



Using the MultiValue Features of Caché

Version 2008.1
29 January 2008

Using the MultiValue Features of Caché
Caché Version 2008.1 29 January 2008
Copyright © 2008 InterSystems Corporation
All rights reserved.

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



Caché WEBLINK, Distributed Cache Protocol, M/SQL, N/NET, and M/PACT are registered trademarks of InterSystems Corporation.



InterSystems Jalapeño Technology, Enterprise Cache Protocol, ECP, and InterSystems Zen are trademarks of InterSystems Corporation.

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

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

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

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

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

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Customer Support

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

Table of Contents

1 Introduction and Overview	1
2 Installing Caché	3
3 Starting MultiValue	5
3.1 Connecting Via the Caché Terminal on Windows	5
3.1.1 Starting a Terminal Session	5
3.2 Connecting to Caché Via Telnet	5
3.3 Starting the MV Shell	6
3.3.1 Exploring MultiValue on Caché	6
3.4 Importing Data and Programs into Caché	6
3.4.1 Starting the MV Shell	7
3.5 Notes on the MV Shell	7
3.5.1 Issuing MV Basic Commands in the Shell	7
3.5.2 Issuing Caché ObjectScript Commands from MV	7
3.5.3 Command History Editor	8
3.5.4 Starting Directly In The MV Shell	8
4 Using a Caché Class and MultiValue: A Sample Session	9
4.1 Starting the MV Shell	10
4.2 Importing the Class Definition	10
4.3 Creating, Manipulating, and Saving an Object	11
4.4 Running Some Simple MV Queries against the Class	12
4.5 Populating the Class with Test Data	13
4.6 Running Other MV Queries	13
4.7 Using the System Management Portal to Run SQL Queries	15
4.8 Using Studio to Examine the Class Definition	15
Appendix A: Available Flavors / Emulations	17
Appendix B: Configuration Options	19
B.1 Controlling Caché Startup Behavior On Windows	19
B.1.1 Change Caché Start To Manual	19
B.1.2 Remove The Cube From The System Tray	19
B.2 Enabling Telnet	20
B.2.1 Start The Management Portal	20
B.2.2 Configure Telnet	20
B.2.3 Enable Telnet	21
B.3 Increasing The Routine Buffer Size	21
B.3.1 Setting The New Value	21

1

Introduction and Overview

Caché provides MultiValue developers with access to its application development and system management features. This set of features provides a native MultiValue environment as part of Caché. It also allows existing MultiValue applications to take advantage of the features of Caché, including:

- Caché objects
- Both MultiValue queries and SQL queries
- Studio, an integrated development environment
- Caché Web-based features, including Caché Server Pages (CSP) and Web services generally
- Browser-based system management
- Advanced security management

This document provides basic information on installing Caché and getting started with the MultiValue (MV) shell.

CAUTION: Major changes have been made since the previous edition of the MV Developer's release. Due to these changes upgrading from previous versions to this one is NOT allowed. If you have an existing installation, export any data and programs you wish to transfer to the new edition. Then stop Caché and uninstall the existing edition. Install this new version in a new directory. You can then import the data and programs you saved.

For further information on installing, starting, stopping and removing Caché, please consult the [Caché Installation Guide](#). There are separate sections for [Windows](#), [OpenVMS](#), [UNIX and Linux](#), and [Macintosh](#).

2

Installing Caché

This chapter covers the basics of installing Caché for those who are new to InterSystems and its products. To find out if the system you intend to use is one that InterSystems supports for Caché, please check the [Supported Platforms](#) document. This lists the details of hardware and software requirements for successful operation.

A full description of the Caché installation process for experienced (and/or more curious) users is available in the [Installation Guide](#).

If you are new to Caché, there are some things to keep in mind about installation:

- It is possible to install multiple copies of Caché on a given platform and have them operate independently of one another. In this case, the individual installations are referred to as “instances” .
- Each instance of Caché must be installed in a location (“destination folder”) separate from all the others.
- Each instance of Caché communicates with the outside world over two TCP/IP ports, (the “superserver port” and the “webserver port”). The installer for your platform will choose available unused ports. Each can be overridden, but in most cases it is not necessary.
- Caché provides security facilities that can be tuned to the level of protection needed by a particular installation. You can defer learning about the details of security by choosing “Minimal” as the level of security for your MV installation.

