Monitoring Intersystems Cache Database

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

About eG Innovations

eG Innovations provides intelligent performance management solutions that automate and dramatically accelerate the discovery, diagnosis, and resolution of IT performance issues in on-premises, cloud and hybrid environments. Where traditional monitoring tools often fail to provide insight into the performance drivers of business services and user experience, eG Innovations provides total performance visibility across every layer and every tier of the IT infrastructure that supports the business service chain. From desktops to applications, from servers to network and storage, from virtualization to cloud, eG Innovations helps companies proactively discover, instantly diagnose, and rapidly resolve even the most challenging performance and user experience issues.

eG Innovations is dedicated to helping businesses across the globe transform IT service delivery into a competitive advantage and a center for productivity, growth and profit. Many of the world's largest businesses use eG Enterprise to enhance IT service performance, increase operational efficiency, ensure IT effectiveness and deliver on the ROI promise of transformational IT investments across physical, virtual and cloud environments.

To learn more visit www.eginnovations.com.

Contact Us

For support queries, email support@eginnovations.com.

To contact eG Innovations sales team, email sales@eginnovations.com.

Table of Contents

ABOUT EG INNOVATIONS 2

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: PRE-REQUISITES FOR MONITORING AN INTERSYSTEMS CACHE DATABASE SERVER 3

2.1 Configuring the Intersystems Cache Database to work with the eG Agent 3

CHAPTER 3: ADMINISTERING THE EG MANAGER TO MONITOR THE INTERSYSTEMS CACHE DATABASE 14

CHAPTER 4: MONITORING THE INTERSYSTEMS CACHE DATABASE 16

4.1 The Cache Server Layer 17

4.1.1 Cache Database Test 18

4.1.2 Cache Console Log Test 21

4.1.3 Cache Global Stats Test 23

4.1.4 Cache Processes Test 25

4.1.5 Cache System Test 27

4.1.6 Ecp Application Server Test 30

4.1.7 Ecp Data Server Test 33

4.2 The Cache Memory Structures Layer 37

4.2.1 Cache Buffer Test 37

4.2.2 Cache Locks Test 42

4.3 The Cache Service Layer 44

4.3.1 Cache Performance Test 45

4.3.2 Cache Resources Test 52

4.3.3 Cache Network Test 54

CHAPTER 5: TROUBLESHOOTING 57

CHAPTER 6: CONCLUSION 59

Table of Figures

Figure 1.1: Access to the data stored by the data engine 2
Figure 2.1: The Services page 4
Figure 2.2: Enabling service monitoring 5
Figure 2.3: Switching to a different namespace 6
Figure 2.4: Specifying the Namespace to switch to 6
Figure 2.5: Executing the routine %MONAPPMGR 7
Figure 2.6: Selecting the Manage Monitor Classes option 7
Figure 2.7: Choosing to activate/deactivate a monitor class 8
Figure 2.8: Specifying the name of the class to activate/deactivate 8
Figure 2.9: Activating the Monitor Class 9
Figure 2.10: Selecting another class for activation 9
Figure 2.11: Activating the Database, Processes, and SystemMetrics class 10
Figure 2.12: The Monitor Settings page of the System Management Portal 11
Figure 2.13: Enabling the SNMP Agent to start on system boot 12
Figure 3.1: Managing a Cache Database server 14
Figure 3.2: A table displaying the list of unconfigured tests for the Cache Database server 14
Figure 4.1: Layer model of the Cache database server 16
Figure 4.2: Tests associated with the Cache Server layer 18
Figure 4.3: The tests associated with the Cache Memory Structures layer 37
Figure 4.4: The tests associated with the Cache Service layer 44

1

Chapter 1: Introduction

CACHÉ is a multidimensional database that combines robust objects and robust SQL, thus eliminating object-relational mapping. Cache enables rapid web application development, quick transaction processing, scalability, and real-time queries against transactional data.

Caché is a post-relational database - the “relational” part of “post-relational” refers to the fact that Caché is a full-featured relational database. All the data within a Caché database is available as true relational tables and can be queried and modified using standard SQL via ODBC, JDBC, or object methods. The “post” part of “post-relational” refers to the fact that Caché offers a range of features that go beyond the limits of relational databases, while still supporting a standard relational view of data. These features include:

- The ability to model data as objects (each with an automatically created and synchronized native relational representation) while eliminating both the impedance mismatch between databases and object-oriented application environments as well as reducing the complexity of relational modeling.
- A simpler, object-based concurrency model.
- User-defined data types.
- The ability to take advantage of methods and inheritance, including polymorphism, within the database engine.
- Object-extensions for SQL to handle object identity and relationships.
- The ability to intermix SQL and object-based access within a single application, using each for what they are best suited.
- Control over the physical layout and clustering used to store data in order to ensure the maximum performance for applications.

Another unique feature of Cache is its Unified Data Architecture. Whenever a database object class is defined, Caché automatically generates a SQL-ready relational description of that data. Similarly, if a DDL description of a relational database is imported into the Data Dictionary, Caché automatically generates both a relational and an object description of the data, enabling immediate access to objects.

At the core of the Cache architecture, is the Cache Multi-dimensional Database Engine that provides the complete set of services— including data storage, concurrency management,

transactions, and process management—needed to build complex database management systems. Data can be stored and accessed from the database engine using objects, SQL, or through direct access to multidimensional structures. Figure 1.1 depicts the access methodologies.

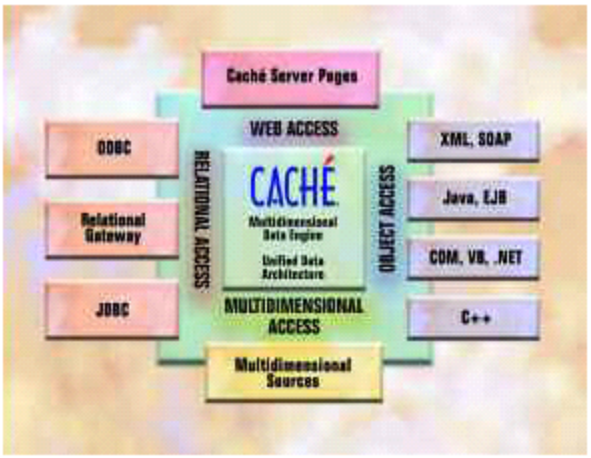

Figure 1.1: Access to the data stored by the data engine

Regardless of the access method, all data in Cache's database is stored in Cache's multidimensional arrays. These multi-dimensional arrays enable efficient data storage; further, the absence of joins and table-hopping hastens query execution. In addition, its ability to distribute data and application logic across multiple server systems, simplifies the development of reliable applications. Owing to these capabilities, Cache is a key ingredient in the design and delivery of a wide variety of applications ranging from single-user embedded systems to large, multi-server, multi-user installations (such as those required by banks, hospitals, etc.) providing essential services to end-users.

This dependence on Cache for performing business-critical tasks and for developing mission-critical applications could only mean that even a wafer-thin deviation in its performance could cause an enterprise to lose millions. Database administrators are thus faced with the daunting task of ensuring the 24x7 availability of the Cache database and the optimal performance of all its components. The eG Enterprise helps administrators in this task.

2

Chapter 2: Pre-requisites for monitoring an Intersystems Cache Database Server

While the eG agent extracts some of these critical statistics from the SNMP MIB of Cache (eg., metrics related to free space in the database, buffer usage, ECP application and data servers, etc.), for a few other key metrics (eg., metrics related to the console log, resource seizes, lock counts, etc.) the agent executes the **cstat** utility that is provided with Caché. This utility, which is available in the install directory of a Cache instance, provides a wealth of information about the running Caché system.

Where the eG agents needs to contact the SNMP MIB for performance data, make sure that the Windows SNMP service is installed and started either automatically or manually on the Cache host. In addition, ensure the following:

- The **Cache Monitoring Service** should be enabled. The steps for enabling this service are explained in the Section **2.1**.
- Activate critical SNMP base classes, so that eG agent can monitor some of the most significant aspects of Cache performance
- Configure the Cache SNMP agent to start automatically at Cache startup

2.1 Configuring the Intersystems Cache Database to work with the eG Agent

1. Follow the steps given below to enable the **Cache Monitoring Service**:
   - Navigate to the Home -> Security Management -> Services page of the System Management Portal.
   - Click the **%System_Monitor** service in the **Services** page (see Figure 2.1).

