2 Rapid Application Development

9.1.2.1 iKnow Enhancements

In this release, the iKnow API has been extended with the new Semantics package. This package contains two major new feature sets.

Semantic Dominance

Semantic dominance is a term used to describe how important a specific element (can be a word, a concept, a set of concepts or a sentence) is in the context of specific text and in relation to the other information in the text. The semantic dominance API allows applications to identify:

- the semantically most dominant elements in a single source, that is, the most dominant elements in a domain or a domain subset;
- how dominant elements are shared by different texts; the most typical sources in a domain-the most atypical sources in a domain.

The uniqueness and added value of the semantic dominance API resides mainly in two elements. First, of all the algorithm implemented works based on the context of a single source only and doesn't need a reference corpus to identify the base line importance of a text element. Second, the algorithm generates values that are comparable between different texts and between different text elements. For example the semantic dominance value of a single concept can be compared with the semantic dominance value of a pathway consisting of multiple concepts and relations.

The semantic dominance API can be used to easily find out what the important common content of a series of text is or to find out what texts are the most typical for a given set of texts. It allows also to quickly find out what the important new information is in a text.

Proximity

With the new capabilities in the proximity API, it is easy to find out which concepts appear in the context of a chosen reference concept. The functions in this API are used to find out what concepts are associated with the reference concept and how strong the association is. Based on the contents of the domain for which the semantic proximity is calculated the semantic proximity profile for the concept “bakery” will contain concepts such as “muffins”, “belgian waffles”, “cake” and “pie” each with a different value to express the strength of the association with bakery.

9.1.2.2 HTML5 Is The Default

Beginning in this release, the default doctype for all Zen pages is HTML 5 for browsers that support HTML 5 and CSS 3. Also included is a new generated css file that will be served only when Caché detects an HTML 5 supported browser and the output contains CSS 3 syntax. You may override this behavior by changing the **%OnDetermineCSSLevel()** callback to return 2 instead of 3 or by modifying a global node in the managers database. With this change, we also start using the native SVG renderer in IE 9 and above.

9.1.2.3 Improved Arabic Rendering In Reports

Starting with this version, Caché officially supports Arabic output in Zen Reports using Apache FOP.

9.1.2.4 Regular Expressions

With 2012.2, InterSystems introduces a feature-rich alternative to ObjectScript pattern matching, Regular Expressions. Using Regular Expressions not only detects if a complex pattern is present, but also provides location and results of successful matches. This allows more complex manipulation of strings in less code. For details on usage, please refer to the class documentation for %Regex.Matcher.

9.1.2.5 Support For Node.js

This release adds support for Node.js. Node.js a platform built on the Google Chrome JavaScript runtime. It uses an event-driven, non-blocking input/output model for building web and data-intensive applications. The Node.js integration with Caché is native and has direct access to Caché globals. More information on Node.js is available at nodejs.org.

9.1.2.6 Unlimited Local Arrays

As part of version 2012.2, InterSystems offers a simple way for application developers to dynamically change the memory allocation for processes during runtime, or define large memory allocations during process startup time. Available system memory can be by fully utilized to perform complex and high performance operations by keeping arrays in memory.

While this feature is most useful on 64–bit operating systems, process private memory can also be changed in 32–bit environments. Customers need to verify that use of this feature does not trigger unexpected performance profiles in the operating system, for example, excessive swapping.

9.1.2.7 .NET eXtreme

This release includes support to .NET for the Globals API. This enables direct global access and manipulation of globals from .NET. This API enables .NET and Caché to run intraprocess on the same computer providing high speed data access.

9.1.2.8 Java eXTreme Over TCP

This version of Caché allows XEP to run over TCP. XEP is a lightweight Java object persistence technology for Caché. Initially it executed intraprocess because it was built only on top of the Globals API for Java. While it still leverages the Globals API concepts in many respects, being able to use TCP means that it is now optionally intraprocess.

9.1.2.9 .NET 4.0 Components For Object Provider And .NET Gateway

This release provides .NET 4.0 compiled versions of the Object Provider for Caché and the .NET Gateway. Though previous releases were compatible with .NET 4.0, there is now an option to only have .NET 4.0 components in applications.

9.1.2.10 Long Strings Are The Default

Beginning with 2012.2, support for long strings is now enabled by default for new installations as well as upgrades.

9.1.3 Performance And Scalability

9.1.3.1 ECP Rollback

Beginning in this release, rollback operations issued by a process on an ECP Application Server will be addressed by the Database Server asynchronously. This approach allows continued parallel processing of other requests. Previously, a rollback operation was handled synchronously: the Database Server would serially handle the entire rollback before continuing with other requests from a given ECP Application Server. The new technique prevents a long rollback operation from disrupting other application work on a given ECP Application Server.

9.1.3.2 Use AES Hardware Encryption When Available

Many processor vendors now include support for encryption algorithms directly in hardware. Beginning with this release, Caché takes advantage of this feature (when present) in the Advanced Encryption Standard (AES) calculations. In doing so, it provides increased performance by using the instruction set of the processor.

The initial use of this option will take place on Intel 64-bit processors, beginning with the Intel® Xeon® Processor X5680 (Westmere). On such systems, Caché makes direct use of the hardware instructions to perform AES encryption. The systems for which this is true are: Microsoft x86-64, Red Hat Enterprise Linux for x86-64, SuSE Linux Enterprise Server for x86-64.

9.1.4 Reliability, Availability, Maintainability, Monitoring

9.1.4.1 Simpler Disaster Recovery Failover And Failback

In previous releases, Caché Database Mirroring has provided an Asynchronus Mirror configuration for Disaster Recovery purposes. In this release, InterSystems has made it easier to switch over to an Asynchronous DR Mirror node in the event of a disaster (or for testing purposes); and also made it easier to switch back to the main production failover mirror at a later date.

9.1.4.2 Configurable UNIX® Installation

In this release UNIX® installations can now be done in a manner similar to Windows installs that use command line options. New configuration parameters are available with the new **cinstall_silent** installation script to define the instance name, installation directory and packages to install.

9.1.5 Security

9.1.5.1 Public Key Infrastructure

Increased concerns about information security in many industries have correspondingly increased the need to encrypt and sign documents using the X.509 certification. However, generating, safe-guarding, and distributing X.509 keys and certificates is a complex and time-consuming process.

In this release, InterSystems users have the ability to automate the process that generates certificates by defining a server to act as a certificate authority. Clients cannot only request certificates from the certificate authority, but automatically receive them for their local use.

9.2 Caché 2012.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2012.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2012.1 and 2012.2.

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

9.2.1 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 2012.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

9.2.1.1 Version Interoperability

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

9.2.1.2 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 addition of pages to assist with database mirroring and the separation of roles.

9.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

Extended Memory

In prior releases, there was a limit on the maximum a Caché process could consume. By default, the limit was 16MB, but could be increased to 49MB.

Beginning with this release, the limit has been extended. It is now 2TB, subject to limitations imposed by the platform and operating system. For example, most 32–bit systems will be restricted to 2GB or less. Also, larger Caché partition sizes may need larger swap space allocations in the underlying platform.

Administrators should note the following changes:

- bbsiz: This parameter is now renamed to "Process Private Memory".
- Installation defaults: New installations will set the value of Process Private Memory to 256MB. For upgrades from previous releases, the value will not be changed.
- Management Portal: The Process Private Memory value is shown with the label, "Maximum Per-Process Memory".

There are two system variables, *$STORAGE* and *$ZSTORAGE*, available for developers to determine how much process private memory is still available and what the maximum value is. At process, start both system variables have the same value derived from the Process Private Memory setting, Application developers can increase the value of *$ZSTORAGE* at any time during process execution, for processes that are designed to require larger than the default process private memory allocation. In addition, the exhaustion of process private memory resulting in a <STORE> error, no longer results in process termination. Application developers have now the options on how to proceed in this case:

- Increase memory by increasing the value of $ZSTORAGE and continue running.
- Increase memory to capture error/stack information and HALT the process.
- HALT the process.

Note: InterSystems utilities to capture error information will automatically increase the value of *$ZSTORAGE* to allow the recording of the error and stack information, and then HALT the process.

Extended Memory — Platform Details

Windows and OpenVMS systems have no configuration controls for virtual memory. For UNIX® platforms, there are two configuration parameters of interest:

- Data space: This is controlled by the "ulimit –d" command.
- Virtual memory (or process) space: The "unlimit –v" command sets the value.

For specific versions of UNIX®, the changes are

- HPUX (32– and 64–bit); IBM AIX

  In prior versions, application data space was controlled via the "ulimit -d" command. Now, the command has no effect. The only control is via the Caché *$ZSTORAGE* value.

- Linux (Red Hat and SUSE)

  No change. Both data space and virtual memory space are controlled by the operating system via the "ulimit -v" command.

- Mac OS X

  No change. Neither data space nor virtual memory space are controlled by commands.

- Solaris

  In previous versions, the data space was controlled by the "ulimit -d" command. Starting with this version, the application memory is taken from the virtual memory space and is controlled via the "ulimit -v" command.

Collection Indexes May Need Rebuilding

Prior to this version, if an application used a collection with a non-exact collation (for example, MVR) and allowed NULL elements, an SQL INSERT or UPDATE of the collection could store the wrong value in the index. This is also true for applicatons that have an index on a collection(ELEMENT) or collection(KEYS) value, and the collection value can be NULL.

Applications with such indexes must rebuild them in this version in order to index the existing NULL collection values properly.

Mirroring-Related Changes

- Mirroring .CPF Changes

  All mirror members are now configured in the MapMirrors section of the CPF file. The [MirrorAsyncMemberSources] section in CPF file is obsolete, it exists onlyafter the instance is upgraded to 2012.2. When the async member starts up and connect to the failover member, the [MirrorAsyncMemberSources] section is automatically converted to Mirrors and MapMirrors sections

  The MapMirrors section in CPF file contains all types of mirror members (failover and async). Caché adds a member to the MapMirrors section if the incoming connection from a failover or async member is not in the MapMirrors section yet.

  The AsyncMemberGUID in MirrorMember section indicates whether the system is an async member (is a non-null string) or failover member (is a null string). The JoinMirror in the MirrorMember section will be 1 for async member using MapMirrors/Mirrors to configuration the mirror. Previously, the JoinMirror had to be zero for async member when it used MirrorAsyncMemberSources section to configure the async member.

  The AsyncMemberType is now part of the MirrorMember section for all mirror sets in an async mirror member; previously it resided in MirrorAsyncMemberSources. A value of 1 indicates this is a reporting async member and allows the mirrored databases to be read-write; the value of 0 indicates this is a disaster recover async member which implies all the mirrored databases are read-only.

- Primary Mirror Identification Changes

  In previous versions, the result from **##class(%SYSTEM.Mirror).IsPrimary** or **$SYSTEM.Mirror.IsPrimary()** was only TRUE if the system was the active primary. In this version, it is now true if the system is the primary mirror member, and will remain the primary mirror member until it is shutdown (that is, regardless of its trouble state). If

$SYSTEM.Mirror.GetInfo() contains both "PRIMARY" and "ACTIVE", then **$SYSTEM.Mirror.IsPrimary()** will return TRUE.

Note: There is a small window during primary startup where **$SYSTEM.Mirror.GetInfo()** contains "PRIMARY" but not ACTIVE (it contains either FAILOVER or RECOVERY at this point). During this time **$SYSTEM.Mirror.GetInfo()** returns FALSE.

If an application was using **$SYSTEM.Mirror.IsPrimary()** to detect whether the primary is up and healthy, the logic involved will have to be rewritten for the new situation.

- Async Mirror Members May Now Use Journal Purge Interval

  Async Mirror members can now be configured to purge journal files based on the system journal file purge setting (System>Configuration>Journal Settings) rather than as soon as the async member is done with journaling. This means the system to be configured to retain the journal files for longer if there is some reason to do so.

  In 2012.2, InterSystems removed the [MirrorMember] DaysBeforePurge parameter from the .cpf file. When upgrading a 2011.1 async member to 2012.2, if this parameter is non-zero then the upgrade propcess sets the new AsyncUseSystemPurgeInterval parameter is set to 1. This means that mirror journal files on the async member will be purged according to the system setting for purging journal files.

  Important: If the time period for the system setting is less than the prior value of the DaysBeforePurge parameter, mirror journal files are not retained for as long as the system manager had intended.

Change to Journal Restore Dialog

In prior releases, if journal restore got errors updating a database, it would dismount it to prevent users from accessing the database inadvertently. With this release, it no longer automatically dismount the databases upon errors; instead, at the end of journal restore, it gives the user an option to dismount them:

```
Some updates of the following database(s) were skipped during journal restore,
due to errors. The database(s) might be in an inconsistent state.
<list of skipped databases>
Do you want to dismount the above database(s) (yes/no)?
```

The prompt is present only if journal restore gets into the situation as described above. It does not count non-journal target databases, which are always skipped. The prompt must be answered with either y(es) or n(o); no default is presumed.

Users who script journal restore based on input/output may need to adapt the script to this potential prompt. Users of the journal restore API (Journal.Restore class) may set the Dismount property specifically to have the skipped databases dismounted. The default for this property is No.

Temporary Files Deleted When FileStream Is Removed

This version changes the behavior of the %IO.FileStream class so that, if it is used to create temporary file instances, the corresponding temporary disk file is deleted when the file object instance is killed or goes out of scope.

This change also introduces a new property of the %IO.FileStream class, IsTemp. Setting IsTemp to 0 preserves the behavior of previous releases.

%Admin_Manage Role Needed for Broadcast

The ability to broadcast messages using the **Broadcast** method of %System.Process(**$ZU(94)**) to terminals and processes now requires the application doing so to have the **%Admin_Manage** role.

The Caché Control Process Now Spawns Processes Under the Service Userid

The Caché control process serving the local connection now spawns jobs ($ZF(-1)) in the same user/permission context that all other Caché processes do. That is, it uses the userid declared for the Caché service.

Null Character Conversion from Unicode to Caché Locale Unified

In previous releases, the conversion of a null string from Unicode to the current Caché locale equivalent differed between UNIX® and Windows systems. The resulting string was zero characters long on UNIX®, and a single NUL character on Windows. In this release, both platforms return a NUL character.

Unauthenticated Access and Two-Factor Authentication

When the Terminal or Bindings service is configured to allow Unauthenticated access and also to require two-factor authentication, the second factor (security token sent to mobile phone) will not be performed for the UnknownUser unless that user is configured with a mobile phone number.

Health Monitor And Application Monitor Functionality Now Under System Monitor

In this release, Health Monitor and Application Monitor are unified under a single entity, System Monitor. System Monitor encompasses the following:

- Management Portal Dashboard (formerly implemented as an Application Monitor class %Monitor.System.Dashboard run by **^%MONAPP** in the %SYS namespace)
- any other Application Monitor class running in %SYS
- Health Monitor

In addition, ^%MONAPP and Health Monitor are no longer started at startup; instead, System Monitor is started, which runs these applications.

System Monitor is configured via a new utility, **^%SYSMONMGR**. Application Monitor configuration for %SYS(%MONAPPMGR), and Health Monitor (MONHEALTHMGR) can also be run from this single entry point.

System Health components can only run in %SYS. This requirement affected **%Monitor.Health.Period.Create()** and **%Monitor.Health.Period.Modify()** which have been renamed to **SYS.Monitor.Health.Period.Create()** and **SYS.Monitor.Health.Period.Modify()**, respectively.

Name And Format Changes For Monitor Log

In this release, the Health Monitor log file, HealthMonitor.log, is renamed to SystemMonitor.log. This log is acccessible through the Management Portal under System > System Logs.

Also,to improve searching in the HealthMonitor.log for details of Health Monitor notifications, the datetime format used is the same as that in the cconsole.log file.

Long Strings Enabled By Default For New Installs

Beginning with this release, new installations of Caché have long strings enabled by default.

New Maximum Size And Default Values For Configuration Parameter "bbsiz"

The *bbsiz* parameter in the CPF file now allows a maximum of 2147483648KB. The default on a new install is now 262144KB. On an upgrade, the value remains unchanged.

New Default Size For gmheap

In this release the default size for gmheap has been increased to account for the iKnow Spanish and English language models which are loaded as part of the default configuration. When upgrading to this release. if the existing value for gmheap is less than the default, it will be set to the default. If the value is greater than the default, it will be left unchanged.

Change Listening Strategy For .NET Gateway

The DotNetGatewaySS.exe process takes three arguments: port, host and logfile. The port argument is required, but the others are optional.

The default value for host had been 0.0.0.0; the gateway listens on all TCP/IP adapters for a connection. Starting with this version, the default is now the loopback port 127.0.0.1.

This is being done to restrict access to the TCP/IP port from anyone other than the machine the DotNetGatewaySS.exe is running. The gateway default is now consistent with the Management Portal. Users can still configure the host to listen on other IP addresses, with additional firewall configurations as they wish..

Journal File Integrity Checking Now Verifies File Chronology

The class reference documentation for **CheckIntegrity()** of %SYS.Journal.File has been updated to make it clear that the journal file paths that are passed as an array to the method are expected to be in chronicle order of their creation. The method has also been updated to check the headers of the given journal files. If it is found that the journal files are specified out of order, a single value 0 is returned and the top node of the array is set to the number index of the offending element.

Change Locale Defaults Of Locale JPUWFrom EUC To UTF8

Beginning with this release, applications running in locale `jpuw` (Japanese/Unix) that open files or printers without specifying a translation table (that is, relies on the locale default) will show different results. To produce the results from prior releases, the application must explicitly set the desired locale.

Changes Made To Czech Locale

Czech locales (`csw8`, `csy8` and `csyw`) now have localized strings for currency, weekdays and months properly set. In addition, they have the proper date format (for example, 23.03.2012) and number format (1 234 567,89).

Question Sequence Change To Journal Restore

Running ^JRNRESTO on a failover mirror member triggers a new question regarding whether the user wants to restore mirrored or non-mirrored databases. Mirrored database restore is handled via catchup, non-mirrored database restore continues to use the existing ^JRNRESTO interface. This release adds a menu option to ^JOURNAL to restore mirrored databases.

Customers who have scripted the journal restore process will need to account for this change when ^JRNRESTO/^JOURNAL is executed on failover mirror members.

CACHE Database Is Always Read-Write

The CACHE database will now be treated like the CACHESYS database. An attempt to change it from read-write to read-only using ^**DATABASE**, the Management Portal, or the `SYS.Database` object will be ignored. The database always remains in read-write mode.

Delimiter Change In Listing Namespaces

During the field test versions of this release, the method **##class(%SYS.Namespace).ListAll(.x)** returned an implicit namespace string with the system name enclosed between two at-sign characters, "@". For the general release, the delimiters are now "^" characters.

Package Mapping Overrides Routine Mapping

When routine has a name containing a period (.) and the name preceding the period is the same as a name involved in a package mapping, the package mapping overrides routine and the routine will not be visible. Routine names must not collide with the names of packages involved in package mappings.

9.2.1.4 Platform-specific Items

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

Windows

- Do Not Rollback If Database Update Fails In Upgrade

  The Windows installation activity will not initiate rollback during an upgrade if there is an error in performing the DatabaseUpdate action. If there is an error during DatabaseUpdate, there will be an error message displayed and update will proceed to the final dialog.

  This change does not affect new installs where, if there is an error in DatabaseUpdate action, it will still be rolled back.

- Second Ethernet Address Unavailable

  This release removes **EthernetAddress(2)** from the class $SYSTEM.INetInfo on Windows because the underlying support is not available on this platform.

- Timeout For Windows Startup Removed

  Previously, when restarting an instance on Windows, Caché would wait up to 180 seconds for Windows to create the shared memory segment it needed to run. When the requested segment was larger than 5GB and memory was fragmented, this limit could be exceeded and the instance failed to start. This release removes the time limit altogether.

- RoseLink Files No Longer Installed

  In this release, Caché no longer installs and registers RoseLink-related files.

Windows And UNIX® / Linux

- Callin Linking

  Beginning with this release, some Caché kernel components are now implemented in C++. Therefore, any callin module linking with the Caché static library/object (cache.o) must include the platform standard C++ lib to their link command. (This is not required when linking with the Caché dynamic library, libisccache.dylib).

- OpenSSL Version

  In this release, InterSystems has updated the openssl library to version 1.0.0e for Windows and Unix. In addition, the distribution includes the International Data Encryption Algorithm (IDEA) cipher. All InterSystems projects depending on openssl have been modified to use new version. Applications affected by changes in the new version should be updated as well.

Linux RedHat 32-Bit And 64-Bit

In this version, InterSystems has rebuilt httpd for the 32-bit and 64-bit platforms to remove a dependency on libexpat.so.0 library. This library was previously installed as part of a normal installation but may no longer be present depending on the initial system configuration.

MacOS 64

Upgrading a macOS 64–bit installation deletes the following LDAP libraries:

- libldap-2.4.2.dylib
- libldap-2.4.2.1.0.dylib
- libldap.dylib
- liblber-2.4.2.1.0.dylib
- liblber-2.4.2.dylib
- liblber.dylib

OpenVMS

Sites using HPSWS on OpenVMS must configure the CSP Gateway to generate a content-length for all Management Portal responses. To set this, select Application Access for `/csp` in the System Portal pages for the CSP Gateway, and verify that "Response Size Notification" is set to "Content-Length".

9.2.2 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.

9.2.2.1 Routine Compiler Changes

$ZTRAP Inside a TRY Block Not Allowed

The use of $ZTRAP inside a TRY block has always been illegal because the error handling order is ambiguous., but until this version, it was not detected by the routine compiler when the TRY occurred inside a procedure. Beginning with this version, the compiler reports such usage as an error.

9.2.2.2 Routine Changes

$COMPILE-Generated Object Code Will Not Use Routine Path Setting

Customers who are setting $SYSTEM.Process.UserRoutinePath() (formerly $ZU(20)), will see a change in behavior from version 2011.1. The change re-adopts the behavior of version 2010.2 and earlier.

Before 2011.1, routine utilities such as $SYSTEM.OBJ.Load() would save the compiled object code in the current namespace. In 2011.1 and later, if a UserRoutinePath has been set, the compiled object code would be saved in the routine path namespace instead of the current one. This version restores the previous behavior so the routine utilities will ignore the UserRoutinePath.

Enhanced Check For Illegal Global References

A previous release enforced the rule that global names cannot end with a period. However, the check did not also examine indirect references with subscripts. In this release, that case is also checked so that

```
Set name = "^zzz.(3)"
Set @name=99
```

will result in a <SYNTAX> error.

9.2.2.3 Class Changes

Func() Method Added To Query Classes

A new query member method is available for use, **Func**. **Func()** accepts actual values corresponding to the formal parameters defined by the query. It returns an instance of %SQL.StatementResult. When the **Func** method executes successfully that instance of %SQL.StatementResult is a result set.

If an application has a class with a method whose name is the same as a query name concatenated with "Func", then a member method name collision will be reported at compile-time. Refer to the %Library.Query class for more information.

Callbacks Added For Persistent Class Index Maintenance

Four new callback methods are now available for persistent classes — **%OnBeforeBuildIndices**, **%OnAfterBuildIndices**, **%OnBeforePurgeIndices** and **%OnAfterPurgeIndices**. These callbacks are invoked by **%BuildIndices** and **%PurgeIndices** when the callback implementation is runnable.

The "Before" callback is called prior to doing any purge or build work but after any validation, index list processing and extent lock work. The "After" callback is called after all work is complete but before the extent lock is released. If a "Before" callback returns an invalid status value, then the Build/Purge method is exited immediately with no build/purge work performed and the invalid status value is returned to the caller of the Build/Purge method. For more information refer to the %Library.Persistent class documentation.

%ResultSet.* Classes Superceded

The %ResultSet.SQL and other classes in the %ResultSet package have been superseded by the Caché Dynamic SQL implementation. Refer to the class documentation for %SQL.Statement and the Dynamic SQL section of the Caché documentation for more information.

Note: %Library.ResultSet continues to be used by a number of Caché features but for Objectscript InterSystems recommends using %SQL.Statement.

Extent Manager Reports Deletion Of A Non-Existent Extent As An Error

In this version, attempting to delete a class extent using the **DeleteExtentDefinition(<extentname>)** of %ExtentManager.Utilnows return an invalid status error if the extent name does not exist. Applications that depend on the previous behavior must add a check for the existence of the extent before attempting to delete it.

Global Stream Classes Now Report Status Properly

The **%SaveData** method of the legacy global stream classes (for example, %Library.GlobalBinaryStream or %Library.GlobalCharacterStream)was not properly returning a failure status if an error occurred during save. This omission is fixed and **%SaveData** returns an accurate status value indicating success or failure.

%Exception.AbstractException — DisplayString Method Must Return %Library.String

The %Exception.AbstractException class **DisplayString** method must return the string to display for this exception.. However, previously, the method signature did not have a return type defined at all. The method is now defined to return an instance of %LibraryString.

Applications that have subclasses of the exception class that have overridden DisplayString must change those classes to add %String as the return type.

Changes To Export And Import Selectivity

In prior releases, Caché did not include selectivity information as part of a class export unless the caller passed in the `/exportselectivity` qualifier. As a consequence, exporting a class from one namespace and importing it in another did not by default not move the selectivity information. This caused problems because this selectivity information is needed to analyze SQL query issues. Without this information, the compilation of a query could result in choosing a sub-optimal query plan.

Starting with this release, Caché uses different defaults. On export, Caché will export the selectivity information by default as the exported class should fully reflect the class we have stored in the database. On import, Caché will use `/importselectivity=2` by default, that is, keep any existing selectivity value if the class already has some but if a property does not have an existing value then use the selectivity from the import XML file.

Note: This test is done on a property by property basis.

Class Deletions

The following classes were present in the previous version and have been removed in this distribution:

- %Library.Storage — %SQLDelete
- %Monitor.Health — AbstractCallback, AbstractSensor, Chart, Control, HealthAlert, Period, Rule, SensorClass, SensorReadings, SystemSensors, Test
- %Net.Remote.Java — ReentrancyTest
- %ResultSet — SQLQuery
- %cspapp.exp — utilexpclasscompile, utilexpclassdelete, utilexpclassexport, utilexpclassimport, utilexpclasslist, utilexproutinecompile, utilexproutinedelete, utilexproutineexport, utilexproutinefind, utilexproutineimport, utilexproutinelist, utilexpviewroutine %cspapp.op utilcspsessions, utilensembledashboard, utilopaction
- %cspapp.sec — utilsysresource, utilsysresources, utilsysssl, utilsysssllist
- %iKnow.UI.Zen — extendedDataCombo, iKnowSuggestBox
- %iKnow.Utils.MaintenanceWSAPI — AddEntity

- %template — soapclientwizarddone, webservicepolicy, webservicepolicycreate, webservicepolicydone, xmlschemawizarddone, xsltwizard, xsltwizarddone
- Config — Mirror, MirrorSetMembers
- SYS.DataCheck — LocalDatabasePath
- Security — Authentication, BooleanYN, Password, Permission

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CPT.CalloutIndex	Method	ModuleDirectory
%CPT.CalloutShell	Method	StandardLanguage
%CPT.CalloutTypeIndex	Method	ShowChild
%CPT.HPT.LoadingState	Property	NodeIndex
%CPT.Tree.Fragment	Method	MatchPlaceholder, Root
	Property	RootNodeIndex
%CSP.Util.SMTitlePane	Method	DrawQuickJumpList
%CSP.UI.Portal.About	Parameter	APPLICATION, CSPURL
%CSP.UI.Portal.Dialog.ChangePassword	Property	UserRoles
%CSP.UI.Portal.Dialog.ZenReportServerAction	Method	ondialogFinish
%CSP.UI.Portal.NLS	Method	DrawHelpText
%CSP.UI.Portal.ObjectGateway	Property	HasModified
%CSP.UI.Portal.TaskInfo	Method	GetPIDOBJ
%CSP.UI.Portal.ZenReportServer	Property	IsModified
%DeepSee.Component.chartLegend	Method	onloadHandler
%DeepSee.KPI	Method	%OnGetFilterMembers
%DeepSee.Query.query	Method	%ProcessFilterSpec
	Property	%filterIncludes, %slicerIncludes
%DeepSee.Report.UI.chartInfoWizard	Property	seriesTxt
%DeepSee.UI.Analyzer	Parameter	CSPURL
%DeepSee.UI.Architect	Parameter	CSPURL
%DeepSee.UI.ArchitectSA	Parameter	CSPURL
%DeepSee.UI.Dialog.WidgetBuilder	Method	ChangeWidget, adjustWidgetSize
%DeepSee.UI.Dialog.WidgetCatalog	Method	%OnDrawHTMLHead, adjustWidgetSize, dashboardEventHandler
%DeepSee.UI.FolderManager	Parameter	CSPURL

Class	Type	Name(s)
%DeepSee.UI.LogViewer	Parameter	CSPURL
%DeepSee.UI.MDXQuery	Method	setQuery
	Parameter	CSPURL
%DeepSee.UI.Settings	Parameter	CSPURL
%DeepSee.UI.WorksheetBuilder	Method	getColorsPopup, getColorsWidget
	Parameter	CSPURL
%Dictionary.ClassDefinition	Method	ClassTypeIsValid, LanguageIsValid
%Dictionary.MethodDefinition	Method	LanguageIsValid
%Exception.AbstractException	Property	Code, Data, InnerException, Location, Name
%Library.CacheLiteral	Method	IsModified, SetModified
%Library.DynamicQuery	Method	SQLClose
%Library.EnsembleMgr	Method	setupHealthShare
%Library.Storage	Method	%SQLLogicalToOdbcFields, %SQLOdbcToLogicalFields
%Net.Remote.Gateway	Method	%CallServer
	Property	Proxies
%Net.Remote.Java.JavaGateway	Method	%JavaBindingAttach, %JavaBindingDetach
	Parameter	JAVABINDINGATTACH, JAVABINDINGDETACH
%SQL.Statement	Method	%MetadataGet, %MetadataSet
%SQL.Statement	Property	%StatementClass
%SYNC.Transporter	Method	OpenFile, TimeTransport
%SYS.Task.DiagnosticReport	Method	AuthPassDecode, AuthPassEncode
%SYSTEM.Help	Method	%ClassName
%Stream.GlobalCharacter	Method	%Exists, %GetLockReference, %IsModified, %LogicalToStorage, %NormalizeObject, %ObjectModified, %OnClose, %OnConstructClone, %OnNew, %OnRollBack, %StorageToLogical, %ValidateObject, BuildValueArray, Clear, CopyFrom, Flush, IsNull, LastModifiedGet, MoveToEnd, OutputToDevice, Read, ReadIntoBuffer, ReadLine, ReadLineIntoStream, Rewind, SizeGet, Write, WriteLine
	Parameter	BUFFERLEN

Class	Type	Name(s)
	Property	Buffer, IOSize, LineTerminator, MaxNodeNo, Mode, NodeNo, Position, RemoveOnClose, TempGbl, mLastModified, rollback
%Stream.TmpCharacter	Method	CopyFromAndSave
%TSQL.ResultSet	Method	%Execute, Execute
%UnitTest.Result.TestAssert	Method	SaveIndices
%UnitTest.Result.TestCase	Method	SaveIndices
%UnitTest.Result.TestInstance	Method	SaveIndices
%UnitTest.Result.TestMethod	Method	SaveIndices
%UnitTest.Result.TestSuite	Method	SaveIndices
%WebStress.Scripts	Method	ShowProgress
%XML.ImportHandler	Property	HandlerType
%ZEN.Dialog.routineSelect	Method	DrawRoutineItem
%ZEN.Report.Display.controller	Parameter	NAMESPACE, XMLFORMAT
%ZEN.Report.Display.node	Method	%GenerateCode, %QuoteValue, %QuoteValueEmbedded, %QuoteValueL10N
	Property	composite, id, parent
%ZEN.Report.Display.pagefooter	Property	orientation
%ZEN.Report.Display.pageheader	Property	orientation
%cspapp.op.utilsysjobinfo	Method	GetTitlePane
	Parameter	PARENTPAGE
%iKnow.DirectInputDO	Property	IKSCRC, IKSENT, IKSPATH, IKSTYPE
%iKnow.Matching.MatchingWSAPI.GetDictionaryMatches	Property	profileId
%iKnow.Matching.MatchingWSAPI.GetDictionaryMatchesById	Property	profileId
%iKnow.Objects.DictionaryElement	Property	MinMatchType, Position, Role
%iKnow.UI.AbstractPortal	Method	BuildFilterForm, StoreFilter, onChangeOperatorClient, onToggleFilterClient
%iKnow.UI.IndexingResults	Method	DeleteCurrentVirtualSource, ProcessInput, displayInputScreen, realTimeInputClient
	Property	Loader
%iKnow.UI.MatchingResults	Method	DoMatching
%iKnow.Utils.MaintenanceQAPI	Method	AddEntity
%iKnow.Utils.MaintenanceWSAPI	Method	AddEntity

Class	Type	Name(s)
Config.CommonMapMethods	Method	Download, UploadClose, UploadExecute, UploadFetch, Upload
	Query	Upload
Config.CommonMultipleMethods	Method	Download, UploadClose, UploadExecute, UploadFetch
	Query	Upload
Config.CommonSingleMethods	Method	Download, Upload
Config.MirrorAsyncMemberSources	Method	UpdateDatabasesForRWAsyncMember
Config.MirrorMember	Method	DaysBeforePurgeSet
	Property	DaysBeforePurgePresent
Security.System	Property	BypassSecurity

Method Return Changes

The following methods have different return values in this version of Caché:

- %CPT.CalloutCommon — NodeLineRange
- %DeepSee.Query.scalarFunction — %EvaluateScalar, %IsScalarFunction
- %Exception.AbstractException — DisplayString
- %Exception.SQL — SQLMessageString
- %Exception.StatusException — DisplayString
- %Library.CacheLiteral — SQLCompute
- %SQL — DynamicStatement
- %XSQL.Log — logSQLCODE
- SYS.DataCheck.Destination — Connect
- SYS.MirrorConfiguration — GetOtherFailoverSystemConfig, RetrieveMirrorMemberConfig
- SYS.Mirroring.GetMirroringInformationSoap — GetFailoverMemberInformation
- SYS.Mirroring.GetMirroringInformationSoap.GetFailoverMemberInformation — Invoke
- SYS.Mirroring.MirroringWebServices — GetFailoverMemberInformation

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CPT.CalloutCommon	BuildAllIndices
%CPT.CalloutTesting	SetupSettingsString
%CPT.SQLCallout	Compile
%CPT.Tree.Fragment	%OnNew, IndexTree
%CSP.StudioTemplateMgr	AddTemplate

Class Name	Method Name(s)
%CSP.UI.Portal.ZenReportServer	SaveData
%CSP.UI.SQL.UserPrivPane	LoadPriv, LoadRole, LoadUserRole
%Collection.ListOfDT	LogicalToOdbc, OdbcToLogical
%Compiler.COS.Refactor	ChangeClassNameExecute, oneClass
%Compiler.XML.Generator.Adaptor	GenLiteralImport, GetSimpleExport
%DeepSee.AbstractKPI	%GetFilterMembers, %GetKPIValue, %OnGetFilterMembers
%DeepSee.Component.pivotController	getIconHTML, updateState
%DeepSee.Component.pivotTable	GetQueryStatus, selectCellRange
%DeepSee.ComputedDimension.Base	%GetComputedMembers
%DeepSee.ComputedDimension.iKnow	%ProcessIKnowFact
%DeepSee.Generator	%BuildCubeInfo, %GetSQLFieldExpression
%DeepSee.KPI	%GetFilterMembers
%DeepSee.KPIWorksheet	%GetKPIValue
%DeepSee.Query.Engine	%AggregateEnd, %Consolidate, %GetBranchesForSlicer
%DeepSee.Query.query	%RewriteCompoundQuery, %RewriteGroup
%DeepSee.Query.scalarFunction	%EvaluateScalar
%DeepSee.Report.UI.dataPickPool	setDataTree, setDataTreeFromDSS
%DeepSee.Report.UI.reportPreviewer	GenerateDataSet, execute
%DeepSee.Report.dataPresenter	CreateOutputStream
%DeepSee.ResultSet	%ExecuteListing, GetDefaultFormat, %GetQueryStatus, %GetStatus
%DeepSee.TaskMaster	DequeueTask
%DeepSee.UI.Dialog.Analyzer	SavePivotTable
%DeepSee.UI.Dialog.WidgetBuilder	AddWidgetToDashboard
%DeepSee.UserPortal.Utils	%ClearDashboardSettings
%DeepSee.Utils	%GetCubeLevels, %GetCubeList, %SaveSubjectAreaDefinition, %SynchronizeCube
%ExtentMgr.GlobalRegistry	FindReference, IncompatibleUse, LockUse, RegisterReference, UnLockUse
%ResultSet.Static	%OnNew
%Installer.AbstractLogger	flushIO
%Library.EnsembleMgr	OnSystemStartup
%Library.FilemanTimeStamp	LogicalToDisplay

Class Name	Method Name(s)
%Library.Persistent	%KillExtent
%Library.ProcedureContext	Prepare
%Library.RegisteredObject	%OnNew
%Library.Routine	normalizeName
%Library.SerialObject	%Open
%Library.Storage	%SQLDelete
%Net.Remote.Gateway	%Connect
%SOAP.Security.Policy	CheckEncryptedSupportingTokens, FindToken, GetSecurityToken
%SQL.DynamicStatement	Prepare, findStatement
%SOAP.WST.RequestSecurityToken	CreateIssueResponse
%SOAP.WebParameters	ParseParameters, ParseParametersElement
%SOAP.WebService	Fault, ReturnInternalStatusFault
%SYNC.Transporter	ExportFile, Import, getTransporter
%SYS.Journal.File	GetNext, GetPrev, MirrorByTimeReverseOrderExecute
%SYS.NLS.Locale	%OnNew
%SYS.Portal.Resources	%SetCustomResource
%SYS.ProcessQuery	CONTROLPANELExecute, NextProcess, VariableByJobNumberExecute, VariableByPidExecute
%SYS.Task.History	WriteLog
%SYS.TaskSuper	DisplayGUID
%SYS.ZENReportExcelExporter	ExportToExcel, ExportToXlsx
%SYSTEM.SQL	ImportDir
%SYSTEM.Security	GetGlobalPermission
%Studio.SourceControl.ItemSet	Import, Load, LoadToNS, LoadToOS
%UnitTest.SQLRegression	verifyResults
%WebStress.Playback	RunStop
%WebStress.Record	Run
%WebStress.Scripts	CacheRecorder
%XML.Implementation	AnalyzeTiming
%XML.ImportHandler	%OnNew
%XML.Namespaces	PushNodeForExport
%XML.Security.Signature	ComputeSha1Digest
%XSQL.DSI.GlobalPrivateTable	%OnAfterGenerate

Class Name	Method Name(s)
%XSQL.DSI.TempTable	%OnAfterGenerate
%ZEN.Generator	%DoesXDataExist
%ZEN.Report.Display.bidioverride	%DrawToXSLFO
%ZEN.Report.Display.block	%DrawToXSLFO
%ZEN.Report.Display.body	%DrawToXSLFO
%ZEN.Report.Display.br	%DrawToXSLFO
%ZEN.Report.Display.call	%DrawToXSLFO
%ZEN.Report.Display.callsvg	%DrawToXSLFO
%ZEN.Report.Display.caption	%DrawCellFO, %DrawCellFO1
%ZEN.Report.Display.category	%DrawToXSLFO
%ZEN.Report.Display.class	%DrawToXSLFO
%ZEN.Report.Display.container	%DrawToXSLFO
%ZEN.Report.Display.cssinclude	%DrawToXSLFO
%ZEN.Report.Display.div	%DrawToXSLFO
%ZEN.Report.Display.fo	%DrawToXSLFO
%ZEN.Report.Display.foblock	%DrawToXSLFO
%ZEN.Report.Display.group	%DrawToXSLFO
%ZEN.Report.Display.html	%DrawToXSLFO
%ZEN.Report.Display.img	%DrawToXSLFO
%ZEN.Report.Display.init	%DrawToXSLFO
%ZEN.Report.Display.inline	%DrawToXSLFO
%ZEN.Report.Display.inlinecontainer	%DrawToXSLFO
%ZEN.Report.Display.item	%DrawToXSLFO
%ZEN.Report.Display.line	%DrawToXSLFO
%ZEN.Report.Display.link	%DrawToXSLFO
%ZEN.Report.Display.masterreference	%DrawToXSLFO
%ZEN.Report.Display.node	%StyleHTML, %StyleXSLFO
%ZEN.Report.Display.p	%DrawToXSLFO
%ZEN.Report.Display.pagefooter	%DrawToXSLFO
%ZEN.Report.Display.pageheader	%DrawToXSLFO
%ZEN.Report.Display.pagemaster	%DrawToXSLFO
%ZEN.Report.Display.section	%DrawToXSLFO
%ZEN.Report.Display.smallMultiple	%DrawToXSLFO
%ZEN.Report.Display.table	%DrawToXSLFO

Class Name	Method Name(s)
%ZEN.Report.Display.tableOutput	%DrawCellFO, %DrawCellToHTML, %DrawFooterFO, %DrawFooterToHTML, %DrawHeaderFO, %DrawHeaderToHTML
%ZEN.Report.Display.tbody	%DrawToXSLFO
%ZEN.Report.Display.td	%DrawToXSLFO
%ZEN.Report.Display.th	%DrawToXSLFO
%ZEN.Report.Display.thead	%DrawToXSLFO
%ZEN.Report.Display.timeline	%DrawToXSLFO
%ZEN.Report.Display.tr	%DrawToXSLFO
%ZEN.Report.Display.xslinclude	%DrawToXSLFO
%ZEN.Report.Ping	ping
%ZEN.Report.reportPage	%DrawToXSLFO, %GetFileByAbsoluteURL, %GetFileByRelativeURL, %MakeToXSLFOFile, CreateOutputStream
%cspapp.exp.utilexpfindreplace	Cleanup, SaveData, SaveRoutine
%iKnow.DirectInput	ProcessPath
%iKnow.Filters.GroupFilter	%OnNew
%iKnow.Filters.SimpleMetadataFilter	%OnNew
%iKnow.KB.Knowledgebase	LoadDir
%iKnow.LB.Languagebase	LoadDir
%iKnow.Matching.DictionaryAPI	CreateDictionaryTerm
%iKnow.Matching.DictionaryQAPI	CreateDictionaryTerm
%iKnow.Matching.DictionaryWSAPI	CreateDictionaryTerm
%iKnow.Matching.MatchingAPI	GetDictionaryMatches, GetDictionaryMatchesById, GetTopItems, GetTotalDictionaryScoresBySource, GetTotalDictionaryScoresBySourceId, GetTotalItemScoresBySource, GetTotalItemScoresBySourceId, ProcessAggregate
%iKnow.Matching.MatchingQAPI	GetDictionaryMatchesByIdExecute, GetDictionaryMatchesExecute, GetTopItemsExecute, GetTotalDictionaryScoresBySourceExecute, GetTotalDictionaryScoresBySourceIdExecute, GetTotalItemScoresBySourceExecute, GetTotalItemScoresBySourceIdExecute
%iKnow.Matching.MatchingWSAPI	GetDictionaryMatches, GetDictionaryMatchesById, GetTopItems, GetTotalDictionaryScoresBySource, GetTotalDictionaryScoresBySourceId, GetTotalItemScoresBySource, GetTotalItemScoresBySourceId

Class Name	Method Name(s)
%iKnow.Queries.EntityAPI	GetByFilterInternal
%iKnow.Queries.CrcAPI	GetCountByDomain
%iKnow.Queries.CrcQAPI	GetCountByDomain
%iKnow.Queries.CrcWSAPI	GetCountByDomain
%iKnow.Queries.EntityQAPI	GetCountBySource
%iKnow.Queries.EntityWSAPI	GetCountBySource
%iKnow.Queries.MetadataAPI	AddField, AddListOfValues, GetFieldId, GetValue, GetValueById, UpdateField, UpdateFieldById
%iKnow.Queries.MetadataQAPI	UpdateField, UpdateFieldById
%iKnow.Queries.MetadataWSAPI	UpdateField, UpdateFieldById
%iKnow.Queries.SentenceAPI	GetLanguage
%iKnow.Queries.SentenceQAPI	GetLanguage
%iKnow.Queries.SourceAPI	GetTopLanguage
%iKnow.Queries.SourceQAPI	GetTopLanguage
%iKnow.Source.Lister	BuildExtIdFromName
%iKnow.Source.Loader	ProcessBuffer, ProcessVirtualBuffer
Config.Journal	JournalFilePrefixIsValid
Config.MapShadows	ListExecute
Config.NLS.Locales	ImportDir
Config.NLS.SubTables	ImportDir
Config.NLS.Tables	ImportDir
Ens.Adapter	OnKeepalive
Ens.Alarm	CheckAlarmTasks, RemoveAlarm
Ens.BPL.UI.Diagram	GetBPLStream, Open
Ens.Config.Production	addSchema, getDTLSchemas, getItemSchema
Ens.Job	Launch, RecoverActiveMessage, Start
Ens.Rule.Model.expression	parseExpression, parseToken, test
Ens.Util.Documentation	BuildURL, CreateDoc
Ens.Util.LookupTable	%Import
SYS.DataCheck.Destination	%OnNew, Connect, ManagerWorkflow
SYS.DataCheck.Phase	%CreateQuery
SYS.DataCheck.Query	Answer
SYS.DataCheck.RangeList	ListElementsExecute
SYS.DataCheck.System	%OnNew

Class Name	Method Name(s)
SYS.Database	Copy, Defragment
SYS.Mirror	CatchupDB, PurgeAsyncMemberJournalFiles
SYS.MirrorConfiguration	AddFailoverMember, CheckMemberConnection, CheckNewMirroredDB, CreateNewMirror, CreateNewShadow, JoinExistingMirror, RetrieveMirrorConfig, RetrieveMirrorMemberConfig
Security.Events	Create, Set
Security.Resources	Create
Security.Roles	Import
Security.Users	Import, UpdateOne

9.2.2.4 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

Change to Internal Representation Of Signatures

Beginning with this release, the internal representation of non-literal formal argument initial values may be delimited by "{" and "}". Applications, such as programming tools, that do their own interpretation of parsed formal specifications in the class dictionary must be rewritten to handle this case.

ONDELETE Checking Improved

The ONDELETE keyword is only supported for use with foreign keys, or in relationship declarations specifying "Parent" (the declaration of the child table) or "One" (the declaration of the "many" table). Its use is invalid in all other contexts.

This restriction was not enforced in prior releases, but it now is. For incorrect uses, the class compiler will report an error similar to:

```
ERROR #5331: OnDelete keyword value 'cascade' is only valid for a relationship: XXX
```

with the appropriate ONDELETE value, class name and property name replacing the "XXX".

9.2.2.5 Language Binding Changes

Java XEP Changes

- Java XEP: Event Persister importSchema Removed

  This version removes all methods that accepted an InterfaceResolver as one of several arguments. Applications that use these methods should be changed by replacing

  ```
  persister.importSchema(className,resolver);
  ```

  with

  ```
  persister.setInterfaceResolver(resolver);
  persister.importSchema(className);
  ```

- Java XEP: Event Persister Method Changes

  In this release, the following three event persister methods have been removed: **callCOSProcedure**, **callCOSFunction**, and **executeUpdate**; the following two new methods have been added:

```
void callProcedure(String procedureName, String routineName, Object ... args)
Object callFunction(String functionName, String routineName, Object ... args)
```

The method, **Event getEvent(String className, int indexMode)** is new in this release and is used to control indexing. The default setting for *indexMode* is `Event.INDEX_MODE_ASYNC_ON`.

The method, **Event.close**, no longer conditionally calls stopIndexing. Now it never calls it, so if that behavior is desired, the users will need to call **stopIndexing** explicitly prior to calling **Event.close**

Finally, the **startIndexing** and **waittForIndexing** methods now check to see if synchronous indexing option is enabled, and if so, they will throw an exception.

- Java XEP: Event Persister Option Changes

  The following options have been removed:

  – OPTION_INDEXING_GET_EVENT_*
  – OPTION_INDEXING_CLOSE_EVENT_*
  – OPTION_FETCH_LEVEL_*
  – OPTION_REFERENCES_*
  – OPTION_INHERITANCE_*

  The following options have been added: INDEX_MODE_ASYNC_ON, INDEX_MODE_ASYNC_OFF, INDEX_MODE_SYNC.

  Because of the option removal, applications can select the JNI query flavor via the method,

  ```
  void useJNIQuery(boolean jni_flag);
  ```

  where *jni_flag* selects JNI querying when true, and Java querying when false.

  Finally, these events have been moved from Event Persister to Event: FETCH_LEVEL_ALL, FETCH_LEVEL_DATATYPES_ONLY, FETCH_LEVEL_NO_ARRAY_TYPES, FETCH_LEVEL_NO_OBJECT_TYPES, FETCH_LEVEL_NO_COLLECTIONS.

XDO (And Dynamic LCB) Cleanly Prohibit Accessing Subclasses

These bindings do not support subclasses. However, in prior releases the eXTreme Dynamic Objects (XDO) and dynamic Light C++ Binding failed to throw an exception if an attempt was made via one of them to create a dynamic object for a class which inherits from another persistent class (that is, a subclass), and also failed to prohibit data to be stored via this dynamic object.

In the release, if an attempt is made to create a dynamic object for a class which is a subclass, an XDOException (for XDO) or Db_err (for dynamic LCB) is thrown with the message "Subclasses are not supported by this binding".

Changes to Java Packages

In this release, CacheClassBuilder is now contained in com.intersys.objects; before this it resided in com.intersys.cache. Package com.intersys.objects is considered to be public package for application developers while com.intersys.cache contain only internal classes intended for direct use InterSystems modules. InterSystems provides JavaDoc for classes in com.intersys.objects.

Class CacheClassBuilder defines an API that can be used by a Java application to create and modify Caché classes inside a Caché server. It is used by some advanced customers and therefore its place is in the public package.

The old class, com.intersys.cache.CacheClassBuilder, exists as a stub for compatibility. It is now deprecated but usable. BuildClass.java is now part of the Java samples and shows how to use CacheClassBuilder

Jalapeño Schema Incompatibility

Applications created with versions of Jalapeño earlier than 2012.2 are incompatible with the server for Jalapeño 2012.2 or higher. To update an older application, use Jalapeño 2012.2 to generate new schemas and recompile the application on this release.

9.2.2.6 SQL Changes

TO_CHAR Now Honors Locale Decimal Separator

This release now correctly uses the NLS configured DecimalSeparator for the format option "D" of the function, **TO_CHAR**, hen converting numbers to strings.

Changes to Lock Behavior for CREATE INDEX

When the execution of a DDL statement locks the class definition to be edited, the lock timeout is nopw set equal to the SQL Lock Timeout setting for the process. The old behavior used a lock timeout of 0 meaning the lock would not wait at all to see if the lock became available.

Also, for a CREATE INDEX statement execution, the class definition lock now remains until the completion of the execution of the entire statement, including the population of the index data. In preceding versions, the class definition lock was released before the index was built. This caused problems if the building of the index failed and the create index had to be rolled back.

SQL Gateway Now Supports BINARY and VARBINARY in %SQL.JDBCResultSet

The SQL Gateway now supports the datatypes BINARY and VARBINARY for use in %SQL.JDBCResultSet. Prior to this release, columns declared as binary would be retrieved as hexadecimal strings.

SQL Shell Displaymode Change

The SQL shell no longer opens statement results automatically when the displaymode is not set to the current device. Instead, the Shell allows the user to specify a path and file name where statement results are stored. If the file name is not given, a random file name is generated and used. A list of files used for a particular statement is displayed on the current device after the statement is executed.

Suppress PKEY/UNIQUE Constraint Creation on IDENTITY Column

In prior versions, when creating a table through DDL that specifies an IDENTITY column, if the column definition also included a UNIQUE or PRIMARY KEY column constraint, Caché SQL would suppress the PRIMARY KEY or UNIQUE index definition in the class; this was done because the IDENTITY column is already equivalent to the IDKEY and is projected as the PRIMARY KEY of the table. However, if the PRIMARY KEY or UNIQUE constraint was defined as a table constraint instead of a column constraint, Caché SQL did not suppress the definition of the PRIMARY KEY or UNIQUE index definition in the class.

Beginning with this release, it does; but it also means Caché has a limitation in this area. If you attempt to alter a table and drop a primary key or unique constraint where Caché did not previously create an index for the constraint because the field is the IDENTITY field, the drop of the constraint will return an error because the key could not be found. InterSystems intends to correct this in a future version with enhancements to the constraint model in the class definitions.

Round Scalar Function Now Rounds Scale Factor

The second argument to the SQL ROUND scalar function is now itself rounded to the nearest integer. For example:

```
ROUND(1234.567,2.5)
```

will be the same as

```
ROUND(1234.567,3)
```

This change makes the ROUND function consistent with the TRUNCATE function behavior.

Correction to DATETIME handling in Data Import Wizard

In prior versions, when that Data Import Wizard converted a DATETIME value with a format similar to "June 24 1947 03:04:05:006PM", the milliseconds portion of the time was not handled properly when ":" separated the seconds from the milliseconds. The millisecond value is now properly converted.

Increase Lock Timeout for Cached Query Metadata

In previous versions, the lock timeout used when the system needs to lock cached queries was the same as the SQL Lock Timeout, which defaults to 10 seconds. However, it is sometimes desirable to have a longer lock timeout for cached queries. In this release. there is a new API that a system administrator can invoke to change the lock timeout for cached query locks. It is:

```
Set StatusReturn = $SYSTEM.SQL.SetCachedQueryLockTimeout(timeout, .oldvalue)
```

In addition, this release cvhanges the default timeout for cached query locks to 120 seconds.

9.2.2.7 CSP Changes

Correct Header Display by $SYSTEM.CSP.Show

Until this release, the *showhttp* flag was interpreted the wrong way in the $SYSTEM.CSP.Show function. Now, when it is true, it shows the HTTP headers; when it is false, it does not not show them.

New Default for javascript Static Files with No Charset

In order to align Caché behavior with major web vendors such as Apache and IIS InterSystems is changing the default way static javascript files are rendered. This change only effects *.js files if they are rendered by Caché using the stream server; javascript files served from the web server are not affected.

In this release, javascript files are now be marked as Content-Type of `application/javascript` where as before they were marked as `text/javascript`. (`application/javascript` has superseded the previous content type and is the correct type defined in the HTML standard.) In addition, Caché will default to not sending a charset for javascript files; this is consistent with the default behavior of the major web servers.

This means that if the file contains a BOM (byte-order mark), the browser will automatically detect this and use the correct charset; if the javascript file does not contain a BOM, then the browser will default to reading this in utf-8. The previous behavior sent a charset based on the default file translate table which, if the file contained a BOM, prevented the browser from detecting and using this information.

AN application that wishes to override this new behavior to specify a charset for the javascript files by setting the global, *^%SYS("CSP","MimeFileClassify","JS")* to the list value **$listbuild(contenttype, binary, charset)**. For example,

```
SET ^%SYS("CSP", "MimeFileClassify", "JS") = $listbuild("text/javascript", 0 ,"ISO-8859-1")
```

which sets the older content-type and uses the iso-8859-1 charset. In addition, if the CSP translate table is defined to be something other than " ", or if the *^%SYS("CSP","DefaultFileCharset")* global is set to a null value, Caché will use this charset for javascript files as well as other text files. By default, neither of these items are set.

9.2.2.8 XML Changes

Reset Current Device Translate Table When %XML.Writer Finishes

If an invocation of %XML.Writer uses a different translate table from the process invoking it, Caché re-establishes the process translate table as the default when %XML.Writer returns. Previous versions did not do this.

Validity Checking Now Done for NUL-Terminated Empty Strings by XMLImport

In prior releases, a defect bypassed validity checking for imported strings consisting of the value $CHAR(0). In this release, Caché checks the validity of a %String value that has the value, $CHAR(0).

This change treats $CHAR(0) as having a length of zero for purposes of enforcing MINLEN and MAXLEN comparisons. This also applies to VALUELIST: to include $CHAR(0) as a valid value, two consecutive commas are needed in the VALUELIST, for example,

```
VALUELIST = ",,a,b,c"
```

9.2.2.9 SOAP Changes

Control Inheritance of Header Data

Previously all headers in the parameters XData were ignored if the parameter *SOAPMETHODINHERITANCE* was set to 0. Beginning with this release, when the web service or web client parameter *SOAPMETHODINHERITANCE* is 1, then header information from the XData for the parameter is always inherited. This change also inherits the header information which is not specific to a method (request or response element as direct child of parameters element) even if *SOAPMETHODINHERITANCE* is 0.

New, Single Method to Add to SOAP Security Header

In previous releases, when you added items to the SOAP security header, you had to choose the method to use: **AddToken()** or **AddElement()**. In this release, these methods are deprecated and replaced by the new method **AddSecurityElement()**. You are not required to make any changes to existing web services or clients. If you want to update them to use the new method, you can do so easily; it is not necessary to adjust the calling sequences.

In previous releases, if the item being added just carried data, you used **AddToken()**. This rule means that you used this method to add the following items: `<BinarySecurityToken>`, `<DerivedKeyToken>`, `<EncryptedKey>` (if it does *not* include a `<ReferenceList>`).

If the item being added causes either signing or encryption, you used **AddElement()**. This rule means that you used this method to add the following items: `<Signature>`, `<EncryptedKey>` (if it includes a `<ReferenceList>`), `<ReferenceList>`.

Argument Order Correction for WebMethod

In past versions, WebMethods created by the SOAP Wizard had the Output arguments always at the end. In this version, a ByRef argument is now placed correctly, based on the WSDL, after an Output argument.

The incorrect ordering worked for client WebMethods because XMLSequence=0 was specified for the descriptor class. However, it did not work for web services. An existing client which is rebuilt will, in this case, have its arguments reordered and require called routine to change calling sequences. This change has no effect at runtime unless the client is rebuilt using the wizard.