These instructions assume that you have obtained a CDROM with a version of Caché that will run on your system. If you are new to Caché, you are encouraged to review the introductory material on Caché noted previously. Installation instructions for specific platforms can be found in the [Installation Guide](#) for

- [Windows](#)
- [Linux and Unix](#)

- [Macintosh](#)

3

Starting MultiValue

There are two principal paths to accessing the Caché MultiValue shell: via a built-in terminal emulator (on Windows systems only), or via a telnet connection.

3.1 Connecting Via the Caché Terminal on Windows

Once Caché is installed and operational, you can run the MultiValue shell via a Caché terminal session. To do this:

3.1.1 Starting a Terminal Session

Click on the Caché icon in the System tray. Choose **Terminal** from the menu displayed. A new terminal session starts and displays a window in which to enter commands.

3.2 Connecting to Caché Via Telnet

The MultiValue shell can also be run via a telnet connection using your favorite terminal emulator. By default, Caché is installed with its telnet capability disabled; the instructions to [enable telnet](#) are given in the appendix.

3.3 Starting the MV Shell

Regardless of how you connect to Caché, in the terminal window at the “USER>” prompt enter the command:

```
MV
```

followed by **ENTER**.

If this is the first invocation of MV, some initialization is required. For example, MV will create the SYSPROG and USER accounts automatically. Once initialization is complete, MV displays a welcome message and then the colon (:) prompt.

You may begin working in the default account just set up. Or, you may create one or more other accounts using the CREATE-ACCOUNT command, specifying which MultiValue flavor you wish to use, for example,

```
CREATE-ACCOUNT DEMO <flavor>
```

where <flavor> is the name of the flavor to use. The complete list of available flavors is in the [appendix](#).

3.3.1 Exploring MultiValue on Caché

Once the shell is running, you may explore the MultiValue environment. Other documents will help you import your applications and data, as well as expand your capabilities to include the features that Caché provides.

Once you logout from the shell, you will returned to your terminal or telnet session. To finish, you should **QUIT** from your telnet session, or issue the **HALT** command if you are in a Caché terminal session.

3.4 Importing Data and Programs into Caché

Note: At present, Caché only provides support for importing backups made on UniVerse, D3/PICK, and jBase systems. For other MV systems, InterSystems provides example export and import programs that can be modified as needed by the user to transfer the relevant data to Caché.

3.4.1 Starting the MV Shell

This is the procedure described [previously](#). There is an implicit assumption that importing applications and/or data is an administrative function that should be done under the SYSPROG account. Once the shell is started, enter the command:

```
MVIMPORT <BackupFilePath>
```

where BackupFilePath is the location of the backup file. It must be supplied in machine-specific format. For example, on a Windows system it would be something like:

```
MVIMPORT C:\MVTEST\Acct_Main\Backup.uvb
```

MVIMPORT will read the backup file, create the necessary dictionary and files entries, and store the programs and data there. See the [Operational Differences](#) document for details on MVIMPORT.

3.5 Notes on the MV Shell

This section provides additional information related to interacting with Caché from the MV shell.

3.5.1 Issuing MV Basic Commands in the Shell

The MV shell provides the ability to execute Basic commands directly by preceding them with a semicolon. For example,

```
; PRINT "Hello, World"
```

will result in “Hello, World” being echoed to the terminal window.

3.5.2 Issuing Caché ObjectScript Commands from MV

The MV shell provides a way to issue ObjectScript commands without having to terminate shell operation. The COS command sends the remainder of its line to Caché for execution. For example, at the MV prompt, the command

```
COS WRITE "Hello, World", !
```

will write the expected string to the output device followed by a newline. Since multiple Caché commands may be placed on a line (separated by single blank spaces), a single invocation of COS may execute multiple Caché commands.

The character “[” may be used in place of the “COS” .

3.5.3 Command History Editor

Within the MV shell, it is possible to recall previously entered commands, perform simple edit operations on them and submit the edited version as a new command. This is similar to the functionality provided by the UniVerse TCL stacker. All the commands that operate in this fashion begin with a period, (.). The available commands are:

Command	Description
.A{nn} string	Append the characters of string to the end of stack number 'nn'
.C{nn}/From/To	Change “From” to “To” in string 'nn'
.DNN{-mm}	Delete stack 'nn' (possibly through stack 'mm')
.L{nn}	Display the last 'nn' lines of the stack
.U{nn{-mm}}	Convert stack 'nn' (through stack 'mm') to uppercase
.X{nn{-mm}}	Execute stack 'nn' (through 'mm')
.?	Display a summary of the available commands

3.5.4 Starting Directly In The MV Shell

Caché provides the capability for an MV user to start up in the MV shell. This is done by modifying the definition of the user to specify a namespace and initial routine that will be executed when the login is complete.