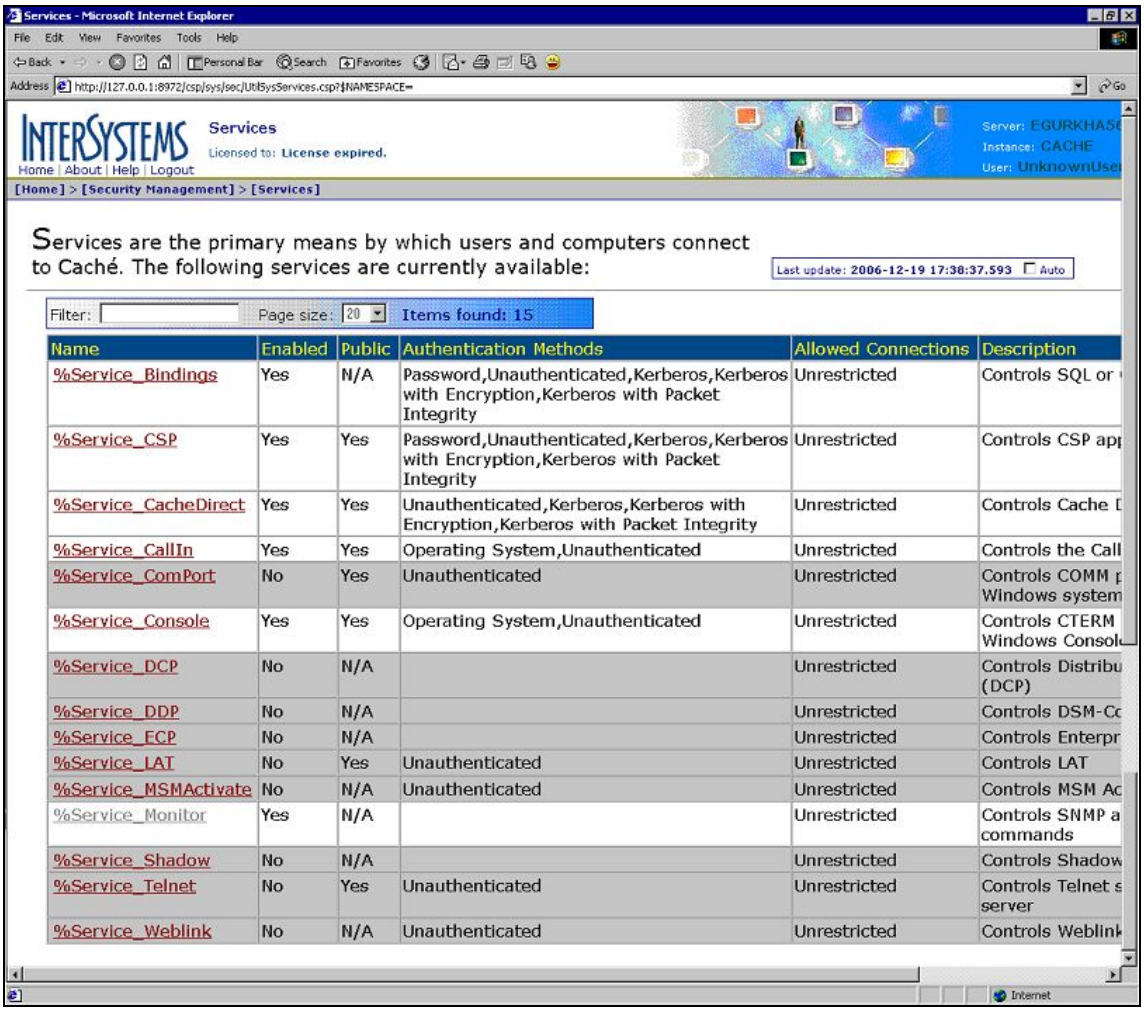

Figure 2.1: The Services page

- Figure 2.2 then appears. The **Service enabled** check box will be deselected, by default. Click on the check box to select it and click the **Save** button to register the changes.

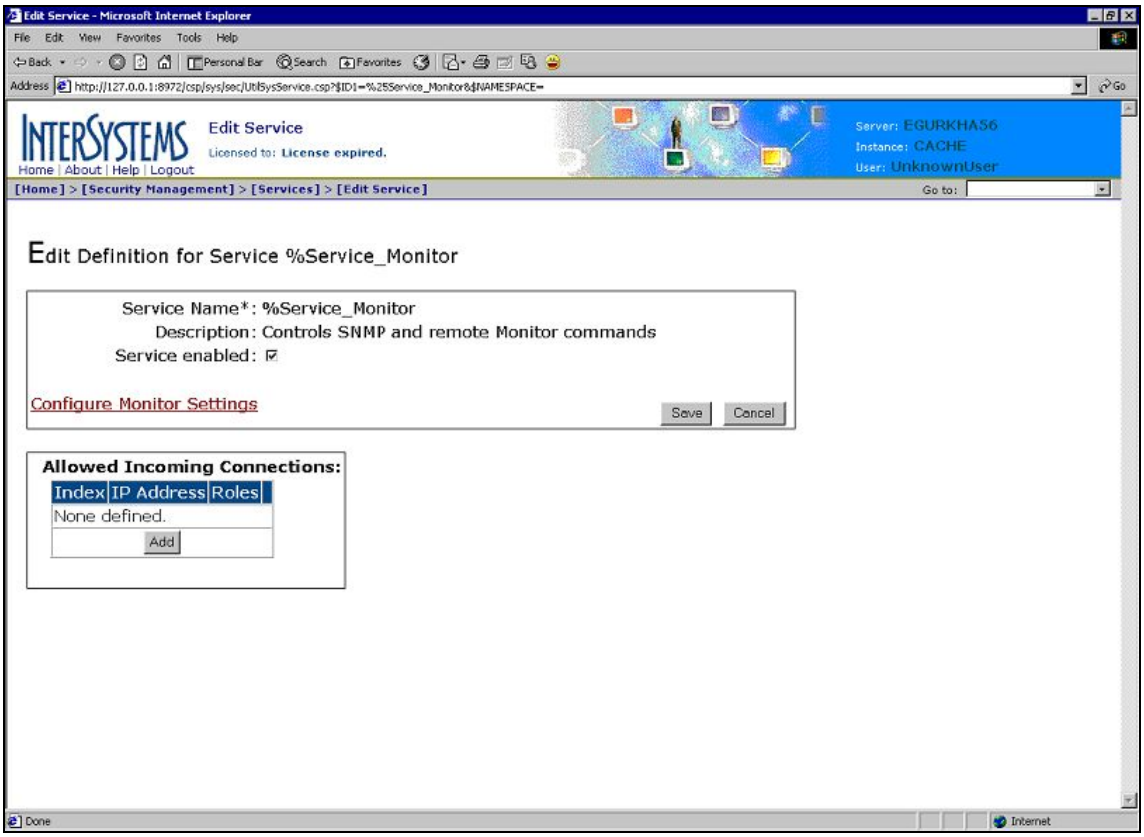

Figure 2.2: Enabling service monitoring

- You will then return to the **Services** page, where you can verify whether the **%System_Monitor** service is enabled or not.

2. To enable the SNMP base classes that monitor critical aspects of Cache performance, do the following:

   - Start the Cache Terminal
   - From the command prompt of the Terminal, issue the command, **DO ^%CD**, to switch to a different namespace.

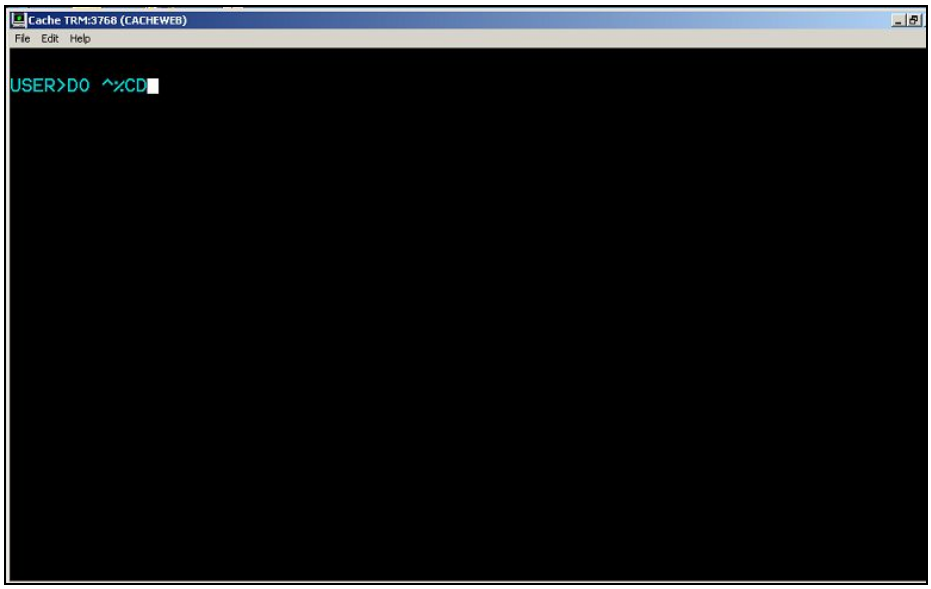

Figure 2.3: Switching to a different namespace

- You will now be prompted for a namespace. Type **%SYS** therein.

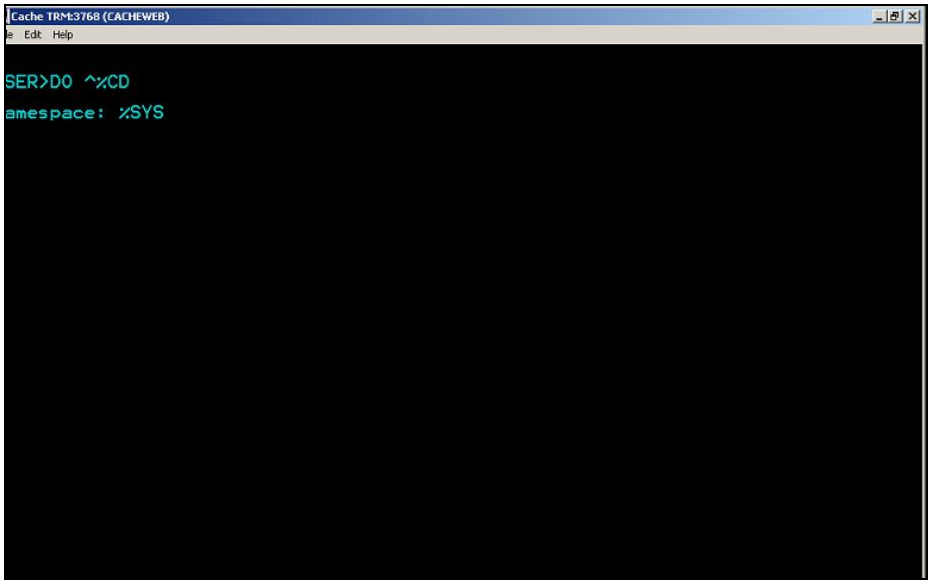

Figure 2.4: Specifying the Namespace to switch to

- Execute the routine **%MONAPPMGR** as indicated by Figure 2.5.