9.2.2.10 BASIC and MVBASIC Changes

MVBASIC Now Passes Multidimensional Array Elements by Value

Prior to this release, the parameter passing behavior for using array element variables as actual parameters in MVBASIC calls to subroutines and methods was inconsistent. When an MVBASIC application called a subroutine or function passing an array,

- it passed a subscripted array element variable of an MVBASIC array declared in a DIM statement with explicit integer dimensions by reference;
- attempting to pass an array element variable of a COS multidimensional array containing 1 or 2 subscripts generated a run-time error.

Furthermore, when an MVBASIC application called a method, it passed all subscripted array element variables by value.

Beginning with this version, when an MVBASIC application calls either a subroutine, a function or a method, a subscripted array element variable of an MVBASIC array declared in a DIM statement with explicit integer dimensions is passed using by-reference while a subscripted array element variable of a Objectscript multidimensional array is passed using by value.

This means there is no longer an error when passing an Objectscript multidimensional array element variable as a parameter. This is alsoa change to the way arrays are passed to methods.

If an application wishes to preserve the previous behavior where a change to a formal parameter in the method did not change the value of the corresponding actual parameter, it should enclose the parameter in parentheses; the compiler then interprets that actual parameter as an expression value instead of a variable.

Note: There is no change to passing any variable (array or scalar) that does not include subscripts.

Change The Definition Of IFS() When The Condition Includes Singletons

In this release, the definition of the MVBASIC **IFS(B, X1, X2)** function has changed. In previous versions, it always produced a dynamic array with same number of fields, values, and subvalues as the shape of the boolean, *B*. The new definition covers the following three situations:

1. If argument *B* contains is a singleton (that is, it does not contain any field marks, value marks or subvalue marks), either the result is all of *X1* or the result is all of *X2* depending on whether *B* is true or false.
2. If *B* is a dynamic array that contains a field component which is a singleton, the corresponding field of the result is all of the corresponding field of *X1* or *X2* (including all values and subvalues) depending on whether the singleton Boolean field component is true or false, respectively.
3. If *B* is a dynamic array that contains a value component which is a singleton, the corresponding value of the result is all of the corresponding value of *X1* or *X2* (including all subvalues) depending on whether the singleton Boolean field component is true or false, respectively.

SUM Function Now Returns Dynamic Array

The **SUM()** function is defined as returning a dynamic array string. However, in previous releases the compiler was incorrectly treating it as returning just a single number; only the first component of the dynamic array was returned. This defect has been fixed and **SUM()** now returns all the components of the dynamic array result.

MVBASIC IO Status Handling Changes

Formerly, MVBASIC placed the error status from a failing I/O statement into `@SYSTEM.RETURN.CODE`. Unfortunately, this conflicted with other uses of this same variable so MVBASIC I/O statements no longer put I/O status values there.

MVBASIC I/O statements now place the status value into the system variable, `@IO.STATUS`. If the I/O statement succeeds, a zero is placed in `@IO.STATUS`; if the I/O statement fails then a non-zero error code is placed there. I/O error codes are also placed into the system variable, `@IO.ERROR`, but this variable is unchanged on a successful I/O statement. Therefore, `@IO.STATUS` gives the status of the most recently executed I/O statement, while `@IO.ERROR` gives the status of the I/O statement that most recently failed.

Some I/O statements also put the error code into the `@STATUS` system variable; this behavior has not changed. The I/O error message codes correspond to the items in the ERRMSG file.

MVBASIC programs that used I/O error codes stored in `@SYSTEM.RETURN.CODE` must be rewritten to instead use `@IO.ERROR` (and possibly `@IO.STATUS`.)

9.2.2.11 xDBC Changes

Support Binary Values In JDBC Linked Procedures

In this version, output arguments declared as binary will be returned that way. In past versions, they were returned a hexadecimal values. The same is true of values in a select list.

Avoid Rounding In Return Values From ODBC

Previously, a C++ application read decimal values from ODBC by calling GetData(). That method requested the value as a string value which the C++ binding then converted to decimal. When the data was processed in this way, ODBC converts

it to a number and then back to string, and may lose some precision in doing so. In this release, the data is requested in a way that avoids the intermediate conversion to a string, so the original precision is preserved.

9.2.2.12 DeepSee Changes

Change to Interpretation of Combined Filters

In this release, DeepSee combines filters differently than in previous releases, in a subset of cases. This is not a change in the behavior of MDX, but instead a change in how DeepSee interprets a pivot table definition as an MDX query.

Previously, if you dragged and dropped multiple members of the same level to the **Filters** box (in Analyzer), DeepSee combined those members with a logical OR. Now it combines them with a logical AND.

To combine the members with a logical OR, you must instead do the following:

1. Drag and drop the level to the **Filters** box.

   This action adds a drop-down list to the bar above the filter table.

2. Select the members from this drop-down list.

This change potentially affects the results shown by your existing pivot tables, depending on how you created them. InterSystems suggests that you examine your pivot tables and adjust any as needed. You can do this in the Analyzer.

Alternatively, you can do this in Studio, by carefully editing the applicable .dfi items in the **Other** folder in the **Workspace** window (or by editing the container class that you use to hold the pivot tables). If you dragged and dropped multiple members of a single level to the **Filters** box, the pivot table definition contains multiple `<filter>` elements that refer to the same level. For example:

```
<pivot ...>
...
  <filter spec="[ColorD].[H1].[Favorite Color].&[Blue]" key="Blue" value="" text="Blue" ...>
  </filter>
  <filter spec="[ColorD].[H1].[Favorite Color].&[Green]" key="Green" value="" text="Green" ...>
  </filter>
</pivot>
```

To change this to the other form, which is in interpreted as logical OR, replace the two `<filter>` elements with a single `<filter>` element that has a combined value for `key`:

```
<pivot ...>
...
  <filter spec="[ColorD].[H1].[Favorite Color].Members" key="{&amp;[Blue],&amp;[Green]}" ...>
  </filter>
</pivot>
```

If you are not comfortable with making such manual changes, open the pivot table in the Analyzer and make the change there.

For examples that show the two approaches, see "How DeepSee Combines Filters" in the chapter "Filtering Pivot Tables" in *Using the DeepSee Analyzer*.

Change to Handling of Local Overrides to Pivot Table Definitions

This release changes how DeepSee handles overrides made to pivot table definitions from within dashboards.

A dashboard widget can include the Mini Analyzer button, which permits users to override the pivot table definition. The override was saved as the `localDataSource` attribute within the XML for the pivot table, and this override was visible to all users. DeepSee now saves local overrides separately for each user, within a global. The `localDataSource` attribute no longer has any effect.

9.2.2.13 Zen Changes

Changes to Generated Names For CSS3 Stylesheets

Until his release, the filenames generated for CSS3 stylesheets were not compatible with all OpenVMS releases. This is due to the inclusion of an extra “.” character in the filename. Starting with this release, the filenames that Caché generates for CSS3 stylesheets will end with “_3.css” instead of “.3.css”.

This should be completely transparent for most applications, however, it is possible that application code explicitly references the previous filenames. If that is the case, the application must be changed to use the new filename.

9.2.2.14 Zen Reports Changes

Replace Apache FOP with Skynav FOP

This release replaces Apache FOP with Skynav FOP. Skynav FOP was developed by Glenn Adams, who is sponsored by Basis Technologies. Skynav FOP is a branch of Apache FOP though it exists under a separate source repository. It has an Apache license. It offers complex script support for languages like Arabic and Hebrew which is why InterSystems made the switch.

This change requires no change to the administration of FOP. There is no change to configuration settings. Our software acts as before except now supports complex scripts if the font for the complex script is in a supported font, for example, Arabic. This information is obsolete. Current version of Apache FOP supports complex script languages.

Change for SVG Processing

This release of Caché installs revision 1229814 of the batik trunk. This is done to address a defect in version 1.7 of batik that prevents ZEN Report COS Charts and ZEN Page Charts from using the same SVG files.

The RenderServer Now uses Environment Variables

Beginning with this release, runwithxep.bat will now use the environment variables `JAVA_HOME` and `XEP_HOME` in determining location of Java binary files and the XEP library.

Zen Pages Uses CSS Level 3

This release modifies the base Zen page class so that the CSS level defaults to 3 for any browser whose user agent contains Mozilla/5.0 or above (basically, any modern browser including IE9 and above; false for IE8 and below. This tells the browser the page uses HTML5. Some pages may not work well in HTML5 mode (usually due to invalid CSS values). Developers should either correct the pages or set cssLevel to 2 as described in this section.

If the application wants to force CSS level to 2 (because of incompatabilities), it can override the **%OnDetermineCSSLevel** method for the page and return 2, or change the cssLevel globally by setting the *^%ISC.ZEN.cssLevel* global to 2.

GenerateReport Produces HTML5

GenerateReport now generates HTML files that are HTML5 compliant. This may lead to problems in displaying the results of GenerateReport on a non-HTML5 compliant browser such as IE 8.

Note: To generate an HTML file that opens in an HTML 5 browser, the file extension must be ".xhtml" not ".htm" or ".html".

Element bidioverride Changes

In prior releases, Zen Reports generated bidioverride as a div for HTML. In this version, it generates a fo:bidi-override instead. The direction attribute of Zen Reports element, bidioverride, sets the *dir* attribute of fo:bidi-override. The *unicode-bidi* attribute of bidioverride is ignored.

Elements <write> and <inline> Do Not Insert Spaces

This release corrects the processing of the <write> and <inline> elements so that they no longer emit extra spaces. Application that previously relied on this behavior will need to be modified.

Section Now Inherits Writing-Mode From Containing Report

The fo:page-sequence generated by a section now inherits the writing-mode from the containing report if the writing-mode is not specified for the section.

Remove Orientation Attribute

The “orientation” attribute (used to set portrait or landscape mode) was not used in a meaningful way in previous releases. If report definitions are using orientation within pageheader or pagefooter, the developer should remove it. The report should have no change in its visual appearance.

10

Caché 2012.1

This chapter provides the following information for Caché 2012.1:

10.1 New and Enhanced Features for Caché 2012.1

The following major new features have been added to Caché for the 2012.1 release:

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in "Caché 2012.1 Upgrade Checklist."

10.1.1 Rapid Application Development

10.1.1.1 iKnow Technology

iKnow is a new technology addition to Caché that considerably enriches the ability of applications to analyze, handle, and use unstructured (textual) data. Without needing any predetermined expertise or knowledge about the data, iKnow automatically discovers the most important information locked in your unstructured data and opens it up for automated interpretation and exploitation.

Classic approaches to unstructured data analysis typically use keyword-based searching to match possible word groups to pre-built dictionaries and language models. Keeping this data current with the text requires continued, and often intensive, effort and ongoing maintenance.

The vision behind the iKnow approach is that unstructured data is composed of two different types of elements: concepts and relationships (expressing links between concepts). iKnow automatically discovers this information in your unstructured data and opens it up for interpretation and exploitation by end users, business intelligence analysis, and business processes. It can be used for any situation where there is the need to automatically transform unstructured data into structured views of concepts and how they relate topics to one another.

The iKnow technology is accessed through an easy-to-use interface that consists of a set of system classes defined in the %iKnow package. The capabilities are exposed in three different ways: as a set of Objectscript methods, a set of SQL stored procedures, and a set of web services.

10.1.2 Performance And Scalability

10.1.2.1 Improvements To Stream Performance

The %Library.GlobalBinaryStream, %Library.GlobalCharacterStream, %Stream.GlobalBinary, and %Stream.GlobalCharacter stream implementations have been optimized for performance by reducing the amount of data copied during certain operations, and by leveraging CACHETEMP-based storage more effectively. In benchmarking heavy stream-based activity, the new stream implementation shows up to a factor of 1.5 improvement over previous versions.

10.1.3 Reliability, Availability, Maintainability, Monitoring

10.1.3.1 Manage ZEN Report Render Servers

This version introduces a new background process (external to Caché) that will be triggered automatically if you are generating Zen Report PDFs. This process instantiates a Java Virtual Machine (JVM) that will run FOP, the PDF generator from Apache. Caché will launch this process when a request comes in for a PDF based report, and send it information about the material to be processed.

The process will remain running in memory until it is explicitly terminated. More than one server process may be in operation at the same time. Caché will configure the process(es) using the settings defined in the Management Portal for the Zen Report Render Server. An administrator can specify what port each renderer will listen for requests on, what port(s) heartbeat monitor will keep track of, and logging details. If Caché detects a process crash, or a failure in the heartbeat monitor, then it will automatically relaunch this process.

Each server process is configured from the Management Portal parameters available at the time of its launch.

10.1.3.2 System Monitor

The Caché System Monitor functions as a Multivariate Process Control System for monitoring a Caché system and alerting when it is not running within the statistical boundaries of a “standard” system. The definition of “standard” is defined for each instance based on the application and user workflows in a given Caché environment. Deviations from the norm

are measured using the WECO (Western Electric COmpany) statistical probability rules. The System Monitor provides reporting and alerting based on outlier and non-standard events as defined by the WECO rules.

10.1.3.3 Task Manager Improvements

This release adds a number of improvements to support email notifications, and consistency adjustments. Customers can now specify a port number for the SMTP connection, and use an API to programmatically access task manager information.

10.1.4 Security

10.1.4.1 CSP Gateway To Caché Over SSL

With this release, applications may now request a secure connection between the CSP Gateway and the Caché instance it connects to. This adds an important security layer for connections where the CSP Gateway does not reside on the same machine as the Caché instance.

10.1.4.2 Web Services - Secure Conversation

Many Web services applications accommodate frequent communication between the service and client. When this communication needs to be secured end-to-end using WS-Security, the application encounters additional overhead because WS-Security uses public key encryption to secure each message separately.

To mitigate this overhead, the Web services community has introduced WS-Secure Conversation. WS-Secure Conversation moves the overhead from securing each message to a single handshake. Once the secure session is established, the service and client enter a secure conversation until the session's expiration.

This release provides support for WS-Secure conversations in Caché.

10.1.5 Other

10.1.5.1 Zen And HTML5

With this release, InterSystems has added logic to allow all Zen pages served from its products to produce HTML5 output. Whether HTML5 output is produced or not is controlled by the setting of a global, *^%ISC.ZEN.cssLevel*.

If the value of this global is set to 3, all Zen pages written by InterSystems that are served to browsers that support it will produce HTML5 output (interpreted in strict rather than quirks mode). If the global is missing or has the value 2, HTML output will be the same as in 2011.1. The default for new installations and upgrades is to retain the previous behavior.

Those applications that wish more fine-grained control over HTML5 production can override the method, **%OnDetermineCSSLevel()** for a page or subclass to return the value 3. Certain pages in the management portals work in this mode already; and all will beginning in 2012.2.

Generating HTML5 will also automatically invoke the native SVG renderer built into Internet Explorer 9, bypassing the Adobe SVG plugin should it be installed.

10.2 Caché 2012.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2012.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2011.1 and 2012.1.

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

10.2.1 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 2012.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

10.2.1.1 Version Interoperability

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

10.2.1.2 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 those in the pages associated with database mirroring and a major change in the look of the Portal itself to increase the granularity of control and provide the ability to define a per-user "favorites" list.

10.2.1.3 Operational Changes

This section details changes that have an effect on the way the system operates.

2KB Databases No Longer Supported

Version 2011.1 noted that support for 2KB databases was to be withdrawn in this release. Existing 2KB databases must be converted to 8KB format prior to the upgrade to this release using **SYS.Database.Copy()**.

Default Minimum GMHEAP Size Is Now 22MB for New Installations

Beginning with this release, the minimum size required for the generic memory heap (*gmheap*) in new installations has been raised from 4MB to 22MB in order to allow loading of the initial iKnow language tables, `en` (English) and `es` (Spanish).

Upon upgrade, the installer will examine the value set for the gmheap:

- If the value set for the gmheap < 21184KB, it will be changed to the greater of the previous value + 16384KB, or to 21184KB, which ever is greater.
- If the value is > 21184KB, the installer will leave the setting untouched. In this case, administrators are assumed to have other reasons for setting the gmheap value higher, and this new memory demand should also be taken into account in their calculations of a new value.

  Also in this case, if the value does not permit the loading of the basic language tables, Caché will produce a warning in cconsole.log and attempt to degrade gracefully.

If applications wish to take advantage of languages other than English and Spanish, the gmheap size must be further increased per language desired. The following table gives the available language model and the amount of the increase. For those, with gmheap sizes above 21184KB, the sizes of English and Spanish tables are also included.

Language Code	Language Name	Memory Needed
de	German	4.5 MB
en	English	3.3 MB
es	Spanish	12 MB
fr	French	4.4 MB
nl	Dutch	3.6 MB

Finally, if a site does not intend to use any of the iKnow capabilities, the gmheap value may be set lower after the upgrade / installation. This will require a restart of Caché to take effect.

Emergency Mode Runs Under the Owner Userid, Not As Root

The emergency mode startup code is now run under the owner userid, not "cacheusr". This prevents unnecessary failures in attempting to initialize certain resources, such as the CSP portal.

Defragmenting Globals Now Requires %Admin_Manage

Defragmenting the globals in a database now requires the **`%Admin_Manage`** role. Previously only write access to the database was required.

^DATABASE Functions Extended to Non-Primary Mirror Members

Beginning with this release, the various **^DATABASE** functions related to returning free space, compaction, truncation and defragmentation can now be run on mirrored databases from all mirror members regardless of their current role. Previously, they could only be run on a node when it was the primary mirror member.

^DATABASE Menu Changed

The **^DATABASE** menu was changed in this version to add the option

```
4)* Mirror Set Name:
```

under the menu item

```
1)      Create a database
```

Sites which drive **^DATABASE** via scripts should take note.

License Enforcement Tightened

In this release, the management of licenses, including license limit enforcement, has been improved.

- Licenses Cannot Be Obtained from the CSP Server Process

  Applications that call **$SYSTEM.License.Login()** from a CSP server process (serving a CSP page request or a SOAP request) will encounter a <FUNCTION> error and must be changed to use the **%CSP.Session login** API.

- Use of Management Portal Consumes a License

  The System Management Portal application now consumes a license. There is now no difference, from a license use perspective, whether you log in to Caché through a terminal connection and run character-oriented management/operation utilities, or you access this functionality through the browser based Management Portal.

  Customers who have sites that run with a license key sized to the exact number of users may encounter license limit exceeded issues if the Management Portal is used, consumes the last license unit, and another user attempts to log in to Caché.

- Remote CSP Connections Not Allowed without a License

Caché now enforces the rule that a Caché instance that does not have a valid license key installed will not serve CSP applications to a remote host. Without a valid license key, CSP applications are now only accessible from a browser on the local machine.

- Task Manager Initiated Processes Consume a License with Web Add-On Key

  Processes started by the task manager will now consume a license as web add-ons.

Make LoginCookies Authentication Mode Fully Independent

Beginning with this release, for applications which allow both unauthenticated and login-cookie authentication modes, the login cookie is now checked before the unauthenticated mode. In prior releases, an application could be logged in as UnknownUser even if a login-cookie was available.

Increase Frame Stack Size on 64-Bit Systems

The frame stack size for 64-bit platforms has been increased in this release to account for increased stack usage in recent releases. It may now be larger than 64KB. (The stack size for 32-bit platforms has already been increased in a previous release.) This will prevent <FRAMESTACK> errors when programs are migrated forward.

This change results in an increase in the memory usage for each partition, depending on the 64-bit platform:

- Unicode: 25,032 bytes
- 8-bit: 19,817 bytes

Note: If your application (or application set) was close to running out of space in previous versions, this additional consumption of memory may cause <STORE> errors. If this happens, the partition size must be increased.

Changes to %Manager Role

In this release, the **%Manager** role now has the **%Admin_Journal:USE** resource added to it.

Managing Tasks Now Requires New Role

Beginning with this release, if a Task is to be created, modified, or run, the resource **%Admin_Task:USE** is required. This resource is now added to the **%Manager** role with **USE** permission so users owning this role can manage tasks.

Note: The **%Operator** role is not granted this resource by default, so users holding this role cannot manage tasks, unless the **%Admin_Task:USE** resource is added to the **%Operator** role.

If an administrator has created roles which were based on the **%Operator** role and tghey were used to access the Operator menu in the Management Portal, the administrator will need to modify those roles to add the **%DB_CACHESYS:RW** privilege to it; and, if they wish to manage tasks from this role, must also add **%Admin_Task:USE**.

DeepSee Resource Additions

This release adds new resources to the system for working with DeepSee, namely:

- **%DeepSee_AnalyzerEdit** - Grants full access to DeepSee Analyzer
- **%DeepSee_Analyzer** - Grants ReadOnly access to DeepSee Analyzer
- **%DeepSee_PortalEdit** - Grants full access to DeepSee User Portal
- **%DeepSee_Portal** - Grants ReadOnly access to DeepSee User Portal
- **%DeepSee_ArchitectEdit** - Grants full access to DeepSee Architect
- **%DeepSee_Architect** - Grants ReadOnly access to DeepSee Architect
- **%DeepSee_Admin** - Grants access to DeeepSee configuration and security settings

%Admin_Task Resource Added

In this version, Caché now defines a new resource **%Admin_Task**. If a task is to be created, modified, or run, the resource **%Admin_Task:USE** is required.

%Admin_Task:USE is now added to the **%Manager** role so users owning this role can manage tasks.

The **%Operator** role is not granted this resource by default; users holding this role cannot manage tasks, unless the **%Admin_Task:USE** resource is added to that role. If a user has created roles which were based on the **%Operator** role and used to access the Operator menu in the Management Portal, they will need to modify that role to add the **%DB_CACHESYS:RW** privilege to it. In addition if they wish to manage tasks, they also need to add **%Admin_Task:USE privilege**.

Lastly, if when creating a task, the "User to run the task as" is modified to someone other than the logged in user, this operation requires the **%Admin_Secure:USE** privilege.

BackupScripts No Longer Distributed

The scripts, “cbackup” and “cbackups”have been removed from UNIX® distributions and installations, including RPM and DMG kits.

Dialog Changes for Key Activation at Startup

It is no longer possible to browse for the encryption key file used for database encryption during Caché instance startup. This enforces two-factor security by separating access to the encryption key file from knowledge of the password needed to activate the key.

Changes in the Treatment of the USER Database on Upgrade and Install

Installation processing will no longer modify the USER database and namespace settings on upgrade, nor will it recreate USER database and namespace definitions if they are not present. A new install will create the USER database and namespace by default on all platforms.

The feature “server_user” is no longer visible in custom installs on Windows, but it is possible to disable the USER database installation in new install Windows by disabling the installation of feature “server_user” using command line, creating a transform, or modifying MSI package.

Changes to CSP Gateway Installation for Apache on UNIX®

UNIX® installation scripts will no longer update the external web server configuration if it was already configured for CSP Gateway. If the web server was not previously configured for CSP Gateway, the new configuration will use the shared modules mechanism rather than the CGI-based mechanism used by default in previous versions. This change affects the regular UNIX® installer, the CSPInstall standalone CSP Gateway installation script, and RPM and DMG installations.

The CSP Gateway installation process will not change a web server configuration when it detects that the web server was already configured for CSP Gateway. The current detection mechanism consists of parsing the Apache configuration file for the string “/csp” in any line not started with a comment character(#). If the installer detects that the web server was previously configured for CSP Gateway, it produces a message stating that any required re-configuration should be done manually. Nonetheless, it still upgrades CSP Gateway binary files as needed.

Configuration File Changes

- STU Parameter Removed

  The STU=1 parameter in the CPF file has been removed. To start the system for maintenance, use the Emergency Startup option.

  Emergency Startup has been enhanced so:

  – Neither TASKMGR, Shadowing, Mirroring, nor (if applicable) Ensemble productions are started.

  – ZSTU,%ZSTART,%ZSTOP,ZSHUTDOW are not run when the system starts or stops.

  – User processes which log in using the emergency id do not run %ZSTART or %ZSTOP.

- 2KB Buffer Allocation Reassigned

  In the [config], section, if space is allocated for 2KB buffers in the globals= declaration, an amount of memory equal to the same number of 8KB buffers will be added to that allocated for 8KB buffer, and the 2KB allocation will be ignored.

  In the [Startup] sections, if the DBSizesAllowed parameter had an entry for 2KB buffers, this will be removed.

- MirrorSet Options Expanded

  In the [MirrorSetMembers] section, the new parameters ConnectTo and MemberType have been added. Each parameter allows for the following values: AgentAddress, AgentPort, ConnectsTo, ECPAddress, GUID, InstanceDirectory, MemberType, MirrorAddress, MirrorSSPort, Reserved.

- Method for Retrieving / Storing Mirror Client Name for SSL Authentication Changed

  The handling of SSL identification fields in mirroring has been revised. Commas no longer need to be converted to slashes, and fields no longer need to be re-ordered for comparison. This change can affect applications trying to use a certificate where the Subject field contains slashes (/) in the data. The DN fields are now base64 encoded when stored in the .cpf file to avoid issues of commas in the data within a comma delimited field.

  Note: This change creates an interoperability problem when there is a mix of failover members, some with and some without this change. The mixed configuration is supported for upgrade purposes and it will function correctly. However, while there is a mixed configuration, it is not possible to re-configure the mirror to use SSL. If the mirror is already configured to use SSL, it will continue to do so. In order to enable SSL, all of the failover members must be running compatible versions.

Add Latin5 And CP1254 Translation Tables to Turkish Locale

This release adds I/O translation tables for Latin5 (ISO-8859-9) and Windows CP1254 to the Turkish Unicode locale (turw).

Stopping Journaling Overrides Freeze on Journal Error

Beginning with this release, stopping journaling overrides the setting that freezes the system in the event of a journal error. That is, on a system set to freeze on journal error (FreezeOnError = 1 in cache.cpf), stopping journaling is allowed and does NOT cause the system to freeze thereafter.

CAUTION: Under this circumstance, by stopping journaling, the site accepts the loss of journal data (and attendant recovery issues in case of failure during this period) as being of less consequence than the loss of the ability to continue processing.

Convert 8-Bit Journals to Unicode Before Using

When upgrading from earlier 8-bit release to a Unicode version of this release, two steps are necessary. First, upgrade the earlier version to an 8-bit version of this release and process any journal files needed. Than, upgrade this 8-bit version to a Unicode version.

Stream Processing Improvements

This version contains significant improvements to how streams are handled by Caché. These include the elimination of unnecessary copying of buffers, better use of CacheTemp, and improvements to the order that stream nodes are saved. The net result of these changes for applications with heavy streams loads is an increase in the number of operations from 1600 streams operations per second to 2400 operations per second.

Web Services Security Speedup

In previous releases, the time needed to generate the security header could represent a sizeable fraction of the time to generate the message as a whole — for the average message, as much as three-quarters of the time. In this version, generation of the header has been cut roughly two-thirds resulting in substantially better throughput.

Changes To ECP Rollback

In previous versions, an ECP rollback would inhibit service until the rollback finished; the time involved depended on the amount of data to be reset. In this version, rollback and normal ECP service can each proceed simultaneously. There is no pause in service.

Change in Form of Generated Queries

In some cases, the DeepSee Analyzer generates MDX queries of a different form than previously. This can affect scorecards based on pivot tables, so that the scorecard is no longer functional and must be reimplemented as described here.

The affected queries are ones for pivot tables that you create as follows:

- Optionally drag and drop any items to **Rows**
- Drag and drop members, levels, or dimensions to **Columns**
- Drag and drop one measure to **Measures**

In the previous release, a pivot table that you created this way had an MDX query like the following example:

```
SELECT NON EMPTY [Outlet].[H1].[Region].Members ON 0,
       NON EMPTY [Product].%TopMembers ON 1
FROM [HoleFoods]
WHERE [Measures].[Amount Sold]
```

A scorecard can display the results of a query of this form.

In this release, a pivot table created in the same way has an MDX query like the following example:

```
SELECT NON EMPTY NONEMPTYCROSSJOIN([Outlet].[H1].[Region].Members,{[Measures].[Amount Sold]}) ON 0,
       NON EMPTY [Product].%TopMembers ON 1
FROM [HoleFoods]
WHERE [Measures].[Amount Sold]
```

Because this query has a crossjoin, it cannot be displayed in a scorecard.

If you have existing scorecards that depend upon a query of the former kind, do the following:

1. Create a KPI that is based on the desired MDX query.
2. Use that KPI as the data source for the scorecard instead of the previously saved pivot table.

Rebuild DeepSee Indices

In 2012.1, there is a change to the indices of the generated dimension tables. This change may cause your queries to stop working unless you rebuild the indices for all the dimension tables. To do so, enter the following command in the Terminal, in each namespace that contains DeepSee cubes:

```
Do ##class(%DeepSee.Utils).%BuildDimensionTableIndices("*")
```

10.2.1.4 Platform-specific Items

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

Mac OS X

- Rename libcache.dylib

  Mac OS X 10.7 (codenamed “Lion”) introduced a new shared library called libcache.dylib. In order to avoid the name conflict, the Caché callin library has been renamed to libisccache.dylib. Existing applications that link with libcache.dylib need to be changed to use libisccache.dylib.

- Kerberos Authentication Key File

Kerberos authentication on macOS 10.7 will not work unless a keytab file containing the key for the Caché service principal is present in the file /etc/krb5.keytab. On all other platforms, and on previous versions of macOS, this file should be <installdir>/mgr/cache.keytab. The error returned from the kerberos library that indicates this condition is:

```
ERROR #956: Kerberos error: GSS-API error acquiring server credentials;
No credentials were supplied, or the credentials were unavailable or inaccessible.
(00070000); unknown mech-code 0 for mech 1 2 840 113554 1 2 2 (00000000)
```

UNIX® — Speed Up Spawning Jobs

When spawning a child process to add a job, files open in the parent need to be closed. In this version, more efficient means are used to do his on Linux, HP-UX and AIX. For situations where many files are open and lots of jobs are spawned, managing the open files can be up to an order of magnitude faster.

OpenVMS — Update Xerces And Xalan For OpenVMS

In this release, the XERCES and XALAN utilities have been upgraded to versions 3.1.1 and 1.11 respectively.

Non-OpenVMS Systems — Large Journal Writes

In this version, the maximum size of a journal write has been increased to 4MB. Measurements indicate this speeds up journaling by a factor of 2 for a streams load on non-SAN disks producing 1.5GB of journal.

10.2.2 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.

10.2.2.1 Naming Conventions For Globals

Beginning with version 2012.1, InterSystems reserves for itself global names that meet the following criteria:

- In *user* databases, all names starting with "^ISC.".
- In CACHETEMP, all names beginning with "^CACHETEMP.ISC.".
- For process-private globals, all name starting with "^||ISC.".

10.2.2.2 Routine Compiler Changes

Check For Too Many Arguments In A Multiple SET

A **SET** command that sets multiple variables to the same value could overflow the argument stack and cause unpredictable errors. The compiler will now correctly count the number of argument stack entries that the command will use and give a compile error if there are more than 255.

10.2.2.3 Routine Changes

Error Stack Now Initialized For Each Error

Prior to this release, the error stack information persisted until $ECODE was set to the null string. This means that it could contain levels for an older error that happened at a deeper stack level, mixed with levels of a newer error that happened at a less deep level. This made it difficult to interpret the stack display for the newer error.

Beginning with 2012.1, any existing error stack information will be cleared when a new error happens, and the new error stack will contain only entries that show the state at the time of the current error.

This change should make error analysis code simpler and avoid confusion that could happen with the previous arrangement. Existing code should not have to change unless it has special provisions to try to compensate for the previous behavior.

Charset Is Now In %RO Export And Used On Import

The **%RO** export utility and **Export^%apiRTN** will now insert into the routine header the charset the routine was output as. On import, from either %RI or **Import^%apiRTN**, this charset information will be used to ensure that the routine imported is identical to the one exported. If the importing system does not have the required translate table, it will report a suitable error message stating this.

If the files to be imported does not have the charset information (for example, because it was createdon an earlier release), the behavior will be the same as in prior releases.

Routine Compilation Now Uses Multiple Jobs

A call to **^%RCOMPIL**, **##class(%Routine).CompileAll**, **Compile^%R** to compile some routines will now spread the work across multiple CPUs when possible. The behavior of the compilation should be identical except for the improved speed.

The previous behavior, compilation only within the current job, can be restored by executing

```
Set ^%SYS("Compile","MultiCompile") = 0
```

Important: When using multicompile, the worker jobs started may run under a different userid than that of the process that initiated the top-level compile. The userid depends on the platform.

- On Windows, the sub-processes run under the userid set for the Caché server.
- On UNIX® platforms, the userid of the initiating process is used for the sub-process.
- OpenVMS, like UNIX®, runs the sub-process under the userid of the process that started it.

This means that if you wish to use multicompile, you must make sure that the files used in the compilation sources and any include files) have permissions that make them accessible to the sub-process doing the lower-level compiles.

Note: If the class compiler is invoked from within a transaction, the multiple-compilation flag will be ignored because the worker jobs would not be in the same transaction as the master process, and if the user tried to roll back the compile it would only roll back whatever the master process had done leaving the classes in an inconsistent state.

Changes To %Routine Read/Write Behavior

In prior releases, if you used %Routine to write *n* lines of a routine, and then read them back in using the **ReadLine** method, the *AtEnd* property would not be set true until you attempted to read line *n+1*. This is changed in this release to set *AtEnd* true when the last line is read.

10.2.2.4 Class Changes

Class Deletions

The following classes were present in version 2011.1 and have been removed in this version:

- %Compiler.LG — EJBRoot
- %Library.EIVX
- %Library.EIVXBench
- %Library.EIVXBenchNew
- %Library.EIVXBenchNewNew
- %Library.EIVXBenchOld

- %Library.EIVXBrowser
- %Library.EIVXDemo
- %Library.EIVXRecreate
- %Library.EIVXSAXHandler
- %Library.EIVXUnitTest
- %Library.EIVXxmlDOM
- %Projection — EJB, EJBFlexible, EJBJBoss, EJBPramati, EJBWebLogic
- %template — soapclientwizard, soapclientwizardns, soapclientwizardout, soapclientwizardpreview, xmlschemawizard, xmlschemawizardclasses, xmlschemawizardns, xmlschemawizardout, xmlschemawizardpreview
- EMS.UI.Component — emsCheckbox, emsDropLabel, emsListBox, emsTagRemoveButton
- Ens.Enterprise — Portal.LogTablePane, MsgContentsPane, Portal.MsgPane, MsgTablePane, Portal.MsgTraceFilterPane, Portal.MsgTracePane, Portal.MsgTraceSVG

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CPT.CalloutCommon	Method	GetConfig, GetConfigFrom
%CPT.CalloutShell	Method	ClearConfigScope, ConfigScopes, SetConfigScope, ShowConfigScope
%CPT.HPT.LoadingState	Method	GetFirstMatchFor
%CPT.HPT.Reader	Method	GetFirstMatchFor
%CPT.Regen.RegenerateSource	Method	FirstGloss, Gloss, NextGloss, WriteGloss, WriteLineToOutput, WriteSpaceIfNeeded, WriteToOutput
%CPT.Regen.RegenerateSource	Property	LastWrittenChar, OutputStream, WriteSpaceNext
%CSP.Daemon	Method	purgeZombies
%CSP.UI.Portal.Config.SQLDataTypes	Method	editItem
%CSP.UI.Portal.EnsembleMonitor	Parameter	APPLICATION
%CSP.UI.Portal.SSLList	Method	editSetting
%Collection.Super	Method	oidDataId, orefDataId
%Dictionary.CompiledInstanceVar	Property	Slot
%IO.LibraryStream	Method	DeleteAttribute, GetAttribute, IsDefinedAttribute, NextAttribute, SetAttribute

Class	Type	Name(s)
%Library.Storage	Method	%SQLAfterDeleteTriggers, %SQLAfterInsertTriggers, %SQLAfterUpdateTriggers, %SQLBeforeDeleteTriggers, %SQLBeforeInsertTriggers, %SQLBeforeUpdateTriggers
%Monitor.Health.AbstractSensor	Method	%OnNew, Log
%Monitor.Health.AbstractSensor	Property	HealthQueue, Logfile
%Monitor.Health.Control	Method	BuildChart, Enabled, sendmsg
%Monitor.Health.Control	Property	MinSampleSize, PeriodDayTime, SavedReading, SavedReadingNo, WaitTime, trace
%Monitor.Health.HealthAlert	Property	StdDev, ValueList
%Monitor.Health.Period	Index	IDKEY
%Monitor.Health.Period	Property	DayMonth, DayWeek, PeriodID
%Monitor.Health.Period	Query	Periods
%Monitor.Health.SystemSensors	Method	GetDatabase, GetOldestTx
%Monitor.Health.SystemSensors	Property	Curtime, DBSizeI, MirrorB
%Monitor.System.Dashboard	Property	Interval
%SAML.Assertion	Method	Validate
%SOAP.Security.Header	Property	WSBodyLength, WSBodyPosition
%SOAP.Security.KeyIdentifier	Method	Validate
%SOAP.Security.Policy	Property	encryptedBody
%SOAP.Security.Reference	Method	Validate
%SOAP.Security.SecurityTokenReference	Method	Validate
%SYS.Journal.Record	Method	GetRealPIDSYSinFilter
%SYS.NLS.Record	Method	InpRelacedGet, InpRelacedSet, OutRelacedGet, OutRelacedSet
%Stream.FileBinary	Parameter	READCHANGED, READLINE, READNODATA, READNORMAL, READNOTCHANGED, WRITE, WRITEAPPEND, WRITEJUSTSAVED, WRITENORMAL
%Stream.GblChrCompress	Method	Flush, OutputToDevice, ReadIntoBuffer, Write
%Stream.GlobalCharacterSearchable	Method	%SaveData, ReadIntoBuffer
%Stream.GlobalCharacter	Parameter	READCHANGED, READNODATA, READNOTCHANGED, WRITE

Class	Type	Name(s)
%Stream.GlobalCharacter	Property	TempNode
%Studio.Project	Method	ItemCountSet
%Studio.Project	Property	ItemCount
%XML.Security.EncryptedData	Method	InitializeForService
%XML.Security.EncryptedData	Property	ElementId
%XML.Security.EncryptedKey	Method	IsBodyEncryptedGet, SetEncryptionMethod
%XML.Security.EncryptedKey	Property	IsBodyEncrypted, X509Credentials
%XML.Security.EncryptedType	Property	EncryptionOptions
%XML.Security.KeyInfoClause	Method	Validate
%XML.Security.KeyInfo	Method	Validate
%XML.Security.KeyValue	Method	Validate
%XML.Security.RSAKeyValue	Method	Validate
%XML.Security.ReferenceList	Property	IsBodyEncrypted
%XML.Security.Signature	Method	XMLBeforeExport
%XML.Security.Signature	Property	Length, Position
%XML.Security.X509Certificate	Method	Validate
%XML.Security.X509DataElement	Method	Validate
%XML.Security.X509Data	Method	Validate
%XML.Security.X509IssuerSerial	Method	Validate
%XML.Security.X509SKI	Method	Validate
%XML.Security.X509SubjectName	Method	Validate
%ZEN.Dialog.finderDialog	Method	ondialogFinish
%ZEN.Portal.selector	Method	GetDropdownContent
Config.config	Property	Wdstrategy, WdstrategyPresent
Ens.BPL.UI.BPLDocument	Method	Delete, GetClassName, GetEditorURL, GetOther, HasExtension, ListClose, ListExecute, ListFetch, Load, TimeStamp
Ens.BPL.UI.BPLDocument	Parameter	DOMAIN
Ens.BPL.UI.BPLDocument	Query	List
Ens.Config.Production	Method	getRoutingRuleDelegates
Ens.Config.Production	Method	getRoutingRuleTransformations
Ens.DTL.UI.DTLDocument	Parameter	DOMAIN
Ens.Enterprise.Portal.MonitorStatus	Method	DrawTitle
Ens.Enterprise.Portal.MonitorStatus	Method	GetQuickLinks

Class	Type	Name(s)
Ens.Enterprise.Portal.MonitorStatus	Parameter	APPLICATION
Ens.Enterprise.Portal.MonitorStatus	Parameter	DOMAIN
Ens.Enterprise.Portal.MsgBankEventLog	Method	BreakUpDescriptionText, BreakUpStackText, CreateDataSet, DrawLocalType, GetAndUseDefaults, GetWhereClause, GiveAdviceString, SaveDefaults, changeRefresh, countReset, enterKey, expandoState, formReset, onAfterPageChange, onSearchHandler, onSelectItem, showTrace, timeout
Ens.Enterprise.Portal.MsgBankEventLog	Property	detailsWidth, pageNumberId, pageSizeId, resultsTableId
Ens.Enterprise.Portal.MsgBankViewer	Method	BreakUpDescriptionText, BreakUpStackText, CreateDataSet, DrawLocalType, GetAndUseDefaults, GetColumnsAndFrom, GetWhereClause, GiveAdviceString, SaveDefaults, changeRefresh, expandoState, formReset, onAfterPageChange, onSearchHandler, onSelectItem, timeout
Ens.Enterprise.Portal.MsgBankViewer	Property	detailsWidth, pageNumberId, pageSizeId, resultsTableId
Ens.Enterprise.Portal.SystemList	Method	DrawTitle, GetQuickLinks
Ens.Enterprise.Portal.SystemList	Parameter	APPLICATION, DOMAIN
Ens.VDoc.SearchTable	Method	BuildIndex, GetExtentSuperclass, IsASub, RemoveIndex, SearchHeader
Ens.VDoc.SearchTable	Parameter	DOCCLASS
SYS.Mirror	Method	ActivatedMirroredDatabase

Method Return Changes

The following methods have different return values in this version of Caché:

- %IO.LibraryStream — ReadLineIntoStream
- %Monitor.Health.Control — ClearConfig
- %Net.POP3 — WalkParts
- %SOAP.MsgDescriptor — InvokeService
- %SOAP.Policy — WriteOneAlternative
- %SOAP.Security.Policy — ApplySendAlternative, ApplySupportingTokens, SignEncryptParts
- %ZEN.Portal.standardPage — DoLogout

Method Signature Changes

The following methods have different signatures in this version of Caché:

Class Name	Method Name(s)
%CPT.HPT.ParseNode	WriteAnnotation
%CPT.Regen.RegenerateSource	%OnNew
%CPT.Tree.Pair	RegenSource
%CSP.Session	endSession
%CSP.UI.Portal.Config.Devices	editItem
%CSP.UI.Portal.Config.ZenReport	SaveData
%CSP.UI.Portal.X509Credential	SaveData
%DeepSee.Component.pivotTable	selectCellRange
%DeepSee.Query.query	%RewriteForCurrentMember, setFunction
%DeepSee.ResultSet	%GetFiltersForCellRange, %GetOrdinalLabel
%DeepSee.Utils	GetMemberTree
%IO.FileStream	reopen
%IO.LibraryStream	%OnNew
%Installer.Installer	CSPApplication
%Library.ClassDefinition	ClassInfoExecute
%Library.EnsembleMgr	createPortal, createPortalApp, loadMessages
%Library.File	NormalizeDirectory, Read
%Library.RegisteredObject	%OnValidateObject
%Library.SyntaxColorReader	%OnNew, FromCode
%Monitor.Health.Chart	%OnNew
%Monitor.Health.Control	CheckAllSensors, ClearCharts
%Monitor.Health.Period	%OnNew
%Net.SSH.SFTP	Get, Put
%SOAP.Addressing.Properties	GetDefaultResponseProperties
%SOAP.Security.Header	AddToken, Perform
%SOAP.Security.Policy	AddSupportingTokens, AnalyzeSamlToken, AnalyzeToken, AnalyzeX509Token, ApplySupportingTokens, GetTokenType, ValidateAsymmetric, ValidateEncryption, ValidateTokenReference, WriteAlternative
%SOAP.WebBase	LogOutput, ProcessSOAPEnvelope
%SOAP.WebService	InvokeMsgClass, WSAddSignatureConfirmation
%SQL.DynamicStatement	referencedObjects
%SYS.Journal.System	GetAlternateDirectory, GetPrimaryDirectory
%SYS.PTools.SQLQuery	NewQuery

Class Name	Method Name(s)
%SYS.ZENReportServer	%ServeTransform
%SYSTEM.OBJ	GetDependencies
%SYSTEM.SQL	TuneTable
%SYSTEM.Security.Users	SSLPeekClientHello
%Standards.AU.eHealth.HI.SignatureContainerType	Perform
%Studio.SourceControl.ISC	P4Cmd
%XML.Security.EncryptedData	ComputeCipherData, Create, CreateFromEncryptedKey, Encrypt
%XML.Security.EncryptedKey	AddReference, CreateX509, Perform
%XML.Security.EncryptedType	InitializeKey
%XML.Security.ReferenceList	InitializeForService, Perform
%XML.Security.Signature	AddReference, ComputeSha1Digest, InitializeValue
%XML.Writer	CanonicalTree, CanonicalTreeInternal, Canonicalize
%ZEN.Component.abstractPage	onServerMethodError
%ZEN.Report.reportPage	%PerformTransform
%cspapp.sec.utilsysapplication	SaveConfig
Ens.BPL.Transform	isProperty
Ens.BPL.UI.BPLDocument	SaveBPLClass
Ens.BusinessService	CallProcessInputAsync, CheckProcessInputAsyncStatus
Ens.Config.Item	GetBusinessType
Ens.Enterprise.MsgBank.TCPService	OnProcessInput
Ens.Enterprise.MsgBankOperation	exportEvents
Ens.Enterprise.Portal.MonitorStatus	GetLocator
Ens.Enterprise.Portal.MsgBankViewer	showTrace
Ens.ScheduleHandler	ValidateScheduleSpec
Ens.Util.LookupTable	%Import
Ens.Util.XML.Reader	ChangeXMLStreamEncoding
SYS.Database	CreateDatabase, Defragment, PackZU27Error
Security.Events	Start
Security.Services	StartStopTerminalDaemon

Server-Only Classes

The following classes have been marked in this release as server-only:

- %MV.Adaptor.xml
- %MV.EnumClass.xml

- %MV.File.xml
- %MV.PropertyParameters.xml
- %MV.SelectList.xml
- %MV.StudioRoutines.xml
- %MV.Verbs.xml

New $SYSTEM.Bit Class

This releases adds a new class, $SYSTEM.Bit with two entry points:

- **StringToBit(str)** converts a string of bits into a $BIT format bitstring.
- **ZBitToBit(str)** converts a legacy DTM-style $ZBIT* string into a $BIT format bitstring.

These methods will give an <ILLEGAL VALUE> error if the argument cannot be converted into a valid $BIT string.

Changes To Platform Names

The platform names returned by $System.License.KeyPlatform() have been modified to provide consistent format. They are:

Value	Old Name	New Name
0	XXX	Invalid Platform ID
19	HP RISC/32-bit	PA-RISC-32(HP UX)
28	Alpha (OpenVMS)	Alpha(OpenVMS)
29	Alpha (UNIX)	Alpha(UNIX)
30	Solaris/SPARC	SPARC-32(Solaris)
32	Intel (Windows NT)	X86-32(Windows)
36	Linux/Intel	X86-32(Linux)
37	IBM PowerPC/32-bit	p-32(AIX)
40	Alpha (Open VMS - DSM)	Alpha (Open VMS - DSM)
41	VAX (Open VMS - DSM)	VAX (Open VMS - DSM)
44	Cache PC	X86-32/64(Linux, OS X, Windows)
45	Solaris (UltraSPARC)	UltraSPARC(Solaris)
46	HP RISC/64-bit	PA-RISC-64(HP UX)
47	Heterogeneous	Heterogeneous
48	IBM PowerPC/64-bit	p-64(AIX)
49	HP-UX/Itanium	Itanium(HP UX)
52	Linux/x86-64	X86-64(Linux)
53	VMS/Itanium	Itanium(OpenVMS)
55	Win64/x86-64	X86-64(Windows)
57	Solaris/x86-64	X86-64(Solaris)
58	Mac OS X/x86-64	X86-64(OS X)

Value	Old Name	New Name
59	(new)	ARM(Android)

All other values are either obsolete or unused.

ExportToStream Now Uses Raw File I/O

When an application calls **$SYSTEM.OBJ.ExportToStream**, Caché exports the items to a file and then construcst a file stream pointing to the file so the caller has the data in a stream. In previous releases, the file stream used was a %FileCharacterStream. This meant the default file character IO table would be in effect when reading the stream back in which, in some circumstances, would corrupt the data in the file. Beginning with this release, Caché uses %FileBinaryStream which will preserve the original format of the data.

%Open OREF Checking Improved

In previous releases, if there were two separate subclasses of %Library.Persistent, ClassA and ClassB, with object IDs OIDA and OIDB, respectively, the following statements would execute without error

```
Set InstanceB = ##class(ClassA).%Open(OIDB)
Set InstanceA = ##class(ClassB).%Open(OIDA)
```

In this release, %Open now checks that the OID matches the class doing the open. In instances where the exact subclass is unknown, developers should use the closest, common superclass, for example:

```
Set InstanceB = ##class(%Library.Persistent).%Open(OIDB)
```

Config.Databases:List Query Changed To Split Out Mirrored Database Info

The Config.Databases:List query has been updated so it no longer generates SYSLOG entries every time it is executed. Config.Databases:List query no longer returns the Mirrored flag nor the MirrorStatus field which indicated whether the database is active or not in the current mirror. A new Config.Databases:MirrorDatabasesList query has been added which returns information about local mirrored databases in the configuration file.

10.2.2.5 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

Removed The "s" System Flag

This release removes the use of the "s" system flag for filtering lists of classes in functions like CompileAll. The functions that return lists of classes that were previously using the "s" system filter will now return classes whether the class has the "system" keyword or not.

This resolves a problem where some customers used the class "system" keyword in order to define the compile order of the classes. Then when they tried to compile their classes, none of the classes with "system" defined were compiled because the compile request did not have the "s" flag. Now all classes will be included regardless of whether the class has the system keyword.

Improvements To Class Dependency Recognition

The class compiler sorts the list of classes to compile into two main types: hard dependencies where one class needs another class to be fully compiled and runnable in order for it to compile (indicated by the keyword DependsOn); and soft dependencies where the other class needs to be compiled as part of the same set, but it does not need to be runnable (such as superclasses).

The dependency resolution code first sorts out the hard dependencies. Then it resolves the soft dependencies in each group. The resolution of these hard dependencies requires that we bring over any soft dependencies into the same group. For example if class A DependsOn class B, then class B must be compiled earlier than class A. However, if class B has superclass C, then C must also go into this earlier group too, otherwise Caché cannot compile B.

The code that resolves the hard dependencies now also recursively checks each item's soft dependencies to ensure that none of these soft dependencies have a hard dependency before it inserts the item into the proper order. Under some rare circumstances, it is possible this change could result in some classes which appeared to compile cleanly in prior releases reporting a compile dependency error. If this occurs, it is because there is a real dependency cycle, but the previous version of the code was not detecting this correctly.

Resolution Of Ambiguous References

In the case where a multidimensional property and a method share the same name, for example,

```
Class Objects.Test Extends %Persistent
{
  Property Uncertain As %String [ MultiDimensional ];

  Method Uncertain(a As %String, b As %String) As %String
  {
     Quit "Uncertain Method Return"
  }
}
```

the ambiguity of the expression "Uncertain(foo, bar)" will depend on context. When the expression is the receiver of a value (left of the equal sign), it will be treated as a multidimensional value, that is, the statements

```
Set obj = ##class(Objects.Test).%New()
Set obj.Uncertain("A", "B") = "Some string"
Write obj.Uncertain("A", "B")
```

will store the value "Some string" in the multidimensional array *Uncertain* of the instance referenced by *obj*.

But when it is the supplier of a value (right of the equals), the same reference will be resolved to the method. Thus, in the example given, the value, "Uncertain Method Return", will be written to the output device.

Note: Though a questionable programming practice because of the ambiguity in usage, the class compiler will not report this as an error.

Re-Enable Checking For Instance References In ClassMethods

If an application makes a reference to a local method, such as,

```
Write ..Method()
```

the class compiler checks that this method exists, and that it is invoked from an instance method, not a classmethod. Similar checks are done for properties. Due to an error in version 2011.1, this checking was disabled. It is now operational again, with the effect that some classes that (incorrectly) compiled successfully in 2011.1 will now fail to compile in this version.

10.2.2.6 Language Binding Changes

Omit Trailing Zeroes In Timestamp Fractional Seconds For Light C++ Binding

The Light C++ Binding no longer stores trailing "0" digits in the fractional seconds component of fields of type, d_timestamp. Furthermore, if the value of the fractional seconds component of a timestamp is zero, then the "." separating seconds from fractional seconds is also not stored. This corrects a problem in which timestamps stored by the Light C++ Binding could not be correctly compared to timestamp constants in SQL.

Rework MDS API To Use A More Java-Friendly Style

The MDS API has been substantially reworked in preparation for making it visible as a primary API for the Globals product. The primary motivations for this are: to reintroduce visibility of the term and concept "global"; to make the style more in line with the expectations of the Java community; to simplify/clarify the interface; and to add ease-of-use enhancements. The MDS API was released in Cache 2010.1, but was used internally in the implementation of XEP, the interface employed by customers.

Refer to the Javadoc in the com.intersys.globals package for detailed documentation of all classes and methods.

Globals API In Caché

With this release, several related areas of functionality in the Globals API not supported in GlobalsDB are made available when used in Caché:

- Thread-specific connections; these are in addition to the default non-thread-specific connections which are the only type supported in GlobalsDB.
- The Globals API and other eXTreme APIs (XDO, eXTreme JDBC) ensure that they do not open incompatible connection types (thread-specific for one API, non-thread-specific for the other) when both APIs are used in the same application in Caché.
- The underlying JNI code throws the correct exception types based on which API it was called from, even when calls to different APIs are interspersed in the same application.

This means that functionality present in MDS (the deprecated precursor of the Globals API) is provided in a manner that maintains independence of Globals packages from other eXTreme packages. This in turn means that the same Globals code can ship independently of other eXTreme packages in globalsdb.jar in the Globals DB product.

Enterprise Java Bean (EJB) Projection No Longer Supported

With this release, support for the Enterprise Java Bean (EJB) has been removed from Caché.

The Python Binding Now Supports The %Decimal Type

This release corrects a oversight in the Python binding by introducing the %Decimal type. It is support for both Python 2.7 and Python 3.0. A decimal is introduced through the Python class intersys.pythonbind.decimal. The constructor is **intersys.pythonbind.decimal(significand, exponent)** and the value of the decimal is significand * 10 ** exponent.

Unicode Is The Default Encoding For d_list

In previous releases, the datatype d_list used the thread locale by default. In certain cases, that resulted in loss of information when adding an element to the list. Now the default is Unicode, and no information is lost.

This change may make some **$LIST** elements become of type 2 (Unicode). this means that binary comparisons for entire lists may now give false negatives since the data is logically the same, but is stored differently, such as comparing **$LISTBUILD("1")** and **$LISTBUILD(1)**.

Java Extreme XEP Changes

The ability to delete an entire extent via a Java method is a powerful tool, especially during the development and testing stages. It can also be extremely dangerous if used without caution. The three changes introduced by this release improves usability and decreases the possibility of abuse:

1. `OPTION_IMPORT` with either of the `OPTION_IMPORT_PRESERVE_EXTENT` and `OPTION_IMPORT_DELETE_EXTENT` qualifier has been removed. This means it is no longer possible to delete parts of the schema during the schema import process. Calling get/setOption(OPTION_IMPORT) will result in an exception. Schemas will be preserved by default, as long as the object model has not changed. If necessary, extents should be deleted using the newly added **EventPersister.deleteExtent** method (see next item).
2. There is a new method in the EventPersister class that is the preferred way to delete an entire schema. It has the declaration

   ```
   void deleteExtent(String eventName) throws XEPException
   ```

3. The **Event.deleteExtent** method has been deprecated. Applications using it should be altered to call **EventPersister.deleteExtent**. **Event.deleteExtent** will be removed in a future release.

All the XEP examples have been changed to use this new approach.

10.2.2.7 SQL Changes

Support For Includes And Macros In CREATE PROC Statements

This release changes the behavior of the DDL CREATE PROCEDURE, CREATE FUNCTION, CREATE METHOD, CREATE QUERY, and CREATE TRIGGER statements when compiled as embedded SQL statements or prepared as dynamic statements. This change is not fully backward-compatible and may require modifications to applications, especially when code bodies of type ObjectScript are used in the CREATE statement.

Consider the following statement:

```
&sql;(CREATE PROCEDURE SquareIt(in value INTEGER) RETURNS INTEGER LANGUAGE COS
     {
        #define Square(%val) %val*%val
        QUIT $$$Square(value)
     }
)
```

Prior to this change the #if, #define and $$$Square macro references would be expanded and processed when the CREATE PROCEDURE *STATEMENT* was compiled. After this change, the processing and expansion will be included in the procedure's method definition, and get processed and expanded when the *METHOD* is compiled.

Multi-Index Optimization Now Uses Indexes For All "Efficient" Equality Conditions

If a query has several equality conditions with good indexes on the same table, they will now always be used if that table is chosen as the first to process. Previously, if one or two provided "good enough" selectivity, additional indexes might be ignored.

Some query features may override this behavior. For example, for an ORDER BY, DISTINCT, or GROUP BY query, the benefit of indexes assisting these operations will be compared to the multi-index strategy, and may supersede it. Similarly, since TOP optimizes for time to first row, it may also override this new default behavior. Finally, producing rows in order to improve performance on a later join may also cause a different query strategy to win.

Changes In Storage Methods Associated With SQLStorage

Internal methods generated by the compiler for classes that use Caché SQLStorage were not properly NEWing some %-variables used by SQL code. These methods will now NEW *%msg*, *%ok*, and *%ROWCOUNT*; *%ROWID* and *SQLCODE* were being properly NEWed.