Note: In order to modify the definition of a user, you must be an administrator. That is, you must be a member of a role that has **%Admin_Secure** or **%All** privileges. If you installed Caché using the “Minimal” security setting as directed, you should automatically have the proper permission.

If you have the proper authorization, you can modify the user definition via the Management Portal page **[Home] > [Security Management] > [Users]**. Select the user you wish to modify and select the “Edit” option. When the **Edit User**, page appears, fill in the **Default Namespace** text box with the namespace you wish the user to be in when the MV shell starts. Enter “**^%SYS.MV**” into the **Default Tag^Routine** box as the initial Caché command to be executed when login succeeds.

4

Using a Caché Class and MultiValue: A Sample Session

Caché comes with the file MVDEMO.PERSON.xml, which contains the MVDEMO.PERSON class. This section provides a sample session of loading and manipulating this class. Its steps provide examples of the MultiValue features in Caché, and how the native Caché environment interacts with these features.

The steps are:

1. [Starting the MV shell.](#)
2. At the MV shell prompt, [importing the class definition.](#)
3. Using MV Basic to perform [basic object manipulation](#): instantiation, setting some properties, and saving an instance.
4. Examining the data in the MV shell with [a simple MV query.](#)
5. Using the [Caché data population utility](#) to instantiate a larger number of objects.
6. [Running more complex MV queries](#) against the data.
7. In the Caché System Management Portal, [running SQL queries](#) against the data.
8. Using Studio to [examine the class definition.](#)

In addition to supporting traditional MV Basic file manipulation (such as with OPEN, WRITE, and so on), Caché allows you to treat data as objects. This sample highlights the use of Caché object features.

4.1 Starting the MV Shell

In a Caché Terminal window, make sure you are in the USER namespace; this is indicated by a prompt of `USER>`. A Caché namespace is similar to a MultiValue account.

To go from another namespace to the USER namespace, use the **ZNamespace (ZN)** command — in this example, from the SAMPLES namespace:

```
SAMPLES> ZN "USER"  
USER>
```

Then start the MV shell:

```
USER> MV
```

followed by **ENTER**.

More on the MV Shell can be found in the [Operational Differences](#) documentation.

4.2 Importing the Class Definition

The definition of the MVDEMO.PERSON class is in the file MVDEMO.PERSON.xml. To import the class definition into Caché, use the **Load** method of the %SYSTEM.OBJ class in the MV shell:

```
USER: ; "%SYSTEM.OBJ"->Load(<class definition file>, "c")
```

where

- The leading semi-colon (“;”) in the MV shell specifies that the rest of the line is MV Basic code (not an MV command). The Cache MV shell has been enhanced to allow for lines of MV Basic code to be executed from the MV shell.
- The first argument specifies the quoted name of a XML class definition file, including a full path. The location of this file relative to the Caché installation directory is
<cache-installation-dir>\Dev\mv\samples\MVDEMO.PERSON.xml (Windows) or
<cache-installation-dir>/dev/mv/samples/MVDEMO.PERSON.xml (Unix, Linux, MacOs).
- The second argument specifies the required compiler flags (“c” means compile).

Note: All examples of invoking commands from the MV shell in this chapter include the MV shell prompt (“USER: ”).

When you run this command, a series of output statements appear in the Terminal indicating that the file has been imported and compiled.

For a standard Windows installation, the following will do the import:

```
USER: ; demo = "C:\InterSystems\Cache\Dev\mv\samples\MVDEMO.PERSON.xml"  
USER: ; stat = "%SYSTEM.OBJ"->Load(demo, "c")
```

When you run this, a series of output statements appear in the Terminal indicating that the file has been imported, compiled, and exported to a MultiValue file.

With the MV shell, you can specify:

- MV shell commands.
- MV Basic commands. Lines containing MV Basic commands begin with a leading semi-colon.
- Caché ObjectScript calls. Lines containing Caché ObjectScript begin with a leading “COS” or a leading left bracket (“[”).

4.3 Creating, Manipulating, and Saving an Object

Once the compiled class is available, you can invoke its **CreateSome** method:

```
USER: ; "MVDEMO.PERSON"->CreateSome()
```

This command invokes the **CreateSome** method of the MVDEMO.PERSON class. This method performs the following actions:

- Creates an instance of MVDEMO.PERSON
- Sets the values for the properties of that instance.
- Saves the instance.
- Creates ten other instances of the class using the Caché data population utility (described in the following section on populating the class with test data).