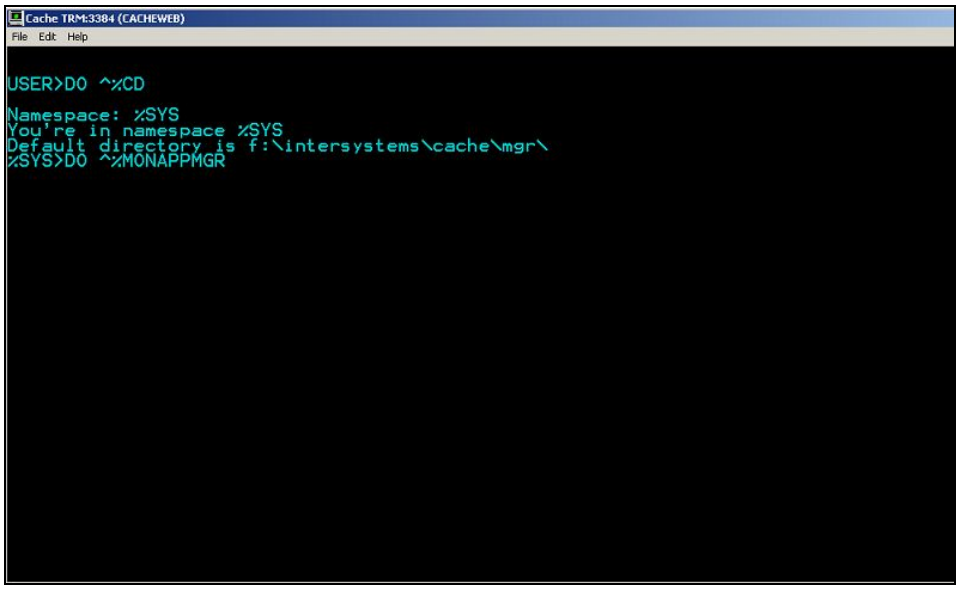

Figure 2.5: Executing the routine %MONAPPMGR

- Upon execution, the routine requests you to select from 4 options. Select the **Manage Monitor Classes** option, by pressing **2**.

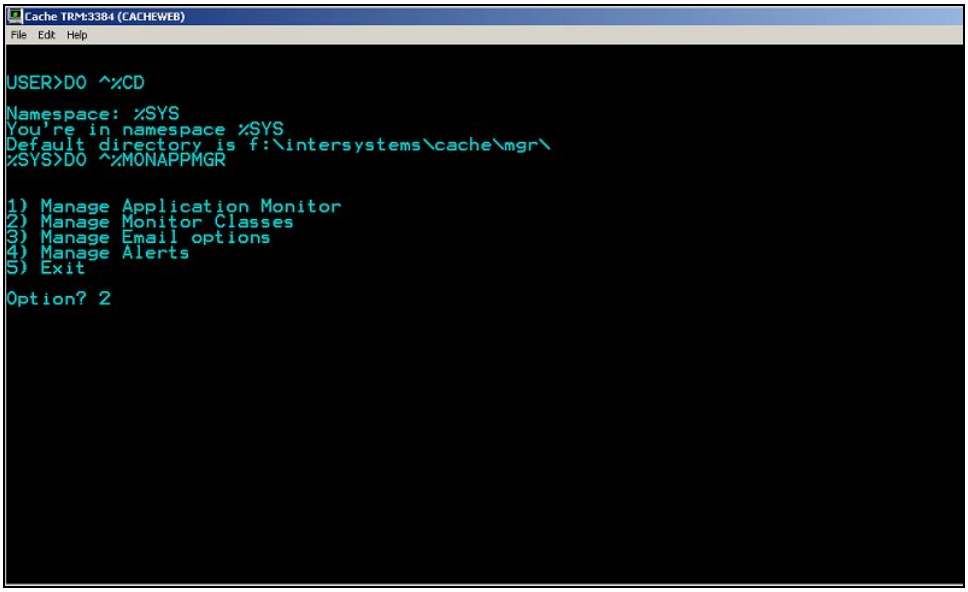

Figure 2.6: Selecting the Manage Monitor Classes option

- You will now be prompted to choose the action that you want to perform on the Monitor Classes. Since we need to activate a monitoring capability (i.e., Freespace monitoring), select the **Activate/Deactivate a Monitor Class** option, by pressing **1** (see Figure 2.7).

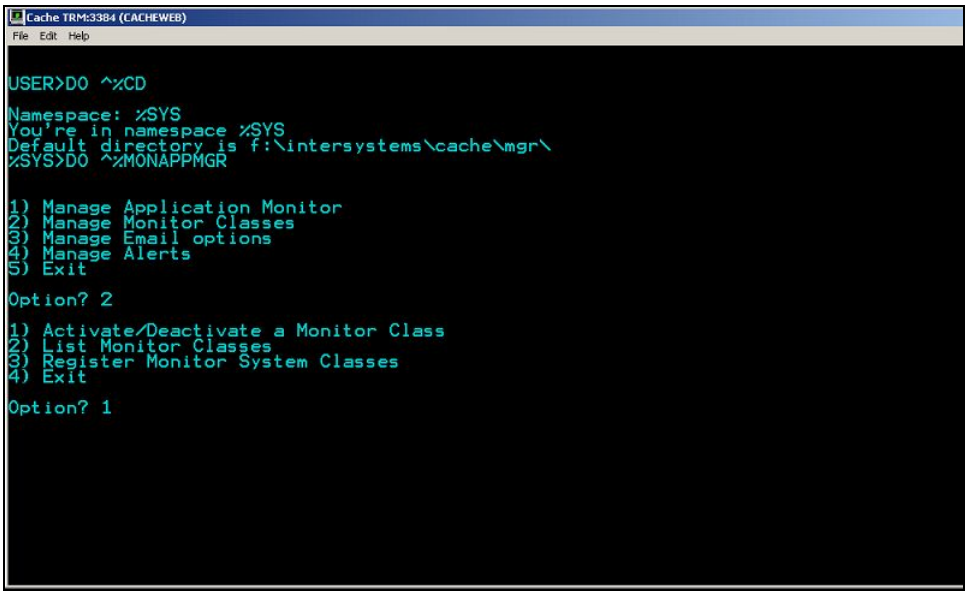

Figure 2.7: Choosing to activate/deactivate a monitor class

- When prompted for the monitor class to activate/deactivate, specify **%Monitor.System.Freespace** as indicated by Figure 2.8. This class monitors the free space on every Cache database.

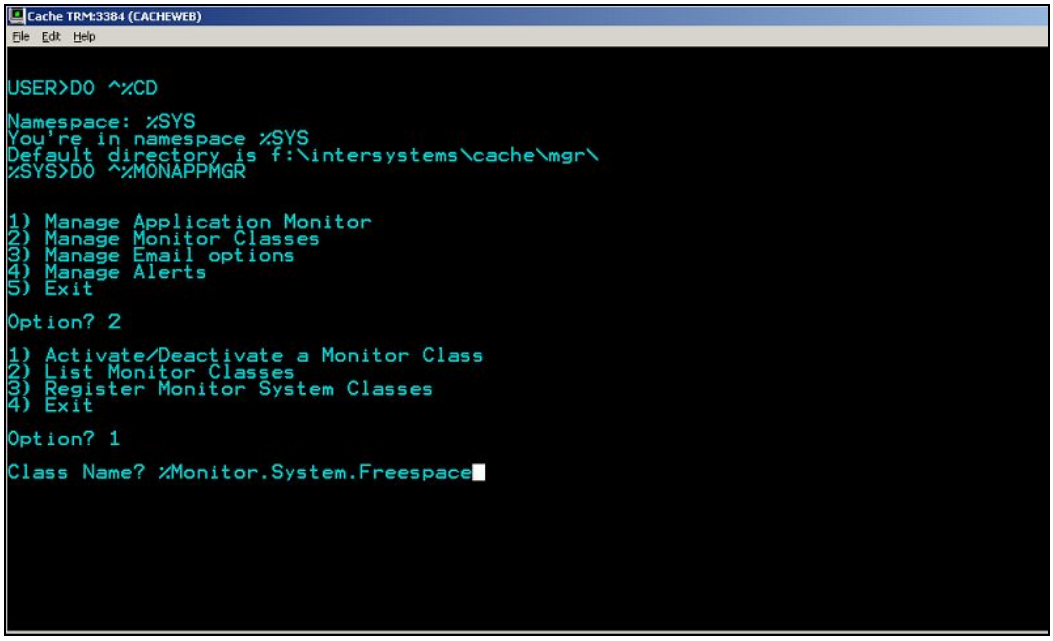

Figure 2.8: Specifying the name of the class to activate/deactivate

- Then, choose to **Activate** the **%Monitor.System.Freespace** class by pressing **A** (see Figure 2.9).

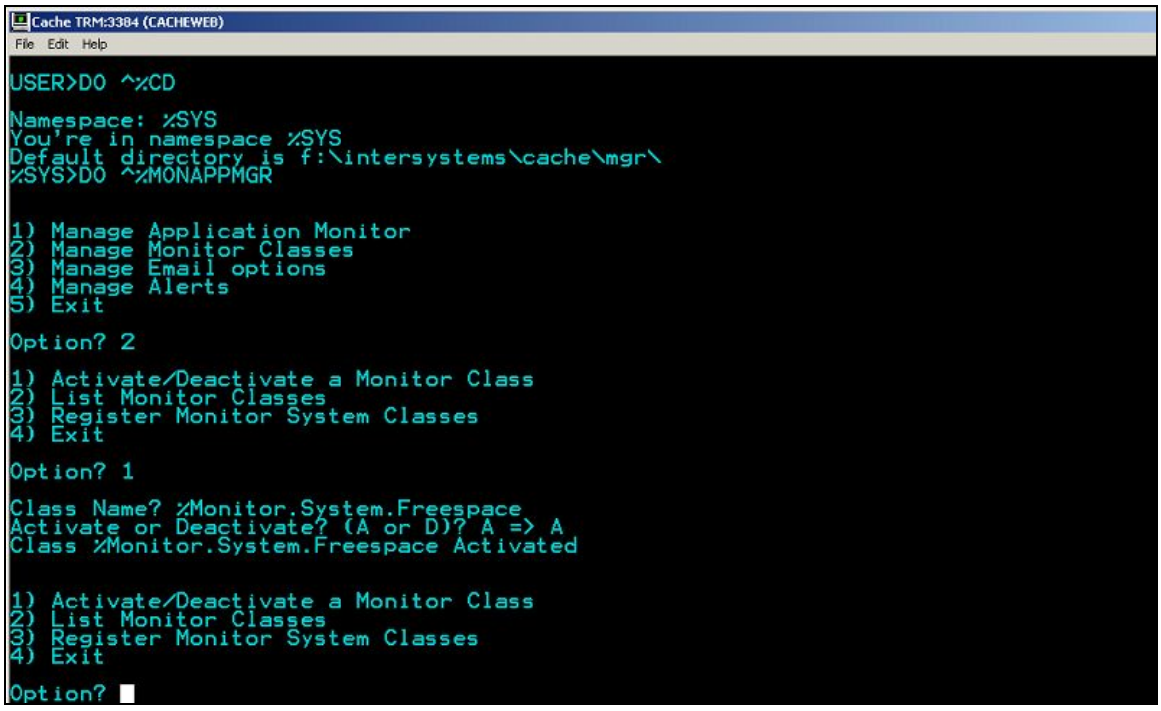

Figure 2.9: Activating the Monitor Class

- Upon successful activation, a message to that effect will be displayed, followed by a list of options. Press **1** this time to activate another class.