Applications that make use of the fact that these variables were getting "left behind" will have to be changed.

Trigger Definition Enhancements

Changes have been made to SQL triggers to make {field} references perform better now that Caché supports trigger events of multiple types. The code generated by a trigger will now only be emitted one time, even if the trigger had more than one event. In order to make this change possible, the rules regarding the behavior of {Field*Flag} references are now as follows:

- {Field}:
  - For INSERT, resolves to the value inserted for the column.
  - For DELETE, resolves to the on-disk, pre-delete value for the column.
  - For UPDATE, resolves to the new value the column is being updated to.
- {Field*N}:
  - For INSERT, resolves to the value inserted for the column.
  - For DELETE, resolves to the on-disk value for the column from the row being deleted.
  - For UPDATE, resolves to the new value being updated for the column in this row.
- {Field*O}:

  - For INSERT, resolves to NULL. This is new behavior.
  - For DELETE, resolves to the on-disk value for the column from the row being deleted.
  - For UPDATE, resolves to the old value for the column being updated.

- {Field*C}:
  - For INSERT, TRUE if the inserted value is non-NULL, otherwise FALSE. This is new behavior.
  - For DELETE, TRUE if the value being deleted is non-NULL, otherwise FALSE. This is new behavior.
  - For UPDATE, TRUE if the new value differs from the old/on-disk value, otherwise FALSE.

Furthermore, the following also applies to Trigger code: Triggers may contain line labels. However, the trigger code is generated outside the scope of any procedure blocks. This means the label must be unique in the class definition. Any other code compiled into the class must not have the same label defined, including code in other triggers, non-procedureblock methods, SqlCompute code, and so on.

TSQL: UPDATE STATISTICS

The Caché TSQL compiler now supports the UPDATE STATISTICS statement; it generates a call to $SYSTEM.SQL.TuneTable() for the it. This statement is also supported as a dynamic statement.

Change Processing Of Non-%ObjectSelectMode Select Statements

The result of executing a dynamic SQL statement is a result object. This result object may be a simple result (indicating success or failure), a result set object, or a procedure result object. If the statement was prepared with *%ObjectSelectMode* = 1, the result set columns can be automatically swizzled when referencing the row type properties if the type of the column corresponding to the property is a swizzleable type. For example,

```
SELECT Spouse from Sample.Person where Spouse IS NOT NULL
```

If that statement were prepared as a dynamic SQL statement with *%ObjectSelectMode* = 1, retrieving the <resultORef>Spouse property will trigger swizzling and an oref will be returned. However, if %ObjectSelectMode is 0 then the property will be returned as a simple ID value.

For streams, SQL will return a fully qualified Stream OID (SOID) value. This is true for embedded SQL and dynamic SQL when *%ObjectSelectMode* is 0. The fully qualified SOID value can be swizzled manually using **##class(%Stream.Object).%Open(<SOID>)**. Stream columns selected by executing a dynamic SQL statement prepared with %ObjectSelectMode = 1 will automatically swizzle.

An error in dynamic SQL caused stream columns to automatically swizzle when %ObjectSelectMode was not 1. This has now been fixed. Applications that took advantage of this behavior will have to change their processing to use **%OBJECT(streamcol)** or to manually swizzle the stream SOID using **%StreamObject.%Open**.

Dynamic SQL %Print Changes

The **%Print** utility method implemented for result sets will now quote the output value if the value contains a line feed. Before this, the value would only be quoted if the value contained the delimiter character.

Changes To SQL DECODE Statement

A change has been made to the SQL **DECODE** function. Prior to this change, **DECODE** returned the same type as the first return argument in the **DECODE** function call. It has been changed in this release to return the most compatible type of all the potential return values from the function.

Consider the following function call:

```
DECODE(999, ' ', 1, 0.7)
```

Caché considered the return type of this to be INTEGER. When .7 was returned as the value through the xDBC driver, it was truncated to 0 because the type was supposed to be INTEGER. (By contrast, Oracle does not consider the value 1 to be an INTEGER, but rather FLOAT(38) which allows the .7 to be returned to the client intact.)

Now, the possible return values in the function call above are 1 and .7, so in this case the function would return NUMERIC(1,1) The following data types are compatible and are specified in order of precedence (highest to lowest): VARCHAR, DOUBLE, NUMERIC, BIGINT, INTEGER, SMALLINT, TINYINT. A **DECODE** function call that could return a NUMERIC, INTEGER, or TINYINT; the return type will be NUMERIC because it has the highest precedence of the three types. Due to the type given to literal replaced arguments in xDBC queries, if the above function was included in an xDBC query column, the type reported would be NUMERIC(18,9).

Host Variables Disallowed In The View Of A Query

Host Variables have never been allowed in the query of a view. Their attempted use sometimes caused unexpected and non-descriptive errors like, *SQLCODE* = -51, when attempting to compile these statements. If you attempt to reference a host variable in a view's query, or in a CREATE VIEW or ALTER VIEW statement, you will now get an *SQLCODE* = -148 and an appropriate error message in %msg.

Changes To ROWCOUNT In TSQL

A problem has been corrected with the TSQL SET ROWCOUNT feature. If any application relies on the current behavior of *ROWCOUNT* in Caché TSQL, it will have to be modified.

For TSQL code such as:

```
ClassMethod RowCountTest() As %Integer [ Language = tsql,
                                         ReturnResultsets,
                                         SqlName = RowCountTest,
                                         SqlProc ]
{
    set ROWCOUNT 1
    select * from Cinema.Film
    set ROWCOUNT 0
}
```

In previous releases, the result set returned by executing the procedure would return all the rows. Now it returns only the first one.

%IGNOREINDICES Changes

The deprecated keyword `%IGNOREINDICES` now gives a syntax error if the index name given is non-existent. Since `%IGNOREINDICES` is deprecated, all occurences should be changed to `%IGNOREINDEX`.

10.2.2.8 CSP Changes

Improved HyperEvent Error Processing

Developers making use of the **OnErrorSetup()** method of %CSP.Login to handle issues with hyperevent requests should note that hyperevents error handling is now performed by the CSP server, and they should make use of the client-side error processing if they wish to take action based on these error conditions. Errors which occur during HyperEvents now capture more information about the cause of the error and allow users the ability of making more informed decisions about actions they should take at an application level.

This release introduces a new Javascript object of type cspHyperEventError which has the following properties with the listed values:

- code: corresponds to an HTTP response code OR a response code from the XMLHttpRequest object in use. XMLHttpRequest codes may be browser-specific.
- text: a free text field which corresponds to the current text returned to the cspRunServerMethodError() callback function.
- serverCode: corresponds to the error number on the server, if available. This value may be null.
- serverText: the error message from the server, if available. This value defaults to the empty string, that is “ ”.

- exception: an exception which triggered the error. This value may be null.
- arguments: the list of arguments to the function where an exception was trapped. This value may be null and will only be populated if exception is defined.

When an error occurs during a HyperEvent call, including server-side exceptions to Zen methods, the new **cspHyperEventErrorHandler()** function in cspxmlhttp.js will check whether the user has defined the **cspRunServerMethodError()** callback for handling hyperevent errors. If so, **cspRunServerMethodError()** will be called with an additional argument containing a *cspHyperEventError* object. If the **cspRunServerMethodError()** function has not been defined for the page, the default error handling function will log an alert as it always has.

The behavior for Zen is slightly different. Zen always defines the **cspRunServerMethodError()** callback. Zen uses the **onServerMethodError()** callback function inherited from %ZEN.Component.abstractPage to allow users the option of handling hyperevent errors. This callback function will also be supplied a second argument containing the enhanced *cspHyperEventError* object so Zen developers can actively control how hyperevent errors should be handled. Server-side exceptions and errors will now be reported as hyperevent errors and will be reported as HTTP 500 responses. Those being triggered by status errors will include the server-side error code and text. If the **onServerMethodError()** callback is not implemented, the Zen code will check whether the error indicates the user has been logged out and could not be logged in automatically - this check is based on an HTTP code of 401 and a server code of 864. If this is the case, the Zen framework will trigger a page refresh using the following Javascript:

```
self.document.location.reload()
```

This will present the user with the login page specified for the current CSP application; the user will not be able to perform any further activities without logging in again. If applications have implemented the **onServerMethodError()** callback, they should take note that a return value of false from this method will still check for whether the page should be refreshed. An undefined return value or a return value of true will bypass any further Zen processing of the error. Note that many more Zen errors from the server will now be reported through these callbacks, and application developers should be aware of the increase in errors reported this way.

As an extension of these changes, the CSP server will no longer attempt to redirect requests for %CSP.Broker to a login page in situations where

- previous authentication information (such as for a group-by-id CSP application group or a sticky context) is not available,
- a username has not been supplied, and
- unauthenticated access is not permitted for the current application.

This kind of situation is relatively common, and would occur when a session has expired and the hyperevent includes no authentication. If this does occur, then the CSP server will NOT redirect to the login page, but will report an HTTP 401 error to the client and supply a server code of 864 - authentication required. If a login is attempted for the HyperEvent (which is uncommon), the login failure will NOT be directed via the **OnErrorSetup()** method of the login page, but will be reported directly to the client as an HTTP 401 error with the error code encountered on the server.

10.2.2.9 Web Services Changes

Allow 200 As Valid Return Value

In prior releases, Caché accepted only 202 as a return from one-way operations. According to WS-I Basic Profile 1.1, it is perfectly OK to return 200 from one-way operations, so this is now allowed.

10.2.2.10 xDBC Changes

Change Handling Strings Containing NULs

In theory, data returned from Caché and reported as a SQL_VARCHAR and converted to SQL_C_CHAR would be null terminated, and would not contain internal null values. Caché strings, however, can contain NULs and this caused problems for an ODBC application that tries to perform SQLGetData on the value. Previous releases would reported the data truncated when Caché detected the NUL before the full length of the data. The ODBC application usually responded to the truncation error by trying to get the value over and over again, unsuccessfully, in a loop.

With this release, Caché converts and returns all the data to SQL_C_CHAR, even if it contains embedded $CHAR(0). While this is different from what you would expect for a normal null terminated C string, it is now left up to the ODBC application to decide how to deal with this corrupted data.

In the case of SQL_BINARY data, it will be converted to SQL_C_CHAR that contains $CHAR(0) characters and can be fully read and converted to multibyte values since Caché will not return falsely truncation errors that suggest a bigger buffer is needed.

Note: This change affects string output data converted from Caché. Input strings in ODBC are generally null-terminated, and no change has been made to how they are handled.

Return Username On UNIX® And OpenVMS Platforms

In prior versions, Caché returned UID value of the user on UNIX® and OpenVMS for $USER. Beginning with this version, $USER returns the username.

10.2.2.11 MultiValue Changes

Corrections To SYSTEM(1001), SYSTEM(1051) And SYSTEM(1056) Functions

SYSTEM(1001) is now emulation dependent. For a program compiled with jBASE emulation, it returns the command line as an attribute delimited string. In all other emulations, it will return the emulation number that the program was compiled with (as it currently does).

SYSTEM(1051) continues to returns the emulation number that the program was compiled with as it has in previous releases.

SYSTEM(1056) is new and returns 4 attributes describing the emulation as follows:

1. The emulation number that the program was compiled with.
2. The emulation name that the program was compiled with.
3. The emulation number that currently exists for the MV account.
4. The emulation name that currently exists for the MV account.

Change R95 emulation name to POWER95

Any references to R95 as an emulation should be replaced with POWER95.

Support Unidata-style Global Catalog Verbs

Now, if the first word of an MV command starts with "*" and the current emulation is "UNIDATA" or "UDPICK", then the "*" is removed and the global catalog is accessed for that verb.

Can Now Set Port Numbers For Phantom Processes

Starting with this version, phantom processes will now use the custom port number assigned in ^||*%MVPortNo*.

COPY.FILE Now Updates @ID

The destination of a **COPY.FILE** command will now have its *@ID* dict item updated to reflect the new filename.

MV Trigger Wrapper Name Changed

The MV trigger wrapper routine is now named by taking the name of the global that holds the data section of the file and prefixing it with "MVTW".

Projected Storage Changes

Beginning with this release, an index on a collection in an MVENABLED class will now project as full; previously it had been marked as conditional. Furthermore, the child table projected from a list collection in an MVENABLED class will now include one empty row if the collection (multivalue) is null.

MVBASIC Changes

- MVBasic Defaults For Missing Exponent Term

  The MultiValue BASIC **PWRS(A,B)** built-in function and the vector ** operator (same as the vector ^ operator) have been modified. If any component of the dynamic array B is missing, or is the empty string, then that component is treated as the value 1.

  Note: This behavior is different from the **DIVS(A,B)** and **MODS(A,B)** built-in functions and the vector / operator. These vector operations treat a missing component in the second operand as a 1, but they treat a component containing the empty string as a 0.

- DIVSZ Function Added To MVBASIC

  There is a new dynamic array function, **DIVSZ**, in MV BASIC. The **DIVSZ** function is identical to the **DIVS** function except that it does not generate an error when an a dynamic array component is divided by zero. If a component of the second operand of **DIVSZ** has the value 0 then the corresponding component of the result dynamic array will be 0.

  Note: Applications that define a user-written routine or variable using the name **DIVSZ** will need to rename the routine or identifier.

- MVB System Variables Not Allowed When Descriptor Is Required

  In MVBASIC, calling **$DATA(@ME)** or **$GET(@ME)** would cause the compiler to crash. Applying **$DATA()** and **$GET()** to any other system variable (for example, **$DATA(@ACCOUNT))** would always return a result as if the system variable were undefined. System variables are not implemented as variables but are instead special cased and are usually implemented as expressions. They cannot used as arguments to functions or statements that require a variable (as opposed to requiring an expressions value.) The compiler now detects these situations and gives appropriate compile-time errors messages.

  Note: It is still legal to write calls such as **$DATA(@ME->Property)** because a class property can usually be referenced anywhere a variable may be referenced.

- Disallow Nested Dynamic Array References

  In previous releases, the MVBASIC compiler did not give a syntax error when the program contained a nested dynamic array reference such as VAR<i><k,j>. Trying to compile this illegal syntax would often lead to a crash of the MVBASIC compiler. If the compilation were successful, executing the program would give unpredictable results. This syntax is no longer accepted.

  Where the MVBASIC syntax requires a variable reference (such as a left-hand-side of an assignment, or the first operand of a READ statement), the variable reference can contain at most one dynamic array reference (E.g., A<i,j>) followed by at most one substring reference (A<i,j>[s,l]). These rules were not enforced previously and variable references containing multiple dynamic array references and/or multiple substring references would often produce erroneous object code.

  It is permissible to use multiple dynamic array references and/or substring references when that variable was being evaluated as part of an expression, for example,

```
LET X = A<I><1,J><1,1,K>[s1,l1][s2,l2]
```

Previously, a sequence of more than two dynamic array references inside an expression evaluation may have produced a syntax error.

- IFS() Function Corrected And REUSE() Allowed

  The function **IFS(ifarray, array1, array2)** takes 3 dynamic arrays as parameters and it returns a dynamic array result. The argument *ifarray* contains boolean values; the shape of the function result is identical to the shape of *ifarray*. Where *ifarray* contains a TRUE component, the corresponding component of the result is the corresponding component of *array1*. Where *ifarray* contains a FALSE component, the corresponding component of the result is the corresponding component of *array2*.

  If the selected component of *array1* or *array2* does not exist and **REUSE()** is not used on the selected argument, then the result component is the empty string. If the selected corresponding component of *array1* or *array2* does not exist but REUSE() is used on that argument, then the result component is the closest, existing *array1* or *array2* component preceding the missing component position.

  It is an error if REUSE() is applied to the first argument, *ifarray*.

  Note: If an application calls **IFS()** using arrays with mismatched shapes, then there is the possibility the results will be different among different MultiValue vendors. The application may need to be altered to require that parameters to **IFS()** to be reshaped in order to get the desired and consistent behavior.

- MVB Ultimate Defaults To -FULL.LOGICAL.EVALUATION

  The default behavior for short-circuit Boolean evaluation for Ultimate emulation is now

  ```
  $OPTIONS -FULL.LOGICAL.EVALUATION
  ```

  Previously, it was

  ```
  $OPTIONS  FULL.LOGICAL.EVALUATION
  ```

  The new behavior will not evaluate the *B* operand of *A* & *B* (*A* AND *B*) when the *A* operand evaluates to FALSE and it will not evaluate the *B* operand of *A* ! *B* (*A* OR *B*) when the *A* operand evaluates to TRUE.

- MVB OPENSEQ Requires A TO Clause

  The OPENSEQ statement requires that there be a TO clause. This restriction is now correctly enforced.

10.2.2.12 Zen Changes

Improved HyperEvent Error Processing

Please refer to the changes detailed in the CSP HyperEvent section.

Add Flag To Selectively Enable Toggle Behavior In tablePane Row Select

A previous enhancement enabled the user to unselect the current row by clicking on it, making the row click option effectively a toggle and allowing the user to 'select nothing' without having to reload the page. This was a departure from the prior behavior where subsequent clicks on the selected row were ignored and the only way to clear the current selection was to either select something else or go through the API. This change adds the flag enableToggleSelect to tablePane, effectively making the previous enhancement optional. The default is to mimic the pre-2010.2 behavior in the name of backwards compatibility.

Existing early adopters of the newer behavior will need to add a enableToggleSelect="true" attribute to any tablePane tag where unselecting is desired.

10.2.2.13 Zen Reports Changes

PageHeaders In PDF Generation

In PDF generation, if a report with multiple sections contains a top-level pageheader, that pageheader will not appear in the output. For a different first page when using multiple sections, specify a masterreference with pagePosition="first". In addition, only a masterreference with pagePosition="any" will be included in the HTML output.

10.2.2.14 ActiveX Changes

Support For ITypeInfo In CObjInstance Removed

In prior versions, CObjInstance returned the default ATL ITypeInfo for the class. This was incorrect because the actual interface for CObjInstance is much more than what the default implementation returns. Removing it means that CacheActiveX clients will not make decisions based on erroneous meta- information for CObjInstance.

11

Caché 2011.1

This chapter provides the following information for Caché 2011.1:

11.1 New and Enhanced Features for Caché 2011.1

The following major new features have been added to Caché for the 2011.1 release:

- Managed Encryption Keys

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

11.1.1 Rapid Application Development

11.1.1.1 Multiple Session Callback Events

In previous versions, the only user-defined Session Events triggered at the beginning, end or timeout of a CSP session occurred based on the last CSP application accessed by that user. In this version, this behavior has changed. Caché will now execute the SessionEvent logic for the current CSP application, plus the most recently accessed CSP Application that was used by this session prior to the event, if more than one application was accessed.

11.1.1.2 WebStress Testing Facility

This version introduces a new core utility called WebStress. InterSystems has used this tool in prior releases to record, randomize and playback HTTP-based scripts against various applications for the purpose of QA, scalability, and network load testing. The tool runs on a Caché or Ensemble system on any supported platform and can test any web-based application. It includes additional hooks required for correctly benchmarking CSP- and Zen-based applications making use of hyperevents. For recording scripts, users must employ a supported browser and the ability to define a proxy server.

11.1.1.3 New DeepSee Implementation

This release of Caché introduces a new version of DeepSee (previously referred to as "DeepSee II") with the following improvements:

- Data Modeling

  Data modeling has been simplified. This version uses Caché classes that reference application transactional classes. Therefore, there is no need to modify application classes before use as in the preceding version. DeepSee models are defined via these reference classes and can be edited using the DeepSee Architect or Studio. Furthermore, data models now support many MDX concepts including multi-level hierarchies.

- Query Engine

  This version of DeepSee uses the MDX query language for all queries. Its query engine has been optimized to support parallel query execution which takes advantage of the power of multi-core architectures. Multi-level result caching improves query performance by retaining the results of queries so the results can be used when the queries are run again.

- User Interface

  Built with the InterSystems Zen technology, the new DeepSee user interface supports multiple browsers including IE, FireFox, and Chrome. Control of the user interface is done via the DeepSee option on the Management Portal. The options include: Architect for creating DeepSee data models, Analyzer for exploring the data, User Portal for creating and viewing dashboards.

11.1.2 Performance And Scalability

11.1.2.1 Improved Class Compiler Performance

As a result of changes we have introduced in previous versions, and by moving performance-critical components into the system level, InterSystems has noticeably improved the performance of the class compiler.

11.1.2.2 Compilation Using Multiple Jobs

In addition to the gains from the Improved Class Compiler Performance in this release, Caché can now be directed to use multiple processes for the compilation of classes and the import of XML files. The number of jobs started will depend on licensed CPU cores and upon observed efficiency (more than 16 gains no added advantage).

11.1.2.3 Support For Large Routines And Classes

Large Routines

In prior releases, the maximum size of a routine was 64KB. Starting with this release, the maximum routine size has been extended to 8MB. For routines larger than 32KB, Caché will use up to 128 64KB routine buffers to hold the routine. These buffers will be allocated and managed as a unit.

The class compiler and the SQL processor have been changed to use this new limit. Customers can take advantage of this improvement merely by recompiling.

Large Classes

Beginning with this release the system now supports a larger class descriptor. Among the consequences are that classes now can contain a larger number of members to be declared in the class. The limits on class inheritance depth, and the number of superclasses allowed have also been defined. For a complete list of the applicable bounds, see "Guide General System Limits" in the *Caché Programming Orientation Guide*.

Please consult the Caché 2011.1 Upgrade Checklist for further information.

11.1.2.4 Journaling Additions

This version of Caché now provides an API for journal restore. See the Journal.Restore class for information on how to use it. In addition, the process Id (PID) is now once again part of each journal record; journal restore in this release has been changed to deal with this difference in format across release boundaries.

11.1.3 Reliability, Availability, Maintainability, Monitoring

11.1.3.1 Management Portal Improvements

The Management Portal now provides access to all functions using one interface, including DeepSee and Ensemble (for Ensemble installations). By providing a new path to each of the functional components, there is now a mechanism to specify access control on each navigational option and granular control for security-critical operations. In addition, users can now specify the most commonly used areas as "favorites" for even faster navigation.

11.1.3.2 Mirroring Enhancements

In this release, several enhancements have been added to mirroring:

- Asynchronous mirror members now purge mirror journal files that have been applied locally and are no longer needed
- The mirroring communication / data transfer process has been optimized for performance by sending larger chunks of data from the primary to the backup failover member
- The mirror Virtual IP (VIP) now supports IPv6 addresses

11.1.3.3 Caché Monitor History Database

The Caché Monitor History Database introduces a facility to capture and analyze historical system statistics such as performance metrics, and errors reported. It supplies a baseline for analyzing performance anomalies and provides historical data to facilitate capacity planning.

A default set of metrics is defined and the schedule for capturing these metrics can be defined by the user. These metrics are fully SQL-enabled, and an API is provided to query the results stored in the database.

11.1.4 Security

11.1.4.1 Web Services Control Separate From CSP Control

In this release, for each web application (formerly known as CSP application), users can specify if CSP and or Web Service access is enabled as part of the web application definition.

11.1.4.2 Two-Factor Authentication For CSP

In this version, InterSystems has broadened the use of two-factor authentication, introduced with 2010.2, to also be used with CSP applications. If enabled, users will, after successful authentication, be challenged to enter an additional code, which has been separately transmitted to their mobile device.

11.1.4.3 Web Service Licensing

Beginning with this release, Caché will now consume a license unit for anonymous connections, and will hold this license unit for a grace period of ten seconds, or for the duration of the web service connection, whichever is longer.

Web service connection with named users (login), already consumed a license unit, and there is no change for these type of connections.

11.1.4.4 Managed Encryption Keys

Based on the existing Caché implementation of data-at-rest (for example, database encryption) keys, this release enables application developers to use the same strong keys, and key management capabilities on more granular data. Managed encryption keys are loaded into memory and applications refer to these keys via a unique key identifier, therefore protecting access to the key itself.

The system is designed to load four keys into protected memory. In addition, Caché now provides a new encryption function which will embed the key identifier in the resulting cipher text. This enables the system to automatically identify the corresponding key, and allows application developers to design re-encryption methods, which are completely transparent to the application, without causing any down time.

This new mechanism is designed to encrypt special data elements (such as credit card numbers), and may or may not be used in conjunction with database encryption.

11.2 Caché 2011.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2011.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2010.1 and 2011.1.

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

11.2.1 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 2011.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

11.2.1.1 Version Interoperability

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

11.2.1.2 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 addition of pages to assist with database mirroring.

11.2.1.3 Packaging Changes And License Keys

The new product packaging announced in 2011 by InterSystems offers more capabilities and features for your current license types. Previously issued license keys will continue work with 2011.1. You do not require a new license key if your application uses only the capabilities offered by the previous product terms and conditions.

If, however, you wish to take advantage of the additional capabilities available for your license type, please contact your InterSystems sales representative to obtain a new, equivalent license key.

11.2.1.4 Operational Changes

This section details changes that have an effect on the way the system operates.

Licensing

In prior versions, license units were not consumed for CSP sessions when $USERNAME equalled "UnknownUser". Now, if the Caché license key has the Web Add-On feature enabled, it is necessary to explicitly declare the Web application public in order to avoid consuming a license key; otherwise, CSP sessions and SOAP sessions consume a license unit.

This can be accomplished for CSP applications by creating a subclass of the %CSP.SessionEvents class and defining a method to handle the **OnStartSession** event, and invoking **$SYSTEM.License.PublicWebAppUser()** from it.

Furthermore, anonymous SOAP requests (those where no Caché login occurs) now consume a license unit for a minimum of 10 seconds. Applications do not require modification, but customers may need to purchase additional licenses if they service SOAP requests from Caché.

Calling **$SYSTEM.License.Login(LicenseId)** from a CSP server process or from a SOAP Web Service process will consume a license unit for the Caché server process inappropriately. This license unit is associated with the CSP or SOAP session rather than with the server process because subsequent requests may be fulfilled by different Caché processes.

The $SYSTEM.License.Login(LicenseId) API consumes a license unit that is associated with the Caché server process. Calling this API from a CSP or SOAP server process effectively creates two license instances, one for the session and one for the process. The process license instance is not released unless the process exits, which may never happen. The appropriate way to explicitly designate the license identifier for a CSP session is by calling the **%CSP.Session:Login** method.

2K Databases Mounted ReadOnly

Beginning with this release, 2K databases will no longer be mounted as writable by Caché. This is the next step in the announced removal of support for 2K databases. If you wish to write data to such a database, you must convert it to the 8K format. The method **##class(SYS.Database).Copy()** can be used to convert a database to the larger format.

Changes To Configuration File

Two parameters have been added to the .cpf file:

- The LibPath parameter is added to the `[config]` section.

  It is used for Unix® only and sets the LD_LIBRARY_PATH environment variable used to search for third-party shared libraries. It is ignored on Windows and OpenVMS. It is a string property with no required or maximum length. This settings take effect immediately; no system restart is required.

- The QueryProcedures parameter is added to the [SQL] section.

  This defines whether or not all class queries project as SQL Stored Procedures regardless of the SqlProc setting of the query. The default is 0, that is, only class queries defined with SqlProc=1 will project as Stored Procedures. When set to 1, all class queries will project as stored procedures. This settings take effect immediately; no system restart is required. However, you must recompile the classes with the class queries in order for this change to have an affect.

- In the [Debug] section, setting dumpstyle=3 will prevent shared memory from being included the core dump

Audit Records Contain Operating System Userid

Starting in this release, the username is now part of the audit record,. When displayed, it is truncated to 16 characters.

The real operating system username is only returned when connecting to UNIX® or OpenVMS systems; On Windows, it will return the username for a console process; for telnet it will return the $USERNAME of the process; for client connections, it contains the username of the client.

Failure To Mount Required Database At Recovery Is Now Fatal

In prior releases, Caché recovery would skip required databases that failed to mount and continue processing; but startup would later fail when processing the database section of cache.cpf which resulted in shutting down the instance. With this version, failing to mount a required database during Caché recovery is now a fatal error that will cause startup to abort at that time. This allows the underlying issue to be addressed sooner.

TaskManager Jobs Now Use Append IP Address

Customer tasks started via the Cache Task Manager use the "Run As" user name as the license identifier. Beginning with version 2010.1, Caché appended the peer IP address to the user name for such jobs. However, there is no peer address for processes started by the task manager, and no address was appended. This caused an inconsistency in license consumption between jobs started by the task manager and those started by other means. Beginning with this release, jobs started by the task manager now append the local IP address.

Emergency Login Policy

With this version, the Emergency Login policy has been expanded to accommodate two-factor authentication.; the policy is now:

1. During emergency access only the emergency user may log in. Console, Terminal, and CSP are the only services enabled.
2. For enabled services, only authenticated access is permitted. Caché uses its own password authentication for the services, where the emergency access username and password must be used.
3. If the an application has a custom login page that page is used during emergency login.
4. For /csp/sys applications, the standard login page (%CSP.Login.cls) will be used during emergency access even if there is a custom login page available. Using the system default assures that the user has access to the Management Portal in emergency mode.
5. Two-factor authentication is ignored in emergency access mode; applications with two-factor authentication enabled will be inaccessible to the emergency user.

Collation Checking On Upgrade

In this version, during an upgrade installation, the new method **##class(SYS.Database).FixDefaultGlobalCollation(Directory)** is run on all user databases which are defined in an instance, and are mountable by the instance. This method will report to the cconsole.log any errors in collation of system globals.

If any errors are detected, the user should run the method again from a programmer prompt, and pass the modify database flag as the **second argument** which will recollate the global in the correct order.

System Freezes When Journal Daemon Is Hung

On a system set to freeze on journal error, if the Caché control process detects that journal daemon (JD) is hung (no activity for 10 seconds) while there is journal data to write, it will stop the write daemon and the system will freeze.

11.2.1.5 Platform-specific Items

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

All platforms

The OpenSSL libraries built and installed with our products have been updated to version 1.0.0b. All InterSystems projects dependent on OpenSSL have been updated to use new version.

FOP has been updated to Version 1.0 with a specific InterSystems patch for processing in Arabic.

Mac OS X — JOB Command Changes

The mechanism for the JOB command on Mac OS X has changed. It solves a problem but may introduce side effects. Due to the way that the underlying kernel interacts with Mac OS X processes, and to the existence of GUI sessions, the traditional UNIX® way of creating daemons (fork/exec) is not enough for Mac OS X. Apple recommends the use of launchd (or launchctl, which is its user interface) to start all background daemons.

This release implements that recommendation. The JOB command on Mac OS X now calls launchctl to start the Caché JOBs.

AIX — Change Direct I/O Handling

On AIX systems when opening databases, journal files and/or the WIJ for "direct I/O" Caché specifies the `O_CIO` option to open the file for concurrent I/O rather than direct I/O. The use of `O_DIRECTIO` allows other openers which can cause problems if the other process employs buffered I/O.

OpenVMS

- Changes to $ZF

  On OpenVMS, user-supplied $ZF() functions may be written in either C or MACRO. Because of a mismatch in the definitions of PRIV and NOPRIV between the two different header files (cdzf.h and czf.m64), the PRIVS=YES feature in czf.m64), was set opposite of what it should be. User $ZF() functions written in C that depend on the PRIV/NOPRIV feature must be recompiled.

- Changes To Compiler Version

  Due to support requirements, OpenVMS compilers have changed. They are now at Version 7.2. Executables built under the previous compilers are not compatible with the new runtimes.

  This in turn implied that Xalan and Xerces needed to be recompiled. InterSystems has taken advantage of this to upgrade Xalan to version 1.10 and Xerces to version 2.7 and incorporated them as libraries (.olb) which are compiled into our executables and no longer distributed separately.

- Changes To Xalan, Xerces, and unixODBC

The change in compiler version implies that Xalan, Xerces and unixODBC needed to be recompiled. InterSystems has taken advantage of this to upgrade Xalan to version 1.10 and Xerces to version 2.7 and incorporated them as libraries (.olb) which are compiled into our executables and no longer distributed separately.

The OpenVMS version of unixODBC has been upgraded to version 2.2.12 which is the same used by other platforms.

- Informix SQL Converter Not Supported

  The SQL converter from Informix to Caché is not supported on OpenVMS. Attempts to run the Informix conversion on an OpenVMS system will now produce an error instead of logging a message in the console log.

- SHA-2 Functions Not available On OpenVMS Versions Prior To 8.4

  On OpenVMS 8.2-1 and 8.3, the functions **$System.Encryption.RSASHASign()** and **$System.Encryption.RSASHAVerify()** do not support the SHA-2 hash functions when using bitlengths of 224, 256, 384, or 512 bits. The HP-supplied OpenSSL libraries in these releases are based on OpenSSL 0.9.7e, which does not include support for the SHA-2 functions.

- Upgrade Quotas For Background JOBs

  In this version several defaults for OpenVMS process quotas for JOBbed processes have been updated. The primary one is PGFLQUOTA, which limits allocation of virtual memory and was causing problems processing very large XML files. BYTLM and FILLM have also been raised to bring them in line with recent vendor recommendations.

Windows — Installer Change

If the installer finds that the IIS virtual directory, /csp, is already configured, it will no longer update the IIS configuration data.

In addition, new properties have been defined to control updating Apache and IIS:

- CSPSKIPIISCONFIG
- CSPSKIPAPACHE20CONFIG
- CSPSKIPAPACHE22CONFIG

Setting any of these to a value of 1 will result in installation updating of the corresponding CSP binary files, but will not make any changes to the corresponding web server configuration. Setting the property to 0 will make the installer update the appropriate web server configuration regardless of the existence of the /csp virtual directory.

11.2.2 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.

11.2.2.1 System Operational Changes

Compiler Version Changed Due To Support For Large Routines

The internal compiler version has been incremented to reflect changes in the object code to support large routines. This means that routines and classes compiled on this version cannot be copied to and executed on previous versions. Attempts to do so will result in <RECOMPILE> errors.

Caché Fully Qualified Domain Names And Kerberos

The normal form of Kerberos server principal names is specified in RFC 4120, Kerberos V5, section 6.2.1. The principal name is composed of several pieces. They are:

- the name of the service
- a "/"
- the Internet domain name of the host
- an "@"
- realm of the key distribution center (KDC) where the server is registered

An example of such a name is `cache/oakland.domainname.com@KEYDISTRIBUTION.COM`.

In previous versions, Kerberos authentication for non-terminal connection to Caché on platforms other than Windows used an ambiguous format: (cache/host@KDC-realm) which was incompatible with the usage when accessing Caché with csession. In this version, Caché has been changed to always generate the correct form of the service principal name.

In most cases, this should have no impact because it is thought that the vast majority of sites will have used the FQDN form when defining the server principal name in the instance keytab. However, it is possible that some sites have defined the server principal name using just the host name, for example for host `oakland`, the value `cache/oakland@KEYDISTRIBUTION.COM`. These sites will experience a problem after upgrading to a version of Cache with this revision.

To correct this error, a keytab entry for the server principal should be created; in this example, `cache/oakland.domainname.com@KEYDISTRIBUTION.COM` should be created to replace the non-standard `cache/oakland@KEYDISTRIBUTION.COM`.

ECP Will Now Use Process ID In Place Of Job Id

In this release, ECP will log the PID instead of the job ID in the journal entries when the job is not a thread. This means that the ECP session will not be backward compatible; in mirror or cluster configuration a new version of the master cannot failover to an earlier version of the product.

Note: The ECP protocol will remain backward and forward compatible, however.

Shadowing Initiation Requires Start Event

When starting a shadow in the Management Portal, or via **^SHADOW**, the user is required to select a source event to start shadowing at. This is true regardless of the value of the StartPoint property of the shadow configuration object, which is deprecated as of this change. One should always specify the StartPoint parameter in **##class(SYS.Shadowing.Shadow).Start()** method to start a shadow non-interactively.

Shadow Information Is Now In The CPF

When a customer upgrades to version 2011.1 or later, and there are shadow systems defined, the shadow information is converted and moved to the CPF file. There are now two new sections in the file:

- The `[Shadows]` section defines the name of the shadow and its properties.
- The `[MapShadows.NAME]` contains the shadow directory mappings.

Exporting Globals No Longer Checks Name Format

An application which relied on %Library.Global.Export() to reject names which did not end in ".gbl" may no longer work as expected. Names that do not end in that suffix will be accepted and, if globals with those names exist, they will be exported. If not, they will not be part of the export, but no error will be generated. **$SYSTEM.OBJ.Export()** can be used in situations where the caller wants the "type" to be required.

%GSIZE Output Format Alterations

Applications which parse **%GSIZE** output and expect a fixed number of columns may now have problems. The numberof columns in the output is now a constant for a given run of **%GSIZE** (prior to this it was variable), but the number of

columns can vary from run torun depending on the size of the longest global name in the output. Rather than parsing the **%GSIZE** output, applications should use the **Size** query in %SYS.GlobalQuery to retrieve global size information.

Journaling Turned On For Multiple Compiles

In previous versions, Caché defaulted to disabling journaling while doing a class compile to avoid filling up journal files and to improve the speed of the compile slightly. Due to recent changes in the class compiler, such as multiple compilation, this is no longer necessary (or desirable when using mirroring).

Beginning with this version, class compiles will now be journaled by default. This will add more data to the journal files if many class compliations are done. On a development system, administrators may wish to consider changing the default /journal qualifier setting to disable journaling. On a production system, however, administrators almost certainly want the new default of journaling the class compile.

Library Path Now Part Of cache.cpf

Using *LD_LIBRARY_PATH* per user can lead to spoofs that could execute code at root level. Beginning with this version, the *LD_LIBRARY_PATH* data will be part of cache.cpf. Other than <install_directory>/bin, nothing will be in the current library search path for the session other than what is in the `LibPath` field in cache.cpf. The field will be updatable via the Management Portal. All applications relying on the *LD_LIBRARY_PATH* environment variable to set search paths for third-party shared libraries will be affected.

Device Aliases Must Be Unique

This release checks the aliases specified for devices. If the same alias is used for more than one device, Caché will report an error at startup and ignore the second definition.

Changes To Emergency Startup

Emergency Startup has been enhanced in this version so that the following occurs:

- TASKMGR is not started.
- Shadowing is not started.
- Ensemble productions are not started.
- Mirroring is not started.
- ZSTU,%ZSTART,%ZSTOP,ZSHUTDOW are not run when the system starts or stops.
- User processes which log in using the emergency id do not run %ZSTART or %ZSTOP.

In addition, the STU=1 parameter in the CPF file has been removed. If you need to start the system for maintenance, use the Emergency Startup option.

Improved Key Management

Beginning in this version, it is no longer necessary to have a database or managed encryption key activated in order to manage a key file. It is, however, now necessary to know a valid encryption key file administrator username and password in order to add new administrators to a key file or to configure unattended database key activation at startup.

Note: This is yet another reason why it is critical to have a backup key file containing an administrator entry stored along with a copy of that administrator's password in a physically secure location.

TROLLBACK Does Not Initiate Database Mount

One of the general principles of Caché is that when a database has been explicitly DISMOUNTed, a database access attempt should not implicitly cause it to be MOUNTed; it must be explicitly mounted by operator action. TROLLBACK has been corrected to be consistent with this principle.

New Locales

Slovenian2

A new collation is available for the Slovenian locale. Slovenian2 is similar to Slovenian1, except that upper and lowercase letters collate together (merged cases).

Turkish (Unicode)

A Unicode Turkish locale is now available ("turw"). By default it uses the Turkish1 collation.

New DDL Type Mapping For VARCHAR(Max) And NVARCHAR(Max)

Beginning with this version of Caché, there are new default system-defined DDL mappings for VARCHAR(Max) and NVARCHA(Max); both of these map to %Stream.GlobalCharacter. Prior to getting a version with this change, systems can simply add these mappings to the user-defined DDL Mappings. This change allows VARCHAR(Max) and NVARCHAR(Max) to be used as an argument to a procedure in a TSQL CREATE PROCEDURE DDL statement.

11.2.2.2 Routine Compiler Changes

Support For Larger Routines

Beginning in this release, the compiler will now allow routines up to 8MB in length. When a compiled routine exceeds 32KB, Caché will use up to 128 64KB routine buffers to hold the routine. These buffers will be allocated and managed as a unit. Therefore, the use of large routines compiled in this release will affect routine buffer allocation. Generally, more 64KB buffers will be required than in previous releases; however, the distribution of memory among the buffer pools will depend on the realtime distribution of routine sizes in use at a specific site.

Note: Routines compiled on this version will not run on earlier versions of Caché.

11.2.2.3 Routine Changes

Implement Japanese Datetime Formats

Two new date formats have been added to the list of *dformats* for $ZDATE and related functions:

- 16 – year, month, and day values with Japanese kanji for year, month and day following each one, respectively; that is, *YYYY*$CHAR(24180)*MM*$CHAR(26376)*DD*$CHAR(26085)

  .

- 17 – like format 16 with the addition of a space after the year signifier, and another after the month signifier.

Make URL Translation Symmetric For Non-Latin1 8-Bit Character Sets

In 8-bit locales not based on Latin1 (for example, "ruw8", the CP1251-based Russian locale),

```
$ZCVT("%XX", "I", "URL")
```

now interprets *XX* as the hex code of a character in the current character set. In previous releases, this was assumed to be a Unicode character; in some character sets this codepoint did not have a corresponding value in the current character set and was replaced by a default character such as "?".

The new behavior means that in CP1251

```
$ZCVT($C(192), "O", "URL") = "%C0"
```

and

```
$ZCVT("%C0", "I", "URL") = $C(192)
```

make a round trip using the URL translation valid for all characters.

11.2.2.4 Class Changes

Larger Class Limits

Beginning with this release the system now supports a larger class descriptor. This means that classes now can support a larger number of members declared in the class. The limits on class inheritance depth, and the number of superclasses allowed have also been defined. For a complete list of the applicable bounds, see "Guide General System Limits" in the *Caché Programming Orientation Guide*.

Error Reporting Changes

The standard Caché mechanism for returning an error is to use the $$$ERROR macro with a standard message. In many cases, the message contained only a description of the error without any context information. Several messages, including the "object to load not found" message now include the classname where the error was encountered. It is possible that some SQL storage applications may have to be changed to recognize the new format.

Update Of Class Dictionary To Level 25 – LegacyInstanceContext

This version of Caché updates the version of the class dictionary to level 25. Among opther changes, this introduces the *LegacyInstanceContext* class keyword wose presence indicates that the class relies on generated code that passes *%this* as the first argument to instance methods. This was previously announced in 2009 in the Compatibility Blog.

As an aid, the class dictionary upgrade looks for references to *%this*. It scans both code and comments in case there are usages of **$XECUTE** and **$TEXT** even though this may result in false positives. If it finds any such references, it marks the class as needing *LegacyInstanceContext* so the compiler will continue to generate code to pass *%this* as the first argument to instance methods.

If no instances of *%this* are found, then the class is not marked *LegacyInstanceContext* so instance methods will no longer assume *%this* is passed as the first argument.

This approach does not, however, uncover separate code that assumes *%this* exists and is properly set. Consider a class with a method that calls an external routine such as:

```
method Test()
{
    Quit $$Func^Routine()
}
```

where **Routine** is:

```
Func () public {
    Quit %this.Name
}
```

Because the use of *%this* will not be discovered in the scan of the class (it is external to that source), it will not be marked as *LegacyInstanceContext*. Subsequent execution may result in an <UNDEFINED> error; in much harder-to-debug situations, *%this* may be pointing to the wrong context or not at an object instance at all.

All new code should rely on *$this* for context; *%this* will not be set for new classes as it is deprecated.. All older classes marked as LegacyInstanceContext=1 will continue to behave the same as in previous releases.

No change is necessary to existing classes because of *LegacyInstanceContext*. However, if you do want to update a class, the steps are as follows:

1. Replace all occurrences of *%this* with *$this* in a given application.
2. Remove the *LegacyInstanceContext* keyword from all classes in the project or application.
3. Recompile the application.

Class Deletions

The following classes were present in version 2010.2 and have been removed in this version:

- %CSP.UI.Portal — ObjectGatewayStartStop
- %ExtentMgr — Extent
- %Library — CppApi
- %OSQL — Debugger, Transformer
- %SQL — Routine, RoutineColumn
- %Studio.SourceControl — ISCCheckin
- %WebStress — DataTransfer, Page
- %WebStress.UI — GridDetails, Attributes, Columns, SelectOptions, Tags, Transfer, SaveData
- %WebStress.UI — Input, Menu, Previous, Definitions, Root, Search, Criteria, Displays, SearchPage
- %XML — SupportCode
- %ZEN.Report — sort
- %cspapp.op — webstress, appservers, appsystem, blank, encrypt, generators, nopagedelay, noresultstore, proxysetup-mozilla, proxysetupmsie, scriptedit, scriptrecorder, scriptrecorderstatus, showerrors, showurls, testappstats, testedit, testprint, testrun, visual, visualdisplay, visualdisplayoptions, webservers
- SYS.Info — Advertiser
- com.intersys.jdbcgateway — JDBCGateway
- java.io — InputStream, OutputStream, Serializable
- java.lang — Class, ClassLoader, Object, Package, AccessibleObject, Constructor, Field, Member, Method
- java.net — ContentHandler, ContentHandlerFactory, FileNameMap, URL, URLConnection, URLStreamHandler, URLStreamHandlerFactory
- java.security — CodeSource, Guard, Key, Permission, PermissionCollection, Principal, ProtectionDomain, PublicKey, Certificate
- java.util — Collection, Enumeration, Iterator, Map, Set

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CSP.UI.Portal.AdvancedSettingsTemplate	Method	DrawTitle
%CSP.UI.Portal.AdvancedSettingsTemplate	Parameter	APPLICATION
%CSP.UI.Portal.AdvancedSettingsTemplate	Property	ParentURL
%CSP.UI.Portal.Application	Method	DrawTitle
%CSP.UI.Portal.Application	Parameter	CSSINCLUDES
%CSP.UI.Portal.Config.Device	Property	ParentURL
%CSP.UI.Portal.Config.SQLDataTypes	Method	DrawLocatorExtra
%CSP.UI.Portal.Config.SQLDataTypes	Property	SingleSubject
%CSP.UI.Portal.Config.SQLDataType	Method	DrawLocatorExtra
%CSP.UI.Portal.Config.ZenReport	Method	DrawTitle

Class	Type	Name(s)
%CSP.UI.Portal.Config.ZenReport	Property	LocatorParent
%CSP.UI.Portal.DatabaseFreespaceCleanup	Method	DrawTitle
%CSP.UI.Portal.DatabaseFreespaceCompact	Method	DrawTitle
%CSP.UI.Portal.DatabaseFreespace	Method	DrawTitle
%CSP.UI.Portal.DatabaseFreespace	Parameter	APPLICATION
%CSP.UI.Portal.FileManTemplate	Method	DrawTitle
%CSP.UI.Portal.FileManTemplate	Parameter	APPLICATION, DOMAIN
%CSP.UI.Portal.FileManTemplate	Property	ParentURL
%CSP.UI.Portal.FileMan	Parameter	APPLICATION
%CSP.UI.Portal.FileMan	Property	ParentURL
%CSP.UI.Portal.ISQL	Method	DrawTitle
%CSP.UI.Portal.ISQL	Property	ParentURL
%CSP.UI.Portal.Mappings	Method	DrawTitle
%CSP.UI.Portal.NLSEdit	Method	DrawTitle
%CSP.UI.Portal.NLSEdit	Parameter	APPLICATION
%CSP.UI.Portal.NLS	Method	DrawTitle
%CSP.UI.Portal.NLS	Parameter	APPLICATION
%CSP.UI.Portal.ObjectGatewayActivities	Method	DrawTitle
%CSP.UI.Portal.ObjectGatewayStart	Method	DrawTitle
%CSP.UI.Portal.ObjectGatewayStop	Method	DrawTitle
%CSP.UI.Portal.ObjectGateways	Method	DrawTitle
%CSP.UI.Portal.ObjectGateway	Method	DrawTitle
%CSP.UI.Portal.RoutineCompare	Method	DrawTitle
%CSP.UI.Portal.SSLList	Parameter	APPLICATION
%CSP.UI.Portal.SSL	Method	CheckAllBlanks, DrawTitle
%CSP.UI.Portal.SSL	Parameter	APPLICATION
%CSP.UI.Portal.Template	Method	DrawTitle, GetQuickLinks
%CSP.UI.Portal.X509Credentials	Method	DrawTitle
%CSP.UI.Portal.X509Credentials	Parameter	APPLICATION
%CSP.UI.Portal.X509Credential	Method	CheckAllBlanks, DrawTitle
%CSP.UI.Portal.X509Credential	Parameter	APPLICATION
%Dictionary.ClassDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledClass	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledConstraintMethod	Method	%AcquireLock, %ReleaseLock

Class	Type	Name(s)
%Dictionary.CompiledConstraintMethod	Property	RuntimeImplementation
%Dictionary.CompiledConstraint	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledForeignKey	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledIndexMethod	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledIndexMethod	Property	RuntimeImplementation
%Dictionary.CompiledIndexProperty	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledIndex	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledInstanceVar	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledInstanceVar	Property	RefSlot
%Dictionary.CompiledMethod	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledMethod	Property	RuntimeImplementation
%Dictionary.CompiledParameter	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledProjection	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledPropertyMethod	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledPropertyMethod	Property	RuntimeImplementation
%Dictionary.CompiledPropertyUDLText	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledProperty	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledQueryMethod	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledQueryMethod	Property	RuntimeImplementation
%Dictionary.CompiledQuery	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageDataValue	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageData	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageIndex	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageProperty	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMapData	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMapRowIdSpec	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMapSubAccessvar	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMapSubInvalidcondition	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMapSub	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorageSQLMap	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledStorage	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledTrigger	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledUDLText	Method	%AcquireLock, %ReleaseLock
%Dictionary.CompiledXData	Method	%AcquireLock, %ReleaseLock

Class	Type	Name(s)
%Dictionary.ForeignKeyDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.IndexDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.MethodDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.ParameterDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.ProjectionDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.PropertyDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.PropertyUDLTextDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.QueryDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageDataDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageDataValueDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageIndexDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StoragePropertyDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapDataDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapRowIdSpecDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapSubAccessvarDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapSubDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.StorageSQLMapSubInvalidconditionDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.TriggerDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.UDLTextDefinition	Method	%AcquireLock, %ReleaseLock
%Dictionary.XDataDefinition	Method	%AcquireLock, %ReleaseLock
%Installer.User	Property	Password
%Library.CacheSQLStorage	Method	%SaveDataInsert, %SaveDataUpdate
%Library.EnsembleMgr	Method	check4Install
%Library.GlobalStreamAdaptor	Parameter	TEMPGLOBALNAME
%MV.StudioRoutines	Method	Compile
%Net.Remote.Proxy	Method	%Execute0, %Execute0R, %Execute1, %Execute1R, %Execute2, %Execute2R, %Execute3, %Execute3R, %Execute4, %Execute4R, %ExecuteGetter, %ExecuteOL0, %ExecuteOL1, %ExecuteOL2, %ExecuteOL3, %ExecuteOL4, %ExecuteOLR0, %ExecuteOLR1, %ExecuteOLR2, %ExecuteOLR3, %ExecuteOLR4, %ExecuteSetter

Class	Type	Name(s)
%Net.RemoteConnection	Method	ExecuteCode
%Net.abstractMQ	Property	mqCOptID, mqCloseID, mqCommitID, mqConnID, mqDisconnID, mqErrLogID, mqGetID, mqGetLastErrID, mqGetStreamID, mqInitID, mqMsgDescID, mqMsgDescIID, mqMsgDescSetID, mqPutID, mqPutStreamID
%SOAP.WebBase	Method	GetElementQualified
%SOAP.WebClient	Property	MethodName
%SOAP.WebService	Property	ImportHandler
%SQL.DynamicStatement	Method	LookupCache
%SQL.Shell	Method	load
%SYS.Audit	Property	ZPad3, ZPad4, ZPad5
%SYS.PTools.Stats	Method	Init
%SYS.PhoneProviders	Method	DeleteId, Save
%Studio.SourceControl.ISC	Method	DisplayUncommitted, GetUncommitted, ListUncommitted, RefreshUncommitted, RemoveAllUncommitted, RemoveUncommitted, SetUncommitted, UpdateUncommitted
%Studio.SourceControl.ItemSetWS	Parameter	CCRSrc
%Studio.SourceControl.ItemSet	Method	%OnBeforeSave, MarkCommitted
%WebStress.Control	Method	BuildCode, CheckGeneratorDataValid, DownloadData, GeneratorDataValid, GetExportType, GetIterationData, GetNextRunNumber, GetRunData, GlobalData, GlobalDataBuild, IterationData, ResetControlData, RunData, SetControlData, Transfer, Upload, UploadAllData, UploadCode, UploadData, UploadMiscData
%WebStress.Control	Property	ValidData
%WebStress.Utils.Recorder.Summary	Property	lines
%XML.DataSet	Method	TypeToXSD, XMLExport, XMLImport
%XML.Security.EncryptedType	Property	CipherData, EncryptionProperties
%XML.Utils.SchemaReader	Property	TargetElementQualified
%ZEN.Report.Display.Chart.plot	Method	needsDataMax, needsDataMin
%ZEN.Report.Display.atthtml	Method	%DrawToHTML
%ZEN.Report.Display.atthtml	Property	name, value

Class	Type	Name(s)
%ZEN.Report.Display.attxslfo	Method	%DrawToXSLFO
%ZEN.Report.Display.attxslfo	Property	name, value
%ZEN.Report.Display.td	Property	colspan, rowspan
%ZEN.Report.Display.th	Property	colspan, rowspan
%ZEN.Report.reportPage	Parameter	AGGREGATETAG, CONTENTTYPE, COUNTREPEATINGELEMENT, DATASOURCE, DEFAULTMODE, EMBEDXSL, ENCODING, EXCELMODE, EXCELSTYLESHEET, HANG, HTMLSTYLESHEET, INDENT, NLSIOTABLE, PDFSWITCH, PRESERVESPACE, PS, REMOVENULLS, RENDERSERVER, RENDERTIMEOUT, REPEATINGELEMENT, REPORTDIR, REPORTXMLNAMESPACE, REPORTXMLNAMESPACEPREFIX, RESOURCE, SPLITANDMERGE, SQLCACHE, STRIPPI, STRIPSPACE, STYLESHEETDEFAULTMODE, SUPPORTMACROS, TABLEALTCOLOR, TOOLONGTEXT, USEINSTALLEDFOP, USEINSTANCEHOSTNAMEONRELATIVEURLS, USETEMPFILES, XMLSWITCH, XSLFOSTYLESHEET, XSLSWITCH, XSLTMODE, XSLTVERSION
%ZEN.SVGComponent.chart	Method	buildLabelArray
Config.CPF	Method	ConvertTo201022
Config.Mirror	Property	Heartbeat
Config.MirrorMember	Method	StartMirror, StopMirror
SYS.WSMon.wsen.Items	Parameter	NAMESPACE
Security.System	Method	LDAPSearchPasswordGet, LDAPSearchPasswordSet

Class	Type	Name(s)
Security.System	Property	DefaultSecurityDomain, InactiveLimit, InvalidLoginAction, InvalidLoginLimit, LDAPAttributeComment, LDAPAttributeFullName, LDAPAttributeNameSpace, LDAPAttributeRoles, LDAPAttributeRoutine, LDAPAttributes, LDAPBaseDN, LDAPCACertFile, LDAPClientTimeout, LDAPDomainName, LDAPFlags, LDAPHostNames, LDAPSearchPassword, LDAPSearchUsername, LDAPServerTimeout, LDAPUniqueDNIdentifier, PasswordExpirationDays, PasswordPattern, PasswordValidationRoutine, SecurityDomains

Method Return Changes

The following methods have different return values in this version of Caché:

- %CSP.UI.System.Index — ProcessIndexZen
- %Library.ProcedureContext — AddContext
- %SYSTEM.Encryption — RSAEncrypt
- %UnitTest.TestCase — AssertSkippedViaMacro, AssertStatusNotOKViaMacro, AssertStatusOKViaMacro
- %WebStress.Control — StartMonitor
- %ZEN.Datatype.boolean — XSDToLogical
- %ZEN.Report.Version — getVersion

Method Signature Changes

The following methods have different calling sequences in this version of Caché:

Class Name	Method Name(s)
%CSP.Response	SetCookie
%CSP.Session	%OnNew, GetSession, Login, SetContext
%CSP.UI.Portal.Config.ZenReport	SaveData, doBrowse
%CSP.UI.Portal.NLSEdit	SaveInternalDefaults
%CSP.UI.System.ExpResultPage	IntegrityCheckBack
%CSP.UI.System.ImportPane	DrawContent
%CSP.Util.Librarian	FindDocBookLink
%IO.FileStream	NewTempFilename
%Installer.Installer	CSPApplication

Class Name	Method Name(s)
%Library.AbstractStream	DeleteStream, IODeleteStream
%Library.Collation	MVDLEDate
%Library.File	CopyDir, CopyFile, NormalizeDirectory, NormalizeFilename
%Library.FileStreamAdaptor	IODeleteStream, ReadLine
%Library.GTWCatalog	SQLFieldsExecute, SQLFieldsJExecute, SQLForeignKeysExecute, SQLForeignKeysJExecute, SQLPrimaryKeysExecute, SQLPrimaryKeysJExecute, SQLProcedureColumnsExecute, SQLProcedureColumnsJExecute, SQLProceduresExecute, SQLProceduresJExecute, SQLSpecialColumnsExecute, SQLTablesExecute, SQLTablesJExecute, getIndexInfoExecute
%Library.Global	Export
%Library.GlobalEdit	CheckGlobalIntegrity
%Library.GlobalStreamAdaptor	IODeleteStream
%Library.Persistent	%DeleteExtent
%Library.RoutineMgr	ImportItemListExecute
%Net.FtpSession	sendCommand
%Net.Remote.Proxy	%PostInvoke, %PreInvoke, %ProcessError
%SOAP.Security.SecurityTokenReference	GetX509Data, GetX509KeyIdentifier
%SOAP.WebBase	GetEncodedAttribute, ProcessHeaders, ProcessSOAPEnvelope
%SQL.DynamicStatement	Prepare
%SQL.Shell	%Go, cmdSet
%SQL.Statement	%OnNew
%SYS.Audit	ListByEventExecute, ListByPidExecute, ListByUserExecute, ListExecute
%SYS.AuditString	LogicalToXSD
%SYS.GlobalQuery	SizeExecute
%SYS.PTools.SQLStats	Init
%SYS.PTools.Stats	LogSave, Report, Start, Stop
%SYS.ZENReportServer	%ServeTransform
%SYSTEM.Encryption	RSADecrypt, RSAEncrypt
%SYSTEM.INetInfo	GetInterfacesInfo
%SYSTEM.OBJ	Compile, CompileList, Delete, Export, MakeClassDeployed, SetQualifiers, UnCompile
%SYSTEM.SQL	SetDefaultSchema
%SYSTEM.Util	IsDST, UTCtoLocalWithZTIMEZONE

Class Name	Method Name(s)
%SYSTEM.Version	Format, GetBuildDate, GetBuildNumber, GetBuildOS, GetBuildTime, GetMajor, GetMinor, GetNumber, GetOS, GetPatchId, GetPlatform, GetPoint, GetProduct, GetVersion
%Stream.FileBinary	ReadLine
%Studio.Debugger	WatchListExecute, WatchListOrefExecute
%Studio.Project	FindInFiles, FindInProject
%Studio.SASchemaUtil	loadSchema
%Studio.SourceControl.ISC	GetSharedWorkspace, SetCredentials, SetSharedWorkspace
%Studio.SourceControl.ItemSet	Load, LoadToNS, LoadToOS
%UnitTest.ODBCSQL	runODBCSQLStatement
%UnitTest.TSQL	runSQLStatement
%WebStress.Control	Prepare
%WebStress.Utils.Recorder	Summary
%XML.Adaptor	XMLImportAttributes
%XML.DataSet	ParseSchema
%XML.Implementation	WriteHeaderBinding
%XML.Namespaces	AddNamespace, DefineNamespacePrefix, PushNodeForExport
%XML.Schema	DefineNamespace
%XML.Security.KeyInfo	CreateX509
%XML.Security.Signature	ComputeSha1Digest, CreateX509
%XML.Security.X509Data	Create
%XML.Utils.SchemaReader	AddNS
%XML.Writer	CharsText
%ZEN.Auxiliary.abstractController	getDataByName
%ZEN.Component.calendar	selectDay
%ZEN.Component.page	XMLImportAttributes
%ZEN.Datatype.boolean	LogicalToXSD
%ZEN.Dialog.finderDialog	GetClassInfo, openSuper
%ZEN.FinderUtils	%GetArrayForQuery, sortData
%ZEN.Portal.Application	%DrawSmallMenu, %DrawTitleHTML
%ZEN.Portal.selector	GetDropdownContent
%ZEN.Report.Display.Chart.chart	render, renderLegend
%ZEN.Report.Display.Chart.lineChart	renderMarkers
%ZEN.Report.Display.Chart.plot	calculateRangeValues

Class Name	Method Name(s)
%ZEN.Report.Display.document	%DrawPageToXSLFO
%ZEN.Report.reportPage	%DisplayPDF, %DisplayPDF1, DeleteTempFiles
%cspapp.op.utilsystaskbuildercontent	ContentSave
%cspapp.sec.utilsysapplication	SaveConfig
Backup.General	ExternalFreeze, SuspendWD
Config.CPF	MergeMaps, RemoveDuplicates, UpdateLines, UpdateLinesAgain
SYS.Database	SilentIntegrityCheck
SYS.MirrorConfiguration	ValidateVirtualAddress
Security.Roles	Export
Security.Users	Export

%Library.Decimal Now Defaults To Scale=0

The Caché datatype, %Library.Decimal, now defaults to a SCALE=0. In previous releases, there was no default SCALE. Upon INSERT/UPDATE or Object Save, the Normalize method will now round to the default SCALE=0 when no SCALE is specified or the property.