4.4 Running Some Simple MV Queries against the Class

Since there are now instances of the class, you can use the available Caché tools to run queries against them. Standard MV shell commands are available in the Caché MV shell, so it is easy to run an MV query:

```
LIST MVDEMO.PERSON Name
```

As expected, this query displays the value of Name and ID of each instance of the class. For example, output from this command might look like:

```
LIST MVDEMO.PERSON Name                                02:09:27pm  26 Jul 2006  PAGE    1
ID.....      Name.....
25018          Smith,John
25019          Ott,Barbara T.
25020          Vonnegut,Maria Y.
25021          Sands,Roger J.
25022          Wakefield,Phil U.
25023          Ortiz,Mario O.
25024          Finn,Bob Z.
25025          Jones,Chris K.
25026          Yeats,Phil H.
25027          Goldman,Phil O.
25028          Ng,Valery W.
```

11 Items listed.

Note: Because the populate utility generates data randomly, the results of this command will differ from what appears here.

A query can also refer to collection properties, such as this class' Phone property:

```
LIST MVDEMO.PERSON Name Phone                          02:15:04pm  26 Jul 2006  PAGE    1
ID.....      Name.....      Phone.....
25018          Smith,John          555-123-1234
                555-432-4321
25019          Ott,Barbara T.    460-920-4165
                661-911-2114
                650-886-1018
25020          Vonnegut,Maria Y. 874-843-5492
                294-588-4007
25021          Sands,Roger J.    781-540-8221
                897-722-5975
25022          Wakefield,Phil U. 635-827-9697
25023          Ortiz,Mario O.    229-472-7806
                989-512-1785
25024          Finn,Bob Z.      214-929-6674
                483-685-7455
25025          Jones,Chris K.    874-204-8174
25026          Yeats,Phil H.    345-959-9979
25027          Goldman,Phil O.  423-344-9508
```

25028

Ng, Valery W.

533-923-2919

481-245-8553

479-793-4879

4.5 Populating the Class with Test Data

Because `MVDEMO.PERSON` is a subclass of the `%Library.Populate` class (often known simply as `%Populate`), it has built-in functionality for generating test data. Running the **CreateSome** method creates ten instances of the class using this functionality.

To create one hundred thousand instances of the class, the command is:

```
USER: ; PRINT "MVDEMO.PERSON"->Populate(100000)
```

This code uses a number of special features of the Caché MV shell:

- The leading semi-colon (“;”) specifies that the subsequent code is an MV Basic command.
- The quoted string specifies the name of the class being invoked.
- The arrow syntax “->” allows you to refer to a property or method of a class. (If a property is itself an object, this syntax may appear more than once, for example in syntax such as `"MYAPP.EMPLOYEE"->SPOUSE->BIRTHDAY`.)
- Since the **Populate** method returns the number of instances it populated, the MV Basic **PRINT** command uses this value as an argument.

Regarding the call itself, the data population utility generates test data for the class properties specified in the `%Library.PopulateUtils` class. For more information, see the chapter “[The Caché Data Population Utility](#)” in the book *Using Caché Objects*.

4.6 Running Other MV Queries

More sophisticated queries are also available. These may be more useful when there are larger amounts of data.

For example, queries can attempt to match parts of strings:

Using a Caché Class and MultiValue: A Sample Session

```
LIST MVDEMO.PERSON Name Phone WITH Name LIKE Yeats, Qui...
                                02:23:27pm 26 Jul 2006 PAGE 1
ID..... Name..... Phone.....
26040      Yeats, Quigley S.      201-507-5264
                                946-663-7004
41436      Yeats, Quigley A.      348-807-3906
                                859-639-9381
```

2 Items listed.

They can perform multiple matches, with exact and partial matches:

```
LIST MVDEMO.PERSON Name WITH Hair = brunette AND Name LIKE Presl...
                                02:27:20pm 26 Jul 2006 PAGE 1
ID..... Name.....
25070      Presley, Mary N.
26245      Presley, Michael M.
26307      Presley, Orson Q.
26455      Presley, Amanda R.
28654      Presley, Chad L.
28727      Presley, Thelma Q.
29922      Presley, Clint B.
30459      Presley, Richard B.
30799      Presley, Filomena M.
31313      Presley, Wilma E.
...
```