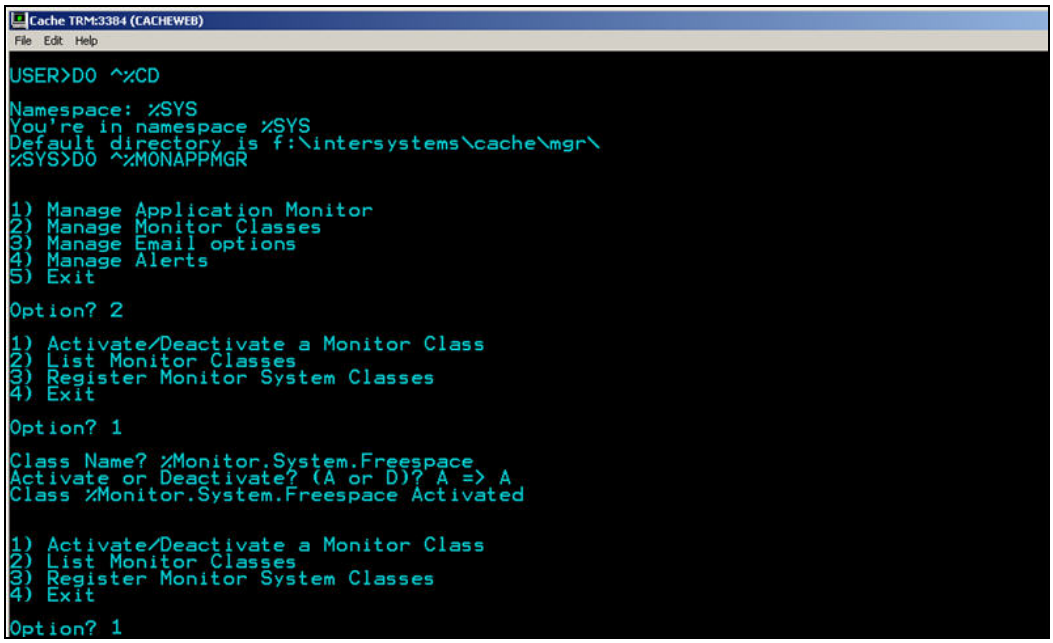

Figure 2.10: Selecting another class for activation

- This time, activate the **%Monitor.System.Database** class so as to enable Cache database monitoring (see Figure 2.11).

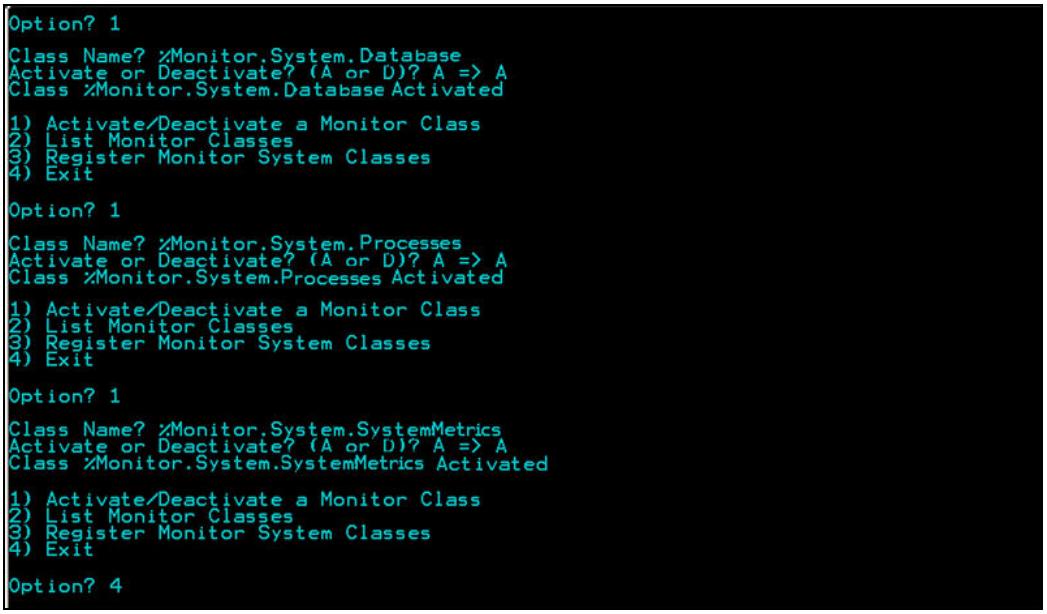

Figure 2.11: Activating the Database, Processes, and SystemMetrics class

- Subsequently, activate the **%Monitor.System.Processes** and **%Monitor.System.SystemMetrics** classes (see Figure 2.11), so that the eG agent is able to extract Cache process-related metrics and system-related metrics.
- Finally, exit the Cache Terminal by selecting option number **4** as indicated by Figure 2.11.

3. To configure the SNMP sub-agent to start automatically on startup, do the following:

- Navigate to the Home -> Configuration -> Monitor Settings page of the System Management Portal (see Figure 2.12).

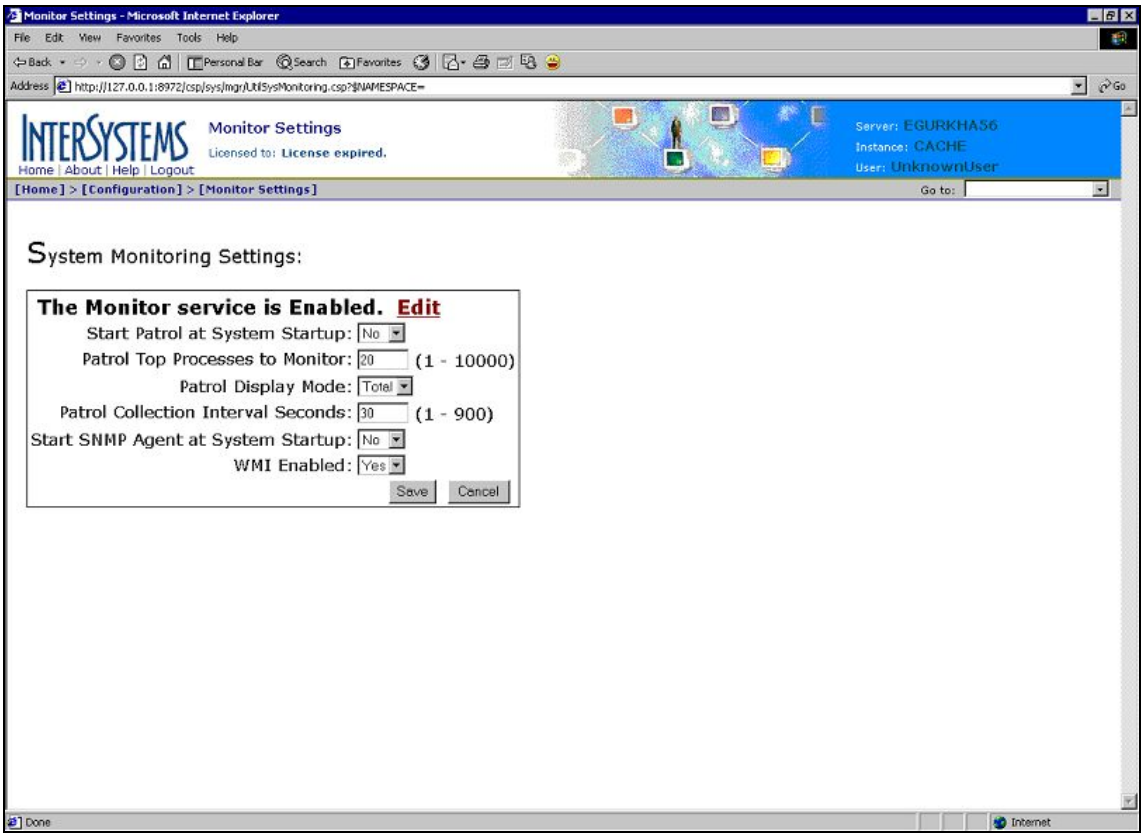

Figure 2.12: The Monitor Settings page of the System Management Portal

- Set the **Start SNMP Agent at System Startup** flag to **Yes** (see Figure 2.13), and click the **Save** button to register the changes.