11.2.2.5 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

Identical Labels In Multiple Methods Of The Same Class

As part of compiling a class, the compiler attempts to pack as many methods of the class into one compiled unit as possible. If two or more methods of that class define a label of the same name, and the methods are marked as PROCEDUREBLOCK = 0, there was the risk that the compiler would include them in the same compiled unit and report a duplicate label error.

Recent improvements in the class compiler have increased the size of the compiled unit and therefore increased the probability that non-procedureblock methods with identical labels could trigger this error. Applications which trigger his condition mustbe written to either use procedurblocks, or to change the label values so there is no overlap.

Avoid Duplicating Properties Inherited From Superclasses In Subclasses

Previously, if an application defined a property *X* in a superclass, and then created a subclass which did not modify property *X* at all (just inherited it from the superclass), the class compiler would redefine this property in the class descriptor of the subclass. This was because Caché needed an internal slot number to reference this property by in the class compiler. This requirement has now been removed. Caché now avoids duplicating the properties in the subclass and allows the system code to dynamically inherit the property from the superclass.

CAUTION: No customer should have applications that depend on the internals of the class compiler (deliberately not publicly documented). Any applications using undocumented internal functions such as $zobjval to access data via slot numbers will have to be rewritten.

Users Must Normalize ID Values

Beginning with this release, passing an unnormalized integer to **%Open**, **%OpenId**, **%Exists**, and **%ExistsId** will no longer work. The applications passing such values must apply normalization prior to calling the method.

Simple ID values are no longer normalized by the various methods that accept an ID as a parameter. ID values passed to the various methods of a class are expected to be in the normalized form that is returned by the **<OREF>.%Id()** method.

If an ID is a simple integer and a value is passed that is not in the integer normal form (01 vs. 1, for example) then the methods named here will fail.

SQL Storage Compiler Now Recognizes SQLCHILDSUB Name

A class that defines a relationship with a cardinality of `PARENT` is often referred to as a "child class"; and the type class of the relationship is referred to as the "parent class". The ID of the child class is based on the relationship (the ID of the parent) and either a property value or a system assigned value. When the ID is based on a system assigned value, a column is generated in the SQL table projected by the child class. That generated column corresponds to the system assigned value and is referred to as the "childsub"; it is also, by default, the generated column name.

This name can be specified in the storage definition by entering a name in the SQLCHILDSUB keyword of the storage definition. The expression used by the system to assign a value to the childsub (ID in the case of objects) is defined in the SQLIDEXPRESSION keyword.

In prior releases, if an existing child class defined SQLCHILDSUB and compiled the class prior to this release, then the generated childsub column would be named "childsub". Beginning with this release, the value of SQLCHILDSUB will be used. This presents a backward incompatibility. The prior behavior is the result of an error; the new behavior is correct. The typical workaround for this was to define a property representing the childsub; if that is done then this change will have no effect.

Cardinality Relationships Can Not Enforce REQUIRED

Previously, a relationship with a cardinality of `MANY` or `CHILDREN` and also specifing `REQUIRED` would compile cleanly. Now, an error is reported by the compiler indicating that the `REQUIRED` constraint for n-cardinality relationships is not supported.

To compile cleanly, remove the REQUIRED constraint.

Compilation Using Multiple Jobs

Beginning with this version, Caché provides the ability to use multiple jobs for large compilations. This is enabled using the qualifier `/multicompile`; it is disabled by default.

When it is enabled, and the compiler detects that it can employ multiple jobs usefully, it will start up slave jobs which will show up in *%SS* as being in the **%occCompileUtils** routine. It communicates with these slave jobs using a global and *$SYSTEM.Event* to signal that some work should be done. When a slave job completes, it sends back to the main process an indication that the work is complete along with any error information or output to display. So the typical behavior is: workers jobs are started, then work is queued, and the worker jobs process their part of the work. The main process waits for each job to finish its part, and displays the errors or any other output destined for display. When all the work is complete at this level, the main process will loop round and start queuing any remaining work.

Once a worker job is started it will remain around for 10 minutes after the last piece of work it receives in case more work appears to avoid the cost of starting and shutting down jobs.

This code will not improve the speed of compiling a single class, it is only focused on compiling multiple classes in parallel. In addition, the only part of the compilation process that supports this multiple cpu compile is compiling the MAC code into INT code, assembling these into routines, compiling these routines and building the class descriptor. This can only be done in parallel when there is no dependency between the classes being compiled. The compiler detects the dependencies and breaks down the compile based on this automatically. Thus, if classes A and B are not dependent on each other , they can be compiled at the same time. If A is a superclass of B, however, A must be fully compiled before compilation can begin on B; no parallelism is possible.

Use of parallel compilation assumes that all relevant dependencies between classes are expressed in the class declaration. The order classes are compiled in may be slightly different from previous versions of Caché, but the order chosen still satisfies all the dependency rules. If two classes did not specify a dependency, the order in which they are done cannot be predicted; this could potentially cause a problem if two classes were dependent on each other but no dependency had been specified and the sequential compilation order just happened to work correctly. If this occurs, add a CompileAfter or DependsOn dependency between the classes to specify their relation.

Also, the worker jobs will obviously have a different *$JOB* number from the main process. This means that if data being stored by one slave job during the compile, and another slave job is attempting to access that data using $JOB as an index, that attempt will fail because the $JOB numbers of the two processes differ. This situation can occur, for example, in sophisticated generator methods that interact with one another. The solution for this is to use the *%ISCName* local variable which is defined in the compiler context; it is a consistent name between the main job and all the worker jobs and so can be used to share information.

The number of jobs used is limited to 16 jobs maximum as recent benchmarks have shown that more jobs than this do not improve overall performance.

Removal Of InterSystems Internal Items

This version removes the method keyword, RuntimeImplementation. It was only intended for use by InterSystems and is no longer required.

It also removes the $$$cIVARrefslot macro. This was undocumented and should not appear in user code. Any code that uses this macro will fail to compile in this version.

Synchronization Order Correction

Sync sets contain entries that represent object filing events - inserts, updates, and deletes. If a sync set contains more than one entry that affects the same object, those entries must be applied in the same chronological order as they occurred originally.

In prior releases, an error existed that caused some entries to be processed out of order. The cause was an unresolved dependency on import. Unresolved dependencies trigger a sync set entry to be scheduled for processing at a later time. This rescheduling could cause entries to be applied out of order and could introduce data corruption.

That error is now fixed, however, it is possible that an application has made some assumptions about the order in which SyncSet entries are processed. If that is the case, the aplication needs to be examined and retested to make certain problems do not occur.

Control Global Kills On %DeleteExtent

The **%DeleteExtent** method attempts to delete all instances of a class. If all instances are successfully deleted **%KillExtent** is called to kill any globals that might be left defined. Not all globals are killed, especially in cases where multiple classes share the same globals.

%DeleteExtent has a new parameter, *pInitializeExtent*, that, if true, causes **%KillExtent** to be called when all instances of the class are successfully deleted. The default value of *pInitializeExtent* is 1 (true). If *pInitializeExtent* is not true, then **%KillExtent** is not called and some empty globals could still be defined after **%DeleteExtent** returns. If the class uses a global counter to assign new object ID values, then that global counter will also remain defined in most cases.

Extent Query In %Library.Persistent

%Library.Persistent defines a query that is inherited by every class that extends %Library.Persistent. The query, **Extent**, is used to produce a result set containing all instances of a persistent class. The **Extent** query can be overridden, either as a %Library.ExtentSQLQuery or as some other query type. The overridden query must return rows corresponding to each instance of the class and the first column must be %ID.

Change To i%var Handling

The usage *i%var* is used in **<var>Get** and **<var>Put** methods to make direct references to an instance variable. The class compiler previously converted *i%var* references into the internal slot number where the instance variable was stored. In this version, this is done by the system code which allows the class compiler to be more dynamic. A side effect of this is that the *i%var* name is not validated at compile time. The code will compile and a runtime runtime error will be generated if the *var* is not defined in the superclass tree.

SQLROWIDNAME Usage Enforced

The class keyword, `SQLROWIDNAME`, allows the user to define the name of the SQL column generated as the ROWID. This SQL column corresponds to the object ID which is only accessible through a method call such as **%Id()**. It is not valid

to override the SQLFIELDNAME of a property in a subclass because it violates the rule that every instance of a subclass is implicitly an instance of its primary super class. The SQL ROWID column name cannot be overridden for the same reasons.

Previously, this rule was not enforced on the SQLROWIDNAME value. It is enforced beginning in this version. Failure to observe it will result in a failure to compile the class.

%GUID Invalid As Column Name

If the user class has an existing column whose name is *%GUID*, then this will now trigger an error during class compile. If the class has GUIDENABLED as true, then the class cannot implement a method named **%OverrideGuidAssignment()**, a property named *%%GUID*, or a property whose SQLFIELDNAME is *%GUID*.

/foldmethod Qualifier Deprecated

The class compiler /foldmethod qualifier used to detect identical methods preserving only one in the generated code has been deprecated. The qualifier no longer has any effect on the generated code.

Bind Properties With CLASSNAME=1 As %Library.List

Any property that specifies CLASSNAME=1 will be bound to SQL as type %Library.List; CLASSNAME=1 means the value for the property is an OID which is in Objectscript $LIST format. A property defined as:

```
Property OID As %Library.Persistent(CLASSNAME = 1) [ Required ];
```

would, in previous releases, bind to SQL as %Library.Integer. starting with this release, it binds to %Library.List.

Inheriting A Relationship Property From A Secondary Superclass Prohibited

In previous versions, an attempted to inherit a relationship from a secondary superclass would get invalid results due to silent failures of the relationship at runtime. Beginning with this version, the compiler nows detects this in the class compiler and reports an error:

```
ERROR #5572: Can not inherit relationship property 'X' in class 'Y.Z' as a secondary superclass.
```

The failure occurred because a primary subclass of a persistent class shares the same extent as the superclass; but a secondary subclass does not. Inherited queries in the secondary subclass could not find the extent of the originating class to properly reference the class data.

11.2.2.6 Studio Changes

INT/MAC Save Does Not Compile Automatically

In prior versions, there was an option in Studio that allowed a Save of an INT/MAC routine to execute a compile as well. With this version, that feature has been removed because the behavior was already available when using the Compile option; so it was redundant. In addition, not being able to save a modification without having it be projected immediately as executable code, while fine perhaps on a test system, was potentially disastrous on a live environment or a shared one.

A Save for any given document type now simply saves the current version of the document back to the server. A Compile always saves the document and compiles the document into its descendant forms as well. For most users this is simply going to require using a different button or keystroke combination in the Studio. To compile a collection of documents you can make use of a Project and the "Build" option.

11.2.2.7 Language Binding Changes

Refactor Java .jar Files

In this release, InterSystems has refectored its Java libraries into four parts:

- cachejdbc.jar – This contains the Caché JDBC driver and is required for all applications using the Java binding. It includes the following com.intersys packages: jdbc, jsse, jgss, and util (newly added in this release).
- cachegateway.jar – This contains Java and JDBC Gateway. It depends on cachejdbc.jar and includes the packages: com.intersys.gateway and com.intersys.jdbcgate.

- CacheexTRreme.jar – It contains the components for Java eXtreme, namely the com.intersys packages: gloals, mds, xdo, and xep.
- cachedb.jar – This contains the remainder of the InterSystems Java components, including Java Binding, Jalapeno, EJB (Enterprise Java Beans), and so on. It holds the com.intersys packages: cache, classes, codegenerator, EJB, jsp, and objects, as well as com.jalapeno package

For more details, please consult The Caché Java Class Packages.

Jalapeño Configuration

Since Jalapeño was first made available, InterSystems introduced multiple configuration options for it that affect performance. Having the "correct" configuration in most cases can improve Jalapeño application performance, in many cases dramatically. The default configuration, however, was not optimized for performance of a typical application but rather was aimed to preserve full compatibility with original version of Jalapeño.

With this version of Caché, the default configuration for Jalapeño has been changed to make it transparent to the application programmer and end user. Specifically, the following features are now available:

- There is now the ability to set a site-wide default Jalapeño configuration in addition to the existing ability to load a configuration for a given application.
- The sample default configuration file was reworked to make it self-explanatory. This file can be modified for site-wide defaults as well as copied and adjusted for specific applications.
- The default configuration has been tuned to provide better performance for the average application.

Default Jalapeño configuration is now stored in Caché Installation directory in /dev/java/conf/jalapeno.properties. The file is a properties file with comments identifying what options can be configured and how. If the installation lacks this file because it is using an older server version, the hard-wired default is used. However, this configuration file can be added to any Caché server from 2010.1 and later. This file affects all clients connecting to the server, not the clients working on current machine.

Note: The default configuration file now uses a LAZY fetch policy and a GENERATE_HELPERS access method. This requires third-party open source libraries (available under Apache license). If this is not acceptable to a specific site, the configuration file MUST be changed.

Changes To Java Generated Code For Properties

Because of the new object dispatch Java driver in 2010.1 and beyond no longer uses projected values fields ii_<Property-Name>, jj_<PropertyName>, or kk_<PropertyName>. Generated samples will need to have the CacheDB.jar from 2010.1 or later in order to work properly.

Class Name Changes For Caché eXtreme

This change renames the Java package for eXTreme dynamic objects, previously com.intersys.extreme, to com.intersys.xdo (where "xdo" is an acronym for "eXTreme Dynamic Objects", analogous to "xep" for "eXTreme Event Persistence"). The classes within this package are renamed as follows:

From	To
com.intersys.extreme.XTDatabaseConnection	com.intersys.xdo.DatabaseConnection
com.intersys.extreme.XTDatabaseConnectionFactory	com.intersys.xdo.DatabaseConnectionFactory
com.intersys.extreme.XTDynamicObject	com.intersys.xdo.DynamicObject
com.intersys.extreme.XTException	com.intersys.xdo.XDOException
com.intersys.extreme.XTNullValueException	com.intersys.xdo.XDONullValueException

The sample code java/samples/extreme/extreme/XTDemo.java has been changed to java/samples/extreme/xdo/XDODemo.java.

Change To Object Save Methodology In Jalapeño

With this version, if the fetch policy is DISCARD_AFTER_FETCH, and a list of related objects have been never accessed by the application from the parent side, then Jalapeño does not check to see if the objects in this list have been modified even on deep save. This drastically improves performance of deep saving a set of objects with a complex relationship graph.

However, there is a possible loss of data when the fetch policy is DISCARD_AFTER_FETCH in the following scenario:

- Fetch an object (object A) from the database.
- Make some changes to it and objects that it references.
- Keep a referenced object (object B) in the application heap memory.
- Save object A back using deep save to save changes in all related objects.
- Fetch object A back.
- Make some changes to object B using its in-memory heap reference.
- Save object A using deep save.

In this scenario, the modifications to object B might be not saved! Under these conditions, if application expects implicit modifications of objects in the application context after they are saved to be noted, it must not use fetch policy DISCARD_AFTER_FETCH. It should use the policy REFETCH_AFTER_FETCH.

Changes to Java Mappings In Jalapeño

In this version, the mapping of Java integers has been changed from the Caché datatype class, %Integer, to %EnumString. This makes the handling of logical values used in ODBC more intuitive.

Note: This is the default behavior and may be overridden by an `@Extends` annotation.

Use Most Specific Types In Datatype Collection Properties For ActiveX and C Bindings

Before this release, collection properties were projected as collections of strings in the dynamic C++ binding. For example, the property

```
property sysListColn as list of %List;
```

was projected as a collection of strings. This had the unfortunate consequence that if any element of the collection was incorrectly converted to a %String, it could corrupt the entire %List.

Now the dynamic class definition that is returned for the type of a collection property reflects the correct collection element type. For example, the meta information for method GetAt() of the dynamic class definition associated with sysListColn now says that the type id of the returned value is D_LIST_ID.

Note: This change is off by default for backward compatibility. To turn it on in C binding: call **cbind_set_typed_colns()**. To turn it on in CacheActiveX: call **.factory.SetOption("TypedCollections", 1)**

Preserve The Value Of Decimal Values Used In C++ Queries

In prior versions, `d_query` represented `d_decimal` to ODBC as SQL_C_DOUBLE; now it uses SQL_C_CHAR. The string conversion preserves the exact value of the number.

11.2.2.8 SQL Changes

Parenthesized Literal Replacement Following An ID

Because of changes in the SQL parser, in some rare cases, constants that were manually surrounded with parenthesis to prevent constant replacement, would now have to be surrounded by two pairs of parenthesis to achieve the same effect. That is, instances like

```
...
WHERE f4 = ('Hello')
...
```

should be changed to

```
...
WHERE f4 = (('Hello'))
...
```

For backwards compatibility, in certain contexts, a single pair of parenthesis will continue to work as prevention for literal replacement, for example:

```
...
SELECT TOP (5) f6
...
```

Before this change, literal replacement was done even for a parenthesized literal if the parenthesis followed the ID "IN". If you wanted to prevent literal replacement in that case, you would enclose it in 2 pairs of parenthesis. The logic was changed to replace parenthesized literals that follow any identifier, except for the following: TOP, SELECT, LIKE, WHERE, ON, AND, OR, NOT, BETWEEN, %STARTSWITH, CASE, WHEN, and ELSE.

Dynamic SQL Supports Statement Use In Files

The Dynamic SQL Shell now supports **LOAD** and **SAVE** commands to load and save SQL statements from/to files. **SAVE** was used previously to save the currently prepared statement to the statement global. It has been redefined. To save to the statement global, it is now necessary to use **SAVEGLOBAL** or the abbreviation, **SG**.

LOAD will load the contents of the specified file into the current statement buffer and prepare the statement. If *EXECUTEMODE* is `IMMEDIATE` and the statement is successfully prepared then it will be executed.

SAVE will save the currently prepared statement to a file. If the file specified already exists then the user is prompted to overwrite it. If the user chooses not to overwrite, an error is reported and the statement is not saved.

Dynamic SQL And CALL Statement

Prior to version 2010.2, embedded SQL did not support a **CALL** statement when the target was not a function. Functions were callable using embedded SQL. Any application that uses **CALL** for a function will have to be modified in beginning with this release. If the called procedure returns a result set, applications must use the new %SQL.Statement class. If the called routine is a function, and it was supported in early versions, applications should use "SELECT <SQL Routine> <arguments>" instead.

Import Utility For Sybase and MS SQL Server Reimplemented

SQL statement import for Sybase and MS SQL Server has been reimplemented. Sybase and MS SQLServer statements are now processed using Dynamic SQL and the results of preparing the statement and executing the prepared statements are logged to a file and optionally echoed to the current device. The interface has not changed, but the dialog has changed slightly. In addition, the log file format is different and the parser used is the TSQL parser. That means more syntax is handled but it also means that errors reported will be different. For successfully processed statements the end result should be the same.

CALL Is Restricted To Defined Procedures

In this release, a loophole has been closed where a class method/query could be called as a stored procedure with the SQL **CALL** statement, even if the method/query is not specified as a SqlProc. This may require a change to the class definition

if a class method/query is called from an SQL **CALL** statement that is not defined as an SQL procedure; its definition will need to change to declare it as such.

Changes To Handle Queries Against Views With UNIONs

In this release, a problem has been corrected involving queries against a view that has a union as part of the query, for example:

```
SELECT 1 UNION SELECT 'one'
```

The returned query metedata will now report a type for column one of the query as VARCHAR instead of INTEGER.

Use of %vid For Row Selection

As an alternative to TOP, this version introduces a new way to restrict the set of returned rows of a query. This release extends %vid to views and FROM clause subqueries. It represents a sequentially assigned row number. Thus, to get rows 5 through 10 of an arbitrary query, say:

```
SELECT *, %vid FROM (SELECT ....) v WHERE %vid BETWEEN 5 AND 10
```

Note: The phrase "SELECT * ..." does not include %vid; it must be selected explicitly: "SELECT *, %vid ...". Also, while this feature is very convenient, especially for porting Oracle queries (this maps easily to Oracle ROWNUM), performance of queries may change as compared to TOP.

SQL Statement Property Change

The *%Language* property of %SQL.Statement is now named *%Dialect*. The SQL Shell LANGUAGE option is now named DIALECT.

Informix Migration — SUBSTR Function

In previous versions, when a **SUBSTR** function was discovered in an SQL context, the third argument was incorrectly passed as the end position. In this version, the SQL **SUBSTR** function correctly accepts a length as the third argument.

TSQL Unsupported Functions

The TSQL system functions, **IS_SRVROLEMEMBER**, **IS_MEMBER**, and **ServerProperty** are not implemented by Caché TSQL. References to these functions are now reported as compiler errors.

TRUNCATE Collation Added

Caché SQL now supports a new collation called TRUNCATE. TRUNCATE is the same as EXACT, but the application may specify a length at which to truncate the value. This is useful when there is EXACT data that the application needs to index and the data is exceeds the maximum length allowed for a Caché subscript.

Like other collations that support a length argument, **TRUNCATE(len)** will truncate the exact value to "len" characters. If length is not specified for TRUNCATE, the collation will behave the same as EXACT. While it is technically supported, it may make definitions and code easier to maintain if you only use TRUNCATE when you have a length defined, and EXACT when you do not.

Like the other collations supported by Caché, **%TRUNCATE** can be used as a unary function in an SQL Statement. For example:

```
Set MyStringFirst100 = $extract(MyString, 1 ,100)
&SQL (SELECT ALongString
      INTO :data
      FROM MyTable
      WHERE %TRUNCATE(ALongString,100) = :InputValue)
```

When using TRUNCATE in a Map Script expression of a %CacheSQLStorage map definition, define the subscript using **$$TRUNCATE**. For example, the map subscript expression may be:

```
$$TRUNCATE({MyLongStringField}, 100)
```

Changes To $SYSTEM.SQL.TOCHAR And $SYSTEM.SQL.TO_CHAR

$SYSTEM.SQL.TOCHAR(<null>) and $SYSTEM.SQL.TO_CHAR(<null_value>) will now return NULL and not 0 for numeric-to-character conversion.

Support Optional Second Argument Of %inlist() To Provide A Selectivity Hint

The %inlist operator can now also be used as a function with optional second argument. This is intended to give an order of magnitude estimate of the number of elements involved in the query. Thus, using a small number of different cached queries, you can get different plans for different cases, for example:

- small lists – %inlist <list> SIZE ((10))
- medium lists – %inlist <list> SIZE ((100))
- large lists – %inlist <list> SIZE ((1000))
- huge lists – %inlist <list> SIZE ((10000))

The second argument must be a constant when it is compiled. From clients except embedded SQL, this means that parentheses must be used as in the example above.

Changes In TO_CHAR Handling

In this version of Caché, TO_CHAR has been enhanced in order to support conversion of logical %Time values to String values. If the value for the expression top be converted is a numeric value and the format contains only the following TIME related format codes:

- HH – Hour of Day (1 through 12)

  HH12 – Hour of Day (1 through 12)

  HH24 – Hour of Day (0 through 23)

  MI – Minute (0 through 59)

  SS – Second (0 through 59)

  SSSSS – Seconds since midnight (0 through 86388)

  AM – Meridian Indicator (before noon)

  PM – Meridian Indicator (after noon)

The expression will be treated as a logical %Time value and not a Logical %Date value. For example, the selection

```
SELECT TO_CHAR($piece($horolog, ',' ,2), 'HH12:MI:SS PM') AS THE_TIME
```

will result in *THE-TIME* having a value formatted as something lilke `11:43:26 AM`.

Evaluation Of Macros And Functions In SQL Preprocessor

Beginning with this release, there has been a change in the behavior of the DDL CREATE PROCEDURE, CREATE FUNCTION, CREATE METHOD, CREATE QUERY, and CREATE TRIGGER statements, when compiled as embedded SQL statements or prepared as dynamic statements. This change is not fully backward-compatible and may require modifications to your applications, especially when code bodies of type ObjectScript are used in the CREATE statement.

The macro preprocessor evaluates # commands, ## functions and $$$macro references before any embedded SQL statement is processed. Consider the following statement:

```
&sql(CREATE PROCEDURE SquareIt(in value INTEGER) RETURNS INTEGER
                      LANGUAGE COS {
                          #define Square(%val) %val*%val
                          QUIT $$$Square(value)
                          }
    )
```

Prior to this change the #define and $$$Square macro references would be expanded and processed when the CREATE PROCEDURE statement was compiled resulting in a method declaration as follows:

```
ClassMethod SquareIt(value As %Library.Integer(MAXVAL=2147483647,MINVAL=-2147483648))
            As %Library.Integer(MAXVAL=2147483647,MINVAL=-2147483648)
            [ SqlName = SquareIt, SqlProc ]
{
    QUIT value*value
}
```

With this change the processing and expansion will be included in the procedure method definition, and get processed and expanded when the method is compiled:

```
ClassMethod SquareIt(value As %Library.Integer(MAXVAL=2147483647,MINVAL=-2147483648))
            As %Library.Integer(MAXVAL=2147483647,MINVAL=-2147483648)
            [ SqlName = SquareIt, SqlProc ]
{
    #define Square(%val) %val*%val
        QUIT $$$Square(value)
}
```

Code that relies on the old behavior of the macro expansion occurring during the compilation of the CREATE PROCEDURE statement will have to be changed. Alternatively, use %SQL.Statement to prepare and execute the CREATE PROCEDURE statement dynamically.

Finally, in prior releases, ObjectScript program code is enclosed within curly braces, for example, { code }. If material needs to be included, the #include preprocessor command must be prefaced by a colon and appear in the first column, as shown in the following example:

```
CREATE PROCEDURE SP123()
  LANGUAGE OBJECTSCRIPT {
:#include %occConstant
}
```

Beginning with this release, the leading colon (:) is no longer required, but it will be accepted without error if present.

Corrections To Date/Timestamp Comparisons And SQL Categories

The datatype classes %Library.Date,%Library.TimeStamp, %Library.FilemanDate, %Library.FilemanTimeStamp, and %MV.Date are now treated as follows with regard to SqlCategory:

1. %Library.Date classes, and any user-defined datatype class that has a logical value of +$HOROLOG should use DATE as the SqlCategory.
2. %Library.FilemanDate classes, or any user-defined datatype class that has a logical date value of CYYMMDD, should use FMDATE as theSqlCategory.
3. %MV.Date classes, or any user-defined datatype class that has a logical date value of $HOROLOG-46385, should use MVDATE as theSqlCategory.
4. %Library.FilemanTimeStamp classes, or any user-defined datatype class that has a logical date value of CYYMMDD.HHMMSS, should use FMTIMESTAMP as theSqlCategory.
5. A user-defined date datatype that does not fit into any of the preceding logical values should define the SqlCategory of the datatype as DATE and provide a **LogicalToDate** method in the datatype class to convert the user-defined logical date value to a %Library.Date logical value.
6. A user-defined timestamp datatype that does not fit into any of the preceding logical values should define the SqlCategory of the datatype as TIMESTAMP and provide a **LogicalToTimeStamp** method in the datatype class to convert the user-defined logical timestamp value to a %Library.TimeStamp logical value.
7. Finally,
   - The SqlCategory of %Library.FilemanDate is now FMDATE.
   - The SqlCategory of %Library.FilemanTimeStamp is now FMTIMESTAMP.

- The SqlCategory of %MV.Date is now MVDATE.

This version also changes the outcome of comparing FMTIMESTAMP category values with DATE category values. Caché no longer strips the time from the FMTIMESTAMP value before comparing it to the DATE. This is now identical behavior to comparing TIMESTAMP with DATE values, and TIMESTAMP compared with MVDATE values. It is also compatible with how other SQL vendors compare TIMESTAMPS and DATES. This means a comparison of a FMTIMESTAMP 320110202.12 and DATE 62124 will no longer be equal when compared with the SQL = operator. Applications must convert the FMTIMESTAMP to a DATE or FMDATE value to compare only the date portions of the values.

Datatype Of A CASE Expression

When using a CASE expression, if any of the potential return values is of type LONGVARBINARY, the return value of the CASE will be of type LONGVARBINARY; otherwise, if any of the potential return values is of type LONGVARCHAR, then the return value of the CASE function will be of type LONGVARCHAR. After that, the datatype of the value will be the first applicable from among: VARBINARY, VARCHAR, TIMESTAMP, DOUBLE, NUMERIC, BIGINT, INTEGER, DATE, TIME, SMALLINT, TINYINT, BIT.

11.2.2.9 CSP Changes

Error Handling While Changing CSP Applications

An event class can be attached to a CSP application. In particular, a callback is made when the session moves from one application to another. According to the design intent, if an error code is returned, the flow will be redirected to the error page.

In previous releases, this error code was “partially” ignored. If an error code was returned, the user was redirected to the error page. However, if the application change completed and the user was actually in the target application, pressing the reload button would display the target page. Now, the error code aborts the application change. The user sees the same error page appear, but pressing the reload button redisplays the error page.

Sticky Login And Login Tokens For Authenticating CSP Applications

CSP logins are now “sticky”. When reentering a previously-entered application, that application will be the same user as on the exit. (Previously, when sharing sessions, the re-entered user would depend on what other applications had been visited.) When you log in as X, a login token is sent in a 'most-recently-logged-in-user' cookie to the browser. When you enter an application for the first time, if login tokens are enabled for the application, the CSP Server will attempt to log you in using that cookie.

All applications in a session now move in tandem. Logging into a new user within an application moves all applications to that user. Logging out a session, logs out all applications in that session.

Allow CSP Applications To Be Grouped By Browser

Caché has two types of Authentication Groups: by-session and by-browser. CSP applications within a group attempt to keep their authentication in sync when possible. All applications are in an Authentication Group. The default authentication group for an application is by-session. (So applications all by themselves in a Session form a single-entity authentication group.) Explicit grouping takes precedence over implicit. So if a group-by-browser application is forced into a session with some other applications, it will share authentication by-browser, not by-session, with the other applications.

Session-Sharing Depends On Exact Cookie-Path Match

In previous versions, two applications were included in the same session if, when entering an application, the Session-Path-Cookie of the first application was a prefix of the previous one. This rule introduced an inconsistency. If the applications were entered in the opposite order, they were in different (instead of the same) session.

Now, applications can be made to share (run in) the same CSP session if their Session-Path-Cookie matches exactly. Applications sharing a session can share data in the Session object and when possible, keep their authentication in sync (remain logged into the same user and are logged out as a unit.)

Note: The global ^%SYS("CSP", "UseLegacySessionSharing") can be set to 1 to return to old-style session sharing.

Session Events And Security Context Management

There are two important changes to session event classes and their security contexts made in this release. First, a CSP session can have multiple classes notified when an event occurs. That is, the statement

```
SET %session.EventClass = "User.MyClass"
```

will add User.MyClass to the list of classes to callback, and a statement such as:

```
SET %session.EventClass = "User.MyClass1", %session.EventClass = "User.MyClass2"
```

will add both classes to the list. This behavior is both easy to explain and it assures an application does not accidentally remove existing event classes , preventing them from being able to run cleanup code which may result in a resource leak.

Second, when moving between CSP applications in the same session Caché automatically adds the event classes of the new application to the list. In previous releases, Caché ignored the new CSP applications event class preventing cleanup of temporary data created in the namespace associated with this application.

Fix CSP Language Match For * In ACCEPT_LANGUAGE

If the CGI variable, *HTTP_ACCEPT_LANGUAGE*, has a value of "*" (which means any language), and with the same quality rating as a specific language, then use the specific language. This implements the HTTP 1.1 rule: The language quality factor assigned to a language-tag by the Accept-Language field is the quality value of the longest language-range in the field that matches the language tag.

Note: The default quality factor is 1 even for "*".

Change Cookie Timeout Name

In order to distinguish between different timeouts for various sessions, processes and so on, and the expiration time of cookies controlled by the browser, this release has changed the name, "cookie timeout" to "cookie expire time".

11.2.2.10 XML Changes

%XML.DataSet Will Now Use Class/Property Metadata When Available

Before this version, %XML.DataSet only used SQL metadata from the query being run. In particular, this meant supporting only the xsdtype of the base datatype from the %xsd. package and not supporting property parameters (such as VALUELIST for properties) and overrides of **XSDToLogical** and **LogicalToXSD**). Now, %XML.DataSet looks at the class name and property name metadata for a column when it is available (it is not always available, for example, if the column is an expression). This change will affect applications that want to just use the SQL-based data.

Column Names With Embedded Spaces

Beginning with this version, %XML.DataSet will convert embedded spaces in column names to underscores.

Web Add-On Declaration Required

If an application wishes to be identified as an anonymous web application eligible to run under the terms of the Web Add-on license, it must call **$SYSTEM.License.PublicWebAppUser()** to identify itself as such.

11.2.2.11 Web Services Changes

SOAP Fault Handling

In this version, Caché now returns SOAP 1.1 faults with an HTTP status code of 400 for client errors, and 500 for server errors as defined by WS-I Basic Profile. SOAP 1.2 faults already conformed to this use of 400 and 500 status codes. Furthermore, SOAP calls **OnInternalFault** for all %Status-based faults produced by the **Initialize** method. **OnInternalFault** was already being called for %Status based faults produced in other places in the code.

If a client is expecting a status code of 200 for faults, then this will no longer work and the client application must be changed.

Wizard No Longer Generates SOAP Headers

The SOAPHEADERS parameter will no longer be generated by the SOAP wizard. Instead the parameters *XData* block will be used to specify which headers to expect at the method level based on the WSDL.

Additional header information for a web service or web client is added in an *XData* block in the web service or web client class. The *XData* block has the parameters element in the configuration namespace as the root element.

Maximum Method Name Length

In this release, SOAP sets the length of method names to 180 characters. Some methods may end up with different names when recreated by the SOAP wizard because truncation no longer needed.

Do Not Close Streams

Beginning with this release, the files that implement the file streams used by SOAP web service and web client will not be closed until the streams that use them are closed.

11.2.2.12 BASIC And MVBASIC Changes

Alterations To Line Continuation Processing

BASIC and MV BASIC line continuation characters allow a source line to span multiple lines. Placing a line continuation character as the last character on a line continues that statement on the next line. The BASIC line continuation character is an underscore (_); depending on the emulation options, MVBASIC can use either a vertical bar (|) or a backslash (\)as a line continuation character.

Previous versions of BASIC and MVBASIC would sometimes allow a line continuation character to appear at places other than the last character of the line. However, this could cause problems if the line continuation character had an alternative use for another purpose. Beginning with this release, the continuation character must be the last character of the line.

11.2.2.13 xDBC Changes

Removal Of All Support For XA

This change removes some experimental XA code from the InterSystems JDBC driver. Caché does not support the XA protocol, nor did the JDBC driver. However, as this was seriously considered a number of times, some experimental code was added to test whether JDBC could fully support it one day. This feature was documented as unsupported, and InterSystems has now decided to remove this dead code as part of an overall Java cleanup.

Changes To Catalog Queries

The TABLE_TYPE argument for the ODBC Catalog Query SQLTables and the JDBC Catalog query getTables has been enhanced to support the following new types in addition to the types 'TABLE' and 'VIEW' that have always been supported:

- `SYSTEM TABLE` – a table projected from a class that has a System > 0 setting
- `SYSTEM VIEW` – a view projected from a class that has a System > 0 setting
- `GLOBAL TEMPORARY` – a table projected from a class with class parameter:

```
Parameter SQLTABLETYPE = "GLOBAL TEMPORARY";
```

Prior to this version, if an application called SQLTables or getTables with an empty TABLE_TYPE argument, only TABLE and VIEW types would be returned. Now all all types that exist in the catalog will be returned. If an application only wants the TABLE and VIEW types, it must be changed to pass in only 'TABLE', and 'VIEW' for the TABLE_TYPE argument.

11.2.2.14 MultiValue Changes

MVFILENAME Class Parameter

The presence of *MVFILENAME* corrects inconsistency issues that previously might occur between copied or imported classes, and the file references in the VOC. The use of *MVFILENAME* assures that the storage definition in a class that extends %MV.Adaptor closely follows the definition of the MV file. InterSystems strongly recommend the use of *MVFILENAME*.

Debugger Changes For D3 Emulation

When using the Studio debugger in a MultiValue account using D3 emulation, if the debugger does not find a value for a mixed or lower case variable name it will look for an uppercase name. This will help when debugging routines that use the D3 default behavior of converting all variable names to uppercase. This can cause confusion if

```
$OPTIONS -NO.CASE
```

turns off the default for a routine and it uses two variable names that differ only in the case of the names. D3 routines compiled with case sensitivity and with variable names identical except for case may see unexpected values in the debugger.

This change applies only to displaying variables. To modify a value, the true uppercase name must be used.

SUM Verbs Changed To Return Info Via Error Messages

In this version, the SUM verb now generates its results in the form of error messages. Previously, it presented its results via the RETURNING clause.

Behavior Changes For Dynamic Arrays With Unassigned Entries

The current behavior of padding the MATBUILD/MATWRITE dynamic array with null entries for unassigned array nodes at the end is no longer supported. No legacy MultiValue system provides this behavior, so InterSystems believes no existing applications depend on it.

In prior releases, unassigned nodes would be treated as empty strings and the resulting dynamic array could have many empty entries at the end if the higher array subscripts were undefined. Beginning with this release, the behavior of MATBUILD and MATWRITE has changed for arrays that have unassigned nodes. Now the behavior depends on an emulation option, which is set to match the behavior of existing legacy platforms, namely:

- Cache, Universe, and Unidata emulations

  Empty strings will be used for unassigned nodes and the dynamic array will be truncated when the highest subscripts of the array are unassigned.

- jBase, Reality, and D3 emulations

  An <UNDEFINED> error will be thrown when an unassigned array node is encountered.

The default behavior based on the emulation type may be overridden with the a compile option.

```
$OPTIONS MATBUILD.UNASSIGNED.ERROR
```

will cause an error to be thrown, while

```
$OPTIONS -MATBUILD.UNASSIGNED.ERROR
```

will use an empty string and truncate any trailing unassigned nodes.

Timed INPUT And AUTOLOGOUT Behavior Change

In previous releases, **AUTOLOGOUT** was found to be unpredictable. In this release, the **AUTOLOGOUT** and timed **INPUT** statements are consistent. If an application contains a timed-out **INPUT** command, the behaviour has changed slightly. For example, if the timeout was 30 seconds as in

```
INPUT var FOR 300 ELSE ...
```

then in prior releases, the 30 seconds applied to the entire INPUT statement. Starting with this release, every time a key is depressed during the **INPUT** statement, the timeout is reset to another 30 seconds. Similar behavioral changes also apply to **AUTOLOGOUT**; each keystroke will reset the **AUTOLOGOUT** timeout.

jBase And Undimensioned Arrays

In this release, an undimensioned array reference will now be a compile error in JBASE emulation. Previously, it was treated as an implicit FMT operation.

jBase CommandInit And CommandNext Changes

Prior to this release, to call the routines **CommandInit** and **CommandNext** (or **JBASECommandInit** and **JBASECommandNext** in later releases of jBASE), you had to create an F pointer into the samples directory that was provided with the Caché installation and compile and catalog **CommandInit** and **CommandNext**.

Beginning with this release, the routines **CommandInit** and **CommandNext** (and their newer equivalents **JBASECommandInit** and **JBASECommandNext**) will be supplied by default with Caché; no compilation is required by the customer prior to use. The sources for these 2 routines will no longer be provided with the release of Caché as they are no longer needed.

The values returned by **CommandNext** are still defined in a file called CommandInclude and this might still be needed by the customer to decode the returned value. This source will continue to be included with the Caché release at the same location as before,that is,

```
<install_directory>/dev/mv/samples/CommandInclude
```

When calling the **CommandNext** routine, the third parameter is the timeout value. Caché now supports the following timeout values:

- timeout < 0: the routine returns immediately
- timeout = 0: the same as timeout = 1
- timeout > 0: Number of tenths of a second to wait until Caché returns a timeout value.

Note: Windows platforms only support whole seconds for wait times, not fractions. Therefore, a timeout of 1 means one second.

jBase CHAIN Handling

The MVBASIC CHAIN statement was not always correctly handled when doing jBASE emulation. CHAIN under jBASE emulation should not pass the default list (list 0) if that list is local or modified. When MVBASIC was initiated by an EXECUTE statement, then a CHAIN would always pass the default list. Now, the passing the default list is always disabled for programs compiled under jBASE emulation.

Evaluation Order For Boolean Operators

The evaluation order of the Boolean-and operators (AND, “&”) versus the Boolean-or operators (OR, “!”) in MVBASIC has been changed to agree with the MVBASIC specification. Previously, the MVBASIC conjunction operators (AND, “&”) had higher precedence than the disjunction operators (OR, “!”). Without parentheses an AND operator would be evaluated before an OR operator. Now the AND and OR operators have equal precedence; they will evaluated in left-to-right order in the absence of parentheses.

Handling Of Division By Zero

Beginning with this version, the MVBASIC **DIVS()** and **MODS()** array functions now signals a <DIVIDE> error if either encounters a divide-by-zero error. This will end execution of the DIVS() function and execution will start searching for a trap handler. Previously, a divide-by-0 during a DIVS() array operation resulted in a message being sent to the operator console log, a 0 being used for the array component in error and execution of the array divide continuing with the remaining elements.

List Collections For MVENABLED Class Cannot Be Empty

The index on a list collection in an MVENABLED class will always contain at least one entry. When the collection is empty, an indexed element value of NULL, and key value = 1 will be inserted into the index.

DEFFUN CALLING Syntax Changed

The syntax of the CALLING clause of a DEFFUN statement has been changed. The CALLING keyword can now be followed by either an identifier or a quoted string literal. If a quoted string literal is used, the first character of that quoted string literal may be an "*" character. If the leading character is an "*:"

- Under Unidata emulation, when a name begins with a leading "*", the leading * is removed and the name is looked up as a global name. It is an error if the global name is not found.
- Non-Unidata emulations allow a function to have a leading "*" character in its name but the leading "*" does not modify the function name lookup rules in these other emulations.

PRINT ON <Channel> Is Now Emulation-Specific

This release adds MultiValue emulation differences for the use of the statement

```
PRINT ON channel {EXPRESSION}
```

Prior to this, the output would always go to a spooler print job regardless of the use of PRINTER ON or PRINTER OFF. Now, the action depends upon the emulation. For some emulations (Reality, jBASE, D3), the output will go to the screen if the application has not executed a PRINTER ON statement. For other emulations, the behavior remains the same, that is, output to a spooled job.

SPOOLER(2) Function Return Changed

In previous releases, the SPOOLER(2) function call in MVBASIC returns 3 fields that are the same, for compatibility reasons. Beginning with this release, one of the duplicated fields (field 15 or MultiValue 15) is now the Caché user name that a user will log in as when they initially connect to Caché in a locked-down security-enabled system. The following values/fields are now of interest as they share related information:

- 3 – user name; OS login name = Fred
- 14 – user name; OS login name, same as 3 = Fred
- 15 – Caché user name in a security-enabled locked-down system, or "UnknownUser"
- 17 – MV account name; = USER

Support Universe/Unidata Behavior Of Data Stack In PROC P Command

The PROC P command now respects the emulation flag STACK.GLOBAL, so that if set, the data stack is not cleared when a PROC executes the P command but rather the secondary output buffer is appended to it.

SSELECT Result Ordering Change

In previous releases, the MVBasic SSELECT statement provided the same ordering as the SELECT statement: theCaché default collation ordering. Caché default ordering places the empty string first, and then places the canonical numbers before all other strings.

Beginning in this version, it now sorts a list created from file or another select list into ordinary string collation order. If a programmer wants numeric strings sorted into numeric order, then SELECT should be used instead of SSELECT. If the input to a SSELECT statement is a list variable that includes duplicated values, the duplicates will be replaced by a single value as part of the sorting process. Thus, SSELECT of a list variable may generate a new list variable with fewer elements.

Note: Both SELECT and SSELECT with an MVBasic dynamic array as input do not sort the array. The elements of the list occur in the same order as the elements occur in the dynamic array.

Changes to MATBUILD And MATWRITE Behavior For Unidata

For Unidata emulation, MATBUILD and MATWRITE now handle trailing empty values differently. MATWRITE will truncate them; MATBUILD will not.

Dynamic Vector Arithmetic Changes

In earlier releases, when asked to perform a divide by the value 0,

- the DIV() and MOD() functions, and the / division operator issued a <DIVIDE> error; but,
- the DIVS() and MODS() functions (dynamic array vector arithmetic) returned a value of 0.

Beginning with this release, all these operation will perform the same way returning a <DIVIDE> error.

PHANTOMs Now Run The Login Verb

The handling of PHANTOMs has changed in this release. When a PHANTOM starts, it will now execute the LOGIN verb. PHANTOPMs now also supprt the CHAIN and EXIT commands.

11.2.2.15 Zen Changes

Change To showModalDialog Function In Zen For Internet Explorer

Zen defines a utility function, **zenLaunchPopupWindow**, that creates a popup window. One of the options it supports is "modal=true". In prior releases, this function would detect Internet Explorer and use the special IE-extension, showModal-Dialog. This function has proved to be unreliable for his purpose. In this release, Zen implements modal behavior as follows:

1. When **zenLaunchPopupWindow** is called in modal mode, Zen turns on the modal handler for the page, and places a transparent area over the entire page and trap any mouse clicks not in the modal area.
2. **zenLaunchPopupWindow** sets up surrogate callbacks to trap the end of modal behavior. Specifically, when the user clicks outside of the modal popup, Zen gives focus back to the popup.

Enforce Restrictions On seriesNames And labelValues In pieCharts

The following restrictions are now enforced begiinning with this release:

- A seriesNames CANNOT contain literal values in a pieChart.
- A seriesNames cannot contain more than one xpath expression in a pieChart.
- A labelNames cannot contain an xpath in a pieChart.

Rather, use seriesNames if one needs to use an xpath in a pieChart; use labelNames if one needs to use literal values in a pieChart.

Changes To xyChart

When introduced in version 2010.2, colors and marker shapes were determined ordinally in a repeating pattern. This proved to have limited utility. Now, Zen gives points belonging to the same dataField in an xy-Chart the same marker and color.

Note: In seriesNames, the x-coordinate dataField mustbe named so that the legend is marked correctly, even though the x-coordinate does not appear in the legend. It is the (x,y) combinations that appear in the legend and they are colored according to the color the y dataField has in the legend.

Do Not Put "/" Before Report Name In Generated <apply-templates>

To allow a use-case of sub-reports where each sub-report provides a sub-section of the data, Zen no longer inserts a "/" in front of the report name as that name appears in the <xsl:apply-templates> element. This aligns how the PDF works with how HTML works.

This allows a use-case that works in the HTML case to work in the PDF case, but there may be PDF reports that are relying on the <apply-templates> selecting "/reportname" rather than "reportname". The solution for these ZEN Reports is to explicitly put a "/" in front of reportname.

SVG Chart Component

This version introduces a major rework of the way axes are labeled in the chart class (parent of bar chart and line chart) It introduces a new flag, autoScaleText which defaults to true.

When set, the axes renderers mostly mimic the previous chart behavior. In this scenario, text scales with the drawing becoming arbitrarily large and potentially distorted if the aspect ratio of the chart changes. The only significant change in this mode is that, if the X axis has too much text along it (in danger of overprinting labels), it will automatically attempt to angle the text (if a label angle has not already been specified) and, in the case of a value axis, will omit some labels in the name of clarity. If the axis is plotting category names, all labels will print, but the user may need to play with font and rendering sizes to ensure legibility (not unlike the previous release).

If autoScaleText is false, the text scales (or not) independent of the size of the chart itself. In practice, this means that font sizes effective specify maximum font sizes (fonts may still scale smaller if need be on very tiny or crowded graphs). This approach allows Zen to use space more efficiently as graphs are resized and “normal” aspect ratios of text are preserved at different rendering scales.

The labeling of axes also changes under this mode. For the vertical axis, if the dimension is a continuous value scale, the system will automatically decimate the printed label set to convey maximum information in the space allotted without overprinting. For categories plotted on the vertical, labels will be shrunk to fit as needed, but no label will be omitted. On the horizontal axis, labels will first be angled in an attempt to fit more information across the line. If this fails to fit all labels into the available space, then range value labels will be decimated while category labels will be shrunken.

Selection Focus For simpleTablePane

This version alters the base behavior for the selection focus of tablePanes. Previously, clicking on a given row selected it (set the *selectionIndex* property and highlighted the row); subsequent clicks on the selected row were ignored. The only way to change the selection focus was to select a different row in the table. This had the effect of making it impossible to unselect _all_ rows once a selection had been made.

Under the new system, clicking on a selected row, unselects it (the *selectionIndex* for the table is set to -1 and no rows are highlighted). Changing the selection focus works as before, the core difference is the toggling behavior of the currently selected row.

In addition, this version implements an `onunselectrow` callback mechanism, allowing page designers to be notified when a given row is unselected. This event fires both when toggling a single row and when changing the selection focus from one row to another. In the multi-row case, the `unselectrow` event is fired FIRST with a *selectionIndex* of -1. Once handled, the `selectrow` event fires with the *selectionIndex* set to the newly selected row number.

Important: Pages that display information about the currently selected row outside of the table itself (such as in a text box) based on an `onselectrow` callback will display stale information if the current row is toggled off and the page hasn't been updated to clear the old info. The solution is simply to listen for the `onunselectrow` event and clear the supplemental widgets.

Changing Handling Of Invalid Classes Used With Form Controls

In preceding versions, invalid values entered into forms would be tagged with the DOM node classname `zenInvalid` at the expense of any existing class name given via the *controlClass* attribute. All modern browsers, however, support a node belonging to multiple classes simultaneously so this either/or behavior is not necessary and, may actually disrupt the geometry of the page.

This version allows the `zenInvalid` class membership to supplement and existing class designation rather than wholly supplant it. This does introduce the issue that if the both the *controlClass* and `zenInvalid` attempt to set the same CSS style (such as the background color of a text box, which `zenInvalid` wants to turn red), the question of which rule takes

precedence becomes a browser dependency. The easiest way to avoid this issue is to ensure that the CSS rules set for the `zenInvalid` class do not directly compete with any styles associated with developer specified *controlClass* designations.

11.2.2.16 Studio Changes

Save No Longer Compiles

Studio no longer supports "Compile On Save" functionality. The **Save** command will always perform a save operation only, but no compile will be executed. Users must manually select the compile options. The option has been removed from the compiler behavior settings.

Export And Import All Settings From A Project To XML

Before this, Studio only exported certain setting from a project to the XML export format, and only imported specific settings. This has now been changed to export/import all the useful settings to XML.

Note: This change is fully compatible with importing XML exports of projects from older versions of Caché. However, the new export outputs many more fields. Importing the XML into an earlier version of Caché will probably fail validation. This can be worked around by passing the '-i' flag to turn off schema validation during the import so that only the fields that the older system knows about will be imported.

11.2.2.17 %Installer Changes

<Database> Create Attribute Correction

In prior versions, the <Database> tag did not properly handle the value of the Create attribute. It did not distinguish between values of "yes" and "overwrite"; both behaved as if "overwrite" had been specified. In this release, the operation has been corrected to match the documentation for the attribute.

12

Caché 2010.2

This chapter provides the following information for Caché 2010.2:

- New and Enhanced Features for Caché 2010.2
- Caché 2010.2 Upgrade Checklist

12.1 New and Enhanced Features for Caché 2010.2

The following major new features have been added to Caché for the 2010.2 release:

- Rapid Application Development
  - Java Event Processing
  - WS Policy
  - Soap Configuration Wizard
  - MultiValue Indexing Improvements
- Performance and Scalability
  - DB Expansion Improvements
  - Large Local Arrays
  - Improved Dejournal Performance
  - $PIECE Performance Improvements
- Reliability, Availability, Maintainability, Monitoring
  - Caché Database Mirroring
  - WS-Management
  - Database Compaction
  - Cache Database
  - Caché Database Blocksize Conversion
- Security

- Two-factor Authentication
- Support for OpenAM
- Encryption Of Shadow Files

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

12.1.1 Rapid Application Development

12.1.1.1 Java Event Processing

This version of Caché introduces several new high performance Java integrations. Each is enabled by the modular in-process Caché kernel. Java can now communicate with Caché on the same computer process-to-process enabling extremely high data access rates.

A standard JNI (Java Native Interface) is used for the Java and Caché integration. In Java JDK 1.6 and beyond, we use the Java NIO features to leverage the evolution of speed and stability in JNI and thus provide:

- MDS – Multi-dimensional data storage

  This is a raw low-level API to access Caché globals. It is the fastest way to do data access (store / retrieve) between Java and Caché. The API provides support for transactions, locking, iterators, and other features; however, it does not directly enable SQL or object support. The API can be used in multi-threaded environments.

- XEP – Extreme Event Persistence

  XEP provides a lightweight Object API on top of MDS. It enables an application to use a lightweight restricted SQL dialect for fast queries and also provides traditional full SQL in support of complex queries. XEP makes use of the transaction and multi-threaded support built-in via MDS.

- In-Process JDBC

  This feature enables JDBC to run over JNI instead of TCP when the Java virtual machine and Caché are running on the same machine.

12.1.1.2 WS-Policy

To complement the Caché implementation of WS-Security, and to support general security requirements, this release now provides the most common elements of the WS-Policy and allows users to easily define policies for existing Web Services.

12.1.1.3 Soap Configuration Wizard

This wizard assists in the creation of a configuration class that applies to a selected Web Service or Web Service client class. This configuration contains WS-Policy expressions that describe the capabilities and requirements of that service or client. These expressions can refer to WS-Security, WS-Addressing, and MTOM.

12.1.1.4 MultiValue Indexing Improvements

This version of Caché contains major changes to ensure better coordination between the MV index verbs (for example, CREATE.INDEX) and PROTOCLASS. The core property definition code from PROTOCLASS and CREATE.INDEX has been combined into a common class method, %SYSTEM.MV.createMVProperty to ensure that the same property definition is created by either facility. In addition, all functions, commands and verbs that use the name of an index will now use the real name of the index as a preferred alternative to the mangled name generated from the name of the source attribute. This includes the functions:

- INDICES(fv), and INDICES(fv, name)

and the commands:

- BSCAN
- OPENINDEX

Furthermore, errors in any of these should now produce better diagnostic messages.