They can suppress the display of the ID for each row:

```
LIST MVDEMO.PERSON Name WITH Hair = brunette AND Name LIKE Presl... ID.SUP
                                2:28:29pm 26 Jul 2006 PAGE 1
Name.....
Presley, Mary N.
Presley, Michael M.
Presley, Orson Q.
Presley, Amanda R.
Presley, Chad L.
Presley, Thelma Q.
Presley, Clint B.
Presley, Richard B.
Presley, Filomena M.
Presley, Wilma E.
...
```

And they can perform counts:

```
USER:COUNT MVDEMO.PERSON WITH Hair = brunette
8391 Items counted.
```

4.7 Using the System Management Portal to Run SQL Queries

Because Caché provides full support for SQL, you can invoke SQL queries against your data. The Caché System Management Portal provides functionality for running ad hoc queries. To run an SQL query:

1. From the Caché cube menu, choose **System Management Portal**. If you have performed a minimal-security installation of Caché, this displays the main page for the System Management Portal; with Normal or Locked-Down installations, a login screen appears.
2. On the Portal's main page, choose **SQL** from the middle column (**Data Management**). This displays the Portal's **SQL** page.
3. Choose the USER namespace from the **Namespace** list on the left.
4. Choose **Execute SQL Statement** from the left column (**SQL Operations**). This displays the **Execute SQL Query** page.
5. On this page, Caché allows you to create and run standard SQL queries against your data. You can either write a query in the available editing area or click the **Query Builder** button to use the interactive **SQL Query Builder**.

SQL access is also available via any ODBC or JDBC tool.

4.8 Using Studio to Examine the Class Definition

Having previously imported the class definition, you can view and edit it in Studio. Studio is a full-featured development environment. To display the class definition in Studio, the process is:

1. From the Caché cube menu, choose **Studio**. A login screen appears. With a minimal-security installation, simply click **OK** and Studio runs under the UnknownUser account, which has all privileges in this type of installation; with normal and locked-down installations, enter a valid user name and password.
2. You should be in the USER namespace. To determine if this is so, check the string that appears in the Studio window's title bar. This is of the form:

```
Studio-<logged-in user>-<Caché instance>/<active namespace>-<project>
```

such as

```
Studio-UnknownUser-CacheMVTTest/USER-Default_
```

3. If you are not in the USER namespace, select **File-Change Namespace** (or **F4**). This displays the **Caché Connection Manager** screen. Choose USER from the list and click **OK**.
4. Open the MVDEMO.PERSON class. To do this:
 - Select **File-Open** from the menu.
 - Press **Ctrl-O**.
 - Click the Open icon from the menu.

Any of these actions displays the **Open** dialog for the USER namespace.

5. In the Open dialog, look only for class definitions by specifying Class Definition (*.cls) in the Files of type field.
6. Select MVDEMO from the available choices and the PERSON.cls from the choices that then appear.

For more information on Studio, see the book *Using Caché Studio*.

A

Available Flavors / Emulations

The following is a list of the flavors (or emulations) that are accepted by Caché. The emulation setting for an account can be changed via the [CEMU](#) command.

Flavor	Interpretation
CACHE	Our own Caché mode — the default if no emulation is specified
D3	D3 mode — D3 is the latest incarnation of Pick
DEFAULT	Synonym for CACHE
IN2	Intertechnique mode
INFORMATION	Synonym for PRIME
JBASE	jBASE Ideal mode
MVBASE	MV Base, a GA system now owned by Raining Data
PICK	Original Pick R83 mode to which many applications are programmed
PIOPEN	PI/Open
PRIME	The basis for UniVerse and Unidata, but is subtly different
R83	Original Pick R83 mode to which many applications are programmed
R95	Reality R95 - R83 plus a number of enhancements
REALITY	Reality Operating System
ULTIMATE	Ultimate Pick and Ult+
UNIDATA	Unidata Ideal mode

Flavor	Interpretation
UNIVERSE	The UniVerse Ideal

Note: While all of these flavors are accepted, InterSystems recommends that you use the Caché version, if possible. Flavor-dependent behavior for commands, functions and so on, has been most completely implemented for UNIVERSE and JBASE.

InterSystems has been focused most tightly on supporting the flavors listed above. If you are porting to Caché from a MultiValue implementation which is not one of those listed, you may find that commands, options, conversion codes, and so on specific to your MultiValue dialect are not supported. In these instances, InterSystems recommends that you replace them with equivalents from one of the supported flavors.