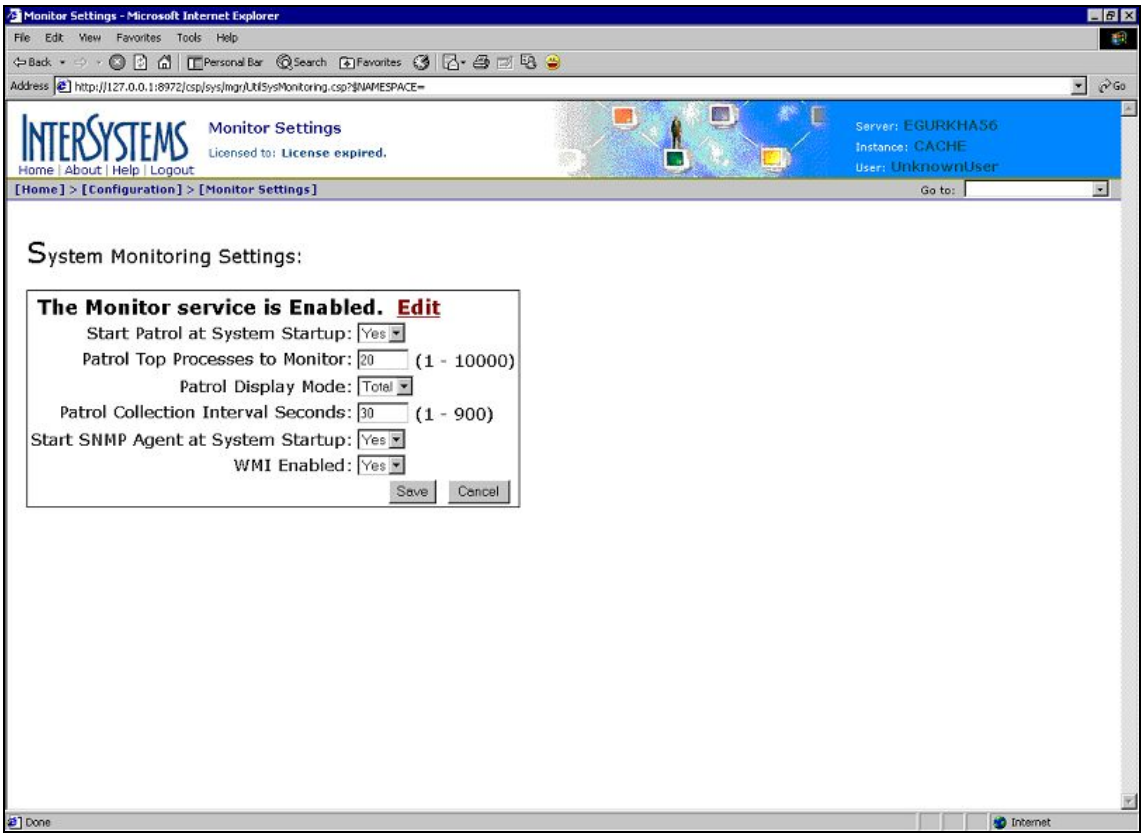

Figure 2.13: Enabling the SNMP Agent to start on system boot

- Next, start the Cache Terminal.
- From the command prompt of the Terminal, issue the command, **DO ^%CD**, to switch to a different namespace.
- You will now be prompted for a namespace. Type **%SYS** therein, as only this namespace is vested with the right to start the SNMP agent service.

```
NAMESPACE: %SYS
```

- The namespace then changes to %**SYS**.

```
Default directory is d:\cache\cachesys\mgr\
```

```
%SYS>
```

- Next, execute the routine that would start the SNMP agent.

```
DO start^SNMP(705,20)
```

Once the SNMP agent starts successfully, the eG agent gathers the required statistics from the SNMP MIB, and reports the performance data so collected to the eG manager.

On the other hand, where the eG agent needs to execute **cstat** for collecting the required metrics, you need to ensure that the corresponding tests are configured with the exact Cache instance name and the correct path to that instance's home directory.

3

Chapter 3: Administering the eG Manager to monitor the Intersystems Cache Database

1. Login to the administrative interface as an administrator (admin).
2. Manually add the Cache Database server using the **COMPONENTS** page (Infrastructure -> Components -> Add/Modify). The eG Enterprise system automatically manages the manually added components.

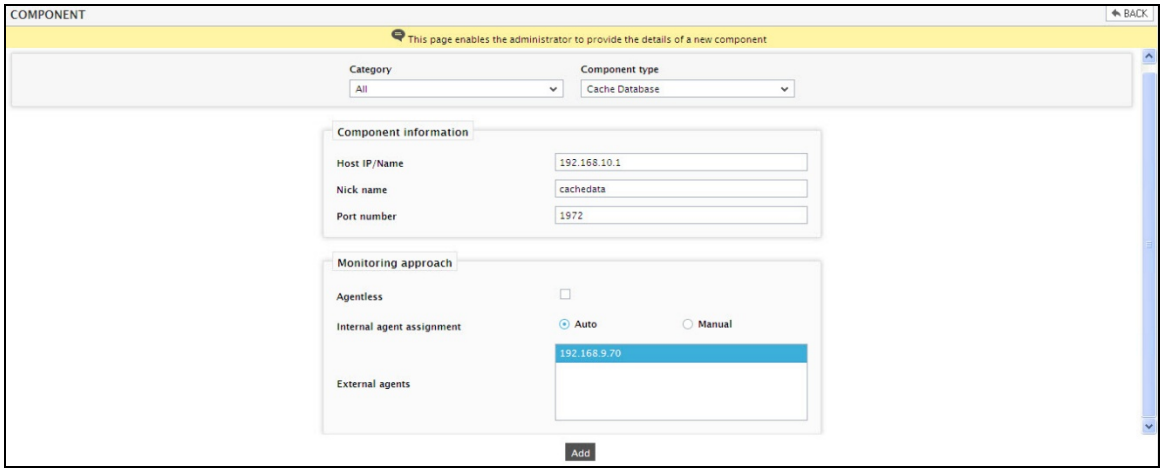

Figure 3.1: Managing a Cache Database server

3. Now, if you try to sign out of the user interface, you will be prompted to configure a series of tests for the managed Cache server (see Figure 3.2).

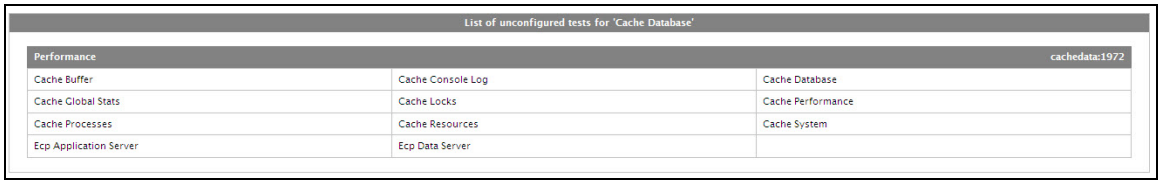

Figure 3.2: A table displaying the list of unconfigured tests for the Cache Database server

4. Click on **Cache Buffer** test to configure it. This test monitors the usage of the buffer pool of every database instance of the Cache database server. To know how to configure the test, refer to Section **4.2.1**.

5. Click the **Update** button to register the changes and try signing out of the admin interface once again. This time, when the list of unconfigured tests reappears, choose to configure the **Cache Console Log** test for the server.
6. This test periodically monitors the console log of every configured cache instance to report the number of normal, severe, and fatal errors encountered by the Cache database server. To know how to configure the test, refer to the Section **4.1.2**.
7. To save the test configuration, click the **Update** button, and then proceed to sign out yet again. This time you will be prompted to configure the **Cache Performance** test. This test monitors the critical determinants of the performance of a Cache database server. To know how to configure the test, refer to the Section **4.3.1**.
8. **Update** the test configuration, and finally, sign out the eG administrative interface.

4

Chapter 4: Monitoring the Intersystems Cache Database

eG Enterprise offers a specialized monitoring model for the Cache database (see Figure 4.1) that monitors the database 24 x 7 and proactively alerts administrators of probable issues in its operations, so that issues are trapped very early and resolved before its too late.

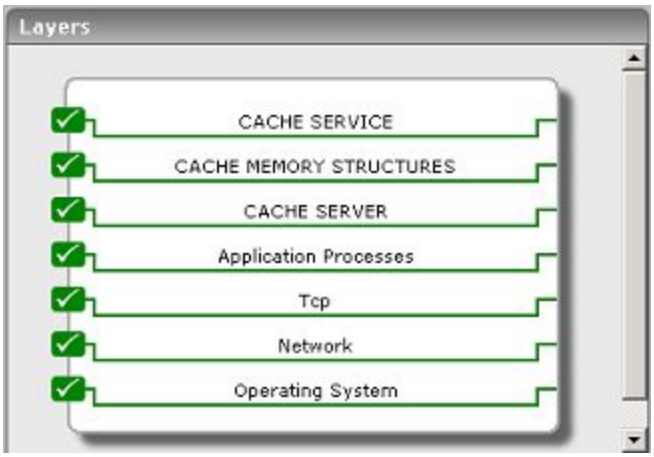

Figure 4.1: Layer model of the Cache database server

Each of the layers depicted by the hierarchical model above, is associated with a wide variety of tests that measure the health of various factors influencing database performance. The performance metrics reported by these tests shed light on the following:

Cache Service/Process Monitoring	• Is the cache instance up and running? • How responsive is the cache database server to client requests? • Are the critical processes running? What is their resource usage?
Cache System/ Database Monitoring	• What is the current status of the database - read-only or read-write? • Is there sufficient free space in the Cache databases? • Are there any severe/fatal errors logged in the console log? • Is license usage optimal?

	• Is data adequately referenced so as to enable easy retrieval? • Is global referencing normal or are there way too many or too little global accesses? • Have too many locks been created on any database instance? If so, what are the lock types? Are there any potentially dangerous lock types? • Are too many processes contending for a lock on the database? • Are resource seizes occurring? If so, what types of seizes are they? • Were any critical changes made to the database, recently? • Was the write daemon able to write all changes to the database? • What is the current status of the write daemon?
Ecp Application/Data Server Monitoring	• Is data transmitted properly from the application and data servers? • Are the global references affecting database performance? • Do the requests from data servers affect traffic?
Cache Performance/Buffer Monitoring	• What is the workload of the data server in terms of lines & routine executed? • Has sufficient memory been allocated to the buffers to reduce physical disk IO? • Have the routine and database caches been adequately sized?

4.1 The Cache Server Layer

The tests associated with the **Cache Server** layer reports a wide variety of statistics related to the following:

- Space usage of each of the Cache databases
- The number and nature of errors logged in the console log file
- Key system and application processes executing on the Cache instances
- Global activity and disk I/O on configured Cache instances
- License usage by the Cache server
- Health of the ECP application and data servers

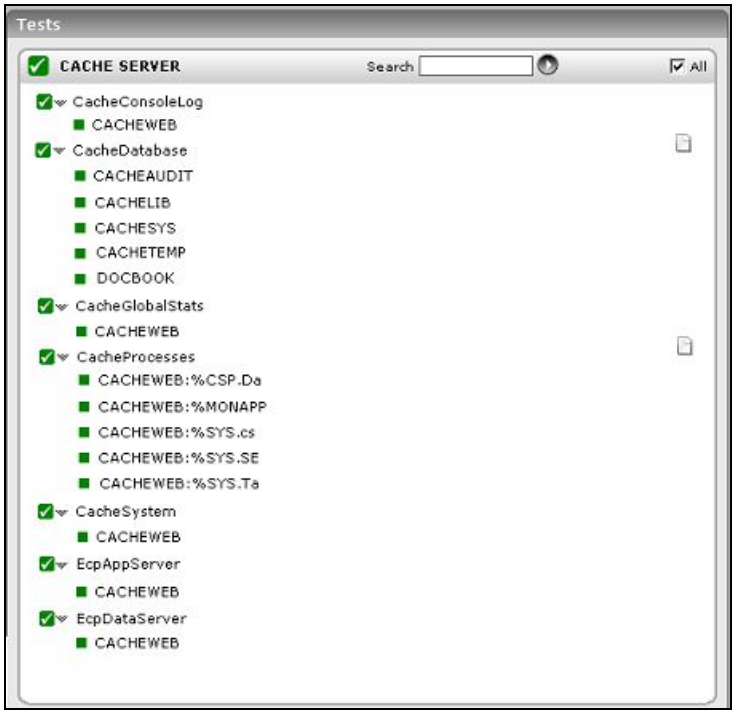

Figure 4.2: Tests associated with the Cache Server layer

4.1.1 Cache Database Test

This test monitors the space usage of every Cache database using SNMP.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every database on the Cache database server

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.
3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION** list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION** chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against the **USERNAME** parameter.
8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.
9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.
10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.