CREATE.INDEX and **PROTOCLASS** will both add a new parameter to any properties that they create so that **DELETE.INDEX** and re-execution of **PROTOCLASS** know that the property was a generated property and can be modified or deleted by them.

12.1.2 Performance And Scalability

12.1.2.1 DB Expansion Improvements

Previously, the Expansion Daemon (EXPDMN) would expand a database by allocating new blocks in 64KB chunks. While this algorithm worked well for smaller expansions, InterSystems found that larger expansions (>50 MB) would take longer than expected; in some cases, it could interfere with the read performance on the system.

Now, for expansions larger than 1MB, the EXPDMN has been enhanced to not only issue larger writes, but to use a sliding-scale expansion algorithm. The new algorithm performs writes with the following schedule: two 1MB chunks followed by one 2MB chunk followed by 4MB chunks till the requested expansion is complete.

12.1.2.2 Large Local Arrays

In previous releases, the performance of operations on local arrays could worsen as the array grew larger. This version of Caché implements a highly optimized algorithm for handling large local arrays that greatly speeds up performance when saving and retrieving values from such large in-memory (local) arrays.

12.1.2.3 Improved Dejournal Performance

Dejournal performance has been improved in this version of Caché by reducing the amount of contention between dejournal pre-fetcher jobs. This is achieved mainly by providing each pre-fetcher job a larger list of journal entries to process. In addition, each pre-fetcher process now works on approximately 100 chunks of the journal file at a time.

12.1.2.4 $PIECE/$FIND Performance Improvements

The $PIECE and $FIND functions have been enhanced to speed up the process of finding the delimiter or the search string within a string by utilizing SSE2 instructions present on all x64 processor chips

12.1.2.5 ZEN Reports Using HotJVM

In this release, the handling of the Java Virtual Machine (JVM) for Zen Reports has changed. In prior versions, Caché would

- Instantiate an instance of a JVM in the process handling the request
- Run the rendering application in this JVM
- Collect the final PDF
- Terminate the JVM

Beginning with 2010.2, the system starts a dedicated back-end process along with the other CSP server-side daemons. This process instantiates a Java Virtual Machine (JVM) and handles all report generation tasks returning the result PDF. The

JVM is never closed. This approach (avoiding the startup and shutdown costs) results in significant reductions in the time needed to produce a report.

12.1.3 Reliability, Availability, Maintainability, Monitoring

12.1.3.1 Caché Database Mirroring

Traditional availability and replication solutions often require substantial investment in infrastructure, deployment, configuration, and planning. Caché database Mirroring is designed to provide an economical solution for fast, reliable, rapid, robust, automated failover between two systems, making mirroring the ideal automatic failover and high availability solution for the enterprise.

Caché Mirroring not only provides an availability solution for unplanned downtime, but also offers the flexibility to incorporate planned downtimes (for example, Caché configuration changes, hardware or operating system upgrades) on a particular system without impacting the overall SLAs for the organization. Combining ECP application servers with Caché Mirroring provides yet an additional level of availability; application servers simply treat a failover as an ECP server restart, and allow processing to continue on the new system once the failover is complete, thus greatly minimizing workflow and user disruption.

By maintaining independent components on the two failover systems, Caché Database Mirroring also reduces the potential risks associated with physical replication, such as out-of-order updates and carry-forward corruption, which are sometimes present in traditional shared-resource failover technologies.

Caché Mirroring also introduces the mirror shadow which can be configured either as a reporting shadow or a disaster recovery (DR) shadow – a reporting shadow can be used for enterprise-wide reporting and business intelligence (BI) purposes; a DR Shadow, which is an up-to-date read-only copy of the production system, can be used as an integral part of an enterprise disaster recovery and business continuity plan.

12.1.3.2 WS-Management

This release adds several Web Service methods to allow monitoring of remote Caché systems via Web Services. The methods are all contained in the SYS.WSMon.Service class, which also includes the details of the implementation.

12.1.3.3 Database Compaction

This version of Caché includes a Database Compaction utility which compacts a database file by moving free-space from within the file to the end of the file. Once this process is complete, it is possible to run the existing Database Truncation utility to return this free-space to the underlying file system.

12.1.3.4 Cache Database

In this version, InterSystems adds a new database to support operational storage requirements of the engine itself. The Cache database will be used to store information like cached queries, or session-related information. At startup, this database is mounted as read-write; no customer application should directly interact with it. Please refer to the System Administration information for further details.

12.1.3.5 Caché Database Blocksize Conversion

This release offers a new class method, **SYS.Database.Copy()**, that customers can call as part of their application deployment scripts. It allows the specification of the database blocksize of the target database, and is a fast and efficient way to upgrade database files from the now-deprecated 2K block format. For details and options refer to the **class documentation**.

12.1.4 Security

12.1.4.1 Two-factor Authentication

Starting with this version of Caché, InterSystems provides two-factor authentication as an added security feature during login for client/server applications, console, and terminal. With two-factor authentication, after authentication using the selected mechanism (such as Kerberos or Caché login), Caché sends a security token to the registered mobile phone of the user. To gain access to Caché, the user must then enter the token at a prompt. This adds the possession of the registered mobile phone as a second factor, along with the secrecy of the user's password as the first.

12.1.4.2 Support for OpenAM

InterSystems has added support to allow Caché to use the OpenAM single-sign on (SSO) component. By using this feature, users that have already successfully authenticated do not have to re-authenticate. InterSystems has demonstrated interoperability with the OpenAM web policy agent, which allows Caché to determine the centrally authenticated ID of a user from the REMOTE_USER environment variable.

12.1.4.3 Encryption of Shadow Files

This augments the data-at-rest encryption provided by Caché. Journal data on the shadow will now also be encrypted if the shadow system is set up to use journal encryption. The primary server and the shadow server do not need to use the same key to encrypt data-at-rest (for example, database files). Each system will use the local key for this form of protection.

12.2 Caché 2010.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2010.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2010.1 and 2010.2.

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

12.2.1 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 2010.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

12.2.1.1 Version Interoperability

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

12.2.1.2 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 addition of pages to assist with database mirroring.

12.2.1.3 Operational Changes

New Database: CACHE

In this version, a new database has been added to the standard set used by Caché. The name of the database is "CACHE" and it holds information such as cached SQL queries or CSP session information. Please refer to the System Administration information for further details.

Note: No customer application should directly interact with the CACHE database.

CAUTION: Those installations that have a pre-existing database with this name must change it, and the applications that reference it, before installing this release. If Caché discovers such a database at startup time, it will note this as an error message to the console log and shut down.

Collation And Resource Required For Database Mount

When mounting a database, Caché now checks to see that its default collation is available. If it is not, the mount fails. If this happens during the system startup and the database is marked as "mount required at startup", Caché will not start. In this case remove this requirement from the configuration and restart. A message is logged to cconsole.log indicating the database that failed to be mounted and its default collation.

Caché will also fail to mount the database if the resource specified for the database is not available. This may be an issue when upgrading older systems.

Automatic Upgrade To 8KB Database

When upgrading to this version, if the CACHESYS database block size is found to be 2KB, it will be automatically converted to 8KB during the upgrade.

Enabling SSL For Secure Communications

Caché can be configured to use SSL for securing its communications with external endpoints. A full administrative interface will be available in a future release. Sites wishing to enable this capability via low-level system settings should contactInter-Systems Worldwide Customer Support. Ask for the information on "Client-side SSL settings for Windows".

12.2.1.4 Platform-specific Items

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

$ZF For OpenVMS Changed Privileges

An error that caused $ZF calls on OpenVMS systems to execute with more privileges than designed has been corrected. If this affects your application, please contact InterSystems Worldwide Customer Support for assistance.

Java Not Supported On Tru64

In this version, Caché no longer supports version 1.4 of the Java VM. Unfortunately, this is the last supported version of the Java VM on HP Tru64 UNIX®. Customers who must have a working Java virtual machine should contact InterSystems Worldwide Customer Support.

The only way at this time to get support for later versions of the virtual machine on Tru64 involves the customer compiling the OpenJDK sources using the GNU GCC toolset.

Windows Installations No Longer Contain Packet Drivers

Beginning with this version, the packet drivers (ispktd2k.sys) are no longer part of the distribution since DDP is no longer supported. This also removes support for LAT.

12.2.2 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.

12.2.2.1 General Operational Changes

Error Trapping Now Strictly Nested

In prior releases, when $ZTRAP was set with an asterisk as the first character, or set to %ETN, an error trap set with $ETRAP or TRY/CATCH at a deeper execution level would be ignored. Now the most recent error trap will always have precedence, that is, be invoked first after the error occurs.

Some Cases Of <MAXSTRING> Are Now <SUBSCRIPT> Errors

In this version Caché now correctly returns a `<SUBSCRIPT>` error when a lock subscript has an encoded string length greater than 256 bytes. Previously, such conditions resulted in an erroneous report of a `<MAXSTRING>` error.

Default $ZDATE / $ZDATEH Format Changed For Japanese Locale

The default date format for the Japanese locales is now 1 (was 3). This format is used by default with $ZDATE/$ZDATEH functions when locale formats are enabled. The format is changed by setting

```
^SYS("NLS","Config","LocaleFormat")=1
```

New Locales Added For German

In order to facilitate the migration of sites using the German Latin1-based locale (deu8) to the equivalent Latin9-based one (dei8), the older German1 and German2 collations have been made available in dei8. Although these collations have the exact same binary encoding in dei8 and deu8, users should be aware that:

1. Latin9 replaces 8 rarely used characters from Latin1 with new definitions: 164, 166, 168, 180, 184, 188, 189 and 190. The most visible one is $C(164) which is now the Euro sign. Subscripts from globals brought from deu8 which contain any of these characters will be displayed with the Latin9 interpretation of their binary values.
2. German3 is the default collation in dei8. If a site requires German2 or German1 to be the default collation, it should set it during startup or a custom locale should be created with the desired default.

New Locale For Slovenia

A new Unicode locale for Slovenia is now available. It includes the Slovenian1 collation and its internal code is “svnw”.

Source Control Setting Now In ^%SYS

The setting to determine which source control class should be used in a namespace was stored in *^SYS("SourceControlClass")* in the users database file, but this means that anyone with write access to this database can modify this setting. The value has been moved into *^%SYS("SourceControlClass", <namespace>)*; access to the CACHESYS database is needed to change this setting.

The *^%SYS("SourceControlClass", <namespace>)* was always in the search path when looking up the source control class, so this will work on older versions of Caché. The change is that the method that updates the class will now set this node and clear the *^SYS("SourceControlClass")* node.

Also when looking up the source control path, the locations examined have been changed to this order:

1. ^%SYS("SourceControlClass", <namespace>)
2. ^SYS("SourceControlClass")

3. ^%SYS("SourceControlClass")

12.2.2.2 Routine Compiler Changes

Macro Preprocessor Return Values Changed

In this version, a change has been made in the macro preprocessor behavior when compiling code that contains SQL statements. Prior to this change if a SQL statement failed to compile, the preprocessor would report the error and stop; no code was produced.

The new behavior is to report the error but continue to compile the code and return the result to the caller of the preprocessor. When a SQL statement fails to compile, the generated code produced is

```
X "** SQL Statement Failed to Compile **"
```

If the method or routine containing the erroneous SQL statement is ever run, the attempt to execute the SQL statement will result in an error message at runtime. This change means when compiling classes with methods that contain embedded SQL statements that fail to compile, the errors will be reported, but the methods will still be generated.

Note: This behavior will not occur for a "&sql(DECLARE CURSOR ...)" statement because DECLARE CURSOR produces no code itself. Furthermore, this new behavior is not in effect when the Caché system compiles cached query routines.

12.2.2.3 Class Changes

BuildValueArray Now Marked As Final

Subvalue indexes can be defined using property elements and keys. If the property is a collection, the element values correspond to the elements of the collection and key values correspond to the position (for lists) or key (for arrays). For properties that are not collections, elements and keys correspond to subscripts and values as contained in a "value array". The value array is constructed by a property method, **BuildValueArray**.

If applications had overridden **BuildValueArray**, the results were ignored when the object was stored, and the index(es) were not updated. This is incorrect.

Beginning with this release **BuildValueArray** is generated automatically for collection properties and marked as Final; therefore, it cannot be overridden by subclasses.

%Text Parsing Of Hyphens And Negative Numbers Improved

This version of Cache' improves the parsing of hyphen/minus sign so that "–<digits>" is parsed as a negative number, and so that "–<digits>–" is parsed as a positive number (digits). In addition, complex non-numeric sequences such as "Section 3-4.2-2" are now parsed as "Section 3 4.2 2"; the sequence 3-4.2-2 is not treated as a numeric just because it contains all numeric characters.

These parsing differences in %Text.Text could cause the parsing of some numeric text sequences involving "–" to be treated differently; therefore, an index on such a field containing text like this should be rebuilt.

Changes To LogicalToStorage And StorageToLogical

LogicalToStorage and **StorageToLogical** are methods that can be implemented for properties, either in the datatype class or as an composite method in the class containing the property. (There is no requirement that both methods be implemented for a property, but if the method is defined then it must be executable.) **LogicalToStorage** method is invoked whenever a logical value is stored (placed in the serialized object in the case of a serial class, written to disk in the case of a persistent class). **StorageToLogical** is invoked whenever a stored (or serialized) property value is extracted from the stored object and placed into a logical container - usually the property's instance variable.

In previous versions, when applied to a collection, the target method was presented with the entire collection. In this version, the method will be called once for each element of the collection.

Changes To %SQL.StatementMetadata

The %SQL.StatementMetadata class models statement metadata for dynamic SQL statements. The metadata instance has an *objects* property that is a collection of instances that describe the characteristics of columns whose values are ID values referencing instances of some class. The structure of the StatementObject class has changed to allow for better linking between the StatementColumn and the StatementObject.

Previously, the StatementObject consisted of *columnName*, *extentName* and *exportCall* properties. The definition of StatementObject now includes a new property, *column*, that is an integer and it corresponds to the element number in the columns collection.

Class Deletions

The following classes were present in version 2010.1 and have been removed in this version:

- Package %JavaScript – Object, Runtime
- Package %SQL – StatementCache
- Package %XSQL.DSI – JPAQuery
- Package %cspapp.mgr – utilconfignethome, utilsysnetworklegacy
- Package Config – DDP, DDPVolumeAndUCIs, DDPVolumeSets, DTMnetBIOS, ETHServers, LAT, LATServices, Config.MirrorReportingAuthorizedIDs, Config.MirrorReportingSources, Net, Servers, UDPServers

Class Component Reservations

The following class components have been reserved for InterSystems use and should not be invoked by customer applications.

Class	Type	Name(s)
%SYS.Journal.System	Method	SetPrimaryDirectory

Class Component Deletions

The following class components have been moved or removed in this version from the class where they were previously found.

Class	Type	Name(s)
%CSP.Page	Method	%ClassName, %PackageName
%CSP.UI.DocLocalize	Method	Net
%CSP.UI.Portal.DatabaseFreespace	Method	GetSize
%CSP.UI.Portal.SSL	Property	PeerLevel0, PeerLevel1
%CSP.UI.Portal.SSLList	Method	DrawTitle
%Debugger.System	Property	DevOpen
%Dictionary.CompiledClass	Property	HasJavaScript
%Dictionary.ForeignKeyDefinition	Property	Origin
%Dictionary.IndexDefinition	Property	Origin
%Dictionary.MethodDefinition	Property	Origin
%Dictionary.ParameterDefinition	Property	Origin
%Dictionary.ProjectionDefinition	Property	Origin

Class	Type	Name(s)
%Dictionary.PropertyDefinition	Property	Origin
%Dictionary.QueryDefinition	Property	Origin
%Dictionary.StorageDefinition	Property	Origin
%Dictionary.TriggerDefinition	Property	Origin
%Library.IResultSet	Method	%GetMetaData
%Library.Persistent	Method	%OnDetermineClass
%Library.ProcedureContext	Method	%Display
…	Property	LTT
%Library.RegisteredObject	Method	%ClassIsLatestVersion, %ClassName, %Extends, %GetParameter, %IsA, %OriginalNamespace, %PackageName
%Library.SerialObject	Method	%OnDetermineClass
%Library.SwizzleObject	Method	%AddJrnObjToSyncSet
%Net.Remote.Java.JavaGatewayService	Method	DefaultClassPath, FindJava, JavaDebugParams, RunJava
…	Parameter	JAVADEBUG, JAVAGATEWAYJARS
%SOAP.WebService	Property	Action
%SQL.StatementColumn	Property	propertyId
%Stream.Object	Method	%OnDetermineClass
%Studio.SourceControl.ItemSet	Parameter	CCRSrc
%SYSTEM.Process	Method	EnableCaching
%UnitTest.JDBCSQL	Method	existJava, getJDK
%UnitTest.Result.TestInstance	Method	NamespaceGet
…	Parameter	READONLY
%XSQL.DS.IResultSet	Method	%GetMetaData
%ZEN.Component.tablePane	Method	onRefreshContents
Config.Cluster	Property	CommPort, NetworkType
Config.Configuration	Method	LATSignOnsStatus, NetActivityStatus
Config.Databases	Method	Mount, SetMIRRORCLI
Config.MirrorMember	Method	isReportingNode
…	Property	ReportingGUID
Config.MirrorSetMembers	Proiperty	VirtualAddressInterface
Config.Miscellaneous	Property	DdpSec, DropRemLocks, EnableCaching, IPV6, MNetSets, NetNullSubs
Config.config	Method	GetLegacyNetConn, UpdateLegacyNetConn

Class	Type	Name(s)
…	Property	LegacyNetConn, MaxClientsPerPort, UniqueClients, gmaxcache, gnetfrac, netcliconmax, vtabsiz
Security.Users	Method	SSLGetCipher, SSLGetCipherList, SSLGetLastError, SSLGetPeerName, SSLGetProtocol, SSLPeekClientHello

Method Return Changes

The following methods have different return values in this version of Caché:

- Package %CSP.UI.Portal.Config.ZenReport – SaveData
- Package %ResultSet.Result – %Display
- Package %SYS.Journal.SetKillRecord – ExistsOldValue
- Package %SYS.TaskSuper – FindId
- Package %UnitTest.Manager – GetTestStatus
- Package %XML.Node – GetText
- Package %ZEN.Template.ObjectGatewayWizard.JavaDone – ImportClasses

Method Signature Changes

The following methods have different calling sequences in this version of Caché:

Class Name	Method Name(s)
%BI.WebAnalyzer	ExportToExcel
%CPT.COSCallout	Compile
%CPT.CalloutTesting	ScoopChunk, TestStream
%CPT.ISQLCallout	Compile
%CSP.UI.Portal.DatabaseFreespace	clearSpace
%CSP.UI.Portal.NLSEdit	SaveIODefaults
%CSP.UI.Portal.SSL	SaveData
%Compiler.Informix.Flo	preparseSQL
%Compiler.Informix.Visitor	nearestLoop
%IO.ServerSocket	ListenJob
%Library.Boolean	DisplayToLogical, LogicalToDisplay
%Library.EnsembleMgr	EnableNamespace, InitializeEnsemble, InitializeHealthShare, Upgrade, UpgradeNamespace, check4Install, createPortal, createPortalApp
%Library.File	SetReadOnly, SetWriteable
%Library.IResultSet	%CreateSnapshot
%Library.PopulateUtils	TimeStamp

Class Name	Method Name(s)
%Net.Remote.Java.JDBCGateway	Test
%Net.Remote.Java.JavaGatewayService	StartGateway
%Net.Remote.Service	OpenGateway
%Net.SSH.SFTP	Test
%Projection.MV	genDictitem
%SOAP.Addressing.Properties	WriteSOAPHeaders
%SOAP.Security.Header	AddElement, GetElementById
%SOAP.Security.UsernameToken	Create
%SOAP.WSDL.Reader	CreateParameter, GenerateService
%SOAP.WebClient	InvokeClient
%SOAP.WebRequest	SendSOAPBody
%SOAP.WebService	Initialize
%SQL.IResultSet	%CreateSnapshot
%SQL.Manager.API	AddUser
%SQL.Migration.Import	Connect, CopyTableStruct
%SQL.Migration.Util	DropTable, DropView
%SQL.StatementMetadata	%GenerateMetadata, %OnNew
%SYS.Journal.File	CheckIntegrity, GetNext, GetPrev
%SYS.Journal.System	GetDefaults, GetHistoryHeader
%SYS.ZENReportServer	ServeTransform
%SYSTEM.CSP	Show
%Studio.AbstractDocument	CompileDocument
%Studio.Extension.Base	OnBeforeCompile
%Studio.General	ConstructCSPSession
%Studio.SourceControl.ISC	Disconnect
%Studio.SourceControl.ItemSet	Import, Load, LoadToOS
%XML.Implementation	WSDLMessageSchema
%XML.Node	GetText
%XML.Reader	Open, OpenURL
%XML.Utils.SchemaReader	ProcessSchema
%XML.Writer	Canonicalize, WriteAttribute
%XSQL.DSI.Call	CompileProcedureContext
%ZEN.Component.dataListBox	%DrawItem
%ZEN.Controller	InvokeClassMethod, InvokeInstanceMethod

Class Name	Method Name(s)
%ZEN.ObjectProjection	CreateProjection
%ZEN.Report.Display.Chart.pieChart	renderGetLabelText, renderSeriesLabels, renderTrigFunctions
%ZEN.Report.Display.report	%DrawToHTML, %DrawToXSLFO
%ZEN.Report.Display.Chart.chart	renderXLabels, renderYLabels
%ZEN.Report.PrintServer	%ServePSTransform, %ServeTransformAndPrint
%ZEN.Report.group	%GenerateAggregates, %GenerateCode, %GenerateOpenTag, %SortChildren, %StoreElements
%ZEN.Report.report	%GenerateCode
%ZEN.Report.reportDataClasses	CreateParentProperty
%ZEN.Report.reportGenerator	CreateReportDefinition, UrGenerate
%ZEN.Report.reportPage	%DisplayPDF, %DrawToHTML, %DrawToXSLFO, %DrawXML, %MakeXMLDataFile, %PerformTransform, DeleteTempFiles, GenerateReport
%ZEN.SVGComponent.chart	renderYLabels
%ZEN.Utils	%GetPhysicalIncludeDi
%cspapp.mgr.utilsysecpcliserv	SaveServer
%cspapp.sec.utilsqluseredit	Save
Config.Configuration	AddDataServer
SYS.Shadowing.Shadow	Delete
Security.Users	Create, GetUsernameAndPassword, PromptForNewPassword, PromptForUsername

12.2.2.4 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

SQL Name Of Index Must Be Unique

The SQLNAME of an index must be unique. This constraint was not enforced in earlier releases. It is now checked and an error reported if it is not.

Normalization Required When Using Simple IDs

Simple ID values are no longer normalized in Caché by the various methods that accept an ID as a parameter. ID values passed to the various methods of a class are expected to be in the normalized form, that is, the form returned by the <oref>.%Id() method. If an ID is a simple integer and a value is passed that is not in the integer normal form (for example, it is 01 rather than 1), methods such as %Open() and %Exists() will fail.

12.2.2.5 Object Binding Changes

Changes To Java Generated Code For Properties

Because of the new object dispatch Java driver in 2010.1 and beyond no longer uses projected values fields ii_<Property-Name>, jj_<PropertyName>, or kk_<PropertyName>. Generated samples will need to have the CacheDB.jar from 2010.1 or later in oprder top work properly.

12.2.2.6 SQL Changes

SQLCODE And %RowCount Renamed

Certain classes (resultsets, procedure context classes and dynamic statements) contained the property, *SQLCODE* whose value is set to indicate the result of executing an SQL statement. This property had other, similar names in past versions. Beginning with this version, it will now be called *%SQLCODE* in all classes but one - %SYSTEM.Error where it will remain *%SQLCode*.

Another property, *%RowCount* is present in some dynamic statement classes and result set classes. This corresponds to the local variable, *%ROWCOUNT*. Beginning with this version, this property is now named *%ROWCOUNT* in all classes.

Dynamic dispatch methods are implemented in these classes to reduce the impact on any existing code. Dynamic dispatch will check for *%SQLCODE* and *%ROWCOUNT* in various uppercase and lowercase forms and normalize the references to all uppercase. In most situations, this will allow existing code to work without requiring any alteration. However, to achieve better performance it is best to update existing code to use the defined property names of *%SQLCODE* and *%ROWCOUNT*.

Qualify Column Names In %ResultSet

If a row type specification received by a %ResultSet instance contains more than one column with the same name, the column name is now changed by qualifying the column name with the column number.

TOP Now Applied By Query

Prior this version, a query of the form

```
SELECT TOP :a <query1> UNION <query2> ...
```

would apply TOP to the result of the UNION. Beginning with this version, TOP will be applied only to the query it appears in. To re-instate the behavior of previous release, the query will need to be rewritten as

```
SELECT TOP :a FROM (<query1> UNION <query2>) ...
```

SQL Compiler Requires Access To Class Data

When compiling SQL statements, the SQL compiler will now acquire a shared lock on any class definition that corresponds with a table the SQL statement is using. This shared lock will be acquired when the compiler accesses internal metadata; it will be released immediately afterwards. If an SQL compilation is unable to acquire a shared lock on a class it needs for compilation, an error will be returned (%SQLCODE = 150) and the compilation of the SQL statement will fail. The lock attempt uses the current SQLLockTimeout setting.

This avoids a situation present in previous releases where the metadata used by the SQL compiler may not have been complete causing the compiled SQL statement to behave incorrectly.

Cached Query Changes

In this version of Caché, a number of significant changes have been made to the way cached queries are stored and used. The most important are:

- The global, *^mcq*, is no longer used to hold cached queries and is no longer a reserved global name. Cached queries are stored in the CACHE database, and are not visible across namespaces as they had been in the Management Portal.
- The RS class package is no longer used for some cached queries and is no longer a reserved package name.
- The "CacheSql" routine prefix is no longer used for cached query routines and is no longer a reserved routine name.
- When setting server initialization or disconnect code, you can no longer set *^mcq("init code")* directly. Instead, you must use the APIs in
  - **$SYSTEM.SQL.SetServerInitCode(code)**: where code contains the initialization code to be executed. Call this function with code="" or missing to delete initialization code for this namespace.

  - **$SYSTEM.SQL.SetServerDisconnectCode(code)**: where code contains the disconnect code to be executed. Call the function with code="" or missing to delete disconnect code for this namespace.

- There is a new function to purge ALL cached queries in all namespaces on the systems: **$SYSTEM.SQL.PurgeAllNamespaces()**. **$SYSTEM.SQL.Purge()** will continue to purge all cached queries in the current namespace only.
- The %SQL.StatementCache class has been renamed %SYS.SQLStatementCache; %SQL.StatementCache no longer exists.
- The %XSQL.DSI.JPAQuery class has been renamed to %SYS.SQLObjectQuery; %XSQL.DSI.JPAQuery no longer exists.

Prior to this version, cached queries from one namespace could be visible in another namespace via the System Management Portal. In this version, cached queries are stored in a separate location, and are not visible across namespaces.

Collation Now Applied To Results Of SQL Function

If an SQL function return value has a collation value, and the function is called from an SQL statement, SQL will now use the collation for the return value. It is possible that SQL results for a query that calls such a function might be different than in previous versions.

An example of such a situation follows. Suppose this function is defined:

```
Class SQLUser.FunctionTest Extends %Persistent
{
Property myString As %String(COLLATION = "SQLSTRING(100)", MAXLEN = 500);

ClassMethod Reverse(Arg1 As %String) As %String
    (COLLATION="SQLSTRING(100)", MAXLEN=500) [SqlName=Reverse, SqlProc ]
  {
      quit $REVERSE(Arg1)
  }
}
```

and it uses the query

```
SELECT myString, SQLUser.Reverse(myString) gnirtSym
FROM SQLUser.FunctionTest
ORDER BY 1, 2
```

Prior to this change, the execution of the query would result in a <SUBSCRIPT> error when the length of the data in *myString* approached 500 characters. This is due to a known subscript length restriction in Caché. Beginning with this version, the SQL query processor will recognize the collation of the return value for the function as SQLSTRING(100) and the ORDER BY will be performed on the collated value of the function return value.

Change In Concatenation Operator

There has been a slight change in behavior of the SQL concatenation operator. Prior to this change, if you had "A || B" in an SQL statement, the resulting value would be a string with exact collation. Now, if *A* and *B* are both strings and both have the same collation, the result will be of that common collation also. This may cause a change in the results returned for some queries.

New Default DDL Datatype Mappings For Large-Object SQL Datatypes

For new installations, the default DDL Datatype mappings for all large objects have been changed:

- Character large objects (CLOBs) have been changed from "CStream%String" to %Stream.GlobalCharacter.
- Binary large objects (BLOBs) have been changed from "BStream%String" to %Stream.GlobalBinary.

Customers with multiple setups that have some new installs and some upgraded systems need to be aware that the newly installed systems may produce different class definitions when executing DDL CREATE TABLE and ALTER TABLE

statements to create stream fields. BStream%String and CStream%String and their synonyms (%Library.GlobalBinaryStream and %Library.GlobalCharacterStream) are still fully supported.

Error Returned Instead Of Silent Failure

Now, error messages will be reported for conversions that used to fail without any message. This change returns SQL_ERROR on a row containing conversion errors that are not due to truncation, to match the behavior of SQLServer reporting. The SQL state reported may not be the same as seen with SQLServer when a clearer message is available in Caché.

More Info Now Returned By Error Handlers

Calls to SQLError and the other error handling APIs will return more complete information than in prior releases. These APIs now have the ability to report multiple errors, so that error reporting can now indicate status on more than a single record. This allows Caché to report truncation errors (among other errors) for a particular row and column of a result set through the application calling SQLGetDiagRec and SQLGetFieldRec ODBC 3.5 APIs.

Changes To User-Defined DATE And TIMESTAMP Handling

Prior to this release:

- if an application had a user-defined DATE datatype, and the Logical value of this datatype was not +$Horolog, it would not accept CURRENT_DATE as input for this field value or in a query as a comparison value for this field. Also, it could not use this field in calls to DATEDIFF, DATEADD, DATENAME, or DATEPART.
- if the application had a user-defined TIMESTAMP datatype, and the Logical value of this datatype was not 'YYYY-MM-DD HH:MM:SS[.sss]', it could not use CURRENT_TIMESTAMP as input for this field value, in a query as a comparison value for this field, or in calls to DATEDIFF, DATEADD, DATENAME, or DATEPART.

Caché SQL will now support these operations as long as the user-defined datatype class contains methods which convert from the user-defined logical value to the %Library.Date logical value. The methods needed for the user-defined date datatype class are LogicalToDate and DateToLogical.

Owners Of SQL Stored Procedures

A correction has been made to the SQL projection of a stored procedure. If the procedure does not have an owner specifically defined for it (the owner of the class that projects the procedure), the owner will default to "_SYSTEM" and no privileges will be granted to the owner (since _SYSTEM has the %All role). This now matches the behavior followed when compiling classes that project tables and views. Prior to this change, the owner would default to the current user (the user compiling the class).

Class References Projection Changes

For a persistent class with a property that is a reference to another persistent class with a CLASSNAME=1, the property no longer projects to SQL as a reference, but as a simple %List type. This corrects behavior where SQL treated this field was an Integer value that contained the ID of the referenced row. When CLASSNAME=1, the field data is instead an Oid value, which is not an integer.

Note: At this time SQL -> syntax cannot be used on referenced fields that are defined with CLASSNAME=1.

New EXTERNALSQLTYPE Property

When a linked table is defined through the Link Table Wizard, a new property type parameter will now be defined called EXTERNALSQLTYPE. This will be set to the type number or name returned from the metadata of the external table.

Correction For CAST(x AS NUMERIC(p,s)) Applied

A problem has been corrected where "CAST(value AS NUMERIC(5,2))" would return incorrect results in display mode when using (European-style) comma as the numeric decimal separator.

The expression CAST (*value* AS NUMERIC(*precision*, *scale*)) used to return a logical value that included the full display scale of the cast value. For example, CAST(1 AS NUMERIC(5,2)) used to return 1.00 as the logical value; now it returns

just 1. The display value will still be 1.00. Any code that relied on the logical value of the CAST result having being 1.00 will have to be modified.

12.2.2.7 CSP Changes

Charset Reporting Changed For 8–Bit Systems

When serving up files with the stream server via CSP on 8–bit systems, Caché now reports which charset is used in a different manner. In the past, Caché used the default character file translate table, but on 8–bit systems this is almost always RAW and so does not provide the information required. Beginning with this version, Caché reports the system default locale charset for these files. The logic is now:

1. If ^%SYS("CSP", "MimeFileClassify", extension) is defined, it is assumed to be of the form "$LISTBUILD(type, binary, charset)" and this will be used to report the type of this file.
2. If the file is not of a character type, the charset will always be set to "" for binary files.
3. If ^%SYS("CSP", "DefaultFileCharset") is defined, then this is the charset to use when serving files.
4. If this is a Unicode installation, Caché will use the default file character translate table.
5. On 8–bit installations, Caché will use the system default charset.

For background information on character translation in Caché, see "Localization Support" in the *Caché Programming Orientation Guide*.

CGI Variable SERVER_NAME

The CGI environment variable *SERVER_NAME* is now in the set of fields that are placed at the beginning of the first request data block dispatched to Caché. This is to enable the CSP server in Caché to identify the application early in the request processing cycle. The order of the first three fields in the data block is now as follows:

1. Session ID (CSPCHD)
2. CGI environment variable, *SERVER_NAME*
3. Request path/file (CSPLIB)

Properly Handle Session Id Between Secure And Non-Secure Pages

If a page loaded via an http link itself has a link to an https page, Caché will now not include the CSPCHD SessionId. This is true even if token-based session management is in effect, or if this is the initial page of a session and autodetection logic is active.

Note: This may be overridden with the CSPSHARE=1 parameter to regain the previous behavior, but is not recommended. Applications should not be sharing a session between secure and insecure pages as this could be a security risk leaking information from the secure pages to the non-secure ones.

12.2.2.8 XML Changes

Use Caché InitialExpression To Represent XML Schema Fixed Attributes

Caché now represents the fixed attribute for elements and attributes in an XML schema using the InitialExpression keyword in Caché classes created by the XML schema wizard. If an application is depending on being able to determine if a fixed element or attribute has been specified, it will no longer be able to do this.

Note: The ability to determine whether an attribute has been specified violates the spirit and letter of the rules for attribute usage in XML. Also, this is not a complete implementation of fixed attributes since the value may later be changed.

XMLImport Efficiency Changes

XMLImport has been changed to process child elements more efficiently rather than the previous method of recursively processing subtrees. Users who have applications that override XMLImport may be required to redesign some of their application processing.

12.2.2.9 Web Services Changes

Correct Handling Of MIME Parts

This version corrects an error in the handling of messages with MIME parts present in prior versions. If the WSDL specifies document/literal and MakeMessageStyle=1 (message format which allows multiple parts), Caché now creates the parts which correspond to MIME content if they are called for by the soap:body element of the <binding> <operation>. Prior versions incorrectly ignored all MIME parts, even if they were in the parts list or defaulted to the parts list.

12.2.2.10 Language Binding Changes

Project %Stream.Object As ICacheObject

In previous versions, if a class used %Stream.Object in its properties or methods, it couldn't be projected to CacheProvider. Beginning with this release, it will appear as ICacheObject in the generated code (a generic interface for .NET proxy classes). The actual object that is returned to the client at runtime is the projection of the specific server stream object. For example, if the object that is returned at runtime is %Stream.GlobalCharacter, the client object will be CacheCharacterStream.

Light C++ Binding Uses Its Own Namespace

Beginning with this version, the Light C++ Binding uses a different namespace ("InterSystemsLCB") than the TCP/IP-based C++ bindings ("InterSystems"). This makes it possible for the same application to use both LCB and TCP/IP-based C++ binding, with each of the bindings using a different shared library (or dll on Windows), and with each library potentially being installed from, and connecting to, a different version of Cache.

Changes To runMethod

The binding server protocol allows nested calls inside output redirection. Before this change, every output redirection call to the client started a nested call to the binding message loop and required a separate message from the client to return to the previous server message.

This change makes the nested call optional. CacheProvider now disallows nested calls to runMethod by default. Applications using nested calls will have to explicitly set AllowNestedCalls to true.

12.2.2.11 xDBC Changes

Change Formatting For %Numeric Scale 2

In earlier releases, Caché returned “.00” for a zero value as a workaround for an Access 2000 defect where a value of “0.00” would be interpreted as #deleted by Access. This workaround is now withdrawn; sites still using Access 2000 should upgrade their versions.

Concatenation Results No Longer Truncated

In prior versions, if a concatenation expression produced a string whose length was greater than 255 characters, it would be truncated to that length. This no longer happens.

Use WVARCHAR SQLTypes For Version 3.5 ODBC With Unicode SQL Types

ODBC 3.5 fully added WVARCHAR SQLTypes support for strings. While a Unicode Caché installation does not care if a datatype is Unicode or not, DotNet and Access applications sometimes need ODBC to support WVARCHAR. In the past for Access Caché used a DSN defined option called “Unicode SQLTypes”; when set to 1, Caché returned -9 instead of 12 for a string SQLType.

"Unicode SQLTypes" can be used via programatic control or in a DSN setting. By default, it is set to 0, meaning that 12 is returned for VARCHAR types. This change allows Caché to dynamically change the metadata reported back from Catalog queries, based on the switch. The server side still considers these data types as 12.

Note: This change is enabled by the Unicode SQLTypes switch and only affects Unicode clients with Unicode Caché using ODBC 3.5 APIs.

12.2.2.12 MultiValue Changes

Change Handling Of EQUATE References

In previous versions, any word in the source which corresponded to an EQUATE definition would be replaced by the body of the definition. Beginning with this release, a sequence of words separated by "–>" will be considered as a single entity for the purposes of matching EQUATE definitions. This means that the sequence

```
EQU VIN TO CAR(7)
AutoCheck->VIN = VIN
```

would previously have been interpreted as

```
AutoCheck->CAR(7) = CAR(7)
```

whereas now it will be treated as

```
AutoCheck->Vin = CAR(7)
```

CREATE.INDEX Key Length Defaults To 150 Characters

The default length of keys created by the CREATE.INDEX command is 150 characters. If an application needs larger keys for the index, the developer must manually edit the index definition in the class.

Secondary Indexes

INDICES(filevar) now returns an array containing the actual index name. Previously, the MultiValue dictionary on which the index was created was returned. This allows the INDICES() function to return indexes created with the MultiValue command CREATE.INDEX as well as those created directly in the class.

Two new attributes have been added to the array returned by INDICES(filevar, indexname). Attribute 7 contains the property name and attribute 8 contains the MVName.

Any MultiValue commands and MVBasic statements and functions can now accept either the actual index name or the MultiValue dictionary name.

Universe Emulation Now Ignores Whitespace In Conversion

Universe ignores leading and trailing white space when converting a string to a numeric value. Now the Caché Universe emulation will do the same.

CAUTION: This change causes the compiler to generated slightly more code for routines compiled for Universe emulation. Very large routines may end up exceeding the maximum space allowed for a routine. In this circumstance, they will have to be refactored into smaller segments.

Extend MVIMPORT To Handle Orphan Directories And Files

MVIMPORT now accepts an optional second parameter on the command line, a path to store any directories and files that are otherwise skipped because they do not directly relate to a VOC. MVIMPORT will examine the backup and determine the lowest point in the tree of directories that includes all these directories and files. MVIMPORT will then import that subtree into the path specified on the command line.

Spooler Now Allows Binary And Text Data In Same Job

This change allows a MultiValue application to have binary data and non-binary data inside the same print job. In prior versions, there were problems with the form-feed character being translated to another sequence (if defined by SP-CONTROL); the translation would be performed on both textual data which did require the translation and binary data that did not. This change also fixes an anomaly earlier versions that might result in fewer form feeds being output at the start of a job.

Use Default Precision Of 4 For SQL Formatting

Users of the MV formatting operations in SQL introduced in 2010.1 ($MVFMT, $MVOCONV, and others) will find that the MultiValue precision now defaults to 4, the same as it does from the MV Shell, rather than zero. In prior versions, the value was not initialized until an MV command was issued.

Logical Operators Now Emulation Dependent

The MVBasic logical operators "AND" and "OR" will now behave properly according to the emulation under which they are compiled. Emulations such as Universe and jBase will stop evaluating arguments as soon as a result for the expression is known.

A new $OPTIONS is provided to control this behavior if the emulation default is not desired:

- $OPTIONS FULL.LOGICAL.EVALUATION means that all the operands will always be evaluated.
- $OPTIONS -FULL.LOGICAL.EVALUATION means that if the result is known after evaluating the first operand, other operands will not be evaluated.

VAR.SELECT Now Default For Ultimate

$OPTIONS VAR.SELECT is now the default for the Ultimate emulation.

Comparison Of Non-Numeric Strings Fixed

Non-numeric strings in the arguments for the comparison functions EQS, NES, GTS, GES, LTS, and LES will now be converted properly according to the emulation under which the program is compiled. This was not the case in prior versions.

Positioning Of READPREV Corrected

In prior versions, the first READPREV after a SELECT ATKEY to a key that does not exist would skip one record. Now, the record position is properly managed and the record is not skipped.

Suppress Newline At End Of Print Job

Normally, when printing in MultiValue, at the end of printing a line of data a newline sequence will be printed. This is normally CR + LF (but can be amended with SP-CONTROL). There are now 3 scenarios when we do NOT print this newline sequence at the end of a line of data:

1. When outputting binary data on a line, as specified with the CHAR(255):"BINARY": sequence.
2. When a line of data is immediately followed by a form-feed on the next line. In this case the new line sequence is not only superfluous,but might also create an unintended blank page.
3. At the end of a print job. Again, this might create a blank page if the page was full when the newline is printed. It is up to the form feed specification using the SP-SKIP setting to decide trailing form feeds.

There is a small chance that print job might have been relying on the previous behavior to throw a page at the end of a print job. If this is the case, then the form queue setting needs to be amended with SP-SKIP to carefully define the end-of-job form feeds.

Named Common Match Checking

If two programs have the same named COMMON area with a different number of variables in the COMMON, there will now be a <COMMON MISMATCH> error thrown when the second program is entered, except in UNIDATA emulation where no error will occur.

Named Common Initialization

Named COMMON variables will now be initialized according to the emulation or $OPTIONS specified. CLEAR COMMON /name/ will now clear all variables in the named COMMON.

Allow TANDEM Usage From Any Account

The use of TANDEM as the master controlling terminal no longer requires SYSPROG account privileges. You can now, by default, TANDEM to a running terminal from any MV account. Before you can become a TANDEM master, the client terminal must enable itself with the following:

```
TANDEM ON
```

This has been enhanced to support:

```
TANDEM SYSPROG
```

which is the same as TANDEM ON, but it now enforces the SYSPROG security i.e. any terminal that wants to be a master controlling terminal must run from the SYSPROG account.

Change MV Login To Use The Caché User Name

When the MV shell starts up, it will now look for a Proc or Paragraph with the same name as the Caché user rather than the OS user. This change means that it is now looking for the @AUTHORIZATION value rather than @LOGNAME value. This change should not affect Windows users; UNIX® MV users may have to change the name of the login proc if they have used the OSUserName rather than the Username.

12.2.2.13 Zen Changes

Add USECOMMONDIRECTORY Parameter To Zen Components

By default Zen classes write out generated js and css files to the CSP directory corresponding to the namespace they are defined in. For packages that are mapped to other namespaces this can be very inconvenient. This change provides a way to change the default behavior: if a Zen Component class sets its USECOMMONDIRECTORY class parameter to 1, then any files generated for it will be placed within the common /csp/broker directory and thus be visible to all namespaces.

This is a somewhat advanced feature and has some important caveats:

- ALL Zen classes within the same class package must define (or inherit) the same value for USECOMMONDIRECTORY. If not, the application will get an error when it is compiled.
- If USECOMMONDIRECTORY is enabled for classes that had previously been compiled without it, priously generated .js and .css files must be deleted from the local CSP directory. If this is not done, they will continue to be used.
- This mechanism is designed for cases where a package is mapped across the entire system. There is no mechanism for avoiding collisions between two packages with the same name in different namespaces that use USECOMMONDIRECTORY. If this is the case, DO NOT USE this feature.

12.2.2.14 Platform-specific Items

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

Changes To Spawned Child Process File Descriptors On UNIX®

External programs started with $zf(-2, <path>) on UNIX® platforms now run with file descriptors 0, 1 and 2 redirected to /dev/null. In prior versions, these descriptors were closed before starting the program.

$FNUMBER Parsing Corrected On AIX

In prior versions, on the AIX platform, illegal format strings passed to $FNUMBER(n,format) were not always detected. The illegal format strings that were not detected consisted of a "P", (parenthesis format) combined with any of "+-TL" (plus, minus, trailing, or leading format). Now these will generate a <SYNTAX> error.

12.2.2.15 Conflicts

Symantec Backup Exec

Symantec Backup Exec solution seems to install an additional driver file (tpfilter.sys) for magnetic tape devices.esence of this driver make it impossible to back up to a magnetic tape from within Caché (first observed for HP Ultrium 920 SAS streamers).

Symptoms that indicate the problem include:

- The "Tape Symbolic Name" tab disappears from device properties in Windows Device Manager.
- The “\\.\Tape0” device becomes unavailable, making `'o 47:("auv":0:2048)'` hang forever and never return.

The only effective remedy is to uninstall the driver (by uninstalling the Backup Exec).

13

Caché 2010.1

This chapter provides the following information for Caché 2010.1:

13.1 New and Enhanced Features for Caché 2010.1

The following major, new features have been added to Caché for the 2010.1 release:

Furthermore, this version of Caché has been improved and enhanced in the following areas:

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

13.1.1 Major New Features

13.1.1.1 Class Compiler And Routine Dispatch Improvements

Version 2010.1 contains many improvements to the performance and reliability of our object implementation. Many of them are changes to the underlying implementation and therefore invisible to developers. Their effects are evident as much improved compilation times – some early testers report up to twice as fast - and faster run-time performance. The class compiler has been improved to support richer development options, better ease of use and faster compile times. The corresponding object runtime and dispatch mechanisms have also been enhanced for better performance and scalability. Version 2010.1 now supports runtime superclass resolution, system implementations for commonly generated property methods (Get, Set, isModified, setModified), and a shared class cache.

There are new compiler and class dispatch mechanisms that avoid errors associated with compiling a class while instances of the class are in use. The compiler also has improved logic for determining when changes to a superclass require recompilation of subclasses; this avoids unneeded recompilation. Details may be found in "Class Compiler Notes" in *Using Cache Objects*.

13.1.1.2 Java Dynamic Object Interface

This version of Caché introduces a new high-performance interface for Java programs that provides the following:

- Multidimensional Storage (MDS) API

  Java applications may now access and manipulate Caché globals directly via a JNI (Java Native Interface) without requiring a conversion to Java objects. It enables the Java application and Caché to execute in the same process space. Among the advanced features of Caché supported by this interface are:

  - Locking (incremental, non-incremental, exclusive, shared ...)
  - Transactions
  - Iterating over globals represented in the multidimensional data structure
  - Automatic parsing of Java arrays or strings to create or update Caché $LISTs

- Java Dynamic Object API

  The API provides very fast Java access to objects stored in Caché via an in-process connection based on JNI, the Light C++ Binding, and Caché callin. The API is dynamic; the classes accessed need not be known at Java application compile time, and no separate code generation step is required.

- In-process JDBC

  This enables JDBC to use a non-TCP/IP connection to Caché like the Dynamic Object API. Applications can execute SQL queries and other SQL statements via JDBC, and in the same transaction context as methods of the Java Dynamic Object API.

13.1.2 Rapid Application Development

13.1.2.1 Dynamic SQL

This release of Caché includes support for a new interface for dynamic SQL, which allows users to define and execute SQL statements at runtime. Dynamic SQL is now implemented through the %SQL.Statement and %SQL.StatementResult classes. %SQL.Statement includes support for the following methods:

- **%New** – for instantiating a new statement object
- **%Prepare** – for preparing an SQL statement, including one that supports runtime parameters
- **%Execute** – for executing a prepared statement

Execution of a dynamic SQL statement creates a result object, which is an instance of the %SQL.StatementResult class. %SQL.StatementResult can hold simple data, a single result set, or multiple result sets, depending on the SQL statement that was executed. It includes functionality for iterating through a result set, handling errors, and so on; it also includes functionality for examining statement metadata, manipulating the form of data for display, and related operations.

For more information on dynamic SQL, see the chapter *Using Dynamic SQL*.

13.1.3 Performance And Scalability

13.1.3.1 ECP Scalability Improvements

In prior versions, if a request buffer from an ECP application server contained a synchronous request (such as a Set, Kill, $INCREMENT, and so on) that resulted in a disk I/O, subsequent requests in that buffer would wait till the disk I/O for the synchronous request completed. Now, the ECP server process daemon on the database server will continue to service subsequent read (or get) instructions that can be serviced from the buffer, in parallel while the disk I/O is completing, thereby speeding up responses to the application server.

13.1.4 Reliability, Availability, Maintainability, Monitoring

13.1.4.1 DataCheck

The DataCheck facility provides a mechanism to compare the state of data on two systems to determine whether the two are consistent. It accounts for the situation where the data on each system may be in transition, and includes logic to re-check discrepant ranges. A typical use of this functionality would be to verify that the source and destination of shadowing are consistent.

13.1.4.2 Compression During Shadowing

Caché version 2010.1 introduces compression of journal data from the primary to the shadow. The source (primary) compresses a source journal block before sending it to the receiver (shadow); the receiver decompresses it upon receipt, prior to saving it to the shadow copy of the source journal file.

13.1.4.3 ECP Responsiveness Metric

Fast responsiveness of a Data Server is key to good performance on the Application Server. To permit measurement of response times, a new property, *ResponseTime*, has been added to the SYS.Stats.ECPAppSvr class that allows you to measure the responsiveness of the Data Server(s) that this application server is currently connected to. If there are multiple connections, the value is the overall response time for all Data Server connections. To compute the average response time, use the number of connections contained in *ResponseConn*.

13.1.5 Security

13.1.5.1 Separate Delegated Authentication and Delegated Authorization

With this release applications can now use different third party technologies for Kerberos and OS authentication using the Delegated Authentication mechanism. This allows customers to use LDAP (Lightweight Directory Access Protocol), for example, to define the Roles of a user, while using Kerberos to authenticate the user.

13.1.6 Documentation

13.1.6.1 Removal Of $ZUTIL Documentation

In this release, InterSystems provides class components (methods, or properties) that have the same information and functionality as each of the documented $ZUTIL functions. This has been done to provide a more modern interface to system services available to a wider range of languages used in Caché applications.

In accord with this transition, the documentation for the $ZUTIL functions has been excised from the ObjectScript reference book. It is now part of a separate document stored in the legacy documentation archive. No further updates to it will be made.

The $ZUTIL functions will remain available for applications to use. However, all future functionality updates which would have been done by adding new $ZUTIL functions will use a class paradigm instead.

InterSystems encourages applications to convert to the new usage. To assist in the transition, a table giving the $ZUTIL function identifier and the new equivalent is given in the conversion checklist for this release. The information will also be reproduced temporarily in the ObjectScript reference work as well.

Note: In a few instances, the $ZUTIL functionality is now provided by system variables ($NAMESPACE, and $DEVICE). Also, this information does not include those $ZUTIL functions affected by the removal of support for DDP and DCP.

13.1.6.2 Change In PDF Page Size

The page size for documentation PDFs has changed in the version. Previously, pages assumed a height of 9 inches and a width of 7 inches. Now they use a height of 11 inches and a width of 8.25 inches. The margins have also been been reduced slightly to 0.75 inches.

This new format permits larger tables and figures, and wider program listings. Waste paper as a result of cropping has been eliminated since the new format will fit on both A4 and U.S. letter-sized paper without need for cutting before binding.

13.1.7 Planned Changes

13.1.7.1 Database Extents Deprecated

The use of database extents was introduced in early releases to work around the fact that files sizes were severely restricted. Databases of that period often exceeded the limits on the size of an individual file. Database extents allowed a logical database to span multiple physical files.

The limits in the underlying technology are no longer relevant – operating systems, mature filesystems and volume managers currently provide support for extremely large files. Database extents have been rendered obsolete. In the interest of simplifying system management, database extents are no longer supported as of this release.

In a future version, Caché, Ensemble, HealthShare and TrakCare will no longer support database extents. A tool will be provided to assist any customers with extents to aggregate them into appropriate files.

13.1.7.2 DCP, DDP And LAT Deprecated

InterSystems introduced Enterprise Cache Protocol (ECP) in 2002. Since then, all of our major customers have adopted ECP to deploy scale-out architectures. Usage of DCP and DDP, earlier technologies used to network databases, has diminished considerably and is now almost exclusively used when data is shared between Caché and earlier M technologies.

We intend that, in the latter half of 2010, Caché and Ensemble releases will no longer support DCP and DDP. LAT support on Windows will be deprecated at the same time.

13.2 Caché 2010.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2010.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 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 2009.1 and 2010.1.

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

13.2.1 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 2010.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

13.2.1.1 Version Interoperability

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

13.2.1.2 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:

13.2.1.3 Operational Changes

Journaling Status Change By Applications Requires Admin Privilege

Setting or clearing the "no journaling" process flag in programmer mode via DISABLE^%SYS.NOJRN or ENABLE^%SYS.NOJRN, respectively, now requires **`%Admin_Manage:USE`** privilege. A <PROTECT> error is generated if the privilege requirement is not met.

Journal History Log File Critical To Journal Management

Beginning with this version, journal restore has been improved with an option to locate journal files to be processed via a journal history log. The history file, journal.log, is located in the manager's directory and should not be modified by customers.

CAUTION: No changes to journal.log should be made by any means other than InterSystems-supplied utilities. Failure to observe this rule can result Caché determining that the file has been corrupted. This will result in journaling being disabled.

If journal.log is deleted, Caché will create a new one when creating a new journal file; however, information on previous journal files will be lost.

Changes In Memory Usage

This version introduces several object-related optimizations which involve additional data elements to be cached (both at a system-wide and per-process level). As a result, there are implications for both system and process memory utilization that must be noted.

System-Wide Routine Buffer

The optimizations in this version require additional routine buffers to be available. As a result, the following should be considered:

- If the default routine buffer pool is in effect, then a 2010.1 Caché installation will consume an additional 8MB system-wide.
- If, however, a custom routine buffer pool is used (for example, one greater than 17MB), the additional amount of shared memory space to be allocated to routine buffers will depend on the routine buffer usage/utilization:
  - If a rough estimate is desired, increase the routine buffer pool by roughly 20%-30% of the current allocation. For example, if the current allocation is 50MB, increase it to 60MB-65MB.
  - If an accurate estimate is desired, the current system routine buffer utilization will need to be reviewed. This can be done by running the **^GLOSTAT** utility and monitoring the "Routine not cached" statistic. The goal is to minimize this value, and this can be achieved by increasing the size of the routine buffer pool.

Per-process Routine Caching

In 2010.1, each process will cache more classes and class info so as to optimize performance. This can result in a 10KB to 30KB memory footprint increase per process.

System-Wide Class Descriptor Caching

In 2010.1, shared class descriptors will be cached. This memory is allocated from the generic memory heap (gmheap). As a result, the following should be considered:

- The default value for gmheap in 2010.1 has been increased from 3200KB to 6400KB. This is to accommodate the additional memory used due to system-wide class descriptor caching. This can occupy anywhere from 1MB to 1.5MB of the gmheap. The amount occupied depends on the number of classes and objects used.
- If a non-default gmheap value is used for an object-based application,
  - If the configured gmheap is within 10% of the default (that is, it is between 2880KB and 3520KB), it will be increased automatically by 1600KB when the instance is upgraded.
  - If the configured gmheap is outside of the above range, a warning will be included in the cconsole.log indicating that the gmheap value should be manually increased. If a rough estimate of the increase is sufficient, InterSystems recommends an increase of 1600KB. For a more detailed estimate, the**cstat** can be utilized to gather system statistics and provide a more accurate estimate. For additional guidance and information, please contact InterSystems Worldwide Customer Support.

GMHEAP Defaults Increased On Upgrade

This version of Caché will increase the gmheap to a minimum value when the system is upgraded. This minimum value is 4800KB for all non-HP platforms; on HP platforms the default is 9600KB.

- If the size of the gmheap is found to be less than the default, the upgrade installation process will increase it to that value.
- If the size of the gmheap is more than the default, upgrade writes a message to the cconsole.log file advising that it should be increased by an additional 1600KB.

Licensing Added To CacheProvider

Client licensing is now standardized in this version to provide consistency in license usage between ODBC, and DotNet.

Exporting Classes To XML Omits Generated Classes

When exporting classes to XML format, Caché now removes any classes that are generated by other classes already in the export/compile list. The reason for this is that compiling the class that generates this other item will recreate the omitted class. Often these generated classes are things such as classes for SQL projection which are also marked as deployed. Before this change, an attempt to export a package which contained a persistent class with a collection property would result in an error message about not being able to export deployed classes. Now this deployed class is automatically removed from the export list and no error is reported.

Check For Global Name Errors On Truncation

When a global name is longer than 31 characters, Caché truncates the name to 31 characters for all operations. This can result in the last character of the name being a period which is invalid as the last character. In prior releases, Caché was not detecting this condition and using the invalid global name in the global or LOCK operation which resulted in errors. In this version, Caché will return a <SYNTAX> error when the long name is truncated.