B

Configuration Options

B.1 Controlling Caché Startup Behavior On Windows

Whether or not there is an icon in the system tray, you can always control a Caché instance via the **Start**, **Programs** and **Caché** menu choices.

Alternatively,

B.1.1 Change Caché Start To Manual

From the **Start** menu, select **Programs**, **Administrative Tools**, then **Services**. On some systems, **Administrative Tools** is found in **Start**, **Settings**, **Control Panel**.

From the Service window, select the item called, “Caché Controller for <instance name>”, where <instance name> is the name you assigned to the Caché you just installed.

From the **Action** menu, choose **Properties**, and change the Startup Type to Manual. Click **OK**.

B.1.2 Remove The Cube From The System Tray

Whether you choose to temporarily or permanently remove the Cube from the system tray, this only affects the small application that is the Cube and not Caché as a whole. If you halt the Cube, you may still control Caché via the **Start** and **Programs** menus.

B.1.2.1 Temporarily

If you click on the Cube in the system tray and choose **Exit**, the cube will stop and its icon will no longer appear in the system tray. The Cube will reappear when the system is rebooted.

B.1.2.2 Permanently

On your Windows boot disk, go to the directory,

```
Documents and Settings\All Users\Start Menu\Programs\Startup\Caché
```

Delete the directory that bears the instance name whose cube you wish to eliminate from the system tray.

B.2 Enabling Telnet

Caché has the capability of communicating with a terminal shell via the telnet protocol. However, when Caché is first installed, the telnet facility is turned off as a security measure. Telnet can be turned on by following these steps:

B.2.1 Start The Management Portal

Caché is administered via a browser-based interface known as the [System Management Portal](#). Make sure the Caché instance you installed is running. Then, start the portal.

B.2.1.1 On Windows Systems

Left-click on the Caché Cube in the system. From the menu, choose **System Management Portal**.

B.2.2 Configure Telnet

When installed, Caché assumes telnet communications will take place on the default port, 23, specified by the Internet Assigned Number Authority (IANA). If you wish to change this port assignment, choose **Configuration** from the System Administration column. In the following page, select **Advanced Settings** from the System Configuration column.

In the **[Home] > [Configuration] > [Advanced Settings]** page that displays, choose **Telnet** from the **Category** scrolling list. The Telnet parameters are shown and you may choose to edit the settings of any of them. Click **OK** to save the new settings. When you have finished, return to the **[Home]** page.

B.2.3 Enable Telnet

When the portal **[Home]** page displays, choose **Security Management** from the System Administration column.

In the new page, choose **Services** from the Security Definitions column. This displays a table of the available Caché services and their respective status. For the table entry named, %Service_Telnet, if the Enabled column has the value “No” , click on the %Service_Telnet name.

The **[Home] > [Security Management] > [Services]** page that displays allows you to modify the properties of the telnet service. The meaningful entries are:

- Service enabled:
Checking this box will enable the service so you can connect to Caché via telnet.
- Unauthenticated
Checking this box means that anyone can connect via telnet as long as they know the IP address of the host on which this Caché instance is installed.
- Password
Check this box if you wish to require operating system authentication (userid and password) of users attempting to connect to this Caché instance.

The remaining settings are for users of more advanced security features. When you have made you selections, click the **Save** button.

B.3 Increasing The Routine Buffer Size

In some cases, Caché may fail to compile a routine because its final size exceeds the allocation size set for routines. The error message displayed is: <ROUTINE TOO BIG TO SAVE>. If this happens, you may have Caché to allocate larger buffers to hold the compiled code. This is done via the Management Portal.

B.3.1 Setting The New Value

From the portal **[Home]** page, choose **Configuration** and on the subsequent page choose **Advanced Settings**.

On the **[Home] > [Configuration] > [Advanced Settings]** page, choose “Memory” from the Category list. Find the Setting, RoutineBufSize, and click **Edit**. In the edit page, change the value from 32 (the

default) to 64. Click **OK**, then **Apply Changes**. Afterward, you may close the system management portal.

Note: The size allocated to routine buffers is used at Caché initialization. The new value will not take effect until Caché is restarted.

B.3.2 Restarting Caché

B.3.2.1 On Windows Systems

Left-click on the Caché Cube in the system. From the menu, choose **Stop Caché**. A popup window appears with options to stop Caché completely, or to stop and restart Caché. Select the **Restart** option and click **OK**.

The icon in the system tray will be grayed out while Caché is not operational. It returns to its normal display when Caché has been restarted.