11. **AUTHTYPE** – This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE** list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:
    - **MD5** – Message Digest Algorithm
    - **SHA** – Secure Hash Algorithm
12. **ENCRYPTFLAG** – This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG** is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.
13. **ENCRYPTTYPE** – If the **ENCRYPTFLAG** is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE** list. SNMP v3 supports the following encryption types:
    - **DES** – Data Encryption Standard
    - **AES** – Advanced Encryption Standard
14. **ENCRYPTPASSWORD** – Specify the encryption password here.
15. **CONFIRM PASSWORD** – Confirm the encryption password by retyping it here.
16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.
17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Read only status:	Indicates the current status of the database - whether read-only or not	Boolean	The value '1' indicates that the database is currently in the "read-only" status, and the value '0'

Measurement	Description	Measurement Unit	Interpretation
			indicates that the database can be read and written into - i.e., 'read-write' status.
Current database size:	The current size of this database	KB	
Free space:	The amount of unused space currently available in this database	KB	
Percent free space:	The percentage of free space currently available in this database	Percent	Ideally, this value should be very high.

4.1.2 Cache Console Log Test

The primary information source for monitoring Caché is the console log (cconsole.log). Caché reports general messages, system errors, certain operating system errors, and network errors through an operator console facility. The console log file is a plain text file and may be viewed with any editor or text viewer. It is found in the MGR subdirectory of the location where Caché was installed. The CacheConsoleLogTest periodically monitors the console log of every configured Cache instance to report the number of normal, severe, and fatal errors encountered by the Cache database server.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every InstanceName configured

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening

4. **INSTANCEDIRECTORY** - Typically, the Cache console log file will be available in the install directory of a Cache instance. Therefore, in the **INSTANCEDIRECTORY** text box, specify the name of the instance being monitored and the install directory that holds the Cache console log file of that instance, in the following format: *InstanceName:InstallDirectory*. In case you want to monitor the console log files pertaining to multiple Cache instances, then provide a comma-separated list of *InstanceName:InstallDirectory* pairs in the **INSTANCEDIRECTORY** text box. For example: *CACHEWEB:d:\Intersystems\CacheWeb,CACHE2:d:\Intersystems\Cache2*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Total errors:	Indicates the number of normal errors/warnings that currently occurred on this Cache instance	Number	The detailed diagnosis of this measure, when enabled, provides the detailed description for each of the normal errors that occurred on this Cache instance. A typical normal error/warning would be, *****WARNING***** : **THE CURRENT CONFIGURATION IS NOT THE SAME AS WHEN CACHE WAS LAST SHUT DOWN**.
Severe errors:	Indicates the number of severe errors that occurred currently on this Cache instance	Number	A typical severe error would be: SNMP server failed to start: error signaling Windows SNMP agent. Some <NETWORK> errors are essentially fatal error conditions. Typically, if email notification of errors is setup on a monitored Cache database server, then, whenever a severe/fatal error is logged in the cconsole.log file, Cache automatically sends out emails to concerned administrators alerting them to the problem condition and enabling them to

Measurement	Description	Measurement Unit	Interpretation
			initiate corrective action. Starting with v5.1 of the Cache database server, if no email notification mechanism is setup, then an additional log file, alerts.log, is created whenever a severe or fatal error is detected in the cconsole.log file. This log contains only the severe and fatal error messages. You can check this log file for details pertaining to such errors. Alternatively, if you enable the detailed diagnosis capability of the eG Enterprise suite, then the complete description of the severe/fatal errors will be available to you in the eG monitoring console itself.
Fatal errors:	Indicates the current number of fatal errors on this Cache instance	Number	

4.1.3 Cache Global Stats Test

This test gathers global activity statistics and displays a variety of information about disk I/O operations.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every InstanceName configured

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening

4. **INSTANCEDIRECTORY** - Typically, the Cache console log file will be available in the install directory of a Cache instance. Therefore, in the **INSTANCEDIRECTORY** text box, specify the name of the instance being monitored and the install directory that holds the Cache console log file of that instance, in the following format: *InstanceName:InstallDirectory*. In case you want to monitor the console log files pertaining to multiple Cache instances, then provide a comma-separated list of *InstanceName:InstallDirectory* pairs in the **INSTANCEDIRECTORY** text box. For example: *CACHEWEB:d:\Intersystems\CacheWeb,CACHE2:d:\Intersystems\Cache2*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Global references:	Indicates the logical count of accesses to globals, including Sets, Kills, $Data, $Order, $Increment, $Query, and global references in expressions, during the last measurement period.	Number	If this number significantly jumps up or declines from the normal, there may be an application issue to research.
Global update references:	Indicates the number of calls to a routine during the last measurement period.	Number	
Routine calls:	Indicates the current number of fatal errors on this Cache instance.	Number	
Routine buffer loads and saves:	Indicates the total number of routine loads and saves as a result of ZLoad, ZSave, and running routines, during the last measurement period.	Number	In a well-tuned environment, this number increases slowly, since most routine loads are satisfied by the routine cache memory without accessing the disk. Each routine load or save transfers up to 32 KB of data (64 KB for Unicode).

Measurement	Description	Measurement Unit	Interpretation
Block I/O reads:	Indicates the number of physical database blocks (2- KB or 8- KB) read from disk during the last measurement period for both global and routine references.	Number	A high value for this measure indicates that direct disk accesses are high. In such a case your database might require some fine-tuning. Consider resizing your buffer pool to increase buffer accesses and reduce data retrievals from the disk.
Block I/O writes:	Indicates the number of physical database blocks (2-KB or 8-KB) written to disk during the last measurement period for both global and routine references.	Number	
WIJ I/O writes:	Indicates the number of 64- KB journal blocks written to the journal file during the last measurement period.	Number	
Logical block requests:	Indicates the number of database blocks read by the global database code during the last measurement period.	Number	In a well-tuned environment, many of these reads are satisfied without disk access.

4.1.4 Cache Processes Test

This test auto-discovers the processes on every configured Cache instance, and reports the number of instances of each process that are currently running; this way, the test verifies whether critical system and application processes are running or not.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every *InstanceName:Processname* pair discovered

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening
4. **INSTANCEDIRECTORY** - Typically, the Cache console log file will be available in the install directory of a Cache instance. Therefore, in the **INSTANCEDIRECTORY** text box, specify the name of the instance being monitored and the install directory that holds the Cache console log file of that instance, in the following format: *InstanceName:InstallDirectory*. In case you want to monitor the console log files pertaining to multiple Cache instances, then provide a comma-separated list of *InstanceName:InstallDirectory* pairs in the **INSTANCEDIRECTORY** text box. For example: *CACHEWEB:d:\Intersystems\CacheWeb,CACHE2:d:\Intersystems\Cache2*.
5. To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

   The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

   - The eG manager license should allow the detailed diagnosis capability
   - Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Process count:	Indicates the number of instances of this Cache process currently running on this Cache instance.	Number	The detailed diagnosis of this measure, if enabled, provides the complete process details such as the namespace to which the process belongs, the lines of code executed by the process, the

Measurement	Description	Measurement Unit	Interpretation
			number of global references made by the process, and the device to which the process belongs.

4.1.5 Cache System Test

Using this test, you can monitor,

- the users to the database instance
- the database and routine caching activities performed by the Cache database instance
- the license usage of the database instance
- errors (if any) that have been logged

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every instance of the Cache database server being monitored

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.
3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION** list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION** chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP

version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against the **USERNAME** parameter.

8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.

9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.

10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.

11. **AUTHTYPE** – This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE**list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:

    - **MD5** – Message Digest Algorithm
    - **SHA** – Secure Hash Algorithm

12. **ENCRYPTFLAG**– This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG**is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.

13. **ENCRYPTTYPE**– If the **ENCRYPTFLAG**is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE**list. SNMP v3 supports the following encryption types:

    - **DES** – Data Encryption Standard

- **AES** – Advanced Encryption Standard

14. **ENCRYPTPASSWORD**– Specify the encryption password here.
15. **CONFIRM PASSWORD**– Confirm the encryption password by retyping it here.
16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.
17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Users connected to Cache:	Indicates the number of users who are currently using this database instance.	Number	This is a good indicator of server load.
Routine cache size:	Indicates the current size of the routine cache	MB	Ideally, this value should be high. A well-sized routine cache can serve a large number of routine loads, thereby considerably reducing direct disk accesses.
Database cache size:	Indicates the current size of the database cache.	MB	Ideally, this value should be high. A well- sized database cache can serve a large number of requests, thereby considerably reducing direct disk accesses.
Licenses in use in Cache:	Indicates the number of licenses currently used by this database instance.	Number	

Measurement	Description	Measurement Unit	Interpretation
Peak license usage:	Indicates the high water-mark of the license usage for this database instance.	Number	An abnormally high value could be a cause for concern, and could hence require further investigation. Alternatively, this could also be an effective indicator of the popularity of the database instance, and might have to be considered while planning the future license requirements.
Error detected in Cache:	Indicates the last 'severe' error message logged in the console log for this Cache instance.	Boolean	

4.1.6 Ecp Application Server Test

One of the most powerful and unique features of Caché is the ability to efficiently distribute data and application logic among a number of server systems. The underlying technology behind this feature is the Enterprise Cache Protocol (ECP): a distributed data caching architecture that manages the distribution of data and locks among a heterogeneous network of server systems. Unlike other "multi-tier" architectures, ECP is primarily a configuration option. That is, you do not have to use special code or development techniques to create distributed database applications.