The behavior in previous versions can be restored for the system by setting the *GlobalNameTruncated* property in the class, Config.Miscellaneous.

Counters For SNMP And WMI Changed

This version, eliminates the use of a class of metric counters which had to be specially enabled, and substitutes counters which are always "on" and more closely integrated into the system code. This should noticeably improve the overhead of using either SNMP or WMI for monitoring. The *Lines* counter is replaced by *Commands*; *GlobalSets* becomes *GlobalUpdates*; *GlobalKills* has been eliminated. The changes are documented in the MIB and MOF.

Note: **^PATROL** has been changed to use the new counters. The old behavior and variable names are available by activating **$$Light^Patrol(0)**.

Changes To Temporary File Management

Beginning in this release, Caché has improved the handling of its temporary files to the point that deleting them at startup is no longer necessary. Therefore, Caché startup will no longer delete any files in the Temp directory.

Important: Startup will no longer delete any user files in the Temp directory. If an application uses this directory to store temporary files, it is the responsibility of the application to manage the files it creates across system restarts.

13.2.1.4 Platform-specific Items

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

$ZF For OpenVMS Changed Privileges

An error that caused $ZF calls on OpenVMS systems to execute with more privileges than designed has been corrected. If this affects your application, please contact InterSystems Worldwide Customer Support for assistance.

Caché Registry Permissions Explicitly Set For Non-Windows Platforms

The Caché registry file, cache.reg, will now have its permissions explicitly set by the installation process. The file will have `root` as the owner and be writable only by it.

Resource Statistics Gathering Off By Default For Some 64-Bit Platforms

Beginning with this version, the gathering of resource usage statistics is disabled by default on Itanium, POWERPC, and x64 platforms. Enabling and disabling the collection of statistics is done via the Config.Miscellaneous class. Set the *CollectResourceStats* property to true to enable the collection of statistics and false to disable it.

LAT Configuration For Microsoft Windows

The [LAT] section of the cache.cpf file is incomplete in this version of Caché. The System Management Portal does not configure the advertised services: ServiceName, ServiceDescription, and ServiceRating. Therefore, the LAT daemon (lat.exe) now it reads its startup parameters from a lat.ini file in the installation directory instead of from the cache.cpf configuration file. Users employing LAT finctionality should copy the [LAT] section from the existing cache.cpf file into a new lat.ini file or create the information with a text editor.

A sample [LAT] section is shown here. There can be a maximum of eight LAT services advertised. This sample has two. Each service consists of a ServiceName, a ServiceDescription, and a ServiceRating.

```
[LAT]
NodeName=
MessageRetransmitLimit=8
HostMulticastTimer=60
NodeGroups=0
UserGroup
s=0
ServiceName_1=CACHE
ServiceDescription_1=Cache LAT service
ServiceRating_1=1
ServiceName_2=Cache2
ServiceDescription_2=Second LAT device
ServiceRating_2=1
```

Note: The LAT service will be deprecated in Caché version 2010.2.

13.2.2 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.

13.2.2.1 General Operational Changes

Interactive And Command Invocations Use SHELL Environment Variable

Previously, when invoking an interactive UNIX™ shell via `$zf(-1)` (or "`!`" or "`$`"), Caché used the setting of the SHELL variable if it was defined, otherwise Caché used just `sh` to start the shell; when a command name was specified, Caché always used `/bin/sh`.

Now, Caché will check for the SHELL environment variable (and use it if defined) in both cases. This can affect any UNIX™ commands that depend on using `/bin/sh`.

Changes In Licensing Display

In the output from licensing display functions, under some circumstances fields could run together (leaving no space between columns). When such output was sent to a file and then a program attempted to parse it, the program could fail because it could not distinguish the field boundaries.

Now, field output will break to the following line if this occurs; new types of parser errors are possible. The recommended way to obtain and work with license data is via queries in the %SYSTEM.License class.

$SYSTEM.License.Login Now Appends Client IP Address

Applications that call `##class(%System.License).Login(UserId)` will find that the license Id has a client IP address appended. This will affect license counting, if the customer previously had users log in to CSP sessions and enabled username-based licensing to cause other types of connections to also use $USERNAME as the license Id. The CSP login will no longer have the same license identifier as other connections.

Also, `$System.License.Login()` will produce different license Identifiers when called on this version of Caché than on earlier versions. This will cause two license units to be consumed.

Kernel Error Codes Made More Exact

When running out of argument stack space, Cache would sometimes report the condition as a <STORE> error, other times as a <STACK> error, and still others as <ASTACK>. Now detection of insufficient argument stack space always reports the condition as <ASTACK>; running out of heap storage will generate a <STORE> error. The error code, <STACK>, is no longer used.

Note: The argument stack and the string stack share the same block of memory. When a program is using an excessive amount of one kind of stack, it is possible that the error will be detected while attempting to allocate space from the other kind of stack. Therefore, <ASTACK> and <STRINGSTACK> are both possible errors when either of these stacks is being heavily consumed.

13.2.2.2 Objectscript Changes

$ZOBJ<xxx> Functions Replaced

The $ZOBJ<xxx> functions have been replaced with standard Caché functions. The $ZOBJ<xxx> functions are still available to applications, but they are no longer documented. The calling sequences for the replacement functions remain the same as before; a textual substitution is all that is required to update the application usage. The replacements are given in the following table:

Previous Function	New (Replacement) Function
$ZOBJCLASSMETHOD	$CLASSMETHOD
$ZOBJMETHOD	$METHOD
$ZOBJPROPERTY	$PROPERTY
$ZOBJCLASS	$CLASSNAME

Note: The function, **$ZOBJCLASS**, was not documented in prior versions.

Remove Descriptions For Obsolete And Deprecated Functionality

In this version of Caché, documentation for legacy networking facilities DDP and DCP has been removed. The functions themselves are classed as deprecated and will be removed in a future release; no replacement functions have been provided. This is also true for several other features because they no longer function as intended. The list of functions that will not have equivalent replacements is:

$ZUTIL Function	Description
67, 1, …	Returns the activity state of a specified process, and resets.
68, 6, …	Enables or disables reliable SET networking mode for the current process.
68, 27, …	Sets or clears network hardening for the current process.
68, 28, …	Restricts or permits kills of root-level global nodes for the current process.
68, 55, …	Selects $X/$Y behavior for TCP devices for the current process.
69, 6, …	Sets reliable SET networking mode system-wide.
69, 13, …	Sets logging of asynchronous SET/KILL errors.
69, 14, …	Sets logging of asynch errors to console.
69, 19, …	Sets DDP password security system-wide.
69, 20, …	Transfers global nodes with null subscripts with DSM-DDP.

$ZUTIL Function	Description
69, 27, …	Enables or disables network hardening system-wide.
69, 28, …	Controls root-level (unsubscripted) global node kills system-wide.
69, 31, …	Sets network locks handling system-wide following a DCP outage.
69, 35, …	Sets silent retry for domainspace connection attempts system-wide.
69, 55, …	Selects $X/$Y behavior for TCP devices system-wide.
69, 67, …	Suppresses or displays the stack and register usage message box system-wide.
90, 4, …	Starts up in a specified namespace (UNIX™/OpenVMS.
100	Determines which Windows operating system is running.
113	Reclaims routine and directory blocks.
130, …	Sets or returns the domain ID or index.
133, …	Maintains a set of metric counters.

Remove $ZUTIL Functions From Documentation

In this version, the Objectscript $ZUTIL functions are no longer documented. The functions remain available for use by applications as in prior releases, but the actions they perform are now also available by other means. The list of functions that have been replaced, their description, and the new means of accomplishing the same objective are given here. Customers should migrate their applications to use the replacement facilities.

Note: The current list of replacements is available in the Caché Objectscript Reference.

Table 13–1: Non-Process- & Non-System-Related Functions

Code	Description	Class	Property / Method
4, …	Terminates a Cache process.	SYS.Process	**Terminate()**
5, …	Returns current namespace or switches to another namespace.	<Special variable>	$NAMESPACE
9, …	Broadcasts a message to a specified device.	%Library.Device	**Broadcast()**
12, …	Converts file or directory name to canonical form.	%Library.File	**NormalizeDirectory()**
15, …	Converts RMS filename to canonical form.	%Library.File	**NormalizeFilename()**
18, …	Sets undefined variable handling for the current process.	%SYSTEM.Process	**Undefined()**
20, …	Specifies the namespace(s) that contains the routine dataset.	Config.MapRoutines	*RoutineSearchPath*

Code	Description	Class	Property / Method
21, …	Returns the location of process-private globals	%SYSTEM.Process	**PrivateGlobalLocation()**
21	Deletes all process private globals	%SYSTEM.Process	**KillPrivateGlobals()**
22, 0, …	Specifies the form feed or backspace control code sequence.	%Library.Device	**SetFFBS()**
28, …	Performs collation conversion.	$SYSTEM.Util	**Collation()**
39, …	Specifies a search path for percent (%) routines.	Config.MapRoutines	**PRoutineSearchPath()**
49, …	Obtains database information.	SYS.Database	<various class properties>
53	Passes TCP device name to child process	%SYSTEM.INetInfo	**TCPName()**
53, …	Returns TCP statistics.	%SYSTEM.INetInfo	**TCPStats()**
55, …	Returns or changes the current language mode.	SYS.Process	*LanguageMode*
56, 2	Locates source file and line of code for last ObjectScript error.	%SYSTEM.Process	*ErrorLine*
56, 6	Returns the operating system error code for a sequential file error.	%SYSTEM.Process	**OSError()**
62, …	Performs a syntax check of command line code.	%Library.Routine	**CheckSyntax()**
67, 0, …	Returns the activity state of a specified process.	%SYS.ProcessQuery	*IsGhost*
67, 4, …	Returns the process state.	%SYS.ProcessQuery	*State*
67, 5, …	Returns the routine name of a specified process.	%SYS.ProcessQuery	*Routine*
67, 6, …	Returns the namespace name for a specified process.	%SYS.ProcessQuery	*NameSpace*
67, 7, …	Returns the device name for the specified process.	%SYS.ProcessQuery	*CurrentDevice*

Code	Description	Class	Property / Method
67, 8, …	Returns the number of lines executed by the specified process.	%SYS.ProcessQuery	*LinesExecuted*
67, 9, …	Returns the number of global references made by the specified process.	%SYS.ProcessQuery	*GlobalReferences*
67, 10, …	Returns the job type of the specified process.	%SYS.ProcessQuery	*JobType*
67, 11, …	Returns the username of the owner of the specified process.	%SYS.ProcessQuery	*UserName*
67, 12, …	Returns the name of the system for a client application.	%SYS.ProcessQuery	*ClientNodeName*
67, 13, …	Returns the name of the executable for a client application.	%SYS.ProcessQuery	*ClientExecutableName*
67, 14, …	Formerly returned the operating system running a client application. Now returns the CSP Session ID	%SYS.ProcessQuery	*CSPSessionID*
67, 15, …	Returns the IP address of a client application.	%SYS.ProcessQuery	*ClientIPAddress*
71, …	Sets date to a fixed value for the current process.	%SYSTEM.Process	**FixedDate()**
78, 21	Searches journal file for open transactions.	%SYS.Journal.Transaction	**SearchOpen()**
78, 22, …	Returns journaling information.	%SYS.Journal.File	**Info()**
78, 23, …	Deletes a journal file.	%SYS.Journal.File	**PurgeOne()**
78, 28	Returns journal directory block information.	%SYS.Journal.File	**DirectoryBlock()**
78, 29	Flushes journal buffer.	%SYS.Journal.File	**FlushBuffer()**
82, 12, …	Redirects I/O operations.	%Library.Device	*ReDirectIO*
86	Returns configuration file pathname and config name.	%SYS.System	**CPFFileName()**
90, 10, …	Tests whether a namespace is defined.	Config.Namespaces	**Exists()**

Code	Description	Class	Property / Method
94, …	Broadcasts a message to a specified process.	%SYSTEM.Process	**Broadcast()**
96, 3, …	Return error number for user-defined command.	SYS.Process	**ThrowError()**
96, 4, …	Sets $TEST to reflect I/O redirection.	SYS.Process	**IODollarTest()**
96, 5, …	Sets the $DEVICE special variable.	<Special variable>	$DEVICE
96, 9	Returns the calling routine name.	SYS.Process	*CallingRoutine*
96, 10	Returns the calling routine database.	SYS.Process	*CallingDatabase*
96, 14	Returns the current device type.	%Library.Device	**GetType()**
110	Returns the name of the system that is running.	%SYS.System	**GetNodeName()**
114, …	Determines Ethernet address.	%SYSTEM.INetInfo	**EthernetAddress()**
115, 11, …	Specifies whether a value can be inserted into an identity column.	Config.SQL	*AllowRowIDUpdate*
128, 1	Returns location of last single step during debugging.	%SYSTEM.Process	**StepInfo()**
132	Makes the last device in use the principal I/O device.	%Library.Device	**ChangePrincipal()**
140, 1, …	Returns file, directory, and disk information and performs file operations.	%Library.File	**(various)**
140, 7, …	Returns a bitmap of file attributes.	%Library.File	**Attributes()**
147, …	Handles spaces in pathnames for the host platform.	%Library.File	**NormalizeFilenameWithSpaces()**
158, 0	Displays currently installed printers.	%Library.Device	**InstalledPrinters()**

Code	Description	Class	Property / Method
168, …	Returns location of current working directory, or sets current working directory.	%SYSTEM.Process	**CurrentDirectory()**
186, …	Sets display in programmer prompt for the current process.	%SYSTEM.Process	**TerminalPrompt()**
188	Returns local date and time with fractional seconds system-wide.	<Function>	$NOW
189	Checks if TCP device is disconnected.	%SYSTEM.INetInfo	**Disconected()**
193, …	Converts Coordinated Universal Time (UTC) to local date and time (and vice versa).	<Function>	$ZDATETIME

Table 13–2: Process- & System-Related functions

Code	Description	Class	Property / Method
68,0	Sets undefined variable handling for the current process	%SYSTEM.Process	**Undefined()**
68, 1, …	Enables or disables use of null subscripts for the current process.	%SYSTEM.Process	**NullSubscripts()**
68, 2, …	Sets sequential file open mode for the current process.	%SYSTEM.Process	**OpenMode()**
68, 3, …	Sets automatic sequential file creation option for the current process.	%SYSTEM.Process	**FileMode()**
68, 5, …	Enables or disables processing of argumentless BREAK commands for the current process.	%SYSTEM.Process	**BreakMode()**
68, 7, …	Retains or strips extended global reference from globals returned to the current process.	%SYSTEM.Process	**RefInKind()**
68, 11, …	Enables or disables read line recall for the current process.	%SYSTEM.Process	**LineRecall()**

Code	Description	Class	Property / Method
68, 15, …	Enables or disables I/O device disconnect detection for the current process.	%SYSTEM.Process	**DisconnectErr()**
68, 21, …	Sets synchronous commit mode for the current process.	%SYSTEM.Process	**SynchCommit()**
68, 22, …	Sets handling of escape sequences when $X is updated for the current process.	%SYSTEM.Process	**DX()**
68, 26, …	Sets namespace display in programmer prompt for the current process.	%SYSTEM.Process	**TerminalPrompt()**
68, 30, …	Sets error handling behavior for the current process.	%SYSTEM.Process	**PopError()**
68, 32, …	Sets date range and invalid date behavior for the current process.	%SYSTEM.Process	**ZDateNull()**
68, 34, …	Sets whether asynchronous errors can interrupt the current process.	%SYSTEM.Process	**AsynchError()**
68, 39, …	Enables or disables caching for the current process.	%SYSTEM.Process	**EnableCaching()**
68, 40, …	Sets sequential file end-of-file handling for the current process.	%SYSTEM.Process	**SetZEOF()**
68, 42, …	Sets $JOB format for the current process.	%SYSTEM.Process	**NodeNameInPid()**
68, 43, …	Sets clearing of global vectors for the current process.	%SYSTEM.Process	**OldZU5()**
68, 45, …	Sets truncation mode for string-to-number conversions for the current process.	%SYSTEM.Process	**TruncateOverflow()**
68, 51, …	Sets whether or not changing namespaces changes operating system directories for the current process.	%SYSTEM.Process	**SwitchOSdir()**

Code	Description	Class	Property / Method
68, 60, …	Sets handling of asynchronous Telnet disconnect errors for the current process.	%SYSTEM.Process	**AsyncDisconnectErr()**
68, 63, …	Enables or disables the use of “e” as scientific notation symbol for the current process.	%SYSTEM.Process	**ScientificNotation()**
68, 66, …	Suppress Telnet NUL at end-of-line for the current process.	%SYSTEM.Process	**TelnetNUL()**
68, 67, …	Suppresses or displays the stack and register usage message box for the current process.	%SYSTEM.Process	**ExceptionLog()**
68, 70, …	Enables or disables $DOUBLE returning INF and NAN values.	%SYSTEM.Process	**IEEEError()**
68, 71, …	Sets IP address format for the current process.	%SYSTEM.Process	**IPFormat()**
68, 72, …	Sets MVBasic handling of undefined variables.	%SYSTEM.Process	**MVUndefined()**
69, 0, …	Sets undefined variable default handling system-wide.	Config.Miscellaneous	*Undefined*
69, 1, …	Sets null subscript mode default system-wide.	Config.Miscellaneous	*NullSubscripts*
69, 2, …	Sets sequential file open mode default system-wide.	Config.Miscellaneous	*OpenMode*
69, 3, …	Sets automatic sequential file creation system-wide.	Config.Miscellaneous	*FileMode*
69, 5, …	Enables argumentless BREAK processing system-wide.	Config.Miscellaneous	*BreakMode*
69, 7, …	Retains or strips extended global reference from globals system-wide.	Config.Miscellaneous	*RefInKind*
69, 8, …	Sets ZA and ZD locking modes system-wide.	Config.Miscellaneous	*ZaMode*
69, 10, …	Sets system behavior when journal is full.	Config.Journal	*FreezeOnError*

Code	Description	Class	Property / Method
69, 11, …	Sets Read Line Recall mode system-wide.	Config.Miscellaneous	*LineRecall*
69, 13, …	Sets logging of asynchronous SET/KILL errors.	Config.Miscellaneous	*NetErrToLog*
69, 15, …	Sets I/O device disconnect detection system-wide.	Config.Miscellaneous	*DisconnectErr*
69, 21, …	Sets synchronous commit mode system-wide.	Config.Miscellaneous	*SynchCommit*
69, 22, …	Sets $X update mode for escape sequences system-wide.	Config.Miscellaneous	*DX*
69, 24, …	Sets $ZF process deletion behavior for OpenVMS STOP/ID system-wide.	Config.Miscellaneous	*StopID*
69, 26, …	Sets namespace display in programmer prompt system-wide.	Config.Miscellaneous	*TerminalPrompt*
69, 30, …	Sets error handling behavior system-wide.	Config.Miscellaneous	*PopError*
69, 32, …	Sets date range and invalid date behavior system-wide.	Config.Miscellaneous	*ZDateNull*
69, 34, …	Sets interruptability of processes by asynchronous errors system-wide.	Config.Miscellaneous	*AsynchError*
69, 35, …	Sets silent retry for domainspace connection attempts system-wide.		
69, 37, …	Sets physical cursor mode system-wide.	Config.NLS.Locales	*PhysicalCursor*
69, 39, …	Sets caching for future processes system-wide.	Config.Miscellaneous	*EnableCaching*
69, 40, …	Sets end-of-file handling for sequential files system-wide.	Config.Miscellaneous	*SetZEOF*
69, 42, …	Sets $JOB format default system-wide.	Config.Miscellaneous	*NodeNameInPid*

Code	Description	Class	Property / Method
69, 43, …	Sets clearing of global vectors system-wide.	Config.Miscellaneous	*OldZU5*
69, 44, …	Sets use of the Nagle algorithm for Telnet transmissions system-wide.	Config.Miscellaneous	*UseNagleAlgorithm*
69, 45, …	Truncates numbers during string-to-number conversion system-wide.	Config.Miscellaneous	*TruncateOverflow*
69, 49, …	Sets logging of transaction rollbacks system-wide.	Config.Miscellaneous	*LogRollback*
69, 51, …	Sets namespace default directory assignment behavior system-wide.	Config.Miscellaneous	*SwitchOSdir*
69, 60, …	Sets handling of asynchronous Telnet disconnect errors system-wide.	Config.Miscellaneous	*AsyncDisconnectErr*
69, 63, …	Enables or disables lowercase “e” as scientific notation symbol system-wide.	Config.Miscellaneous	*ScientificNotation*
69, 66, …	Suppress Telnet NUL at end-of-line system-wide.	Config.Miscellaneous	*TelnetNUL*
69, 68, …	Enables or disables the encryption of journal files system-wide.	Security.System	*DBEncJournal*
69, 69, …	Enables or disables the use of long strings system-wide.	Config.Miscellaneous	*EnableLongStrings*
69, 70, …	Enables or disables $DOUBLE returning INF and NAN values system-wide.	Config.Miscellaneous	*IEEEError*
69, 71, …	Sets IP address format system-wide.	Config.Miscellaneous	*IPV6*
69, 72, …	Sets MVBasic handling of undefined variables system-wide.	Config.Miscellaneous	*MVDefined*
69, 73, …	Set global name compatibility checking	Config.Miscellaneous	*GlobalNameTruncated*

$ZDATEH And $ZDATETIMEH Format 4 Parsing

In version 2008.1, the **$ZDATEH** and **$ZDATETIMEH** function were inadvertently changed to accept any nonambiguous date instead of only a European format date. The correct behavior has been restored; **$ZDATEH(date,4)** and **$ZDATETIMEH(datetime,4)** will only accept strings using European format dates.

For those applications expecting the post-2008.1 behavior, a new date format code, 15, has been added that uses European date ordering, DD/MM/YYYY, but also accepts any other date input that is nonambiguous.

Number Format Parsing Via $INUMBER And $FNUMBER Now Locale-Independent

Prior to 2008.2, the **$INUMBER** and **$FNUMBER** functions worked with European-style numbers: “,” as decimal point, “.” as group separator, “-” as minus sign, and “+” as plus sign. In 2008.2, these functions were changed parsing would use the system default locale to determine the minus sign and plus sign; usually this is “-” and “+”, but not always.

In this version, the behavior has been returned to the way it was before 2008.2, so that is always uses “,” as decimal point, “.” as group separator, “-” as minus sign, and “+” as plus sign.

Improved Detection Of Incorrect $HOROLOG Dates And Times

Some of the routines that accept a **$HOROLOG** string (and/or a **$ZTIMESTAMP** string) do not fully validate their input string. Their behavior on illegally formatted string is undefined. In some cases that undefined behavior may change between Cache versions.

A legally formatted **$HOROLOG** string (or **$ZTIMESTAMP**) string consists of a <date> followed immediately by a comma (,) and then followed immediately by a <time>. Optionally, the comma and the <time> may both be omitted; this is assumed to implicitly represent 0:00:00, the beginning of the day, or a date without reference to a particular time.

A <date> part consists only of decimal digits. A <date> may not include any fractional part nor an exponent part nor any non-digit characters.

The <time> part is a string of decimal digits optionally followed by a decimal point (.) and some fractional decimal digits. A <time> may not include an exponent.

If a routine accepts just a <time> string that is not preceded by a <date> and a comma, a leading minus sign, (-) is permitted to represent negative time intervals.

$INCREMENT Changes

Previously, **$INCREMENT()** of a variable containing a very large number would not change the value of the variable when the increment value is insignificant compared to the original value. Thus, successive calls to **$INCREMENT()** could all return the same value, which is incorrect. Beginning with this version, this situation will cause the increment operation to throw a <MAXINCREMENT> error instead of returning an incorrect result.

13.2.2.3 Routine Changes

KillAllObjects Takes No Action

The function, **KillAllObjects^%apiOBJ()**, used to remove all object memory from the partition has been changed to a no-op. Applications can remove an object by letting the object go out of scope.

13.2.2.4 Routine Compiler Changes

Routine Compiler Skips Generated Code

The routine compiler, **^%RCOMPIL**, will no longer compile code that has been generated. Previously, if you attempted to compile *.mac, and there existed MAC code which was generated by compiling classes, the generated class MAC code would be compiled. This would cause the methods in that class to not be runnable. Other code which is marked as generated will also not be compiled. This includes .INT code which comes from MAC code and classes, and cached query code.

Note: It is possible to compile a "generated" routine, but only if the name of the generated routine is specified exactly as one of the routines to compile.

13.2.2.5 Class Changes

Namespace Changes And OREFs

Assume you create an OREF to a class instance, and then change the namespace the application runs in. In prior versions, you would normally get a <NOROUTINE> error when you tried to access a method of this OREF in the second namespace because it could not find the code to run. Caché was looking for the code in the current namespace and not in the namespace where the OREF was created.

Beginning with this version, an application can create an OREF and change its namespace; when it attempts to use this OREF and it will run the code from the original namespace. However, any references to global data or locks made during execution will be relative to the current namespace. This means that it the executing code does a **%Save()**, the data will be saved it will save the data to the current namespace and not the namespace where the code is running. Similarly, a method that does a **$GET(^Global)** will look up the global in this namespace rather than the original namespace.

When an application changes namespaces, it changes the visibility of data. Cross-namespace access to OREFs does not preserve the global data visible from the original namespace. When an object is opened in Namespace A and it loads data from ^*MyGlobal*, then ^*MyGlobal* is expected to be in Namespace A. In changing to Namespace B with an OREF from Namespace A, then any access by that OREF to ^*MyGlobal* looks for the data in Namespace B.

CAUTION: Caché cannot check for this condition. The use of object references created in other namespace may result in subtle data loss or corruption undetectable at the time of execution.

Class Deletions

The following classes were present in version 2009.1 and have been removed in this version:

- Package %BI – SOAP
- Package %CPT – RegenerateImpl, RegenerateSource
- Package %CSP.UI.System – MappingsTablePane, MappingsTitlePane, MenuTree
- Package %Compiler – JavaScript
- Package %ISQL – SP
- Package %Net – Event
- Package %TSQL – SP, SPContext, SimpleStreamLexer
- Package %XSQL – CacheMetadata, Convert, Debug, DelimitedReader, Settings, Transaction
- Package %cspapp.exp – utilexpglobaledit, utilexproutineinfo, utilsysdbglobalinfo
- Package %cspapp.mgr – utilconfiggblmappings, utilconfigprjmappings, utilconfigrtnmappings, utilsysaccess

Class Reservations

The following classes which were publicly exposed in earlier versions of 2010.1, and/or prior versions of Caché, are being reserved solely for InterSystems use:

- Package %TSQL – FileBuffer, LineBuffer, ResultSet, StreamLineBuffer, StreamLineBufferWF, SysFunc, Transformer
- Package %TSQL.sys – cacheMsgXRef, messages, messagesXRef, snf
- Package %XSQL.DS – Call, ContextStatement, Cursor, DML, IResultSet, Statement, TSQL, TempTable, spexecutesql
- Package %XSQL.DSI – Call, Cursor, DeferredStatement, GlobalPrivateTable, IteratorFunction, JPAQuery, Statement, TSQL, TempTable, spexecutesql
- Package %XSQL – Compile, DLLInterfaceUtil, Dictionary, Exception, Format, Reporting, StatementBatch
- Package %XSQL.Informix – FunctionQuery

- Package %XSQL.System – CacheMessageXRef, Message, MessageXRef
- Package SYS.WSMon – Client, EventSink, Service, wsDatabase, wsEvent, wsResource, wsSystem
- Package SYS.WSMon.wse – Delivery, EndpointReference, Identifier, MyRefProp, ReferenceProperties, Renew, RenewResponse, Subscribe, SubscribeResponse, Unsubscribe, UnsubscribeResponse
- Package SYS.WSMon.wsen – Enumerate, EnumerateResponse, Items, Pull, PullResponse, Release
- Package SYS.WSMon.wsman – Client, Locale, MaxEnvelopeSize, OperationTimeout, RequestTotalItemsCountEstimate, ResourceURI, SelectorSet, Service, TotalItemsCountEstimate

Class Component Reservations

The following class components are being reserved for InterSystems use. They will no longer be displayed in the Class Reference and are subject to change without notice. Sites using these components should contact InterSystems Worldwide Customer Support for assistance in replacing these with other published API components.

Class	Type	Name(s)
%Installer.ClassMapping	Method	%OnGenerateCode
%Installer.ClassMapping	Method	%OnGenerateCode
%Installer.Configuration	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode, generateCode
%Installer.CopyClass	Method	%OnGenerateCode
%Installer.CopyDir	Method	%OnGenerateCode
%Installer.CopyFile	Method	%OnGenerateCode
%Installer.Credential	Method	%OnGenerateCode
%Installer.CSPApplication	Method	%OnGenerateCode
%Installer.Database	Method	%OnGenerateCode
%Installer.Default	Method	%OnGenerateCode
%Installer.Else	Method	%OnGenerateCode
%Installer.Error	Method	%OnBeforeGenerateCode
%Installer.ForEach	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.GlobalMapping	Method	%OnGenerateCode
%Installer.If	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.IfDef	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.IfNotDef	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.Import	Method	%OnGenerateCode

Class	Type	Name(s)
%Installer.Installer	Method	CopyClass, CopyDir, CopyFile, CreateDatabase, CreateNamespace, CreateResource, CreateRole, CreateUser, CSPApplication, DeleteVariable, EnableEnsemble, eval, Evaluate, GetVariable, GlobalMapping, Import, InstallFromCommandLine, IsLoggable, IsVariableDefined, LoadManifestXML, LoadPage, Log, PopNS, Production, ProductionExists, PushNS, RoutineMapping, SetVariable, special, SystemSetting
—	Property	Logger, LogLevel, NSStack, Variables
%Installer.Invoke	Method	%OnBeforeGenerateCode
%Installer.LoadPage	Method	%OnBeforeGenerateCode
%Installer.Log	Method	%OnBeforeGenerateCode
%Installer.Manifest	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.Namespace	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.Production	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.Resource	Method	%OnGenerateCode
%Installer.Role	Method	%OnGenerateCode
%Installer.RoutineMapping	Method	%OnGenerateCode
%Installer.RunInstall	Method	%OnGenerateCode
%Installer.Security	Method	%OnAfterGenerateCode, %OnBeforeGenerateCode
%Installer.Sequence	Method	findChild
%Installer.Setting	Method	%OnGenerateCode
%Installer.SystemSetting	Method	%OnGenerateCode
%Installer.Var	Method	%OnGenerateCode
%Library.ProcedureContext	Property	Results
%Net.FtpSession	Method	endCommand
%Net.POP3	Method	GetHeadersArray
%SYSTEM.Config.SharedMemoryHeap	Method	DisplayUsage
%ZEN.Component.abstractPage	Method	%IsEmbedded

Class Component Deletions

The following class components have been removed in this version.

Class	Type	Name(s)
%Activate.UI.Wizard	Property	lblExit, lblFinish
—	Parameter	DOMAIN, TEMPLATEMODE
%CPT.CalloutDump	Method	DumpParseTreeNodeMap, DumpSourceAndMap
%CSP.UI.Portal.Config.Device	Method	DrawLocatorExtra, onloadHandler, UpdateDetails
%CSP.UI.Portal.Config.Devices	Method	DrawLocatorExtra, saveItem, validate
%CSP.UI.Portal.Config.ValueEditor	Method	DrawLocatorExtra
%CSP.UI.Portal.FileManTemplate	Method	%OnAfterCreatePage
%CSP.UI.Portal.NLS	Method	AddLocator
%CSP.UI.Portal.NLSEdit	Method	AddLocator
%CSP.UI.Portal.ObjectGateway	Method	doStart, doStop, itemChanged
—	Property	msgConfirmSaveStart, msgConfirmStart, msgConfirmStop
%CSP.UI.Portal.ObjectGateways	Method	deleteSetting, editSetting, editSettingD
—	Property	PID
%Compiler.Informix.COS	Method	%OnClose, InformixDT2CacheDT, SeparateName, delimitIdentifier, getTypeCategory, normalizeIdentifier, resolveTableName
—	Property	blockTypeStack, codeStack, contextStack, exceptionDepth, exceptionStack, opStack
%Library.IProcedureContext	Property	Message, RowCount, SQLCode
%Library.IResultSet	Property	%RowCount, %SQLCode
%Library.ODBCCatalog	Method	co25Close, co25Execute, co25Fetch, cr25Close, cr25Execute, cr25Fetch, ek25Close, ek25Execute, ek25Fetch, ik25Close, ik25Execute, ik25Fetch, pc25Close, p25Execute, pc25Fetch, ti25Close, ti25Execute, ti25Fetch
—	Query	co25, cr25, ek25, ik25, pc25, ti25
	Parameter	CHARSET, CONTENTTYPE
%Library.ObjectJournalTransaction	Method	PurgeUpToTxn
%Library.ProcedureContext	Property	ReturnValues
%Library.qccServer	Method	OnPage

Class	Type	Name(s)
%MV.File	Property	%AAA01VarType, %AAA02FileName, %AAA03Namespace, %AAA04DictFlag, %AAA05Global, %AAA06Account, %AAA07TargetAccount, %AAA08TargetFileName, %AAA09IsBinary, %AAA10IsTranslated, %AAA11COSName, %AAA12LockName, %AAA13Class, %AAA14IndexType, %AAA15IndexRoutine, %AAA16IndexName, %AAA17IndexColl, %AAA18BSCAN, %AAA19IndFlag, %AAA20IndNames, %AAA21IndColl, %AAA22IndMV, %AAA23Options, %AAA24TriggerRoutine, %AAA25TriggerOperations, %AAA26FireTriggers, %AAA27CasePreserveName, %AAA28IOTable, %AAA29ClassNamespace, %AAA30SectionName
%MV.PropertyParameters	Parameter	MVITYPERTN
%MV.SelectList	Property	%AAA01Type, %AAA02LastReturnedId, %AAA03NextOffset, %AAA04GlobalName, %AAA05Namespace, %AAA06Values, %AAA07Count, %AAA08IndexName, %AAA09IndexFlags, %AAA10IndexColl, %AAA11InReverse, %AAA12LastReturnedMVPos, %AAA13LastReturnedKey, %AAA14CurSub, %AAA15MaxSub, %AAA16ExplodeFlag
%ResultSet.Static	Method	%RowCountGet
%Studio.Project	Method	deployToGbl
%SOAP.Security.SecurityTokenReference	Property	Id
%SYS.PTools.Stats	Property	CursorName. NameSpace, RoutineName
%SYSTEM.CPU	Property	nTheadsPerCore
%TSQL.Transformer	Method	ExecuteStringAsTSQL, ExpandTable
%UnitTest.Manager	Method	ParseQualifiers, RunOneDirTest, RunTestSuite, SaveLog
%UnitTest.Report	Method	IsSuccess, NoNameSpace
%UnitTest.Utility	Method	processError

Class	Type	Name(s)
%ZEN.Auxiliary.dataController	Method	%GetDataByName, %GetTypeByName, %OnNew, %SetDataByName, autoRefreshHandler, clearAutoRefresh, getData, getDataAsArrays, getDataAsObject, getDataByName, getDimSize, getDimensions, getError, getLabel, getModelId, getPropertyName, getTypeByName, hasData, invokeAction, isModelReadOnly, isPropertyValid, notifyController, onDelete, onloadHandler, raiseDataChange, register, sendEventToViews, setData, setDataByName, setModelId, startAutoRefresh, unregister, update
—	Parameter	DEFAULTVISIBLE
—	Property	alertOnError, autoRefresh, dataBag, dataLoaded, dataReadOnly, defaultSeries, modelChanged, modelError, modelId, oncreate, ondelete, onerror, onnotifyController, onsave, readOnly, timerid
%ZEN.Component.abstractPage	Method	%DrawJSTestIncludes
—	Parameter	SYSTEMPACKAGES, SYSTEMSVGPACKAGES
—	Property	%ComponentIncludes, %IncludePackages
%ZEN.Report.Aggregate.StDev	Property	Sum, SumOfSquares
%ZEN.Report.Display.inline	Property	field
%ZEN.Report.Display.p	Property	field
%ZEN.Report.reportPage	Method	%IsTypeNumeric, OnSVGAttribution
%ZEN.SVGComponent.svgPage	Parameter	SYSTEMSVGPACKAGES
%ZEN.Utils	Method	NeedsIncludeFiles, WriteIncludeFiles
Config.CommonMapProperties	Parameter	SECTIONTYPE
Config.CommonMethods	Parameter	NAMESPACE
Config.Config.MapPackages	Parameter	NAMESPACE
Config.Config.MapRoutines	Parameter	NAMESPACE
Config.MapGlobals	Method	NameIsValid
—	Parameter	NAMESPACE
Config.Miscellaneous	Property	NamespacePrompt
Config.Namespaces	Method	MapPkg, MapSub, MapUserGlobals, MapUserPackages, MapUserRoutines
Config.MapPackages	Parameter	NAMESPACE
Config.MapRoutines	Parameter	NAMESPACE

Method Return Changes

The following methods have different return values in this version of Caché:

- Package %CSP.UI.System.ExportPane – DrawResult
- Package %CSP.UI.System.ImportPane – DrawResult
- Package %Library.CacheIndex – Check
- Package %Monitor.Manager – SystemCountersActive
- Package %UnitTest.Manager – PurgeLog
- Package %ZEN.Dialog.fileSelect – GetSubDir

Method Signature Changes

The following methods have different calling sequences in this version of Caché:

Class Name	Method Name(s)
%BI.WebScheduler	addNewPivotTask
%CPT.COSCallout	Compile
%CPT.CalloutCheckTree	CheckParseTreeNode, CheckParseTreeNodeAnn, CheckParseTreeNodeChild, CheckParseTreeNodeSource, ReportAnn, ReportChild, ReportNode
%CPT.CalloutDump	ExpandAnnIndex, ExpandAnnValue, ExpandChildIndex, ExpandNodeType
%CPT.CalloutIndex	BuildNodeIndex, ListEnumIndex, ListNodeIndex, UpdateNodeIndexForEnum, UpdateNodeIndexForNode, UpdateNodeIndexFromLine
%CPT.CalloutTesting	TestStream
%CPT.CalloutTypeIndex	BuildTypeIndex, IsEqualOrDescendantOf, ListTypeClassHierarchy, ListTypeIndex, UpdateTypeIndexForNode, UpdateTypeIndexForNodeClass, UpdateTypeIndexFromLine
%CPT.ISQLCallout	Compile
%CSP.Session	Unlock
%CSP.UI.Portal.SSL	SaveData
%CSP.UI.System.AutoPageCSP	GetLocator
%Collection.ListOfObj	GetObjectIdNext
%Compiler.Informix.Flo	preparseSQL, parseStream
%Compiler.LG.PropWrapper	SetIIJJKK
%Compiler.TSQL.Flo	preparseSQL, sqlact
%Compiler.XML.Generator.Adaptor	GenArrayCode, GenExportCode, GenListCode, GetSimpleExport
%Debugger.System	StackVars
%Installer.Installer	CreateDatabase

Class Name	Method Name(s)
%Library.ArrayOfObjects	%AddToSaveSet
%Library.CacheIndex	Check
%Library.Collation	FMDLTDate
%Library.GlobalEdit	GetGlobalSize, GetSize
%Library.ListOfObjects	%AddToSaveSet
%Library.ProcedureContext	ResolveTableName
%Library.RegisteredObject	%AddToSaveSet
%Library.RelationshipObject	%AddToSaveSet
%Library.RoutineMgr	UserType
%Library.SyntaxColor	Color
%Monitor.Manager	SystemCountersActive
%Net.FetchMailProtocol	Fetch
%Net.POP3	Fetch
%Net.Remote.Gateway	%Connect
%SOAP.Descriptor	GetSoapParameters
%SOAP.Security.SecurityTokenReference	Validate
%SOAP.WSDL.Reader	GetCacheElement
%SOAP.WebRequest	GetSoapParameters
%SQL.Manager.Catalog	SetFieldSelectivity, SetTableExtentSize
%SYS.PTools.SQLStats	Export, Init, Report, Start, Stop
%SYS.PTools.Stats	Init, Report, Start, Stop
%SYS.ZENReportServer	%ServeTransform
%Studio.Project	Deploy, deployToDatabase, NormalizeName
%UnitTest.Manager	AutoLoad, DebugLoadTestSuite, DebugRunTestCase, LogStateEnd, RunTest
%UnitTest.ODBCSQL	processORSToStream
%XML.Utils.SchemaReader	GetCacheElement, ProcessElement, ProcessSequence
%XML.Writer	Object, RootObject
%XML.XSLT.Transformer	TransformFile, TransformFileWithCompiledXSL, TransformStream, TransformStreamWithCompiledXSL, TransformStringWithCompiledXSL
%ZEN.Component.tabGroup	showNextTab, showPreviousTab
%ZEN.Report.Aggregate.StDev	ProcessValue
%ZEN.Report.Display.caption	%DrawCellFO, %DrawCellToHTML
%ZEN.Report.Display.item	%DrawToHTML, %DrawToXSLFO

Class Name	Method Name(s)
%ZEN.Report.Display.node	%GenerateCode, %StyleXSLFO
%ZEN.Report.Display.tableOutput	%DrawCellFO, %DrawCellToHTML, %DrawFooterFO, %DrawFooterToHTML, %DrawHeaderFO, %DrawHeaderToHTML
%ZEN.Report.group	%SortChildren
%ZEN.Utils	%GenerateIncludeFiles
Config.CPF	MoveToActive, UpdateLines
Config.ComPorts	Load
Config.CommonMapMethods	Load, MoveToActive
Config.CommonMethods	Activate, Load, MoveToActive
Config.CommonSingleMethods	MoveToActive
Config.DTMnetBIOS	Load
Config.Databases	Load
Config.DeviceSubTypes	Load
Config.Devices	Load
Config.ECP	Load
Config.ECPServers	Load
Config.ETHServers	Load
Config.IO	Load
Config.Journal	Load
Config.LAT	Load
Config.MagTapes	Load
Config.Map	MoveToActive
Config.Miscellaneous	Load
Config.Monitor	Load
Config.Namespaces	Load
Config.Net	Load
Config.SQL	Load
Config.SqlSysDatatypes	Load
Config.SqlUserDatatypes	Load
Config.Startup	Load
Config.Telnet	Load
Config.UDPServers	Load

Method Displacements

The following methods have been removed from one class and added to another:

- Method: KPI – from %BI.Soap to %BI.Util
- Method: Pivot –: from %BI.Soap to %BI.Util

DBTIME Implementation Changed

The DBTIME parameter was first introduced in version 2008.2. When set to 1, it caused an index to be defined in the class on a property that was, by default, set to an incremented global on each insert and update; but the index entry for an object was not updated for deletes.

Beginning with this version, DBTIME no longer generates an index definition and property definition. A global, ^OBJ.DBTIME, is set for each insert, update and delete. The first subscript is the incremented global counter - ^OBJ.DBTIME; the second is the class name of the object being filed; and the third is the ID of the object. The value of the global node for each filing event is the type of filing - 0 for update, 1 for insert and 2 for delete. This structure is cumulative. It is used to support various features in Cache and it can be used by user applications for any purpose. Deleting entries from ^OBJ.DBTIME is not automatic and must be done by the user.

Permit Insert Of Trailing Objects In Collections

The SetAt method of a list collection allows the caller to specify an element value and a key. In prior releases, if the element at the key position was not already defined, then the attempt would fail.

This behavior is now changed. The call will still fail if the key is not 1 (the first element position) and the element at the previous position is not defined. Otherwise, the call will succeed and the behavior for calling SetAt(value,key) for a position that is undefined will be the same as Insert. That is, a new element at position "key" will be defined.

Update ID Counter When IDENTITY Set Explicitly On Object Insert

If a value is provided for the IDENTITY property of a new object and the INSERT_IDENTITY option is set TRUE, then the ID of the new object is set to the value of the IDENTITY property.

In previous versions, an error prevented subsequent inserts without the IDENTITY property from succeeding if the ID counter value was the same as a previously inserted object. This has been corrected and an explicit set of an IDENTITY property will update the ID counter as long as the explicit value is greater than the current ID counter value. As part of this correction, another error that prevented a check of the INSERT_IDENTITY option for batch inserts and direct saves has been fixed.

%OnDetermineClass Method Override Blocked

%OnDetermineClass is a class method implemented for persistent, stream and serial objects. It has always been defined to be PRIVATE, but PRIVATE for class methods has never been enforced until this release. Any application class that overrides **%OnDetermineClass** will now have to update the method definition to remove the `PRIVATE` keyword.

Indexes Permitted on %Stream.GlobalCharacterSearchable

Several defects that caused corruption in an index based on a property whose type class is %Stream.GlobalCharacterSearchable have been fixed in this version of Caché. These defects caused updates to the stream to be ignored and resulted in index entries based on the old stream value. Also, deleting a container object did not fully remove the index entries from the stream in all cases.

As part of the correction of these defects, indexing streams is only supported for instances of %Stream.GlobalCharacterSearchable. Instances of this class must be processed through the container. It is not legal to extract the stream OID from the container and process updates to the stream without saving the container.

CAUTION: Existing applications should work well with this change as no change to the existing data structures was made. However, the old behavior almost certainly introduced corrupt index structures. Those indices should be purged and rebuilt after installing this version.

Namespace Prompt Property Changed

The property, *NamespacePrompt*, in the Config.Miscellaneous class has been moved to the class, Config.Startup, and renamed to be *TerminalPrompt*.

13.2.2.6 Class Compiler Changes

This version of Caché continues the work begun in earlier releases of improving the class compiler. The changes that may require changes to applications are detailed in this section.

Class Inheritance Default Order Changed

In version 2009.1, the *Inheritance* keyword was introduced to specify the order of inheritance from superclasses. In that release, the default for classes that do not specify the order was "right", as it had been before the keyword was introduced.

In this version, the default inheritance order for classes that do not explicitly specify an order has been changed to "left".

Superclass Availability Required

In prior versions, subclasses compiled all the method code from their superclasses into themselves. In this version, subclasses contain references to their superclass methods thus avoiding the extra code and saving compilation time. A consequence of this organization is that the first reference to a method in a class requires that all the superclasses be accessible to resolve the reference, that is, either present in the namespace of the calling method or mapped to it.

Flags And Qualifier Removals And Deprecations

In this version of Caché, the following changes made to compiler flags and qualifiers may affect existing operations:

- Optimization flags ignored

  Due to the more dynamic nature of the class and routine dispatching in this version of Caché, the optimization flags ("o1", "o2", ...) are ignored if passed in as arguments. The system code was optimized to make up for any reduction in speed from turning these optimizations off.

- Force flags

  The force flag, "f", is now deprecated.

- SQL compilation deprecated

  The "q" flag that specified SQL-only compilation is now deprecated.

- Keep-valid deprecated

  The "v" flag previously used to keep the objects valid after compilation finishes is now deprecated.

- The following qualifiers are no longer allowed:
  - /deletextent – used to delete the extent associated with a class
  - /version4compatible – used to export CDL files that are version 4 compatible

$GET(..<property>) Require <property> To Be Multi-Dimensional

In previous versions, $GET applied to a class property which is not multi-dimensional may not have been caught as an error when certain optimization flags were set. This has been corrected. The expression

```
$GET(..AClassProperty)
```

is valid only if *AClassProperty* is marked as `Multidimensional`.

Note: This restriction also applies to other commands with respect to class properties: **KILL**, **MERGE**, **$DATA()**, and **$INCREMENT()**.

$PIECE References to ..<property> Require <property> To Be Multi-Dimensional

In previous versions, $PIECE treated a reference to `..<property>` as a string. Because of changes to the compiler, such references will now result in an <OBJECT DISPATCH> error. Applications using this form should either add the `Multidimensional` qualifier to the property, or use a temporary string as in:

```
Set ..MyProperty = "Piece1/Piece2"

Set temp = ..MyProperty
Set $PIECE(temp,"/",2) = "Piece3"
Set ..MyProperty = temp
```

Misuse of %this Prohibited

In prior versions of Caché, it was possible to gain access to the private properties of another object with the sequence

```
New %this
Set %this = OtherOREF
Set Value = ..PrivateProperty
```

This will not work in the new class compiler due to system changes. Any applications using this questionable practice must be updated.

System code public/private protection has now been changed; from a method of class X access to private members of another instance is possible only as long as the origin of the member is class X or a superclass of class X. This will allow most common use of "Set %this=" to be changed to just access the private properties directly.

InterSystems also found code where a classmethod was executing

```
Set value=##this.Property
```

which was implicitly referencing %this on older Caché systems where the value of %this happened to be the last instance method that was called. This is a very risky practice since the last instance is not well-defined; if the classmethod is called as the first thing, then %this will not be defined at all. Applications should be written to pass in the OREF of the object to the classmethod explicitly.

In the new compiler, references to "##this" are converted to **$THIS**, the current class context.

State Of Serial Object Now Held In %%OID

In prior versions, the serial state of an object was being kept internally. This has now been made visible in a real property called %%OID. It is also updated for serial and stream objects. It is used in %Save to correctly handle the case where a serial object is independently serialized so it reports %IsModified()=0 to the save attempt, but the container does not know the new serial value. Using %%OID, the container can always get the correct version of the serial object.

Setting A Serial Property To "" Causes It To Be Initialized

When a class defines a serial property, constructing a new instance of this class and referencing this serial property will cause it to be created using the **NewObject()** method. If the application then sets this property to "", the property is now reinitialized and the next reference will create it again using **NewObject()** just as if theoparent object had been newly created.

NotInheritable Keyword Removed

In previous versions, Caché allowed a user to define a 'not inheritable' property defined with an expression like:

```
Property A As %String [ NotInheritable ];
```

This concept breaks object inheritance because the superclass now has interfaces the subclass does not.

Beginning with this version of Caché, this is no longer permitted. Compiling a subclass of a class with a noninheritable property will result in the an error because property "..A" was not present in the subclass.

When upgrading to this version, the upgrade procedeures will automatically remove "NotInheritable" from any property and thus convert any noninheritable properties into inheritable ones. When it does this, it will add the comment [Previously notinheritable] to the property description.

Any new properties marked as "NotInheritable" will cause an error when they are compiled. Studio will be modified to remove the ability to set or display this keyword.

Private Classmethods Now Supported

Caché now supports private class methods. Private classmethods cannot be called from outside of the class (or a subclass). For backward compatibility, all class methods have been marked "public" so customer code will continue to work correctly. The descriptions of these methods have been altered to add the string, "[Previously private]", to them.

New private classmethods added to classes in the future will be inaccessible from outside the class.

Change In Support For COSEXPRESSION

Prior to this release, the use of `COSEXPRESSION` in a parameter definition would be treated in the same manner as macro substitution. That is,

```
parameter P1 as COSEXPRESSION = """Red"","Green","Blue""";
...
...

Set X = $LISTBUILD(..#P1)
```

would be treated as if it were

```
Set X = $LISTBUILD("Red","Green","Blue")
```

In this version of Caché, a `COSEXPRESSION` is constrained to return only a single value. So the parameter definition must be rewritten as

```
parameter P1 as COSEXPRESSION = "$LISTBUILD(""Red"","Green","Blue"")";
```

or an error will be reported at the time the class is compiled.

Superclass Reference ##super

The usage ..ClassmethodName() now always calls that classmethod in the current class, not that of the superclass. Classmethod in the superclass are now accessed with the expression ##super.ClassmethodName().

Codemode=Call Cannot Using Variable-Length Argument List

Method marked as "codemode=call" are no longer allowed to use variable length argument lists. Such use will cause an error when they are compiled. They need to be rewritten to specify a fixed list of arguments.

Normalize Results of %Exists And %ExistsId For Subclasses

In prior versions, if an application called %Exists passing a fully-formed OID (one that included the classname in it), then Caché always polymorphically dispatched to the **%Exist** method of that class. For example, calling **%Exists** on Sample.Employee passing an OID of a Sample.Person would return true. After analysis, this is considered to be a error since calling **%ExistsId** on Sample.Employee passing the same OID would return false because this is not an instance of Sample.Employee or one of its subclasses.

In this version, the behavior is changed so **%Exists** and **%ExistsId** now both return the correct result in this case. It is possible, but unlikely an application may have been relying on this incorrect behavior.

Limits On Class Descriptor Size

Compilation of a class generates a data structure called a "class descriptor" that contains summary data about the class and its components: properties, methods, parameters, indexes, and so on. Information on each group (such as the methods defined) is stored in its own table and is subject to the maximum limit on Caché strings, 32KB.

In this version of Caché, the table formats have changed. The amount of information stored for each method has increased, while that for a property has decreased. The result is that classes which contain many hundreds of methods or properties may find that the resulting class descriptor exceeds one or more of its table limits resulting in a <CLASS TOO BIG TO SAVE> error.

Note: Classes which encounter this error must be refactored to reduce the number of entries.

The number of methods, properties, indexes and so on includes inherited and generated components not only the ones explicitly visible in the class source. Also, properties may cause the generation of methods, for example, <propertyname>Get and <propertyname>Set.

Names Limited To 180 Characters

Compiled classes are stored in globals indexed by the package name, class name and component name. When each of these is a long string of characters, there is the possibility that the maximum length for a global subscript will be exceeded. To reduce the likelihood of this happening, this version of the compiler imposes a maximum size limit on the names of methods, parameters, and properties declared in a class. That limit is 180 characters.

Stored Procedure Return Parameter Types Checked

Prior to this version, return value type parameters were ignored. Beginning with this version, the return type parameters of a class method are now honored by the stored procedure projection. For example,

```
classmethod myproc() as %String(MAXLEN=100) [ sqlproc ] {
...
}
```

will now project the xDBC column with a maximum string length of 100. Previously, it would have used the default of 50,

13.2.2.7 Object Binding Changes

Change In Properties Projected For Jalapeño

Prior to this version, all properties that are defined in Caché class that are not marked as Private or ServerOnly, or otherwise not projectible, are present in Java POJO class. Now, properties with names starting with "%" will not be projected in POJO. This will affect user classes if they have properties whose name begin with "%".

13.2.2.8 SQL Changes

Changed Handling Of Procedures Names In Embedded SQL

In this version, Caché now treats unqualified procedures names in the &SQL statement differently from prior versions. The schema name used for unqualified procedure names now takes into account any #import value defined. This means if you have an embedded SQL CALL statement ("&SQL(...)") in a class method and the procedure name is unqualified, the default will be the schema that maps to the package of the current class. Previously, the default schema would be the system-defined default schema. The default schema for DDL statements remains the system defined-default schema.

Support For SQL Extension FOR ALL Removed

In this version, support for the FOR ALL statement has been removed. In version 2009.1, it was deprecated. Any application using FOR ALL will need to be modified. If necessary, the FOR ALL statement may be converted into a

```
NOT FOR SOME ...
```

statement with a condition carefully constructed to achieve the desired result.

Date And Time Conversion Errors Now Reported In SQL Statements

In an SQL statement that takes a %Date value as input in either DISPLAY mode or ODBC/JDBC mode, if the input value cannot be converted to a valid Logical %Date value, the SQL statement will return SQLCODE=-146. For similar errors involving %Time values that cannot be converted, SQLCODE=-147 will be returned. For example,

```
&sql(select name into :name from sample.person where dob = '02/29/2009')
```

will return "SQLCODE=-146 Unable to convert Date input to a valid Logical Date value".

Reserved Word Checking More Comprehensive

In prior versions, a reserved word would be accepted as an identifier, for example:

- As an alias following AS in a SELECT or FROM clause,
- As a table reference in a FROM clause

These occurrences will now be reported as errors. This is unlikely to affect many queries since the reserved word could not be used in WHERE, GROUP BY, HAVING, or SELECT clauses, nor as the table qualifying name for a column.

ORDER Is No Longer A Reserved Word

ORDER is no longer a reserved word. It is now a non-reserved keyword. ORDER can be used everywhere a regular identifier is used.

Use Of TOP and ORDER In Subqueries

In prior versions, there were two equivalent ways to apply TOP to a UNION:

```
SELECT TOP ...
FROM (SELECT .... UNION ... SELECT ...)
ORDER BY ...
```

and

```
SELECT TOP ...
...
UNION
...
SELECT
...
ORDER BY
```

Starting with this version, the first syntax form will be required. The second example is considered a defect where TOP always applied to the outermost processing level no matter where it was specified.

CASE Clause Result Type Change

In prior versions, the type reported by a CASE expression would be the same as the first potential return value of the CASE expression. This has been changed to return the best compatible type of all the possible return value types of the CASE expression. Caché will now also use the largest length, precision, and scale of all the potential return values; this applies to numeric return values only.

The precedence of return types (highest to lowest) is:

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

Thus, if a CASE expression may potentially return TINYINT, INTEGER, or NUMERIC values, the type of the column will be NUMERIC because NUMERIC has the highest precedence.

13.2.2.9 SQL Gateway Changes

Settin Auto-Commit

In prior releases, it was necessary to call both **SetAutoCommit^%apiGTW** and **SetAutoCommit^%apiSQL** to change the auto-commit setting. With this release, the **SetAutoCommit^%apiGTW** call is the one that will take effect.

13.2.2.10 XML Changes

Exported xsi:type Now Matches Schema

The *xsi:type* attribute for datatype classes now reflects the actual type of the value. Previously, the *xsi:type* attribute was always set as the base predefined type, for example, `s:string`.

Mixed Content Now Properly Marked In Schema

For all elements identified as "complexType" with an attribute of `mixed="true"`, generate a `CONTENT="MIXED"` property as well as properties for each of the elements in the complexType. This change can cause compatibility problems if regenerating a previously existing set of classes from the XML schema because element properties were not previously generated by the Schema Wizard.

%XML.Reader Now Returns SAX Parser Errors

In this version of Caché, %XML.Reader now returns SAX parser validation errors. Previously, the SAX parser was reporting the error, but there was a defect that prevented the error from being properly reported.