Furthermore, the architecture and operation of ECP is conceptually simple. ECP provides a way to efficiently share data, locks, and executable code among multiple Caché systems. Data and code are stored remotely, but are cached locally to provide efficient access with minimal network traffic.

An ECP configuration consists of a number Caché systems that are visible to one another across a TCP/IP-based network. There are two roles a Caché system can play in an ECP configuration:

- **ECP Data Server** - a Caché system that is providing data for one or more ECP application server systems.
- **ECP Application Server** - a Caché system that is consuming data provided by one or more ECP data server systems.

A Caché system can simultaneously act as both an ECP data server and an ECP application server. However, one Caché instance cannot act as an ECP data server for the data it receives as an application server of another ECP data server.

In an ECP configuration, each ECP application server is responsible for the following:

- Establishing connections to a specific ECP data server whenever an application requests data that is stored on that server.
- Tracking the status of all connections to ECP data servers. If a connection is broken, or encounters any trouble, the ECP application server attempts to recover the connection.
- Maintaining, in its cache, data retrieved across the network

This test monitors the caching and data management functions performed by the ECP application server.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every instance of the Cache database server being monitored

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.
3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION** list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION** chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP

configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against the **USERNAME** parameter.

8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.

9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.

10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.

11. **AUTHTYPE**– This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE**list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:

    - **MD5** – Message Digest Algorithm
    - **SHA** – Secure Hash Algorithm

12. **ENCRYPTFLAG**– This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG**is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.

13. **ENCRYPTTYPE**– If the **ENCRYPTFLAG**is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE**list. SNMP v3 supports the following encryption types:

    - **DES** – Data Encryption Standard
    - **AES** – Advanced Encryption Standard

14. **ENCRYPTPASSWORD**– Specify the encryption password here.

15. **CONFIRM PASSWORD**– Confirm the encryption password by retyping it here.
16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.
17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Global references not from cache:	Indicates the number of global references that were not served by the cache since the last measurement period.	Number	Ideally, this value should be low. A high value of this measure could indicate that many global references were served by directly accessing remote data, thus increasing the network overheads.
Data sent by ECP Application server:	Indicates the amount of data sent by the application server since the last measurement period.	KB	
Data received by ECP Application server:	Indicates the data received by the application server since the last measurement period.	KB	

4.1.7 Ecp Data Server Test

One of the most powerful and unique features of Caché is the ability to efficiently distribute data and application logic among a number of server systems. The underlying technology behind this feature is the Enterprise Cache Protocol (ECP): a distributed data caching architecture that manages the

distribution of data and locks among a heterogeneous network of server systems. Unlike other "multi-tier" architectures, ECP is primarily a configuration option. That is, you do not have to use special code or development techniques to create distributed database applications.

Furthermore, the architecture and operation of ECP is conceptually simple. ECP provides a way to efficiently share data, locks, and executable code among multiple Caché systems. Data and code are stored remotely, but are cached locally to provide efficient access with minimal network traffic.

An ECP configuration consists of a number Caché systems that are visible to one another across a TCP/IP-based network. There are two roles a Caché system can play in an ECP configuration:

- **ECP Data Server** - a Caché system that is providing data for one or more ECP application server systems.
- **ECP Application Server** - a Caché system that is consuming data provided by one or more ECP data server systems.

A Caché system can simultaneously act as both an ECP data server and an ECP application server. However, one Caché instance cannot act as an ECP data server for the data it receives as an application server of another ECP data server.

In an ECP configuration, each ECP data server is responsible for the following:

- Storing data in its local database
- Maintaining the coherency of the various ECP application server system database caches so that application servers do not see stale data
- Managing the distribution of locks across the network

This monitors how well the data server manages data.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every instance of the Cache database server being monitored

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.

3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION**list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION**chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against the **USERNAME** parameter.
8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.
9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.
10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.
11. **AUTHTYPE**– This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE**list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:

- **MD5** – Message Digest Algorithm
- **SHA** – Secure Hash Algorithm

12. **ENCRYPTFLAG**– This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG**is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.
13. **ENCRYPTTYPE**– If the **ENCRYPTFLAG**is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE**list. SNMP v3 supports the following encryption types:
    - **DES** – Data Encryption Standard
    - **AES** – Advanced Encryption Standard
14. **ENCRYPTPASSWORD**– Specify the encryption password here.
15. **CONFIRM PASSWORD**– Confirm the encryption password by retyping it here.
16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.
17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Global references returned by ECP data server:	Indicates the number of global references that were served by the data stored in the ECP data server.	Number	Ideally, this value should be low. A high value of this measure could indicate that the ECP application server cache was ineffective in servicing many requests, and has hence redirected them to data server, thereby consuming a lot of network bandwidth.

Measurement	Description	Measurement Unit	Interpretation
Requests received by ECP data server:	Indicates the number of requests received by the data server since the last measurement period.	Number	
Blocks sent by ECP data server:	Indicates the number of blocks of data sent by the data server.	Number	
Data sent from ECP data server:	Indicates the data sent by the data server since the last measurement period.	KB	
Data received by ECP data server:	Indicates the data received by the data server since the last measurement period.	KB	

4.2 The Cache Memory Structures Layer

Using the tests mapped the **Cache Memory Structures** layer, administrators can assess how well the Cache database server managers its buffer pools and locks.

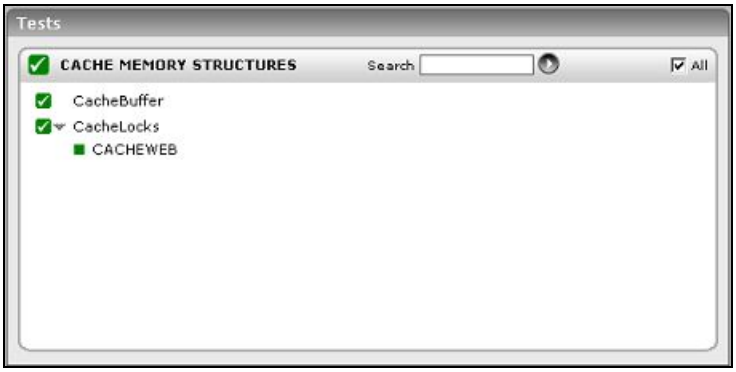

Figure 4.3: The tests associated with the Cache Memory Structures layer

4.2.1 Cache Buffer Test

Caché maintains a buffer pool — an in-memory cache of frequently referenced data blocks — to reduce the cost of fetching blocks from disk. By default, 16 KB is the memory allocated to the

database cache. Close monitoring of the buffer pool is essential to determine the adequacy of the memory allocation to the cache. By increasing the memory allocation, administrators can considerably increase cache hits, reduce disk accesses, and consequently, save the processing overheads.

The CacheBuffer test monitors the usage of the buffer pool of every database instance of the Cache database server. In the process, the test also observes the behavior of the interactive and batch buffer queues. Any Cache process created in the Cache database is set to execute either in the 'interactive' or in the 'batch' mode. Since an 'interactive' process expects user inputs, it has to be run in the foreground and is also, resource-intensive. A 'batch' process on the other hand can only be run in the background, and uses less resources.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every database on the Cache database server

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.
3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION** list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION** chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against

the **USERNAME** parameter.

8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.
9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.
10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.
11. **AUTHTYPE** – This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE**list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:
    - **MD5** – Message Digest Algorithm
    - **SHA** – Secure Hash Algorithm
12. **ENCRYPTFLAG**– This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG**is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.
13. **ENCRYPTTYPE**– If the **ENCRYPTFLAG**is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE**list. SNMP v3 supports the following encryption types:
    - **DES** – Data Encryption Standard
    - **AES** – Advanced Encryption Standard
14. **ENCRYPTPASSWORD**– Specify the encryption password here.
15. **CONFIRM PASSWORD**– Confirm the encryption password by retyping it here.
16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.

17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Buffer size:	Indicates the current size of the buffer	Bytes	A well-tuned environment is one where the buffer is sufficiently sized. A low value of this measure is unhealthy, as light weight buffers force a large number of direct disk accesses causing the database to incur excessive processing overheads.
Buffer count:	Indicates the number of buffers of buffer size	Number	
Batch queue offset:	Indicates the current offset to the start of the LRU (least recently used) queue.	Number	
Interactive buffers:	Indicates the number of buffers currently in the interactive portion of the LRU queue	Number	
Max interactive buffers:	Indicates the maximum number of buffers in the interactive portion of the LRU queue.	Number	
Requeue limit interactive:	Indicates the threshold for requeuing an	Number	

Measurement	Description	Measurement Unit	Interpretation
	interactive buffer.		
Requeue limit batch:	Indicates the threshold for requeueing a batch buffer.	Number	
Interactive write queue:	Indicates the number of interactive buffers in the current write cycle.	Number	
Buffers write queue:	Indicates the number of buffers requeued in the batch LRU queue.	Number	
Write queue wait max:	Indicates the threshold for waking the interactive write daemon.	Number	
Available interactive buffers:	Indicates the current number of available interactive buffers.	Number	Ideally, this value should be high. If the value is low, then the buffer might be unable to serve all interactive job requests, thus forcing direct disk accesses. This could ultimately increase processing overheads.
Minimum interactive buffers:	Indicates the minimum number of buffers in the interactive portion of the LRU queue.	Number	
Buffer usage:	Indicates the percentage of buffer memory utilized.	Percent	A value close to 100% is a cause of concern, as it indicates that the buffer pool is rapidly filling up. This implies that there may not be enough space for additional buffers in the buffer pool; this may force

Measurement	Description	Measurement Unit	Interpretation
			direct disk accesses, which are I/O-intensive. For optimal performance therefore, the value of this measure should be low.