SAX Namespaces Change

In Xerces 3.0 (the version used in this release), there is a change to the way namespaces are processed. The flag to recognize namespaces is on by default. Turning it off will now cause the localname of the returned startElement callback data to be the empty string; previosly, it was a copy of the qname (fully-qualified name).

13.2.2.11 Zen Changes

Client-Side JavaScript Change

Zen applications that use client-side JavaScript methods will need to change the declarations of these methods. Where the method signature formerly began with the "Method" keyword, such as

```
Method test() [Language = javascript]
{
}
```

the declaration must be changed to

```
ClientMethod test() [Language = javascript]
{
}
```

Without this change, Zen classes will not compile or run on this version of Caché.

Changes To Include File Handling

The way that JavaScript and CSS files are generated for Zen components and the way that these files are used at run-time has changed with this release, namely:

- Modules

  Prior to this change, Zen would create a single JavaScript (.js) and stylesheet (.css) file for each class package containing Zen components. With this version, there is a mechanism for greater granularity – within a package, one or more Zen components can be placed into collections called modules.

- Reorganization of JavaScript on pages

  Zen pages are changed so that JavaScript includes and inline JavaScript are placed at the bottom of a page, after the HTML content is served. This should result in faster display of pages.

- Dynamic loading of component JavaScript and stylesheets

  When a component is created, the browser must have its JavaScript and stylesheet definitions on hand. This is accomplished by including the entire library within one large include file. This, in turn, allows dynamic creation of Zen components after a page has been loaded.

- Use of include files for *all* non-page components

  Prior to this, it was possible for a non-page component to have its JavaScript and stylsheet inline within a page. With this change, all non-page components will use include files.

- Removal of Zen background file generation

  The generation of JavaScript and stylesheet files now takes place as part of the compilation; the background file generation mechanism is no longer needed.

Zen dynaTree NodeCount Changed

ZEN dynaTree has a property that counts the nodes in the tree. With this version of Caché, the count now includes all the nodes in the tree, not just those at the top level.

13.2.2.12 Java Binding Changes

Caché Version 2010.1 Is Compatibility Break

Because of internal changes to the way the Java binding manages memory for objects, version 2010.1 represents a break in cross-version compatibility. Java clients on any version of Caché from version 5.0.13 to 2009.1 are compatible with Caché servers in that same range of releases. Those clients will receive an exception if they attempt to access a Caché server running 2010.1.

Java clients attempting to access a Caché server running version 2010.1 must also be running on 2010.1.

13.2.2.13 xDBC Changes

Make Catalog Queries Independent Of Version

For ODBC Catalog queries there are differences between ODBC 3.5 and 2.5. In version 2009.1, this was addressed by increasing the protocol number. This will not continue to work because SQL Server can dynamically switch between returning data for version 3.5 or version 2.5 by calling SQLSetEnvAttr. This sets the SQL_ATTR_ODBC_VERSION option to either 2 or 3 depending on which type of ODBC error messages or catalog information it wishes to present.

In this version, ODBC allows an application to dynamically switch between the two environment settings. Calling SQLAllocEnv sets the environment to version 2.5 by default. If you want to get the version 3.5 catalog information, you need to call SQLSetEnvAttr. If you use SQLAllocHandle (3.5) the driver manager will force you to call SQLSetEnvAttr before making a connection by giving you an out of sequence error.

Preserve Scale For %Numeric Values

In this version of Caché, the JDBC driver no longer ignores the scaling for %Numeric 0 values; it did in previous releases.

13.2.2.14 Callin And Callout Changes

Compatibility Between Caché And External Routines

When using Callin or Callout, the Caché instance and the external program exchanging data must have the same view of the characters exchanged. That is, they must both use a Unicode representation for strings, or both use an 8–bit representation. Mismatched exchanges where one side is Unicode and the other are not permitted. Failure to conform to this requirement may result in segmentation or access violations.

13.2.2.15 MultiValue Changes

Interpretation Of Sleep Value Changed

In prior versions, the MV shell SLEEP command would interpret 2:14 as sleeping for 2 hours and 14 minutes. Now it will be interpreted as a time of day. Beginning with this version the value will be interpreted in one of two ways:

- If the value is of the form, `HH:MM[:SS][AM|PM]`, it will be interpreted as a time of day. The system will sleep until the next occurrence of that time (which may be tomorrow if that time today is past).
- Otherwise the value will be interpreted as the number of seconds to wait from the current time.

Note: This interpretation also applies to the MVBasic SLEEP and RQM statements.

13.2.2.16 Platform-specific Items

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

13.2.3 Operators

13.2.3.1 System Management Portal

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

14

Caché 2009.1

This chapter provides the following information for Caché 2009.1:

- New and Enhanced Features for Caché 2009.1
- Caché 2009.1 Upgrade Checklist

14.1 New and Enhanced Features for Caché 2009.1

The following features have been added to Caché for the 2009.1 release:

This version of Caché has been improved and enhanced in the following areas:

- Performance and Scalability
  - Class Compiler Optimizations
  - XML Performance Enhancements
  - Faster Namespace Activation
- Rapid Application Development
  - .NET Gateway
  - Studio Enhancements
  - Zen Reports
- Reliability, Availability, Maintainability
  - IPv6 Support
  - Configuration File Robustness
  - Truncate Caché Databases
  - Support For 64–Bit Macintosh
  - SNMP and WMI Shadow Monitoring
  - Management Portal Reorganization
  - Embeddable Installation

- Security
  - XML Encryption
- Distribution
  - Weblink

In addition, many more localized improvements and corrections are also included. In particular, if you are upgrading an existing installation, please review the detailed list of changes in the Upgrade Checklist.

14.1.1 Performance And Scalability

14.1.1.1 Class Compiler Optimizations

Several improvements and changes were made to improve compile time for classes. The compiler now generates additional information that is used by later compilations (e.g. incremental) to reuse already compiled code.

Some of these improvement require changes to existing classes in order to execute on this release. Please refer to the Upgrade Checklist to assess the impact on your application.

14.1.1.2 XML Performance Enhancements

The internal content handler for the XML parser has been optimized to interact with Caché more efficiently, which can result in substantial performance improvements when loading complex XML documents.

14.1.1.3 Faster Namespace Activation

Namespace activation has been sped up in this release. The effect is most noticeable in Caché instances with many namespaces and/or complicated subscript-level mappings.

14.1.2 Rapid Application Development

14.1.2.1 Informix Stored Procedure Converter

This version adds the ability to convert Informix SPL routines to Caché class methods. The resulting class methods contain Caché ObjectScript code that replicates Informix SPL behavior, including temporary tables, deferred statement resolution, and exception handling. The converter will optionally log all converted routines, errors encountered, and the name of the class/method generated. A summary of all errors encountered during conversion is also placed at the end of the log.

14.1.2.2 .NET Gateway

The .NET Gateway provides an API for generating proxy classes for custom .NET components or proxy class mappings for packages such as ADO, Remoting, ASP.Net, and so on. The generated proxy classes can be accessed through Caché Basic, MVBasic, or ObjectScript from Ensemble or Caché. Applications can issue method calls out to the Microsoft Common Language Runtime (CLR) on systems located locally, or remotely over a TCP/IP connection.

14.1.2.3 Studio Enhancements

Studio was reimplemented for this version using Microsoft Visual Studio 2008. Because of this, the look and feel of the user interface has been updated to a more modern look. The functionality provided by Studio remains the same but there are new graphics, icons and greater flexibility in how you manage your personal workspace in this version.

Additional significant work was done in 2009.1 to improve Studio performance. This version implements client side class caching and indexing that results in dramatic improvements in class load time. Each time the Studio is launched, the index

is checked against the server. If no changes are detected, the classes are loaded from local cache. Classes that have been changed are invalidated and refetched from the server. This change should benefit most those who use the Studio over a network.

14.1.2.4 Zen Reports

The Zen Reports facility introduced in a prior version has been greatly enhanced for this version. New features include:

- The ability to use external XML data sources and XSL stylesheets
- Conditional Elements and Styles in a report
- Support runtime sorting in the report rather than predefined in the query
- Base level support for Pivot tables

For more details, please refer to Using Zen Reports.

14.1.3 Reliability, Availability, Maintainability

14.1.3.1 IPv6 Support

Caché has been enhanced to support the use of IPv6 addresses. When IPv6 is enabled, Cache accepts an IP address in either IPv6 or IPv4 format or as a host name, with or without domain qualifiers. IPv6 is supported for all TCP/IP connections. You may also wish to review a summary of IPv6 usage and references to related IETF (Internet Engineering Task Force) documents.

14.1.3.2 Configuration File Robustness

This release includes a number of changes to increase the robustness of the configuration file and now contains a reliable internal structure to support various editing techniques to modify the configuration file. Valid files from earlier releases will be automatically converted to the new format upon first upgrade to this version.

WARNING! Invalid CPF files may prevent Caché from installing or starting up.

Important: 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.

If you do not have a userid and password, contact InterSystems Worldwide Response Center (WRC).

14.1.3.3 Truncate Caché Databases

This version of Caché introduces a facility to return unused database space. It works by truncating the trailing free blocks in a database file and immediately releasing the reclaimed space back to the filesystem. The truncation can be requested via the ^DATABASE routine menu, or via the SYS.Database API.

Note: This feature is presently available on Windows and UNIX® platforms only; furthermore, it is not applicable to databases with 2KB block sizes, or databases with raw volumes.

14.1.3.4 SNMP and WMI Shadow Monitoring

This version of Caché adds the capability to monitor Shadow Journaling using either the SNMP or WMI interfaces. The added data objects make available the data from the System Management Portal **[Home] > [Configuration] > [Shadow Server Settings]** page, and include information about each Data Source and Destination Shadow Server connection.

The exact object names, structure, and details about the implementation can be viewed using the Caché MIB (ISC-CACHE.MIB) or MOF (IscProv.mof) files, which are installed with each Caché instance.

14.1.3.5 Support For 64–Bit Macintosh

This release adds Macintosh OS X 64–bit support.

14.1.3.6 Management Portal Reorganization

Several changes and enhancements have gone into the System Management Portal:

- Configuration

  The configuration pages have been reorganized to provide more clarity in the layout. This should assist in locating relevant information.

- SQL

  A separate page now allows for management of FILEMAN data sources.

- Databases

  An option to access the new feature that permits an administrator to reclaim freespace in a database was added. Administrators can also run Integrity check on selected globals inside a database from the Management Portal.

- Licensing

  There is now an option on the licensing page that directs the Managment Portal to display license usage by user.

14.1.3.7 Embeddable Installation

Beginning with this version, Caché installation on Windows will use the native .MSI installation facilities. An MSI editor, such as Orca from Microsoft, can be used to customize an installation to only install required components as well as incorporate application specific components.

In addition, there are two new packages, Config and %Installer that applications can use to create an installation manifest class. This class can be run during product installation to accomplish:

- Creation of namespaces and databases
- Definition and activation of global, routine and package mapping
- Loading of globals, routines and packages
- Compilation of loaded routines and packages
- Execution of specified routines and classes

The packages and their usage are described in the "Creating and Using an Installation Manifest" chapter of the *Caché Installation Guide*.

14.1.4 Security

14.1.4.1 XML Encryption

This release introduces XML Encryption for SOAP messages sent by Caché Web services and Web clients. The implementation is based on WS-Security 1.1 and uses the EncryptedKey element in the message header to use X.509 certificates. Please refer to *Securing Caché Web Services* for further details.

14.1.5 Distribution

14.1.5.1 Weblink

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 \doc\ subdirectory of the Weblink kits. If you have any questions, please contact the Worldwide Response Center.

14.2 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.

14.2.1 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.

14.2.1.1 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 Response Center (WRC).

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.

14.2.1.2 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 Response Center (WRC).

Management Portal Changes

Because of the new rules for resolving 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.

Other Mapping Changes

The mappings of the *^oddPROJECT*, *^CacheMsg*, and *^CacheMsgNames* globals have been changed; they will be stored in the default ROUTINE database for a namespace, rather than in the default GLOBAL database for a namespace. On some systems where the global is already defined, and the namespace had separate routine and global databases for the namespace,

these would be merged from the GLOBALS database to the ROUTINES database defined by the namespace, for all namespaces defined on the system during installation.

If the application has an *^oddPROJECT* global defined in two different namespaces which share the same global database and different routine databases, the end result of this would move the *^oddPROJECT* global into one of the namespaces, and it would not be visible in the other. The solution for this is for the customer to manually copy the *^oddPROJECT* global from one namespace to the other.

14.2.1.3 Version Interoperability

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

14.2.1.4 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.

14.2.1.5 Operational Changes

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.

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.

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.

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 Response Center (WRC) for guidance.

Change In Default Translation For Telnet

Beginning with this release, he following locales had their default translation for Telnet 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

Among other reasons for this change is the fact that the Latin1-based locales in the above list such as deuw, enuw, ptbw, and so on cannot read the Euro sign via telnet using RAW because this character stands above 255.

Note: UTF-8 is totally compatible with ASCII. This means that locales with languages that use ASCII exclusively won't 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.

14.2.1.6 Platform-specific Items

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

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.

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.

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 beODBC 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 "Installing Caché on Microsoft Windows" chapter of the *Caché Installation Guide*.

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.

LAT Configuration

The [LAT] section of the config.cpf file is incomplete in this version of Caché. The System Management Portal does not configure the advertised services: ServiceName, ServiceDescription, and ServiceRating. Therefore, the LAT daemon (lat.exe) now it reads its startup parameters from a lat.ini file in the installation directory instead of from the cache.cpf configuration file. Users employing LAT finctionality should copy the [LAT] section from the existing cache.cpf file into a new lat.ini file or create the information with a text editor.

A sample [LAT] section is shown here. There can be a maximum of eight LAT services advertised. This sample has two. Each service consists of a ServiceName, a ServiceDescription, and a ServiceRating.

```
[LAT]
NodeName=
MessageRetransmitLimit=8
HostMulticastTimer=60
NodeGroups=0
UserGroup
s=0
ServiceName_1=CACHE
ServiceDescription_1=Cache LAT service
ServiceRating_1=1
ServiceName_2=Cache2
ServiceDescription_2=Second LAT device
ServiceRating_2=1
```

Note: The LAT service will be deprecated in Caché version 2010.2.

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.

14.2.2 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.

14.2.2.1 General Operational Changes

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.

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.

%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.

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.

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.

14.2.2.2 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.

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.

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 hostname or specific address.

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.

14.2.2.3 Objectscript Changes

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.

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.

ZINSERT Or ZREMOVE On Current Routine No Longer Allowed

ZINSERT and ZREMOVE cannot 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.

14.2.2.4 Routine Compiler Changes

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 preprocessor 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
```

14.2.2.5 Class Changes

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.

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.

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.

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.

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.

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.

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.

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.

%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.

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.

%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.

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

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

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, DumpParseTreeNodeChild, 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

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 Response Center (WRC) 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

14.2.2.6 Class Compiler Changes

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

%Double.DisplayToLogical Now Returns $DOUBLE Result

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

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.

<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.

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.

Remove IVARmultidimensional Keyword

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

14.2.2.7 Object Binding Changes

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.

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.

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

14.2.2.8 SQL Changes

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.

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.

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
```

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
```

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.

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.

Computed Property Changes

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

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, the system 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.

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.

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.

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'.

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.

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.

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.

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.

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".

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).

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.

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.

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.

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.

14.2.2.9 SQL Gateway Changes

Use %Library.Double For Linked Tables

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

14.2.2.10 CSP Changes

Keep CSP Info Off Nonsecure 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>&CSP-SHARE=1

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.

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".

14.2.2.11 Zen Changes

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.

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.

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.

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;
```

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.

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.

14.2.2.12 Zen Reports Changes

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.

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.

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.

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.

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**.

Large Reports May Fail to Compile

In this version, some very complex report definitions may fail to compile because the code they generate exceeds 32K characters. In this case, please contact the InterSystems Worldwide Response Center (WRC).

14.2.2.13 Web Services Changes

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.

14.2.2.14 xDBC Changes

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()

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)))

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
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

MetaData Table Name	ODBC 2.0	ODBC 3.x
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
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

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.

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.

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.

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.

14.2.2.15 MultiValue Changes

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.

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.

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 cannot be determined.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

14.2.2.16 Jalapeño Changes

NetBeans 5.5 Plugin Removed

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

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.

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.

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.

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.

14.2.2.17 Source Control Changes

%Project Global Renamed

Prior to this version, Studio used to keep a copy of the currently active %Studio.Project in the public variable, *%Project*. This prevented the object from being closed correctly when Studio removed the project. Beginning with this version, the current project name is stored in a process private global, ^||*%Studio.Project*. Functions that need data about this object shold use this new location.

14.2.2.18 Platform-specific Items

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

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.

Terminal Log Files From Unicode Systems Now In Unicode

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

$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.

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.

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.

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.

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.

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.

14.2.3 Operators

14.2.3.1 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.

15

Caché 2008.2

This chapter provides the following information for Caché 2008.2:

- New and Enhanced Features for Caché 2008.2
- Caché 2008.2 Upgrade Checklist

15.1 New and Enhanced Features for Caché 2008.2

This version of Caché has been improved and enhanced in the following areas:

- Performance and Scalability
  - Object Performance
  - Language Performance
  - Binary SOAP
  - Routine Buffer Management
  - Shadow Latency
- Rapid Application Development
  - Studio Enhancements
  - Zen Enhancements
  - Text Search
  - SQL Support For Streams
  - Light C++ Binding
  - JIS2004 Support - Includes support for surrogate pairs
  - Subroutine-level Profiler
  - $DECIMAL
  - Updates To Third-party Software
- Reliability, Availability, Maintainability
  - CSP

  - National Language Support
  - Improved Installation
  - Per Socket Keep Alive
  - Improved Protection Against Starting Multiple Copies of Caché
- Security
  - Superserver Can Accept SSL Connections
  - Telnet over SSL for Windows
  - SQL Column-level Security
  - Web Services Security (WS-Security)

In addition, many more localized improvements and corrections are also included.

15.1.1 Performance and Scalability

15.1.1.1 Object Performance Improvements

In order to improve performance and reduce contention for method dispatch and property access, Caché now caches the locations of these items upon first reference. Subsequent requests for an item first examine the cache and, if found, avoid the effort involved in locating the requested item. This improves the speed of invocation and access to class members. It also eliminates contention on key class metadata when thousands of processes are running.

15.1.1.2 Language Performance Improvements

The routine compiler has been enhanced. The tokens marking the beginning of line have been removed and the bookkeeping of statements in the routines has been improved.

The net result of these changes is that compiled object code should be slightly smaller and faster. There is no longer any timing effects for the use of comments in block structures.

CAUTION: This is a significant change to the ObjectScript compiler and therefore increments the compiler minor version number. Code compiled on earlier versions of Caché will run on this version, but the reverse is not true. Code compiled on this version of Caché will NOT execute on earlier versions.

15.1.1.3 Support For Binary SOAP

This version of Caché enhances the SOAP facilities to allow the binary transmission of SOAP messages between Caché instances. The Web service will then support normal XML-based SOAP or Caché proprietary SOAP format over HTTP. The WSDL produced for a Cache Web service using the binary extension has been enhanced to carry the necessary information for the Cache Web client. This is also a performance improvement since the transmitted data is in a more compact form.

15.1.1.4 Routine Buffer Management Improvements

In prior versions of Caché, all routine buffers were the same size, configured to hold the largest routine to be executed. This release of Caché implements multiple routine buffer sizes, providing separate pools of 4KB, 16KB and 64KB buffers. In most systems, the size of compiled routines span a range; allocating several buffer pools of differing sizes provides better matching of buffers to routines sizes. This, in turn, provides more efficient use of memory, and allows the administrator to better optimize system resources.

15.1.1.5 Shadow Latency Improvements

Shadowing latency has been decreased in this version. Caché now pre-scans the journal file for database updates and pre-fetches the data to be applied to the database.

15.1.2 Rapid Application Development

15.1.2.1 Studio Enhancements

Studio

Studio now has an option "Track variables" which switches on a limited form of variable tracking that allows it to identify variables which are used when they don't have a value. This is called READ-UNDEFINED.

Studio Assist

The following improvements have been made to Studio Assist as aids in program development:

- When the indicator sequence for macros is typed, "$$$", Studio now displays the list of commonly-used system macro names. User-defined macros are also added to this list. Previously, Studio had displayed all defined macros in the system.
- When Studio Assist recognizes the beginning of a reference to a class parameter, (..#), it will allow a selection from among the currently defined parameters for the class.
- Studio will now also provide a list of available local variables for name completion when appropriate. It will also specially color variables used only once in the bounds of a routine or method since these may be typographic errors.

15.1.2.2 Zen Enhancements

In this version Zen has been enhanced. There are new menu formats and desktop components:

- Zen now supports a drag-and-drop API.
- Active groups may now be dynamically resized.
- Zen provides a "lazy" tree control that only loads the data for a sub-tree when it is expanded tree.
- A new tab control has been added.

15.1.2.3 Text Search Enhancements

Text Search in Long Streams

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

Query Optimizations For Text Searches

The SQL optimizer has been improved for searches involving text queries. It now takes account of the fact that text strings are made of individual words that themselves can be indexed as collections.

15.1.2.4 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 ODBC and JDBC gateways now provide this support as well.

15.1.2.5 Light C++ Binding

Caché has changed the semantics of the Light C++ Binding object references. In prior releases, each "open" of an object created a new reference to a copy of the instance. Now the semantics for "open" match the semantics of regular C++ binding object references; all references to the same object point to the same local storage.

This version also provides support for embedded objects and inheritance.

15.1.2.6 JIS2004 Support

This version extends Caché Unicode support to include characters known as surrogate pairs. These are pairs of 16-bit characters that, taken as a unit, represent more than a million Unicode characters outside the Basic Multilingual Plane (BMP). To allow applications to deal with the characters properly, this version of Caché also introduces specialized versions of the regular string functions that correctly handle surrogate pairs.

15.1.2.7 Subroutine-level Profiler

Caché now permits profiling at the level of individual subroutines, procedures, and functions. This feature allows the user to easily see the "busiest" routines, and then drill down to the subroutine line level to analyze and improve performance of any "hot spots." The facility extends the existing **^%SYS.MONLBL** profiler (and coverage analysis).

15.1.2.8 $DECIMAL

A number of improvements to numeric computation have been made in this version of Caché. These include:

- Improved support for $DOUBLE values.
- Better handling of conversions between $DOUBLE values and Caché numbers and strings.
- User control over conversion to Caché floating-point using the $DECIMAL function.

15.1.2.9 Updates To Third-party Software

This version of Caché contains updates to third-party software packages. The following are the version changes from 2008.1 to 2008.2 for all platforms except for OpenVMS:

Package	Version in 2008.1	Version in 2008.2
ICU	3.2	3.6
Xerces	2.6	2.7
Xalan	1.9	1.10+ (contains fixes not in the Apache release 1.10)

For OpenVMS, there are no version changes for these packages in 2008.2. The versions remain as: ICU 3.2, Xerces 2.1, and Xalan 1.5.

15.1.3 Reliability, Availability, Maintainability

15.1.3.1 CSP Enhancements

This version of Caché has several important, new features for CSP and the CSP Gateway:

- It is now possible to enable HTTP authentication programmatically from within Caché.
- This version supports the new native module interface to Microsoft Internet Information Services (IIS) version 7.
- The CSP Gateway now supports connection pooling when using Apache.
- CSP now supports communication using HTTP version 1.1
- The CSP Gateway now supports transmission of messages greater than 32 KB in length via "chunks."
- Updating the CSP password via the System Management Portal now automatically updates the password value in the CSP.ini file.
- In prior versions, requests longer than 32 KB were converted to streams. In this version, they are converted only if the length exceeds the maximum string length. That is, if long strings are enabled, the conversion to a stream will only happen if the request exceeds the maximum long string length.

15.1.3.2 Improved 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.Table, %SYS.NLS.Format, %SYS.NLS.Locale and %SYS.NLS.Device classes now contains the functionality of the previous NLS utilities. See the Windows installation documentation for an example. One can also use the line-oriented routine, ^**NLS**, from a terminal connection.

The Visual Basic-based NLS management application has been removed. The corresponding replacement page is now part of the Management Portal at **[Home] > [Configuration] > [NLS Settings]**.

15.1.3.3 Improved UNIX® Installation

The **cinstall** installer for UNIX® has been updated. It is now possible to embed a Caché UNIX® installation within a partner product and to provide the responses needed by the Caché sub-installation in a script file. This allows a seamless installation of Caché as part of a containing product.

15.1.3.4 Per Socket Keep Alive

This version of Caché refines the TCP keep-alive timeout for Windows and Linux down to 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.

15.1.3.5 Improved Protection Against Starting Multiple Copies of Caché

Under some circumstances, Caché database locking proved insufficient to prevent multiple instances of Caché from being started against the same databases. This version has added safeguards on UNIX® systems against that happening.

15.1.3.6 New Archive Interface Classes

This version of Caché now contains a package, %Archive, that permits connection to an external service for long-term document retention. The package contains classes that 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.

15.1.4 Security

15.1.4.1 Superserver Can Accept SSL Connections

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). Ports can be configured to require the use of SSL or TLS by requesting clients.

The clients that have been upgraded to use this communications capability are:

- Shadowing (configured and enabled via a command-line utility)
- JDBC and Java binding

See the notes in the Upgrade Checklist for further details.

15.1.4.2 Telnet Over SSL for Windows

Caché will now accept Telnet connections if the superserver is configured to use SSL and the %TELNET/SSL configuration has been defined. This is done via the Management Portal.

15.1.4.3 SQL Column-level Security

Caché now permits access control to be applied to SQL table columns. The privileges control the ability of a user to select, insert, and update column values. This allows one to limit access by designated users to only specified columns in a table. See the SQL verbs GRANT, REVOKE, INSERT, UPDATE, and other related verbs for an explanation.

15.1.4.4 WS-Security 1.1

Caché now gives applications the ability to use WS-Security 1.1 to protect web services messages. This capability permits the canonicalization of XML, and the generation and verification of digital signatures for message content according to the X.509 standard (X.509 addresses public key and privilege management infrastructure).

WS-Security is unrelated to HTTP basic access authentication which Caché already supports in this area. The WS-Security support is for UsernameToken Profile with clear text password, as described in the following document:

`oasis-200401-wss-username-token-profile-1.0.pdf`

Online at the Organization for the Advancement of Structured Information Standards (OASIS) site:

`http://docs.oasis-open.org/wss/2004/01/`

InterSystems recommends that you use SSL with this profile because the password is transmitted in clear text. Support for SSL is documented in the "Using SSL/TLS with Caché" chapter of the *Caché Security Administration Guide*. InterSystems also recommends using a SOAP/XML gateway for security purposes; for further details, see *Securing Caché Web Services*.

15.2 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 in the chapter General Upgrade Information in the *Caché Release Notes and Upgrade Checklist*. 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.

15.2.1 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 2008.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

15.2.1.1 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.

15.2.1.2 Prevent Login As UnknownUser

This version 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.

15.2.1.3 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.

15.2.1.4 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 not secure.
- 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 superserver?," change the setting to Accept or None (equivalent, respectively, to Enabled and Disabled in the Management Portal).

15.2.1.5 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.

15.2.1.6 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.

15.2.1.7 Security-Related Changes

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.

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 an unhandled ObjectScript 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()**.

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
```

15.2.1.8 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.

15.2.1.9 Platform-Specific Items

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

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.

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.

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.

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.

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.

15.2.2 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.

15.2.2.1 Compatibility Notices

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.

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.

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.

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.

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.

$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.

15.2.2.2 Character Set Changes

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.

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.

Default For Simplified Chinese Changed

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

15.2.2.3 Routine Changes

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

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

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.

15.2.2.4 ObjectScript Changes

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.

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.

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.

$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.

$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.

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.

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.

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.

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.

$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.

$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.

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.

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.

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.

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.

15.2.2.5 Class Changes

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.

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.

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.

In Caché 2011.1, this change was reversed and journaling of class compiles is on by default in Caché 2011.1 and later versions.

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.

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.

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.

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.

$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.

$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()**.

%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.

%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.

%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()

%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.

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.

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**.

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.

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.

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.

CDL Qualifiers Removed

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

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.

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.

%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.

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.

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.

%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.

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.

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..

%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.

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.

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.

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".

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.

AllowSessions Property Removed from EnsLib.SOAP.Service

In the 2008.1 Ensemble release, the AllowSessions property was removed from the EnsLib.SOAP.Service class settings. It is no longer configurable; instead, you must choose whether the service should use CSP/SOAP sessions at compile time using the *SOAPSESSION* class parameter of EnsLib.SOAP.Service. The default for the parameter is *SOAPSESSION* = 0.

If your subclass of EnsLib.SOAP.Service relied on the AllowSessions setting to control session behavior, you must rewrite it to use the *SOAPSESSION* class parameter. If you are using sessions you must override it to *SOAPSESSION* = 1. If you do not use sessions, you can rely on the default setting.

See the Enabling SOAP Sessions section of the "Creating an Ensemble Web Service" chapter of *Creating Web Services and Web Clients with Ensemble* for details.

15.2.2.6 MultiValue Changes

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.

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.

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.

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.

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.

$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.

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.

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.

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
```

#PRAGMA Added

By default, when an MVBasic routine is compiled, the system 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.

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.

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.

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.

$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.

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.

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.

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.

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.

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.

15.2.2.7 SQL Changes

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.

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.

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.

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];

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”.

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.

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
```

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.

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.

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.

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)
```

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.

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.

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.

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.

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.

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(...)".

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.

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.

15.2.2.8 CSP Changes

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*, and *CSPGatewayRequest*.

Added %request Properties

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

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.

CSP Gateway Changes

The CSP Gateway consists of two modules: A Management Module, CSPmsSys.dll, and a Runtime, CSPms.dll. The runtime 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 runtime 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.

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.
- The 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: This 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.

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.

15.2.2.9 XML Changes

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. Any class 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.

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.

Most methods in %XML.Writer now escape characters as specified by the XML Canonicalization standard. The following table describes how methods in this class escape specific characters, when found in attributes and in elements:

Original Value	How Value Is Escaped When Found in Attribute*	How Value is Escaped When Found in Element†
&	`&`	`&`
<	`<`	`<`
"	`"`	*no escaping*
$CHAR(9) (tab)	` `	*no escaping*
$CHAR(10) (line feed)	`
`	*no escaping*
$CHAR(13) (carriage return)	`
`	`
`
>	*no escaping*	`>`

*This column applies to the method **WriteAttribute()** and to the methods that use **WriteAttribute()**: **Document()**, **DocumentNode()**, **Tree()**, **Canonicalize()**, **CanonicalTree()**, **Element()**, and **RootElement()**.

†This column applies to the method **WriteChars()** and to the methods that use **WriteChars()**: **Document()**, **DocumentNode()**, **Tree()**, **Canonicalize()**, and **CanonicalTree()**.

Note that **Object()** and **RootObject()** do not use **WriteAttribute()** or **WriteChars()**.

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.

15.2.2.10 Terminal Changes

Terminal Local Network Encoding Changed

The 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.

Changes To Terminal Connection Display When Invoked From Scripts

Beginning with version 5.1, a Terminal session would begin by displaying the Node and Instance name that was the endpoint of the connection. It was done so a customer connecting to a machine via telnet 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.

15.2.2.11 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.

15.2.2.12 Language Binding Changes

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.

15.2.3 Operators

15.2.3.1 System Management Portal

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

15.2.3.2 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.

16

Caché 2008.1

This chapter provides the following information for Caché 2008.1:

- New and Enhanced Features for Caché 2008.1
- Caché 2008.1 Upgrade Checklist

16.1 New and Enhanced Features for Caché 2008.1

The following features have been added to Caché for the 2008.1 release:

- Caché Zen
- MultiValue
- Support For RIGHT and Full OUTER JOIN
- New Online Documentation Search

16.1.1 Caché Zen

The Caché Zen application framework makes the development of Web-based applications easier. It provides a simple way to rapidly create complex, data-rich Web applications by assembling prebuilt object components.

Zen is based on the established Caché Server Page (CSP) and Caché Object Database technologies that offer a robust, scalable, and portable platform for hosting Web applications. Zen builds upon the basic features provided by CSP: performance, data access, security, localization, and configuration.

Zen components automatically create standard HTML and JavaScript needed to render complex Web applications. They provide a common object model that is shared between the user's browser and the application logic running on the server.

16.1.2 MultiValue

This version of Caché extends support for MultiValue applications. Existing applications from more than a dozen MultiValue environments may be moved to Caché and run with little or no changes. Caché provides integrated support for MultiValue features such as accounts, terminal types and spooling. In addition, MultiValue applications have unimpeded access to all existing Caché facilities:

- web application development using Caché Server Pages and Zen

- language bindings to Java, Perl, Python, C++, .NET, and ActiveX applications
- web services and service-oriented architectures
- common data storage with Caché applications
- transaction processing
- enhanced security
- the ability to run in distributed, high-availability and/or replicated systems

MutltiValue data is fully available to Caché applications. In addition, Caché classes now support the use of MVBasic as an implementation language for methods along with Caché Objectscript and Caché Basic.

16.1.3 Support For RIGHT and Full OUTER JOIN

With Release 2008.1, Caché SQL now fully supports LEFT, RIGHT, and FULL outer join using the ON clause. Any standard operations and most Caché SQL extensions are allowed in the ON clause, including subqueries. Outer join operations may be nested, for example:

```
Table1 full join Table2 right join Table3 on Table3.x < Table2.y
     on Table1.x > Table3.x
```

Parentheses are supported wherever they are legal. For example, the previous join expression is equivalent to:

```
Table1 full join ( Table2 right join Table3 on Table3.x < Table2.y )
     on Table1.x > Table3.x
```

You may use views of any type anywhere in the join syntax in place of a table.

See the Conversion Checklist for remaining restrictions on JOINs.

16.1.4 New Online Documentation Search

The following changes have been made to the online documentation application, DocBook:

- Prior to this release, the online documentation created and used its own index for the documentation. In this release, the documentation now uses the %Library.Text class for indexing and searching documents. This allows automatic, incremental maintenance of DocBook word indices (so customers that load their own content do not have to rebuild indices).
- The changes implement a new Search page that uses the Caché text searching capabilities built into Caché SQL.
- The new search page is simple to use and better ranks the results of searches. For example:
  - The user interface is much simpler: you can enter a word or words; you can place double quotes around word(s) to force an exact match.
  - The application maintains a cache of recent searches and results (so you can remember what you already looked for).
  - Instead of finding each paragraph, the search finds “documents” (books, chapters, sections, and so on) and may show links to sections within such documents. This makes the results more concise and useful.
  - Search terms that appear in titles are ranked higher in value (and appear earlier in the list) than terms that appear in paragraphs or lists. Among titles, book titles take precedence over chapter titles; chapter titles rank higher than section titles; subsection titles follow sections.
- The search facility user interface is now implemented using Caché Zen classes.

16.2 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 in the chapter General Upgrade Information in the *Caché Release Notes and Upgrade Checklist*. 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.

16.2.1 Announcements

16.2.1.1 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.

16.2.2 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.

16.2.2.1 Version Interoperability

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

16.2.2.2 Management Portal Changes

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]**.

16.2.2.3 Operational Changes

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.

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.

When Upgrading An ECP Configuration SQL Privileges May Be Lost From the Clients

After converting SQL privileges on the database server, the *mpriv* global will be deleted. Beginning with this version, Caché no longer saves the privileges in the namespace, but in the (local) SYS database. Therefore, after the conversion, the application server has the privileges, but the application server does not. The privileges will have to manually be set on the client.

16.2.2.4 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.

16.2.2.5 Platform-specific Items

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

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.

Mac

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

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 multi-threaded CALLIN.

SUSE Linux Enterprise For Itanium

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

16.2.2.6 Limitations

The following known limitations exist in this release of Caché:

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.

DDP Limitations

The following known limitations exist when using DDP:

- You cannot use the **^%GD** utility over DDP — a `<DIRECTORY>` error occurs.
- 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.

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.

16.2.3 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.

16.2.3.1 ObjectScript Changes

$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.

$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.

16.2.3.2 SQL Changes

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).

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
```

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.

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.

16.2.3.3 TSQL Changes

%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.

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 SQL-NAME = 'dbo'.

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.

16.2.3.4 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 ObjectScript.

16.2.3.5 Studio Changes

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.

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."

16.2.3.6 Terminal Changes

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.

Support For Telnet Terminal Types

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")**.

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.

16.2.3.7 Class Changes

%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.

Remove Ability To Force IsValidDT Method Generation For System Datatypes

In a prior release, InterSystems removed the IsValidDT methods from system datatypes to reduce the amount of code generated and improve the speed of the class compiler. At the time, we added a flag that if set would still generate IsValidDT methods for our datatypes in case applications were calling the **IsValidDT** methods of our datatypes directly from their own code.

This switch has now been removed as it causes problems with the SQL projection of our datatypes.

%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.

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.

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 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'
```

%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.

16.2.4 Operators

16.2.4.1 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.

16.2.4.2 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.

17

Caché 2007.1

This chapter provides the following information for Caché 2007.1:

- New and Enhanced Features for Caché 2007.1
- Caché 2007.1 Upgrade Checklist

17.1 New and Enhanced Features for Caché 2007.1

The following features have been added to Caché for the 2007.1 release:

- Call In / Call Out Enhancements
- New Error Handling Syntax
- Long String Support
- Security Enhancements
- SQL Gateway via JDBC
- Objective-C Binding
- New Distribution Mechanism for C++ Binding
- Zen
- SQL Enhancements
- Enhanced T-SQL Support
- Routine Performance Enhancements
- Journal Enhancements
- Light C++ Binding
- CSP Gateway on OpenVMS
- Support for GB18030 Character Set
- Other Changes
- New Supported Platforms

17.1.1 Call In / Call Out Enhancements

The following call in / call out enhancements are included in this project:

- Applications can now call into Caché as a DLL / shared library, rather than as an executable, eliminating the need for static linking. This enhancement will be available for all platforms except OpenVMS.
- Call in is now multithreaded on some platforms. This enables multithreaded applications to call into Caché, with multiple threads effectively executing simultaneously but independently within a single Caché process. Initially, this capability will be available on the following platforms: Windows (32-bit, 64-bit Intel Extended Memory, 64-bit AMD, not Itanium), Linux (32-bit, 64-bit Intel Extended Memory, 64-bit AMD, not Itanium), and Solaris (64-bit SPARC and 64-bit AMD). Other platforms may be added in the future.
- For all other platforms, Caché is now thread safe. It can be called safely from multithreaded applications, with Caché automatically providing any necessary synchronization. While one application thread is executing within a Caché process, other threads are blocked from executing within Caché, although they may be executing other (non-Caché) application code.

17.1.2 New Error Handling Syntax

This project adds new Caché error handling syntax similar to that in Java, C++ and VB. It utilizes three commands: TRY, to identify the scope of an error handler; CATCH, to identify the code to execute on an error; and THROW, to explicitly signal an error.

Note: For those familiar with other language implementations, FINALLY is not supported.

17.1.3 Long String Support

With this project, the maximum length of local and global strings is extended to 3.6 million characters. Storage of long strings is only supported for databases using 8KB blocks.

Enabling support for long strings is done via the Management Portal at **[Home] > [Configuration] > [Advanced Settings]**. The category is "Miscellaneous" and the setting name is "EnableLongStrings".

This can also be set by the function, $ZUTIL(69, 69).

17.1.4 Security Enhancements

- *LDAP User Management* — This enhancement enables user authentication and roles to be managed outside of Caché using LDAP.
- *Delegated Authentication* — The enhancement enables user authentication to be carried out using site-specific Caché code.
- *Row-Level Security* — This enhancement adds extended SQL row-level security capabilities to Caché. Essentially, a column in the table contains a list of roles for each row. When a query is run, the user must have at least one of the row's roles in order to see that row. Typically, the list of roles is calculated (based on other data in the table) and a method is automatically called to perform this calculation whenever a row is inserted or updated. This security column can be indexed to provide row-based security with minimum performance impact.
- *Journal Auditing* — Changes to the state of journaling (both system-wide and process-specific) are now recorded in the audit log.

17.1.5 SQL Gateway via JDBC

In previous releases, the Caché SQL Gateway required an ODBC driver for each "foreign" database on each Caché platform. While this is not an issue for Windows, where good ODBC drivers are available for all databases supported by the Gateway, it has been an ongoing problem for other platforms, particularly UNIX®.

To address this, with this release we will begin using JDBC, rather than ODBC, for Gateway connections. Because all of the databases supported by the Gateway have platform-independent JDBC drivers, this will significantly reduce the testing and deployment of the Gateway and Gateway-based applications.

Note: SQL Gateway is supported on all Caché platforms except OpenVMS. For Caché 2007.1, we will support both the ODBC-based Gateway (on the same platforms as 5.2) and the JDBC-based Gateway. In later releases, for all platforms other than Windows, only the JDBC-based Gateway will be supported. On Windows systems, both the ODBC-based and JDBC-based Gateway will continue to be supported.

17.1.6 Objective-C Binding

This enhancement provides an object binding to the Objective-C language on Mac systems.

17.1.7 New Distribution Mechanism for C++ Binding

With this release we are changing the way that the C++ binding is packaged and delivered, in order to reduce our dependence on specific C++ compilers and libraries.

17.1.8 Zen

Zen is an extensible framework for rapid development of Web applications. Pages are defined via high-level XML-based definitions using a rich set of components. Zen includes a strong, easy-to-use security model; built-in support for multi-lingual applications; server- and client-side event handling; and very good extensibility.

17.1.9 SQL Enhancements

- *Aggregate Optimizations* — This release includes performance enhancements for SQL queries with multiple aggregates or with a combination of aggregates and GROUP BY.
- *SQL Temporary Tables* — This release adds support for SQL temporary tables.
- *Outer Joins* — Left outer joins can now be based on non-equality conditions, e.g. greater than or %startswith.
- *Text Search* — Text search enhancements include %CONTAINSTERM, use of indices for processing %SIMILARITY, and multi-term substitutions in the thesaurus facility.

17.1.10 Enhanced T-SQL Support

This release includes enhancements in T-SQL support, including compilation and runtime performance improvements, support for statement-level triggers, exposure of Sybase-compatible system tables, improved error handling, and some additional T-SQL syntax support.

17.1.11 Routine Performance Enhancements

COS routine management has been extensively revised in 2007.1. As a result, the way routines are managed by Caché is now much more efficient. The amount of overhead spent invoking a new routine may be up to an order of magnitude less

in this release. Furthermore, in prior releases, the number of routine buffers could become a source of contention in systems with dozens of cpus; this is nowfar less likely.

Among the changes is the addition of a new per-process cache identifying recently used routines. The cache speeds up the process of locating routines which are used frequently within the process. In addition, the cache changes the way routine buffers are managed by the system.

In prior releases, only the actual routine buffers being executed were protected against change. All others could be modified. A consequence of this was that prior routines on the call stack could be changed (for example, by recompilation). To illustrate the issue, suppose a routine, A, was executing and then called routine, B. When B started running, A was available to be changed. If A was re-compiled while B was executing, the buffer holding A became obsolete and was released. When B attempted to return to A, Caché would notice this fact and report an <EDITED> error.

With the per-process cache in place, the <EDITED> error no longer occurs. This is because all routines on the call stack of any process have their current versions locked. So the associated buffers do not become obsolete. B always returns to the proper version of its caller, A.

To carry the examination further, if B returns to A and, while A is executing, B is recompiled. The next call from A to B will execute the re-compiled version of B. Similarly, if A is re-compiled while B is executing and B then calls A, the new version of A will be invoked. When the new A returns to B and B subsequently returns, it will return to the prior version of A that was preserved in the routine buffer.

The initial size of the per-process cache (called the "routine vector") is set at system startup. The default is 256. This number can be changed via the function, **$ZU(96,29,<value>)**. The cache can be cleared of all routines not in use with **$ZU(96,31)**. Note that routines are cached by namespace; changing namespaces clears the cache. Changing the routine search path via **$zu(39,...)** also clears the cache.

Because the per-process cache causes system routine buffers to be held until they are no longer in use, it is possible for a large collection of processes to consume all available buffers. When this happens, Caché will force the reduction of each of the per-process caches by some amount, freeing unused buffers. This reduction may recur until sufficient buffers are available for sustained operation.

The command

```
cstat -s<MgrDirectory> -R2048
```

where <MgrDirectory> is the pathname of the Caché manager directory will display the routine buffer usage for running jobs in the “rbuf#” column. If the number of routines in use for most jobs exceeds 30, then the administrator should increase the shared heap size by 512 bytes times the number of expected simultaneously running jobs.

Note: By default each process may cache up to 256 routines. As the system starts running out of available buffers, it will adjust the number of cached routines allowed per process down by 10% every quarter-second. When the number of cached routines per process is 30% of the originally configured number, Caché will log a message to the console, indicating the system may no longer be running efficiently.

When the number reaches 20% of the original setting, Caché will log a warning; and at the 10% threshold will log a “severe” message to the console that indicates the system may hang.

The system will hang when it runs out of routine buffers and cannot free any for use.

CAUTION: The routines held in the cache by dead processes keep the buffers they hold in use until the dead process is removed or the system shuts down.

17.1.12 Journal Enhancements

With this release, a number of internal improvements have been made to the performance and reliability of synchronous journal writes. In addition, system management improvements have been made to enable journal files to be retained on shadow systems for a limited period of time.

17.1.13 Light C++ Binding

The Light C++ Binding is a high-performance object binding that operates “in process”, i.e. the client application executes in the same process as Caché. With this binding, objects are fetched directly from the database to the client, without the requirement to maintain an open object in memory within Caché. The Light C++ Binding is available for the same platforms as multithreaded call in.

17.1.14 CSP Gateway on OpenVMS

With this release, support is added for the CSP Gateway on OpenVMS systems (Alpha and Itanium) running the Apache Web server.

17.1.15 Support for GB18030 Character Set

This release adds support for GB18030, the official character set of the People’s Republic of China in the Simplified Chinese locale (chsw). Since the internal representation used by Caché is Unicode, only those code points in GB18030 that map to Unicode are supported. This includes all the 1, 2 and 4-byte sequences that map to characters in the Basic Multilingual Plane (characters U+0000 to U+FFFF; approximately 64K code points), plus all the 4-byte sequences that map to the additional Unicode planes (U+10000 to U+10FFFF; 1M code points).

This latter range is represented in Caché in UTF-16 as surrogate pairs. This is the same representation used by Microsoft since Windows 2000.

Note: There are approximately 500K code points in GB18030 that don't map to Unicode and are currently unassigned. These code points are not supported by Caché.

17.1.16 Other Changes

- *Increased file name length* — The maximum path length for databases and other files has been increased from 64 to 232 characters.
- *Licensing Improvements* — With this release, the value of $USERNAME can be used to identify users for licensing purposes. This enables more accurate counting in situations where the IP address cannot be used to reliably identify distinct users.
- *Double Literals* — Large numeric literals (>1E146) are now automatically stored as double values.
- *JOB Improvements* — A process started with the JOB command can now be terminated by its parent, regardless of security settings.
- *Visual Studio 2005* — This release makes use of Microsoft’s Visual Studio 2005 for Windows client components. This change affects customers who make programmatic access (e.g. using call in) to Caché executables.
- *MultiNet Support* — We will support the MultiNet implementation of TCP/IP on OpenVMS Alpha systems (but not on OpenVMS Itanium). We may discover and document limitations with this software.
- *Hibernate* — The released version of Hibernate 3.2.1 contains support for Cache2007.1. For further information, or to obtain this version, please visit the Hibernate organization website.

- *Studio Improvements* — A number of smaller changes have been made to the Studio IDE:
  - The toolbar has a View WebPage button that allows the developer to see a preview of the page
  - The toolbar now has Forward and Backward buttons
  - The options dialog has been revised
  - The StudioAssist feature now works with selected XML documents, notably Zen

17.1.17 New Supported Platforms

The following platforms are added:

- Sun Solaris AMD
- OpenVMS 8.3 for Alpha and Integrity
- SUSE Linux Enterprise Server 10

17.2 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 in the chapter General Upgrade Information in the *Caché Release Notes and Upgrade Checklist*. 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.

17.2.1 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 2007.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

17.2.1.1 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.

17.2.1.2 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.

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.

17.2.1.3 Changes in Default Port Choices

This version of Caché chooses the default ports for installation differently from previous 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.

17.2.1.4 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.

17.2.1.5 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.

17.2.1.6 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.

17.2.1.7 System API Changes

%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.

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.

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.

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)

17.2.1.8 Platform-specific Items

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

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.
  - *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.

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.

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.

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.

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.

Dreamweaver No Longer Supported

The release of Caché no longer supports any version of Dreamweaver as a method for generating CSP pages.

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.

17.2.2 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.

17.2.2.1 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.

17.2.2.2 Objectscript Changes

Changes To Pattern Matching

This release makes an incompatible release to the way patterns are matched. If an application makes use of patterns with certain characteristics, it must be recompiled in this version. Otherwise, it will produce incorrect results. Those characteristics are:

- the repetition count is 12, for example, “X?12N”
- the first number of a repetition range is 12, such as “X?12.14N”
- the difference between the two numbers of a repetition range is 12, such as “X?3.15N” or “X?.12N”

17.2.2.3 SQL Changes

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.

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.

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'
```

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 was a documented issue with the Oracle JDBC drivers.

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.

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

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)

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

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

New Comment Syntax

Caché SQL now supports multiline comments with the same /* ... */ syntax used by Caché Objectscript.

17.2.2.4 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.

17.2.2.5 Changes to Callin

In this release of Caché, CALLIN no longer supports msql* functions. These must be changed to Caché SQL.

17.2.2.6 Corrections

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.

17.2.3 Operators

17.2.3.1 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.

18

Caché 5.2

This chapter provides the following information for Caché 5.2:

18.1 New and Enhanced Features for Caché 5.2

The following features have been added to Caché for the 5.2 release:

18.1.1 Jalapeño

Using our traditional object bindings, each client class has, in addition to its developer-defined properties and methods, a set of Caché supplied methods related to object persistence, e.g. Save() and OpenId(). With Jalapeño this paradigm is changed slightly by separating the persistence behavior into a separate "controller" class that can be used with any Java object.

- *Java Object Schema Import*

  As part of Jalapeño, we are adding Java Object Schema Import. Traditionally, our approach to language bindings has been to start with a Caché class definition from which Caché generates one or more "projection" classes. Java object schema import turns this around: it enables Caché class definitions to be created from existing Java class definitions.

- *Database Neutral Deployment*

  Jalapeño provides an additional feature: the ability to deploy applications on any database supported via the Caché SQL Gateway. It does this by substituting SQL requests for the object mechanisms normally used to link the Jalapeño client with the database (Caché) server. Of course the performance of this approach will be inferior to that of Caché, but it provides a convenient way to construct a database-independent application without giving up the power and performance of Caché.

18.1.2 Caché Managed Provider for .NET

The Caché Managed Provider for .NET supplies high performance relational and object database capabilities through a single client — something that no one else delivers for .NET. This release also provides a Visual Studio .NET plug-in that automatically generates .NET assemblies from Caché class definitions.

18.1.3 IEEE 8-byte Floating Point Support

Support has been added for IEEE 8-byte floating point (a.k.a. "double") values, making Caché more attractive for calculation-intensive applications.

This enhancement includes:

- A new internal data type for both scalars and list members
- A new intrinsic function to cast a value to a double
- Assembler optimizations to enable Caché to take advantage of processor-specific floating point instructions
- Object and SQL client enhancements to handle new server data types

18.1.4 Direct FileMan Dictionary Converter

The Direct FileMan Dictionary Converter enables Caché table definitions to be automatically created from FileMan file definitions. (Previously, the conversion was a two-step process, from Fileman to F DBMS and then to Caché.)

A more detailed description is contained in the section on "Converting FileMan Files into Caché Classes" in *Caché Specialized System Tools and Utilities* .

18.1.5 Code Completion in Studio

Code completion simplifies editing in Studio by enabling developers to pick from a list of context-sensitive choices when entering code. For instance, Studio will assist in specifying a class name when "##CLASS(…)" syntax is used or in specifying a property or method name when the type of a variable is known.

18.1.6 Process-Private Globals

Support has been added for process-private globals. Like local variables, they are accessible only by the process that creates them and are automatically deleted when that process halts. However, like globals they are essentially unlimited in size.

18.1.7 Caché Journal File Encryption

Support has been added for encryption and decryption of Caché journal files.

18.1.8 Version Checking (and Optimistic Concurrency)

At the object level, this feature adds automatic support for version checking. If turned on (via a class parameter) it increments the value of a version number property whenever an object is saved. You can use this set of features to implement optimistic concurrency.

18.1.9 Dynamic Dispatch

Dynamic dispatch enables a Caché class to respond to references to properties and methods that are not part of the class definition at compile time.

18.1.10 Free Text Search

Support has been added for indexing and searching textual data, specifically:

- A new %Text data type.
- A new SQL selection operator, %CONTAINS.
- Language-specific parsers for English, Spanish, French, Italian, German, Japanese, and Portuguese.
- Support for multi-word ("n-gram") indices. For instance, with an n-gram length of 1, all individual words are indexed. With an n-gram length of 2, all word pairs that occur together are also indexed.
- Support for "stemming" to map multiple forms of a word (go, goes, going, went, …) to a common root.
- Noise word filtering, to eliminate common words (a, and, the, …) from an index.

18.1.11 ODBC Multiple Result Sets

Support has been added to Caché ODBC for stored procedures that return multiple result sets.

18.1.12 WMI Support

Support has been added for Windows Management Instrumentation (WMI), the Microsoft implementation of Web-Based Enterprise Management. This enables Caché to be monitored by a variety of Windows management tools, including Microsoft Management Console.

18.1.13 Enhanced Debugging Capabilities

Caché debugging capabilities have been enhanced in a number of areas:

- Ability to debug macro routines, class definitions, and CSP files
- New system-level mechanism to make the debugger connection more robust

- New stack trace mechanism
- Ability to BREAK into a READ
- Additional debugging commands / options.

18.1.14 T-SQL Support

A number of capabilities have been added to simplify migration from SYBASE or SQL Server to Caché, including support for the T-SQL language in stored procedures. T-SQL is currently supported on 32 bit Windows, 32 bit Linux (RedHat and Suse distributions), 64 bit AIX®, and 64 bit Solaris.

18.1.15 Device Level SSL and TLS support

Device level support has been added for secure sockets layer communication using SSL 2, SSL 3, or TLS.

18.1.16 Enhanced ECP Performance

Improvements have been made to ECP and the ECP/jrnsync mechanism to enhance performance and scalability.

18.1.17 Enhanced Windows Cluster Resource Management

This enhancement enables Caché to be managed more easily in a Windows cluster.

18.1.18 Improved RPM Linux Installation

A new RPM install package has been created for Caché on Linux.

18.2 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.

18.2.1 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.

18.2.2 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.

18.2.2.1 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.

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.

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.

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.

Editing Globals

The **[Home] > [Globals] > [Edit Global Data]** page now allows the administrator or operator to edit and delete global data values.

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.

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.

18.2.2.2 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.

18.2.2.3 Platform-Specific Items

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

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.

Windows Clusters

InterSystems recommendations for setting up Windows cluster configurations have evolved over the past several versions.

HPUX

The Secure Sockets Layer (SSL) and Transport-Level Security (TLS) features of Caché 5.2 are not available on this version of HPUX.

SUSE Linux

For 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.

18.2.3 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.

18.2.3.1 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.

18.2.3.2 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

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)
  ```

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.

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

18.2.4 Operators

18.2.4.1 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.

19

Caché 5.1

This chapter provides the following information for Caché 5.1:

19.1 New and Enhanced Features for Caché 5.1

Welcome and thank you for using Caché!

This chapter provides an overview of the new and improved features in Caché 5.1.

See the Major New Features section for a description of the important new features and enhancements included in this release.

If you are new to Caché, you can refer to the Getting Started page which contains a variety of links to documentation organized by topic.

19.1.1 Upgrading and Installation

If you are upgrading existing applications and databases from prior versions, please read the Caché 5.1 Upgrade Checklist and the Upgrading chapter of the *Caché Installation Guide*.

For information on installing Caché, refer to the following sources:

- the *Caché Installation Guide* for Windows, OpenVMS, and UNIX®
- the list of Supported Platforms for this release

19.1.2 Major New Features

Caché 5.1 introduces a significant number of new features as well as enhancements to existing features. These features are focused on:

- Advanced Security
- Maximizing Scalability
- Maximizing Development Speed

- Minimizing Support Load

The new features are summarized here. For additional details, refer to the cited chapters or guides.

Caché Advanced Security

A host of new capabilities have been added to give Caché the most advanced security of any mainstream database.

System Management Portal

A new integrated system management interface, built with CSP, replaces Control Panel, Explorer, and SQL Manager. This removes the requirement for a Windows PC in order to manage Caché and, because no Caché client software is required, eliminates potential client/server version mismatch issues and simplifies management of multiple versions of Caché from a single device.

System Improvements

Caché system improvements include many new or enhanced classes and methods, plus major enhancements such as nested rollback and the ability to map class packages to namespaces.

- Nested Rollback — When nested TSTARTs are used, this enhancement enables the innermost TSTART to be rolled back, without rolling back the entire open transaction.
- Namespace Mapping for Class Packages — Namespace mapping has been extended with the ability to map class packages by name, just as routines and globals are mapped.

ObjectScript Language Improvements

The ObjectScript language now provides significantly improved runtime error reporting. Many other enhancements have been introduced, including the following items:

- New **$FACTOR** Function
- New **$LISTNEXT**, **$LISTTOSTRING**, and **$LISTFROMSTRING** Functions
- New **$ROLES** and **$USERNAME** Special Variables
- New Error Trapping Syntax
- More Efficient Code Generation
- Pattern-Match “E” Adapted For Unicode
- Faster **MERGE** Command

New Language Bindings

With this release, Caché introduces new Perl and Python bindings, as well as an improved version of the Caché ActiveX binding.

Object Improvements

The Caché 5.1 Class Library provides many new features and major enhancements.

- Index on Computed Fields — An index definition can now reference properties defined as CALCULATED and SQLCOMPUTED.
- Object Synchronization — Caché can now track records of all object filing events (insert, update and delete) for journaled classes, export the journaled object data, and synchronize it with other databases. Applications with no access to the original database can then resolve references to the synchronized objects.

- Studio Enhancements — New %Studio.Extension classes provide mechanisms for custom menus and user defined data entry. %Studio.SourceControl classes now provide enhanced source control hooks, allowing customized checkout and checkin to a source control system.
- Performance Improvements — Significant improvements have been made to the in-memory performance of relationships.
- Syntax for defining stream and collection properties has been improved, and enhancements have been made to the behavior of streams and collections.

SQL Improvements

Caché SQL support includes many new or enhanced features, including the following items:

- New SQL/XML Support Functions
- JDBC 3.0 Support
- **SAVEPOINT**: New Transaction Processing Feature
- **CREATE TABLE**: New IDENTITY Keyword
- **DROP VIEW**: New CASCADE Keyword
- **INSERT**: New DEFAULT VALUES Clause
- New RowId Counter Validation Option
- New Query Optimizer Plan Verification
- Subquery Flattening
- Enhanced Locking Behavior for Foreign Key References
- READONLY Tables and Fields
- Support for %%CLASSNAMEQ and %%TABLENAME
- CREATE BITMAP INDEX Support for Oracle Import Compatibility

Connectivity Improvements

Caché 5.1 introduces many new options for network connectivity.

- ECP Enhancements — A number of enhancements have been made to the Caché Enterprise Cache Protocol. It is now supported in shared disk cluster configurations with OpenVMS and Tru64 UNIX®.
- SNMP Support — Support for the Simple Network Management Protocol (SNMP) has been added to enable monitoring of Caché by a variety of systems management tools and frameworks.
- LDAP Client — Programmatic access to LDAP servers has been added.
- Mac OS X Server Support — Support has been added for Mac OS X as a server plus the following client components: ODBC, JDBC, Objects, CSP Gateway for Apache.

19.1.3 Caché Advanced Security

With version 5.1, InterSystems introduces Caché Advanced Security. This release of Caché contains a host of new capabilities that provide the most advanced security of any mainstream database. Caché Advanced Security provides a simple, unified security architecture that offers the following advantages:

- It offers a strong, consistent, and high-performance security infrastructure for applications.

- It meets certification standards.
- It makes it easy for developers to build security features into applications.
- There is a minimal burden on performance and operations.
- It ensures that Caché can operate effectively as part of a secure environment and that other applications and Caché can work together well.

See the Caché Security Administration Guide for detailed information on Caché Advanced Security.

19.1.3.1 Key Features

Here are a few of the more important new security features offered in Cache 5.1:

- Kerberos based Security Infrastructure

  Two Authentication models are now available. In addition to Caché Authentication (Username/Password), Cache now provides Kerberos based Security Infrastructure. Kerberos libraries are available on all supported platforms (Windows Single Sign-on for Win32/64 platforms in an Active Directory Domain = Kerberos Realm, since Microsoft uses Kerberos at the heart of their Authentication model).

- Security Management Interface

  The Management Portal's web-based Security Management facility allows complete access to Users, Roles, Services, Resources (including Schemas), Auditing, and all other aspects of Caché security management.

- Security Advisor Utility

  The new Security Advisor utility makes recommendations for securing a Caché DBMS (Security settings, Applications and Auditing).

- Authentication in ODBC/JDBC

  ODBC and JDBC drivers now offer both Caché and Kerberos Authentication. Kerberos mode provides three levels of Encryption: Clear, Integrity (Source and Content Validation), and Encrypted (complete, end-to-end AES Encryption).

- Auditing Facilities

  Caché provides detailed auditing facilities that store audit information in a specially protected Audit Database. Auditing capabilities are available from an Automated/Management and Programmatic/API point of view.

- Encrypted Database Management Facility

  The new Encrypted Database facility allows you to create fully encrypted (AES, up to 256 bit) CACHE.DAT files that stay Encrypted on Disk at all times. I/O is encrypted and decrypted on the fly, with minimal performance impact. The database is encrypted with a Special Key file that is stored on removable devices (like USB Flash Drives) and must be present to mount the DB for use.

Security Advisor

To assist system managers in securing a Caché system, Caché includes a Security Advisor. This is a Web page that 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.

Low-level Security Interfaces

System administrators can exercise low-level control over the security of Caché systems through two character-oriented interfaces:

- **^SECURITY** allows examination and editing of security data related to users, roles, domains, services, applications, and auditing. An overview of **^SECURITY** can be found in The CHUI-Based Management Routines.
- **^DATABASE** provides low-level management capabilities related to Caché databases. An overview of **^DATABASE** can be found in The CHUI-Based Management Routines.

Common Criteria Security Certification

Security certification is becoming an increasingly frequent requirement for government purchases, and is more and more requested for private sector purchases. Because of this, InterSystems has had Caché certified according to the Common Criteria standard. Specifically, effective February 15, 2007, Caché received certification according to the Common Criteria standard (EAL 3).

The Common Criteria provides a set of common security standards for a wide number of nations in North America, Europe, and the Far East. It provides an assurance scale from 1 to 4, where a product's rating indicates the rigor of testing to which it has been subjected; commercially available products are rated from 1 (least rigorous testing) to 4 (most rigorous). Caché is currently under consideration for a level-3 rating. Such a rating indicates that Caché can effectively serve as part of a highly secure operational environment.

19.1.3.2 Caché Advanced Security Concepts

Caché Advanced Security is based on authentication, authorization, and auditing:

- Authentication ensures the verification of the identity of all users.
- Authorization ensures that users can access the resources that they need, and no others.
- Auditing keeps a log of predefined system and application-specific events, to provide forensic information about the database activities.

Authentication: Establishing Identity

Authentication is how you prove to Caché that you are who you say you are. Without trustworthy authentication, authorization mechanisms are moot — one user can impersonate another and then take advantage of the fraudulently obtained privileges.

The authentication mechanisms available depend on how you are accessing Caché. Caché has a number of available authentication mechanisms:

- Kerberos — The most secure means of authentication. The Kerberos Authentication System, developed at MIT, provides mathematically proven strong authentication over an unsecured network.
- Operating-system–based — Available for UNIX® and OpenVMS, OS-based authentication uses the operating system's user identity to identify the user for Caché purposes.
- Caché login — With Caché login, Caché maintains a table of hashed password values for each user account; at login, Caché confirms user identity by comparing the value in the table with a hash of the password provided by the user.

Authorization: Controlling User Access

Once a user is authenticated, the next security-related question to answer is what that person is allowed to use, view, or alter. Authorization manages the relationships of users and *assets* such as databases, Caché services like ODBC access, and user-created applications.

In the most basic authorization model, there are all possible assets, a list of users, and all the relationships between the first group and the second.

Auditing: Knowing What Happened

Auditing provides a verifiable and trustworthy trail of actions related to the system. Auditing serves multiple security functions:

- It provides proof — the proverbial "paper trail" — recording of the actions of the authentication and authorization systems in Caché and its applications.
- It provides the basis for reconstructing the sequence of events after any security-related incident.
- Knowledge of its existence can serve as a deterrent for attackers (since they know they will reveal information about themselves during their attack).

The auditing facility automatically logs certain system events; it also allows you to enable logging for other system events, as well as site-defined application events. All audited events are placed in a tamper-resistant log file. Authorized users can then create reports based on this audit log, using tools that are part of Caché. Because the audit log can contain sensitive information (such as regarding positive or negative values for medical tests), running an audit report itself generates an entry for the audit log. The included Caché tools support report creation, archiving the audit log, and other tasks.

19.1.4 System Management Portal

Caché 5.1 now uses a browser-based interface, the System Management Portal, for system management. This new interface subsumes the functions previously distributed among Explorer, SQL Manager, Configuration Manager, and Control Panel functions of the Windows Caché Cube. In 5.1, these have been removed from the Cube.

An advantage of this approach is that it is no longer a requirement that Caché 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 much easier. No Caché client software is required, simplifying management of multiple versions of Caché from a single device. Cross-release compatibility issues are minimized because both the data and its formatting information come directly from the system being managed.

See Using the System Management Portal for a detailed description of the new interface.

19.1.5 System Improvements

New Caché 5.1 system features and enhancements:

New Features:

- Nested Rollbacks
- Namespace Mapping for Class Packages
- New Method $SYSTEM.Util.CleanDeadJobs()
- New Class $SYSTEM.Monitor.Line
- New Method $System.Device.GetNullDevice()
- New Optional Argument for $ZF(-2)

Enhanced Features:

- Option to Filter Records before Dejournaling on a Shadow
- Callin Enhancements
- 64K Routine Buffer Support
- CVENDIAN Enhancements

19.1.5.1 Nested Rollbacks

This version of Caché introduces multiple transaction levels , which make it possible to roll back part of a transaction without losing all work completed to that point. When nested **TSTARTs** are used, this enhancement enables the innermost **TSTART** to be rolled back, without rolling back the entire open transaction. When two **TSTARTs** are issued without an intervening **COMMIT** or **TROLLBACK**, the transaction level ($TLEVEL) is incremented by 1 (limited to a maximum of 255). When a **TCOMMIT** or **TROLLBACK 1** is issued, the transaction level is decremented by 1. When an unqualified **TROLLBACK** is issued, the transaction level is decremented to 0, and the entire transaction is rolled back.

Transaction commands now work as follows:

- The argumentless **TROLLBACK** command works as usual, rolling back to the very top level transaction and closing the transaction.
- The **TROLLBACK 1** command rolls the current open transaction back one level. All the globals changed within this transaction will be restored, and $TLEVEL is decremented by 1. If there is no open transaction ($TLEVEL is zero) then no action is taken. **TROLLBACK 1** won't roll back globals mapped to a remote system that doesn't support nested transactions unless $TLEVEL is 1.
- The **TCOMMIT** command works as usual. In nested transactions, it decrements $TLEVEL and writes a 'PTL' (pending commit with transaction level) journal record to the journal file.
- The **TSTART** command also works as usual. In nested transactions, it increments $TLEVEL and writes a 'BT'(begin transaction) record in the journal file. If the new $TLEVEL is greater than 1, it writes a 'BTL'(Begin Transaction with level) instead of 'BT'.

Caché SQL now includes standard SQL commands that take advantage of nested rollbacks (see New SAVEPOINT Features).

19.1.5.2 Namespace Mapping for Class Packages

Namespace mapping has been extended with the ability to map class packages from a database to one or more namespaces, just as routines and globals are mapped. Automatic namespace mapping is provided for system classes. All the schemas that begin with '%' from %sys are mapped to all namespaces automatically. These mappings allow the user to access SQL Table, View, Procedures, and classes across multiple namespaces. For example, assume a class %Test that has the following query:

```
Select field1 From %Test
```

Without mapping, attempting to inherit from this class in a user namespace would result in error: "Table %Test not found". With mapping, the class will compile successfully in any namespace.

For detailed information, see Configuring Data in the *Caché System Administration Guide*.

19.1.5.3 New Method $SYSTEM.Util.CleanDeadJobs()

New class method **$SYSTEM.Util.CleanDeadJobs()** is used to roll back a dead job's open transaction (if any) and clean up the dead job's Process Table (pidtab) slot so it can be re-used.

19.1.5.4 New Class $SYSTEM.Monitor.Line

New class $SYSTEM.Monitor.Line is a programmer API for the line-by-line monitor (**^%MONLBL**). It allows integration with Studio, and is also generally useful as a programmable alternative to **^%MONLBL**. For details, see the Programming Interface section in the MONLBL chapter of the *Caché Monitoring Guide*.

19.1.5.5 New Method $System.Device.GetNullDevice()

New class method **$System.Device.GetNullDevice()** returns the name of the null device appropriate for the current operating system type (/dev/null for UNIX®, _NLA0 for OpenVMS, //./nul for Windows). It facilitates development of applications that reference the Null device, and provides an OS-independent method for obtaining the name of the Null Device.

19.1.5.6 New Optional Argument for $ZF(-2)

Function **$ZF(-2)** now has an optional fifth argument that specifies whether or not the spawned process ID should be stored in $ZCHILD. For example:

```
s rc=$zf(-2,"program","","",1)
s childpid=$ZCHILD
```

If the new argument is zero or not specified then $ZCHILD is unchanged, otherwise $ZCHILD is set to the spawned process ID when it is successfully spawned.

19.1.5.7 Option to Filter Records before Dejournaling on a Shadow

To filter journal records before they get dejournaled on a shadow, set the global node ^SYS("shdwcli",shdw_id,"filter") to the name of the filter routine (without the leading "^"). The input parameters of the filter routine are:

- `pid`: process ID of the record
- `dir`: SOURCE (not SHADOW) database directory
- `glo`: global reference in the form of global(subscripts) (without leading "^")
- `addr`: offset of the record in the journal file
- `type`: type of the record: "S" = SET, "s" = BITSET, "K" = KILL, "k" = ZKILL
- `time`: timestamp of the record

In compatible mode shadowing, the pid and timestamp parameters passed to the filter routine always have the value "". The filter routine should return 0 if the record should be skipped; otherwise the record will be dejournaled by the shadow. For example:

```
^SYS("shdwcli","MyShadow","filter")="MyShadowFilter"
MyShadowFilter(pid, dir, glo, type, addr, time) ;
;skip X* globals
If $EXTRACT($qs(glo,0))="X" q 0
Set Msg = pid
Set Msg = Msg _ "," _ dir
Set Msg = Msg _ "," _ glo
Set Msg = Msg _ "," _ type
Set Msg = Msg _ "," _ addr
Set Msg = Msg _ "," _ time
Do ##class(%Library.Device).Broadcast("",Msg)

q 1
```

19.1.5.8 Callin Enhancements

The Callin include files ccallin.h and mcallin.h have been enhanced to merge common functionality and provide greater flexibility for building user-defined C and C++ Callin modules. Defines have been added to make building user Callin modules as independent of interface details as possible. Two features control the selection of interfaces:

#define ZF_DLL

If ZF_DLL is not defined, the Callin module is built for linking with the Caché engine. If it is defined, the module is built as a dynamic shared library using Callback and invoked through the Callout facility. This is the same define employed by cdzf.h.

#define CACHE_UNICODE

If CACHE_UNICODE is not defined, string handling functions and arguments are treated as 8-bit characters. If defined, strings are treated as 16-bit Unicode. String handling functions are available with the "A" suffix, meaning 8-bit (or ASCII), the "W" suffix, meaning 16-bit Unicode (or wide), and no suffix. In the last case the function resolves to either the "A" or "W" suffix according to the definition of CACHE_UNICODE.

New functionality has been implemented to permit NLS translation using the **CacheCvtInW()** and **CacheCvtOutW()** functions for Unicode Callin to 8-bit Caché. They will now convert data within the 8-bit character set range of the Caché engine, instead of reporting an "unimplemented" error. **CacheCvtInA()** and **CacheCvtOutA()** functions for 8-bit Callin to Unicode Caché are not currently implemented.

You can further refine 8-bit argument prototypes with the new macro **USE_CALLIN_CHAR**, which declares them as (char *) rather than (unsigned char *).

19.1.5.9 64K Routine Buffer Support

It is now possible to run with routine sizes up to 64K, by changing the Memory/RoutineBufSize value on the **[Home] > [Configuration] > [Advanced Settings]** page of the Management Portal from 32 to 64. The default and minimum value is still 32 (32K), but now values can be specified from 33..64 (rounded to the nearest 2K increment). Routines or class descriptors greater than 32K will be stored as two global values, the first chunk in ^rOBJ(<routine name>) as currently, and the second chunk in ^rOBJ(<routine name>,0).

19.1.5.10 CVENDIAN Enhancements

The cvendian database endian conversion utility has been enhanced to allow for positive identification of the desired endian orientation, or to optionally just inform the current endian orientation with no conversion. The command syntax is:

```
cvendian [-option] file1 [file2 ... file8]
```

where *option* is one of the following:

- -big — convert the database to big-endian
- -little — convert the database to little-endian
- -report — report the endian orientation of the database

The options may be shortened to their initial letter. If this is a conversion request, and the database already is of the specified endian orientation, a warning message is displayed and no further processing is done. Prior **cvendian** call formats remain supported.

19.1.6 Object Improvements

New Caché 5.1 object features and enhancements:

Object Enhancements

- New Option to Index on Computed Fields
- New Object Synchronization
- New Studio Extension Classes and Source Control Hooks
- New Stream Syntax
- New %SwizzleObject Class
- Extended POPSPEC Syntax
- Performance Improvements for Relationships

- Enhanced VisM OCX

19.1.6.1 New Option to Index on Computed Fields

An index definition can now reference properties defined as CALCULATED and SQLCOMPUTED. The property value calculation must be deterministic, always returning the same value for a given set of parameters. For example, it would be a mistake to use a function such as $Horolog, which returns different values depending on when it is called. Indexing on a property whose computation is nondeterministic will result in an index that is not properly maintained.

To support this option, properties defined as SQLCOMPUTED are now computed in Caché Objects. A new class method, Compute, is called by the property's Get method. The Compute method generates a return value by scanning SQLCOMPUTECODE for field references and converting those references to property or literal values. If the property also has SQLCOMPUTEONCHANGE, the Compute method is called whenever the property is changed.

19.1.6.2 New Object Synchronization

This new feature enables Caché to synchronize objects between databases. All object filing events (insert, update and delete) for journaled classes are automatically tracked. Object synchronization utilities provide methods to export the journaled object data and synchronize it with other databases. Applications with no access to the original database can then resolve references to the synchronized objects.

A new class, %SYNC.SyncSetObject, supplies methods to externalize an object and apply it to the target database. All references to persistent objects from the object being externalized are converted to GUID (Globally Unique Identifier) values. The GUID values are used to look up the corresponding object on import.

Another class, %SYNC.SyncSet, implements methods to manage the set of objects being synchronized. A 'synchronization set' is a set of externalized object values which guarantee that all object references can be resolved, either because the referenced object is in the same sync set, or because it already exists in the target database.

19.1.6.3 New Studio Extension Classes and Source Control Hooks

This release enhances the flexibility of Studio by introducing the %Studio.Extension classes, which provide mechanisms for custom menus and user defined data entry. The %Studio.SourceControl classes now provide enhanced source control hooks, allowing customized checkout and checkin to a source control system.

When the user performs an action in Studio that may require user interaction with the server (for example, attempting to edit a document that is in source control but is not checked out), Studio now calls the **UserAction** method.

```
UserAction (Type, Name, InternalName, SelectedText, .Action, .Target, .Msg)
```

`Type` options are:

- Server defined menu item selected
- Other Studio action

`Name` is the menu item name if `Type` is a menu item, otherwise `Name` indicates one of the following options:

- User has tried to change a document that is locked in source control
- User has created a new document
- User has deleted a document

`InternalName` is the name of the document this action is concerned with.

`SelectedText` contains any selected text in the document that has focus.

`Action` returns an action that Studio should perform:

- Do nothing (this method can still perform some action such as check an item out of source control, but Studio will not ask for user input).
- Display the default Studio dialog with a yes/no/cancel button. The text for this dialog is provided in the `Target` return argument.
- Run a CSP Template. `Target` is the start page name for the template. The template will be passed the current document name, any selected text, the project name, and the namespace.
- Run an EXE on the client. `Target` is the name of an executable file on the client machine.
- Insert the text in `Target` in the current document at the current selection point
- Studio will open the documents listed in `Target`

You can define custom menus for Studio to display. Studio obtains the menus when it first connects to a namespace by running two queries, **MainMenus** and **MenuItems**. **MainMenus** returns the list of top level menu names. After this top level menu is selected, **MenuItems** is used to return the list of items on a specific menu. **MainMenus** can be either a regular menu or a context submenu that is added to all the context menus. The **MenuItems** query is passed the current document name and any selected text in case you wish to vary the menu based on these arguments.

By default, the source control class inherits these queries from %Studio.Extension.Base, where they are defined as SQL queries against prebuilt tables. To load data into these tables, define an XData block called `Menu` in your source control class. When the source control class is compiled, this data is loaded and used automatically. Queries defined in the source control subclass can be changed or completely customized. When data is being returned from the **MenuItems** query, each menu name will generate a call to an **OnMenuItem** method in the source control class, where you may disable/enable this menu item. This allows simple modification of the menus without having to write a custom query.

19.1.6.4 New Stream Syntax

The class hierarchy for current stream classes has been changed so that %Stream.Object is the top class. This change does not alter stream runtime behavior.

In prior versions of Caché, it was necessary to define a stream property as `type = %Stream`, with a collection value of `binarystream` or `characterstream`. Now a stream property is defined by specifying the actual stream class as the type, and the collection keyword values of `binarystream` and `characterstream` are no longer used. A stream class is declared with a `classtype = stream`. This declaration is automatic for any class that extends a new class, %Stream.Object. For backward compatibility, the classes %Library.GlobalCharacterStream, %Library.GlobalBinaryStream, %Library.FileCharacterStream, and %Library.FileBinaryStream have been converted to use the new representation, and are to be used for all existing stream data.

For more detailed information, see the Streams chapter in *Using Caché Objects*.

19.1.6.5 New %SwizzleObject Class

A new class, %SwizzleObject, is now the primary (and only) superclass of both %Persistent and %SerialObject. The purpose of the new class is to define the swizzling interface and implement the parts of that interface that are common to both %Persistent and %SerialObject.

See the %Library.SwizzleObject class documentation for more detailed information.

19.1.6.6 Extended POPSPEC Syntax

The syntax of `POPSPEC` has been extended to allow an SQL table name and an SQL column name to be specified. When they are specified, the **Populate()** method constructs a dynamic query to return the distinct column values from the table. The requested number of values will then be randomly selected from the distinct column values and placed in a value set. The property will then be assigned values randomly from the resulting value set.

See The Caché Data Population Utility for more detailed information.

19.1.6.7 Performance Improvements for Relationships

The in-memory performance of relationships has been significantly improved by using additional in-memory indices to keep track of oref's and oid of items already in the relationship. Previously, when a new item was inserted into the relationship (either using the **Insert** method, or indirectly via the **Relate** method) it would scan the entire relationship to avoid inserting a duplicate item. By keeping an index of the oref's and oid's in the relationship, the cost of checking for duplication items is kept very low even for large numbers of items.

Partition memory used is lower, speed is significantly faster (94x in the second insert of 1000 items) and **%Save** time is faster. When measured with a small number of items in the relationship, there was no measurable slowdown in performance associated with the upkeep of the additional in-memory indices.

19.1.6.8 Enhanced VisM OCX

This release contains a new version of the Caché Direct control (VISM.OCX) that features enhancements such as security upgrades, support for multithreading, and improved error handling.

19.1.7 Language Improvements

New Caché 5.1 ObjectScript features and enhancements:

- Improved Runtime Error Reporting
- New **$FACTOR** Function
- New **$LISTNEXT**, **$LISTTOSTRING**, and **$LISTFROMSTRING** Functions
- New **$ROLES** and **$USERNAME** Special Variables
- New **$ZUTIL(62,1)** Function
- New **$ZUTIL(69)** Configuration Functions
- New **$ZUTIL(158)** Function
- New **$ZUTIL(186)** Function
- New **$ZUTIL(193)** Function
- New Error Trapping Syntax
- More Efficient Code Generation
- Pattern-Match “E” Adapted For Unicode
- Faster **MERGE** Command

New Language Bindings:

- New Perl Binding
- New Python Binding
- New ActiveX Bindings

19.1.7.1 Improved Runtime Error Reporting

Many runtime errors now report additional information. For instance, an "<UNDEFINED>" error will now report the name of the undefined variable.

Error information is stored in the system variable **$ZERROR**, which now returns more information than before. For example, when a routine attempts to use a variable that has not been defined, **$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 (adding " *someinfo"):

```
<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 further information, see the Cache Conversion Guide, and the $ZERROR special variable in the *Caché ObjectScript Reference*.

19.1.7.2 New $FACTOR Function

$FACTOR is a new ObjectScript function for 5.1 that converts a numeric value to a bitstring. Its primary use is for the creation of bitslice indices. For further information, see the $FACTOR function in the *Caché ObjectScript Reference*.

19.1.7.3 New $LISTNEXT, $LISTTOSTRING, and $LISTFROMSTRING Functions

Caché 5.1 adds three new functions for processing list structures: **$ListNext**, **$ListToString** and **$ListFromString**.

$ListNext(`list`, `ptr`, `val`) allows extremely rapid traversing of a list structure (up to 400x faster than doing a loop with **$LIST**).

Before the first call to **$ListNext**, `ptr` should be initialized to 0. After each call, `ptr` will contain the position of the next element in `list` (0 if the end of the list was reached), and `val` will contain the value of the element at that position (undefined if there was no value at that position). **$ListNext** will return 1 if it found another list element, or 0 if it is at the end of the list.

$ListToString(`list`[,`delim`]) takes `list`, and returns the elements as a string separated by `delim` (default ",").

$ListFromString(`string`[,`delim`]) takes `string`, delimited by `delim` (default ","), and returns the pieces as a list.

For further information, see the $LISTNEXT, $LISTTOSTRING, or $LISTFROMSTRING function in the *Caché ObjectScript Reference*.

19.1.7.4 New $ROLES and $USERNAME Special Variables

At Caché 5.1, the **$ROLES** special variable lists the security roles currently assigned to the user. The **$USERNAME** special variable list the user name for the current process. For further information, see the $ROLES, $USERNAME special variables in the *Caché ObjectScript Reference*.

19.1.7.5 New $ZUTIL(62,1) Function

The **$ZUTIL(62,1)** function performs syntax checking on a line of ObjectScript code. It returns the character position of the error and the text of an error message. For further information, see the $ZUTIL(62,1) function in the *Caché ObjectScript Reference*.

19.1.7.6 New $ZUTIL(69) System Configuration Functions

Caché 5.1 documents the following additional system-wide configuration functions: **$ZUTIL(69,19)**, **$ZUTIL(69,21)**, **$ZUTIL(69,31)**, **$ZUTIL(69,35)**, **$ZUTIL(69,37)**, **$ZUTIL(69,44)**, **$ZUTIL(69,49)**, and **$ZUTIL(69,60)**.

Caché 5.1 also supports the new **$ZUTIL(69,63)** and **$ZUTIL(68,63)** functions that control whether a lowercase “e” should be interpreted as an exponent symbol.

For further information, see the $ZUTIL(69) functions in the *Caché ObjectScript Reference*.

19.1.7.7 New $ZUTIL(158) Function

The **$ZUTIL(158)** function can be used to return the number of installed printers and the pathname of a specified printer. For further information, see the $ZUTIL(158) function in the *Caché ObjectScript Reference*.

19.1.7.8 New $ZUTIL(186) Function

The **$ZUTIL(186)** function can be used to specify the information displayed as part of the Terminal prompt. For further information, see the $ZUTIL(186) function in the *Caché ObjectScript Reference*.

19.1.7.9 New $ZUTIL(193) Function

The **$ZUTIL(193)** function inter-converts Coordinated Universal Time and local time values. For further information, see the $ZUTIL(193) function in the *Caché ObjectScript Reference*.

19.1.7.10 New Error Trapping Syntax

This version of Caché implements a special syntax that allows an error trap to pass control up the program stack to a previously established error trap. The syntax is **ZTRAP $ZERROR**. This command will pop entries off the program stack until a level is found with an error trap. Then that error trap will be executed with **$ZERROR** and **$ECODE** unchanged.

This command replaces the two commands `ZQUIT 1 GOTO @$ZTRAP`, which did not work in new-style procedures. This new command syntax can be used in both procedures and old-style subroutines. The old style of passing control up to a previous error trap will continue to work in old-style subroutines. If a **ZQUIT** command is issued in a procedure, it will now result in a `<COMMAND>` error.

The **ZQUIT** command is obsolete as of 5.1, and should not be used for new programming.

19.1.7.11 More Efficient Code Generation

The CacheBasic compiler now uses an improved algorithm that generates significantly smaller and faster code.

19.1.7.12 Pattern-Match "E" Adapted For Unicode

In prior version of Caché, the options used with the pattern-match operator(s) assumed 8–bit characters. This caused the "E" pattern (match every character) to fail when Unicode characters above **$CHAR(255)** were present in the string.

In Caché 5.1, the "E" pattern matches all characters.

19.1.7.13 Faster MERGE Command

The **MERGE** command is now much faster and more efficient when merging two local variables.

19.1.7.14 New Perl and Python Bindings

The Caché Perl and Python bindings provide a simple, direct way to manipulate Caché objects from within Perl or Python applications. They allow binding applications to establish a connection to a database on Caché, create and open objects in the database, manipulate object properties, save objects, run methods on objects, and run queries. All Caché datatypes are supported.

See Using Perl with Caché and Using Python with Caché for more detailed information.

19.1.7.15 Improved ActiveX Bindings

Caché 5.1 includes a new version of the Caché ActiveX binding, CacheActiveX.dll. Internally this new version uses the Caché C++ binding to get object-level access to a Caché server. Using this new binding provides the following benefits:

- access to the client/server security model available within Caché 5.1 (for example, the ability to use Kerberos authentication)
- better performance in some cases due to more sophisticated object caching.

While every attempt has been made to make this new DLL functionally compatible with the older CacheObject.dll it is not 100% binary compatible.

To preserve complete compatibility with existing applications, Caché installs two ActiveX bindings; the newer CacheActiveX.dll as well as the original CacheObject.dll. By default, existing applications will continue to use the original CacheObject.dll. If you wish to use the newer binding you have to modify your existing application to reference this new DLL and test that your application performs as expected.

19.1.8 SQL Improvements

New Caché 5.1 SQL features and enhancements:

New Features

- New SQL/XML Support Functions
- **SAVEPOINT**: New Transaction Processing Feature
- **CREATE TABLE**: New IDENTITY Keyword
- **DROP VIEW**: New CASCADE Keyword
- **INSERT**: New DEFAULT VALUES Clause
- New RowId Counter Validation Option
- New Query Optimizer Plan Verification

SQL Enhancements

- JDBC 3.0 Support
- **GRANT** and **REVOKE** Command Changes
- **CREATE USER** Command Changes
- Subquery Flattening
- Enhanced Locking Behavior for Foreign Key References
- READONLY Tables and Fields
- SQLCODE Changes
- Support for %%CLASSNAMEQ and %%TABLENAME
- CREATE BITMAP INDEX Support for Oracle Import Compatibility
- Extended Support for Milliseconds
- Date and Time Function Enhancements

19.1.8.1 New SQL/XML Support Functions

5.1 implements a collection of new built-in SQL functions for transforming “flat” relational queries into hierarchical XML documents. Application programs that need to generate HTML, or that need to export data in XML format, now have a general and portable interface that has wide industry support (ANSI/ISO SQL-2003 standard).

The following SQL/XML functions are available:

- **XmlElement** – Creates an XML element of the form: <tagName>body</tagName>, with optional attributes. **XmlElement** creates one tagged element that can contain multiple concatenated values.
- **XmlAttributes** – Specifies attributes for an XML element. **XmlAttributes** can only be used within an **XmlElement** function.
- **XmlConcat** – Concatenates two or more XML elements.
- **XmlAgg** – Aggregate function that concatenates the data values from a column.
- **XmlForest** – Creates a separate XML element for each item specified. **XmlForest** provides a convenient shorthand for specifying multiple elements nested within another element, where element instances that are NULL are omitted.

For more detailed information see XMLELEMENT, XMLAGG, XMLCONCAT and XMLFOREST in the *Caché SQL Reference*.

19.1.8.2 New SAVEPOINT Features

With version 5.1, Caché introduces multiple transaction levels (see Nested Rollbacks), which make it possible to roll back part of a transaction without losing all work completed to that point. Caché SQL now offers the following standard SQL commands that take advantage of this ability:

- SAVEPOINT <savepointName> — establishes a savepoint within a transaction.
- ROLLBACK TO SAVEPOINT — rolls back to the most recent savepoint.
- ROLLBACK TO SAVEPOINT <savepointName> — rolls back to the specified savepoint.
- COMMIT – commits only the current sub-transaction when $TLEVEL > 1.

For more detailed information see SAVEPOINT in the *Caché SQL Reference*.

19.1.8.3 CREATE TABLE: New IDENTITY Keyword

Caché SQL now supports the ability to define a column with a system-generated numeric value in a **CREATE TABLE** statement. An IDENTITY column is an exact non-negative integer column whose values are system-generated, and may not be assigned by the user in either **INSERT** or **UPDATE** statements. It may, however, be viewed using **SELECT ***. The syntax is:

```
CREATE TABLE <tablename> (
   [ other-table-elements , ]
   <columnname> [ <datatype> ] IDENTITY
   [ UNIQUE | NULL | NOT NULL |
      DEFAULT [(]<default-spec>[)] |
      [COLLATE] <sqlcollation> |
      %DESCRIPTION <literal>
   ]
[ , other-table-elements ]
)
```

An IDENTITY column is always data type INTEGER with unique non-null values. You can specify a datatype and constraints, but these are ignored by Caché.

This syntax is consistent with Microsoft SQL Server and Sybase syntax.

For more detailed information, see CREATE TABLE in the *Caché SQL Reference*.

19.1.8.4 DROP VIEW: New CASCADE Keyword

Caché SQL now supports the ability to cascade the deletion of a view to also delete any view that references that view. The new keywords are CASCADE and RESTRICT. The RESTRICT keyword is the default and is the same as prior **DROP VIEW** behavior.

For more detailed information, see DROP VIEW in the *Caché SQL Reference*.

19.1.8.5 INSERT: New DEFAULT VALUES Clause

Caché SQL now supports the ability to use default field values when inserting a row into a table. The syntax is:

```
INSERT INTO <tablename> DEFAULT VALUES
```

The statement will insert a single row into the table. Each field that has a default value will have the value assigned to the column. Fields without default values will be NULL for the row.

For more detailed information, see INSERT in the *Caché SQL Reference*.

19.1.8.6 New RowId Counter Validation Option

A new configuration option now makes it possible to validate new system-assigned ID values. The option is activated by setting `^%SYS("dbms","validate system-assigned id")` to 1. Although such validation is not normally necessary, it is possible that the ID could be invalid if the user has modified the value manually, or if objects are inserted into the table without using the object or SQL filer. Other system recovery errors could also allow this condition to exist (bad recovery of a journal file, disk failure, etc.).

When this option is enabled, the table compiler will generate a uniqueness check on insert for the Id value. If validation fails, `SQLCODE=-119`, will be returned to the caller and a message will be written to the console log. After writing a message to the Console.log file and before returning from the filer, the user-defined routine **^%ZOIDERROR** will be called. It is important to review the console log when this error is reported.

When this error is reported, it will be necessary to bring the ID counter back into sync with the data. Each failure will cause the system ID counter to be incremented, so it is possible that the problem will correct itself over time. At the point the error is reported it is not necessarily true that the counter is wrong, since the data itself may be incorrect. It is the responsibility of the user to determine how the counter became invalid.

19.1.8.7 New Query Optimizer Plan Verification

Regression tests based on **TestSQLScript** now have an easy way to verify query plan stability. Defining the class parameter `SHOWPLAN=1` in **%UnitTest.TestSQLScript** will cause the query optimizer plan to be written to an output file.

19.1.8.8 JDBC 3.0 Support

Cache 5.1 supports JDK 1.4 and JDBC 3.0. All required features and most optional features are supported.

19.1.8.9 GRANT and REVOKE Command Changes

Due to the extensive improvements to Caché security at 5.1, the SQL **GRANT** and **REVOKE** commands no longer support the following syntactical forms:

- GRANT ACCESS ON *namespace*
- GRANT %THRESHOLD *number*
- The %GRANT_ANY_PRIVILEGE, %CREATE_USER, %ALTER_USER, %DROP_USER, %CREATE_ROLE, %GRANT_ANY_ROLE, and %DROP_ANY_ROLE privileges

The **GRANT** and **REVOKE** command support the following additional options:

- Granting a role to a role, creating a hierarchy of roles
- The EXECUTE object privilege
- The granting of object privileges to stored procedures, as well as tables and views

- The use of the asterisk (*) to grant EXECUTE object privileges to all stored procedures

For more detailed information, see GRANT and REVOKE in the *Caché SQL Reference*.

19.1.8.10 CREATE USER Command Changes

At 5.1, issuing a **CREATE USER** does not automatically assign any roles or privileges to the user, regardless of the privileges held by the creator. Privileges and roles must be assigned to a new user using the **GRANT** command.

For more detailed information, see CREATE USER in the *Caché SQL Reference*.

19.1.8.11 Subquery Flattening

In many cases the SQL engine will now attempt to “flatten” certain types of SQL queries. That is, a query will be internally converted into an equivalent form that does not contain a subquery. In many cases, it is easier for the SQL optimizer to recognize this equivalent form, and a better execution plan is generated.

19.1.8.12 Enhanced Locking Behavior for Foreign Key References

Locking behavior during table filing has been changed in the following ways:

- During SQL **DELETE**, for every foreign key reference a long-term shared lock will be acquired on the row in the referenced table. This row will be locked until the end of the transaction. This ensures that the referenced row is not changed before a potential rollback of the SQL **DELETE**
- During SQL **INSERT**, for every foreign key reference a long term shared lock will be acquired on the referenced row in the referenced table. This row will be locked until the end of the transaction. This ensures that the referenced row is not changed between the checking of the referential integrity and the end if the **INSERT**'s transaction.
- During SQL **UPDATE**, for every foreign key reference which has a field value being updated, a long-term shared lock will be acquired on the old referenced row in the referenced table. This row will be locked until the end of the transaction. This ensures that the referenced row is not changed before a potential rollback of the SQL **UPDATE**.
- During SQL **UPDATE**, for every foreign key reference that is being changed, a long term shared lock will be acquired on the new referenced row in the referenced table. This row will be locked until the end of the transaction. This ensures that the referenced row is not changed between the checking of the referential integrity and the end if the **UPDATE**'s transaction.

19.1.8.13 READONLY Tables and Fields

Prior to this version of Caché, trying to INSERT, UPDATE, or DELETE into a ReadOnly table would not result in an error until the statement was executed. In this version, an `SQLCODE=-115` error will be raised during compilation.

When a property is defined as ReadOnly, the field in the corresponding SQL table is also now defined as ReadOnly. READONLY fields may only be defined via an initialexpression or SQL Compute code; they may never be explicitly insert or updated via SQL statements. Any attempt to INSERT or UPDATE a value for the field (even a NULL value) will result in an `SQLCODE=-138` error ("`Cannot INSERT/UPDATE a value for a ReadOnly field`").

19.1.8.14 SQLCODE Changes

The following SQLCODE error codes have been added for 5.1:

- -129: This error is raised when you attempt to set a Caché Locale setting to an invalid value. See SET OPTION in the *Caché SQL Reference* for further details.

  ```
  SQLCODE = -129: Illegal value for SET OPTION locale property
  ```

- -138: This error is raised when you attempt to compile an INSERT or UPDATE that references a read-only field. See INSERT in the *Caché SQL Reference* for further details.

  ```
  SQLCODE = -138: Cannot INSERT/UPDATE a value for a ReadOnly field
  ```

- -142: This error is raised when the **CREATE VIEW** command contains a mismatch between the number of columns in the view definition and number of columns in the query. See CREATE VIEW in the *Caché SQL Reference* for further details.

  ```
  SQLCODE = -142: Cardinality mismatch between the View-Column-list and View Query's
  SELECT clause
  ```

- -308: This error is raised when you attempt to define more than one IDENTITY field for a table. See CREATE TABLE in the *Caché SQL Reference* for further details.

  ```
  SQLCODE = -308 Identity column already defined for this table
  ```

- -316: This error is raised when a Foreign key references a non-existent column.

  ```
  SQLCODE = -316 Foreign key references non-existent key/column collection
  ```

- -321: This error is raised when you attempt to drop a view when another view references that view. See DROP VIEW in the *Caché SQL Reference* for further details.

  ```
  SQLCODE = -321 Cannot DROP view - One or more views reference this view
  ```

- -356 and -357: These two errors may be raised by an attempt to use a user-defined SQL function.

  ```
  SQLCODE = -356: SQL Function (function Stored Procedure) is not defined to return
  a value
  ```

  ```
  SQLCODE = -357: SQL Function (function Stored Procedure) is not defined as a function
  procedure
  ```

- -375: This error is raised when you attempt to roll back to a savepoint that was either never established or has already been rolled back.

  ```
  SQLCODE = -375 Cannot ROLLBACK to unestablished savepoint
  ```

- -417: This error is raised when login fails. Usually this is due to username and password checking failure. It can also occur if the username is not privileged.

  ```
  SQLCODE = -417 Cache Security Error
  ```

- -431: This error is raised when you attempt to pass a literal as a stored procedure parameter when the underlying argument type is an object type.

  ```
  SQLCODE = -431 Stored procedure parameter type mismatch
  ```

- -459: This error is raised when you try to connect using Kerberos and security authentication fails. Possible reasons include: the Kerberos security executable cconnect.dll is missing or fails to load; your connection is rejected because of the Kerberos credentials you supplied.

  ```
  SQLCODE = -459 Kerberos authentication failure
  ```

The following obsolete SQLCODE values have been removed:

SQLCODE -340, -341, -342, -343, -344, -345, -346, -347

For a complete list of SQLCODE values, refer to the "SQLCODE Values and Error Messages" chapter of the *Caché Error Reference*.

19.1.8.15 Support for %%CLASSNAMEQ and %%TABLENAME

Caché SQL now supports {%%CLASSNAMEQ} and {%%TABLENAME} references in class definition SQL specific COS code in the following locations:

- SQL Computed field code
- SQL Trigger code
- %CacheSQLStorage conditional map condition expression.

{%%CLASSNAMEQ} (not case-sensitive) will translate to the quoted string for the name of the class which projected the SQL table definition.

{%%TABLENAME} (not case-sensitive) will translate to the quoted string for the qualified name of the table

For example, assume the following trigger in the class User.Person:

```
Trigger AfterInsert1 [ Event = INSERT, Order = 1, Time = AFTER ]
{
Set ^Audit("table",{%%TABLENAME},$j,"AFTER INSERT TRIGGER")=1
Set ^Audit("class",{%%CLASSNAMEQ},$j,"AFTER INSERT TRIGGER")=1
}
```

If User.Employee extends User.Person, the following SQL trigger code will be pulled as an AFTER INSERT trigger in the SQLUSER.EMPLOYEE table:

```
Set ^Audit("table","SQLUser.Employee",$j,"AFTER INSERT TRIGGER")=1
Set ^Audit("class","User.Employee",$j,"AFTER INSERT TRIGGER")=1
```

19.1.8.16 CREATE BITMAP INDEX Support for Oracle Import Compatibility

When loading an Oracle SQL script file through **$SYSTEM.SQL.DDLImport()** or **$SYSTEM.SQL.Oracle()**, Caché SQL now recognizes the **CREATE BITMAP INDEX** statement.

19.1.8.17 Extended Support for Milliseconds

Caché SQL now supports fractional seconds in all date/time functions. The **DATEADD**, **DATEDIFF**, **DATENAME**, and **DATEPART** functions now support a datepart of "ms" or "milliseconds". The ODBC Scalar functions **{fn TIMESTAMPADD()}** and **{fn TIMESTAMPDIFF()}** now support the SQL_TSI_FRAC_SECOND parameter.

See DATEPART in the *Caché SQL Reference* for more detailed information.

19.1.8.18 Date and Time Function Enhancements

- The SQL Scalar functions TO_DATE and TO_CHAR now accept %Library.TimeStamp logical values as input. In addition, the following format codes have been added for support of TimeStamp values:
  - HH – hour of day (1-12)
  - HH12 – hour of day (1-12)
  - HH24 – hour of day (0-23)
  - MI – minute (0-59)
  - SS – second (0-59)
  - SSSSS – seconds past midnight (0-86388)
  - AM – meridian indicator
  - PM – meridian indicator

- There is a new configuration setting for the default format value for the **TO_DATE()** function. The default format is still "DD MON YYYY", but it can be changed using the following commands:

  ```
  Do $SYSTEM.SQL.SetToDateDefaultFormat(<value>)
  ```

  or

  ```
  Do SetToDateDefaultFormat^%apiSQL(<value>)
  ```

  For example:

  ```
  Do $SYSTEM.SQL.SetToDateDefaultFormat("YYYY-MM-DD HH24:MI:SS")
  ```

  The current setting for the **TO_DATE()** default format can be displayed with:

  ```
  Do CurrentSettings^%apiSQL
  ```

  or

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

- The following **CAST** and **CONVERT** operations are now supported for **%FilemanDate** and **%FilemanTimestamp**:
  - `CAST (<%FilemanDate value> AS CHAR)`
  - `CAST (<%FilemanDate value> as DATE)`
  - `CAST (<%FilemanDate value> as TIMESTAMP)`
  - `CAST (<%FilemanDate value> as VARCHAR)`
  - `{fn CONVERT(<%FilemanDate value>, SQL_DATE)}`
  - `{fn CONVERT(<%FilemanDate value>, SQL_TIMESTAMP)}`
  - `{fn CONVERT(<%FilemanDate value>, SQL_VARCHAR)}`
  - `CAST (<%FilemanTimeStamp value> AS CHAR)`
  - `CAST (<%FilemanTimeStamp value> as DATE)`
  - `CAST (<%FilemanTimeStamp value> as TIME)`
  - `CAST (<%FilemanTimeStamp value> as TIMESTAMP)`
  - `CAST (<%FilemanTimeStamp value> as VARCHAR)`
  - `{fn CONVERT(<%FilemanTimeStamp value>, SQL_DATE)}`
  - `{fn CONVERT(<%FilemanTimeStamp value>, SQL_TIME)}`
  - `{fn CONVERT(<%FilemanTimeStamp value>, SQL_TIMESTAMP)}`
  - `{fn CONVERT(<%FilemanTimeStamp value>, SQL_VACHAR)}`

19.1.9 Connectivity Improvements

New Caché 5.1 connectivity features and enhancements:

- New ECP Cluster Support
- New SNMP Support
- New LDAP Client
- New Mac OS X server support

19.1.9.1 New ECP Cluster Support

Enterprise Cache Protocol is now supported in shared disk cluster configurations with OpenVMS and Tru64 UNIX®.

Differences between ECP cluster and failover cluster:

- Faster failover
- Active shared disk(s).
- No network reconfiguration.
- Roll in and out cluster member for repair, upgrade, maintenance and etc.
- All cluster members are live.

Features

- ECP Cluster server will provide higher availability.
- Locks and transactions are preserved during failover.
- Only the cluster master serves the ECP clients.
- The cluster members could be used for other applications.

InterSystems strongly recommends the use of ECP for clustered systems. ECP represents a significant advance over predecessor networking approaches such as DCP. Customers currently using DCP for communications among members of a cluster will see improvements in performance, reliability, availability, and error recovery by converting to ECP.

19.1.9.2 New SNMP Support

To enable monitoring of Caché by a variety of systems management tools and frameworks, support for the Simple Network Management Protocol (SNMP) has been added. The %SYSTEM.MonitorTools.SNMP class allows for control of SNMP agents and functions. This class contains methods to start and stop the Caché SNMP agent, as well as the **CreateMIB()** method which generates a custom MIB file based on an application description in the Monitor Framework.

For details, see Using SNMP to Monitor Caché in the *Caché Monitoring Guide*.

19.1.9.3 New LDAP Client

Programmatic access to LDAP (Lightweight Directory Access Protocol) servers has been added. See the %Net.LDAP.Client.Session class documentation for details.

19.1.9.4 New Mac OS X server support

This version of Caché now installs and executes natively on Mac OS X 10.3. The installation kit is a standard ".dmg" distribution produced by PackageMaker.

Support has been added for Mac OS X as a server plus the following client components:

- ODBC
- JDBC
- Objects
- CSP Gateway for Apache

A native Objective-C binding is also available.

19.2 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.

19.2.1 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 5.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

19.2.1.1 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.

19.2.1.2 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.

19.2.1.3 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.

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).

19.2.1.4 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.

19.2.1.5 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_MSMActivate	No	No	No
%Service_Monitor	No	No	No
%Service_Shadow	No	No	No
%Service_Telnet	No	No	No
%Service_Weblink	No	No	No

19.2.1.6 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.

19.2.1.7 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.

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.

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.

19.2.1.8 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.

19.2.1.9 Web Server Changes

The evolution of security capability in Caché 5.1 has affected the way hypertext information is served to browsers.

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.

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.

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.

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.

19.2.1.10 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 and the directory itself. 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.

19.2.1.11 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.

19.2.1.12 Read-Only Databases

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**.

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.

19.2.1.13 Cluster Changes

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.

19.2.1.14 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.

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.

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.

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.

19.2.1.15 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.

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.

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.

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.

19.2.1.16 CACHETEMP

Caché 5.1 handles CACHETEMP differently from its predecessors. The changes are a result of security requirements and customer requests.

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 be 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.

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.

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).

19.2.1.17 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.

19.2.1.18 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).

Table 19–1: Locales Whose Default Collation Changed

Locale	Collation
chsw	Chinese2

Locale	Collation
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
rusw	Cyrillic1
ruw8	Cyrillic2
zdsw	Japanese1
zduw	Japanese1
zip8	Portuguese2

Table 19–2: 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

Locale	Collation
jpnw	Japanese
jpuw	Japanese (UNIX®)
jpww	Japanese UTF-8
korw	Korean
nld8	Dutch
nldw	Dutch
zdtw	Japanese (DTM-J)

19.2.1.19 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.

19.2.1.20 Upgrading from a Prior Release

This section covers issues related to upgrading an existing Caché system to version 5.1.

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.

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.

$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.

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.

%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.

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.

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.

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.

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.

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.

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.

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 Shadowing chapter of the *Caché Data Integrity Guide* for details.

19.2.1.21 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.

19.2.1.22 Recommendations

InterSystems has several recommendations to administrators setting up a Caché 5.1 system.

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.

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.

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.

19.2.1.23 Limitations

The following limitations apply to Caché 5.1 or specific operating systems running Caché:

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.

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.

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.

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.

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.

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.

Management of Non-% Variables in Embedded SQL

Any non-% variables used by embedded SQL statements within an 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.

Unicode in Global Names

Support for Unicode in global names is not yet fully operational and should be avoided.

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.

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.

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.

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 (WRC) for assistance.

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 (WRC) for assistance.

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.

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.

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.

Perl and Python Language Bindings

The Perl and Python language bindings are supported only on 32-bit versions of Windows.

C++ Language Binding

The C++ language binding is supported only on the Windows platform using Visual Studio 7.1.

19.2.1.24 Platform-Specific Items

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

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.

Windows XP Professional

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.

Windows Enterprise Server 2003

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.

Mac

Support for Xalan, an XSLT (Extensible Stylesheet Language Transformation) processor, is only available on OS x 10.4.

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.

- 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]**.

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.

Linux

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.

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.

Linux / AMD64

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."

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®.

- 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 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.

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.

Tru64 UNIX®

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.

HP-UX

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.

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).

19.2.2 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.

19.2.2.1 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.

19.2.2.2 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.

19.2.2.3 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.

19.2.2.4 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**.

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.

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”.

Permissions On CNLS

The CNLS application is used to change the locale of a Caché installation. Running it now requires **%Admin_Manage:U**.

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.

19.2.2.5 Changes To Routines

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.

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.

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
%LICENSE	%SYS.LICENSE

Previous Name	New Name
%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**

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.

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.

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 (WRC) for assistance.

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).

19.2.2.6 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

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
%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.
%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.

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.

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.

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.

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]**.

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

More detail on these methods can be found in the class documentation for %Library.Persistent.

Note: Changes to the underlying storage mechanisms that affect persistent objects are detailed here.

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.

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.

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.

Methods Supporting SQLCOMPUTECODE

Caché allows classes to define SQL computed properties by declaring them with the attribute, SQLCOMPUTED, 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. SQL-COMPUTECODE 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.

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.

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)
```

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.

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.

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.

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

ObjectScript

```
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.

%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.

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.

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
```

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.

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.

19.2.2.7 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.

19.2.2.8 SQL Differences

In the transition to version 5.1, the following changes were made in SQL that may affect existing programs.

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.

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.

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.

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.

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]**.

%msql Eliminated

This variable was used in 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.

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”)

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.

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.

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.

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.

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.

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.

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
```

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.

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.

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.

^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.

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.

19.2.2.9 System Error Code Changes

New Error Codes

This version of Caché adds news system error codes:

Table 19–3: 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.

Error Codes Removed

This release of Caché no longer supports the system error, <DISCONNECT>.

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.

19.2.2.10 ObjectScript Changes

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.

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.

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.

$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.

$ZERROR Contains 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\
```

$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.

Function Number	Requirement	Explanation
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) Change

  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) Change

  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 "^".

$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.

19.2.2.11 Storage Changes

%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, multi-dimensional, 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.

%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

19.2.2.12 Java

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.

19.2.2.13 Terminal

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.

Argument Changes

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.

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.

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.

Local And Network Encodings Are Now Distinct

The local and network translation settings for 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.

19.2.2.14 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

19.2.2.15 Callin And Callout

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.

19.2.2.16 CSP Changes

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.

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.

19.2.2.17 Collation For Locales Now On By Default

Please refer to the discussion in the Administrator section.

19.2.2.18 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
  ```

19.2.3 Operators

19.2.3.1 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.

19.2.3.2 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.

19.2.3.3 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.

19.2.3.4 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.

19.2.3.5 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.

A

Pre-2014.1 Upgrade Information

A.1 Upgrading Caché

A.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 the InterSystems Global Summit or other events.

A.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.

Note: 2K block size databases must be converted to 8K format before upgrading to 2012.2.x or beyond. See Supported Upgrade Paths in the "Upgrading Caché" chapter of the *Caché Installation Guide* for more information.

Prior to beginning an upgrade, InterSystems recommends that an instance be shut down using standard methods — not as the result of, for example, a system crash.

After each upgrade step, the following conditions apply:

- Pure Routines

  Pure routines are upward-compatible from version to version and do not need to be recompiled. Pure routines are those that:

  – are not generated as a result of class compilation;

  – do not use classes as part of their operation; and

  – do not access routine or class meta-information (such as using source-level debugging on version X of code compiled on version earlier than X.).

  Most versions contain improvements in the routine compiler, in the generated code, and/or efficiencies in the runtime support, and customers may decide to recompile their routines to take advantage of new features.

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

- Classes, SQL, Projections, Proxy Classes

  InterSystems requires that all custom classes be recompiled after an upgrade. Users should recompile their application classes using either their own build procedure, or enter the following command to upgrade and compile the class dictionaries in namespaces with custom code:

  ```
  ZN "namespace"
  Set StatusCode = $SYSTEM.OBJ.CompileAll("u")
  Do $SYSTEM.Status.DisplayError(StatusCode)
  ```

  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.

  It is also good practice to log the output from the preceding command sequence so that a detailed examination of the results is possible.

  You could also use the **$SYSTEM.OBJ.CompileAllNamespaces()** method to recompile all namespaces, but you should not do this when upgrading a HealthShare installation. You should not recompile a namespace used for HealthShare.

- Other routines

  Routines that do not meet the preceding definition of pure routines must be recompiled after classes are recompiled. Make sure you have access to write to the database and execute the command:

  ```
  Do ##class(%Library.Routine).CompileAll()
  ```

  in each namespace of interest.

- 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 cannot 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.

When Recompiling

There are several things to remember when recompiling:

1. Because recompiling necessarily updates system data associated with the routine, users who recompile classes or routine must have write access to this data (^ROUTINE) in each namespace where the object being compiled is mapped. Failure to observe this requirement will result in "ERROR #302: the database is read-only".
2. Everything needed for compilation must be accessible in the namespace where the class or routine is mapped. For example, if a class or routine refers to a user-defined include file, this must also be mapped so it will be found by the compiler when the source is processed.
3. By default, **$SYSTEM.OBJ.CompileAllNamespaces** does not compile the namespaces "%SYS" and "DOCBOOK". If a site has classes in %SYS, these will need to be manually recompiled. Do not use **$SYSTEM.OBJ.CompileAllNamespaces** on a HealthShare installation.
4. The recommendation to use features such as **$SYSTEM.OBJ.CompileAll** assumes that all information about dependencies between items being compiled is entirely contained within the declarations of the items themselves. For some complex applications, for example those that assemble an application "core" and then bootstrap themselves to the final version, this will not be the case. In such instances, users should create scripts to compile the code so that dependencies unavailable to the compiler are accounted for, especially in relation to schemas and domain-specific languages.

5. Although it is not required, if a site has placed routines named [%]Z* in the %SYS namespace, InterSystems recommends that these be recompiled, for example, with

```
ZnSpace "%SYS"
Set sc = $SYSTEM.OBJ.CompileList("Z*,%Z*", "ck")
```

A.1.3 Compatibility With Earlier Versions

InterSystems strives to assure that applications written for a version of Caché will run without change on subsequent versions. While this is not always possible (and therefore the reason for this document), the reverse is not true.

CAUTION: In general, InterSystems does not support moving an application developed for version X to a version prior to X. This applies to both source and compiled versions of the application. Exceptions to this are few and explicitly noted.

Customers writing an application that will be deployed on multiple versions of Caché should design and develop the application to use only the features on the earliest of the intended versions.

A.1.4 Web Application Considerations

As part of your upgrade and deployment plans, please consider the following strong recommendations from InterSystems regarding web applications:

- In secure deployment environments, do not allow Unauthenticated access to any web application. To do this, on the Edit Web Application page for each application, uncheck the `Unauthenticated` box under `Allowed Authentication Mechanisms`.
- The /csp/broker web application should not have any allowed authentication methods. To do this, on the Edit Web Application page for /csp/broker, uncheck the boxes for all the choices under `Allowed Authentication Mechanisms`.