4.2.2 Cache Locks Test

Caché locks are created when a Caché process issues a Lock command on a Caché entity, such as a local or global variable, as long as the entity is not already locked by another process. Periodic monitoring of the locking activity on every Cache instance is imperative to ensure that no application-critical Cache entity is locked, as this can sometimes cause serious application errors. Similarly, the number of locks used by a Cache system can also impact Cache performance. Besides the number, it would be good practice to keep an eye out for the lock type too, so that potentially dangerous lock types are detected and released in time. The CacheLocks test that the eG agent executes on a Cache server serves all the above-mentioned purposes. This test reports the number of locks on every Cache instance, and additionally reveals the lock type, reference, and owner of the lock, so as to aid further diagnosis.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every InstanceName configured

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening
4. **INSTANCEDIRECTORY** - Typically, the Cache console log file will be available in the install directory of a Cache instance. Therefore, in the **INSTANCEDIRECTORY** text box, specify the name of the instance being monitored and the install directory that holds the Cache console log file of that instance, in the following format: *InstanceName:InstallDirectory*. In case you want to monitor the console log files pertaining to multiple Cache instances, then provide a comma-separated list of *InstanceName:InstallDirectory* pairs in the **INSTANCEDIRECTORY** text box. For example: *CACHEWEB:d:\Intersystems\CacheWeb,CACHE2:d:\Intersystems\Cache2*.

5. To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

   The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

   - The eG manager license should allow the detailed diagnosis capability
   - Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Lock entries:	Indicates the current number of locks on this Cache instance.	Number	If the value of this measure is very high or is increasing consistently, then you might have to enlarge the size of the lock table. Also, you might want to view the detailed diagnosis of this measure, so that you can determine the process that is holding each of the locks, the lock reference, and most importantly, the lock type. An 'incremental lock', for instance, is potentially dangerous because it can lead to a situation known as "deadlock". This situation occurs when two processes each assert an incremental lock on a variable already locked by the other process. Because the attempted locks are incremental, the existing locks are not released. As a result, each process hangs while waiting

Measurement	Description	Measurement Unit	Interpretation
			for the other process to release the existing lock.

The lock type/mode can be any one of the following:

Type of Lock	Description
Exclusive	Exclusive lock mode
Shared	Share lock mode
LockZA	ZALLOCATE lock mode
WaitLock	Waiting for exclusive lock mode
WaitShare	Waiting for share lock mode
WaitLockZA	Waiting for ZALLOCATE lock mode
LockPending	Exclusive lock pending, waiting for server to grant the exclusive lock
SharePending	Share lock pending, waiting for server to grant the share lock
DelockPending	Delock pending, waiting for server to release the lock
Lost	Lock lost due to network reset

4.3 The Cache Service Layer

The **Cache Service** layer evaluates how well the Cache server performs critical operations such as global referencing, journaling, locking and unlocking, resource seizing, etc. Besides, external tests are also included in this layer to determine the availability and responsiveness of the Cache.

Figure 4.4: The tests associated with the Cache Service layer

4.3.1 Cache Performance Test

This test monitors the critical determinants of the performance of a Cache database server. This includes:

- Code processing speed
- Routine loads and saves
- Global jobs
- Logical block accesses
- Journal entries
- Locking activity

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every *InstanceName* configured

Configurable parameters for the test

1. **TEST PERIOD** - How often should the test be executed.
2. **HOST** – The IP address of the Cache Database Server.
3. **PORT** – The port on which the server is listening.
4. **SNMPPORT** – The port at which the Cache Database Server exposes its SNMP MIB; the default is 161.
5. **SNMPVERSION**– By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the **SNMPVERSION** list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
6. **SNMPCOMMUNITY** – The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the **SNMPVERSION** chosen is **v3**, then this parameter will not appear.
7. **USERNAME**– This parameter appears only when **v3** is selected as the **SNMPVERSION**. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly

secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against the **USERNAME** parameter.

8. **CONTEXT** – This parameter appears only when v3 is selected as the **SNMPVERSION**. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the *SNMPEngineID* value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a *contextName*). If the **USERNAME** provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the **USERNAME** in the **CONTEXT** text box. By default, this parameter is set to *none*.
9. **AUTHPASS**– Specify the password that corresponds to the above-mentioned **USERNAME**. This parameter once again appears only if the **SNMPVERSION** selected is **v3**.
10. **CONFIRM PASSWORD**– Confirm the **AUTHPASS** by retyping it here.
11. **AUTHTYPE**– This parameter too appears only if **v3** is selected as the **SNMPVERSION**. From the **AUTHTYPE**list box, choose the authentication algorithm using which SNMP v3 converts the specified **USERNAME** and **PASSWORD** into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options:
    - **MD5** – Message Digest Algorithm
    - **SHA** – Secure Hash Algorithm
12. **ENCRYPTFLAG**– This flag appears only when **v3** is selected as the **SNMPVERSION**. By default, the eG agent does not encrypt SNMP requests. Accordingly, the **ENCRYPTFLAG**is set to **NO** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **YES** option.
13. **ENCRYPTTYPE**– If the **ENCRYPTFLAG**is set to **YES**, then you will have to mention the encryption type by selecting an option from the **ENCRYPTTYPE**list. SNMP v3 supports the following encryption types:
    - **DES** – Data Encryption Standard
    - **AES** – Advanced Encryption Standard
14. **ENCRYPTPASSWORD**– Specify the encryption password here.
15. **CONFIRM PASSWORD**– Confirm the encryption password by retyping it here.

16. **TIMEOUT** - Specify the duration (in seconds) within which the SNMP query executed by this test should time out in the **TIMEOUT** text box. The default is 10 seconds.
17. **DATA OVER TCP** – By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set the **DATA OVER TCP** flag to **Yes**. By default, this flag is set to **No**.
18. **INSTANCENAME** – By default, this is set to *All*, indicating that all Cache instances will be monitored. To monitor specific instances, provide a comma-separated list of Cache names.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Lines executed:	Indicates the number of lines of code per second that were executed on the server.	Lines/Sec	This serves as a good measure of the processing capability of the database server.
Routine loads/saves:	Indicates the rate at which routine loads and saves occurred.	Routines/Sec	Routines are small programs that run within a Cache database server. In a well- tuned environment, this number increases slowly, since most routine loads are satisfied by the routine cache memory without accessing the disk. Each routine load or save transfers up to 32 KB of data.
New global references:	Indicates the rate of at which new global references were set in this database instance.	Refs/Sec	A global is a named multidimensional array that is used for storing data in a physical Caché database. Data can be spread across many globals that exist in different databases on the same system or remote systems. Using a

Measurement	Description	Measurement Unit	Interpretation
			namespace, you can group closely related globals and databases, so that data can be easily referenced. Cross-referencing of data ensures better data storage and efficient query execution.
New global sets:	Indicates the rate at which new globals were created in this database instance.	Sets/Sec	This measure considers the rate of Set operations performed on this database instance since the last measurement period.
New global kills:	Indicates the rate at which new globals were killed in this database instance.	Kills/Sec	This measure considers the number of kill operations per second performed on this database instance since the last measurement period.
Logical database block reads:	Indicates the rate at which data blocks were read from the disk.	Blocks/Sec	Globals are stored on disk within a series of data blocks; the size of each block (typically 8KB) is determined when the physical database is created. A high value for this measure indicates that direct disk accesses are high. In such a case your database might require some fine-tuning. Consider resizing your buffer pool to increase buffer accesses and reduce data retrievals from the disk.
Physical database block reads:	Indicates the rate at which physical database blocks (2-KB or 8-KB) were read from disk for both global and routine	Blocks/Sec	A high value for this measure indicates that direct disk accesses are high. In such a case your database might require some fine-tuning. Consider resizing your

Measurement	Description	Measurement Unit	Interpretation
	references.		buffer pool to increase buffer accesses and reduce data retrievals from the disk.
Physical database block writes:	Indicates the rate at which physical database blocks (2-KB or 8-KB) were written to disk for both global and routine references.	Blocks/Sec	
New database journal entries:	Indicates the number of entries recorded in the journal during this measurement period.	Number	To provide database integrity and reliability, Caché includes a number of journaling subsystems that keep track of physical and logical database updates. The journal management technology is also used to provide transaction support (a journal is used to perform transaction rollback operations) as well as database shadowing (a journal is used to synchronize a shadow server with a primary data server). This measure hence aids in effective audit tracking of your database, as it can tell you whether any modifications have been performed on the database - be it a set/kill/transaction event. Since the events for which journaling is needed is configurable, you might want to track critical changes to your database by enabling journaling.
Lock commands:	Indicates the number of	Number	

Measurement	Description	Measurement Unit	Interpretation
	lock commands that are issued in the last measurement period.		
Lock successes:	Indicates the number of lock commands that succeeded in the last measurement period.	Number	When the attempt to lock a transaction succeeds, it means that other processes will be prevented from seeing the modifications to the transaction, until the transaction is committed.
Lock failures:	Indicates the number of lock commands that have failed in the last measurement period.	Number	A lock command typically fails when it is unable to acquire a lock on a transaction - this happens when the transaction to be locked is already locked by another process.
Current global jobs:	Indicates the number of jobs currently counted 'in global'.	Number	
Max global jobs:	Indicates the maximum number of jobs currently counted 'in global'.	Number	
Throttle wait jobs:	Indicates the number of jobs required to wait for a throttle.	Number	
Write update status:	Indicates whether /not the Write Daemon has successfully updated the database.	Boolean	Rather than writing directly from memory to the database, the Caché write daemon system process (WRTDMN) uses an intermediate file, the write image journal. This file is usually named cache.wij. Write image journaling technology uses a two- phase process of writing to the database, as follows:

Measurement	Description	Measurement Unit	Interpretation
			• In the first phase, Caché records the changes needed to complete the update in the write image journal. Once all updates to the write image journal have been entered, a flag is set in the file and the second phase begins. • In the second phase, the Write daemon writes the changes recorded in the write image journal to the database on disk. When this second phase completes, the Write daemon sets a flag in the write image journal to indicate it is empty. When Caché starts, it automatically checks the write image journal and runs a recovery procedure if it detects that an abnormal shutdown occurred. When Caché indicates successful completion, the internal integrity of the database is restored. Recovery is necessary if a system crash or other major system malfunction occurs at either of the following points in the two- phase write protocol process: • Before the Write daemon has completed writing the update to the write image journal. In this case, recovery discards the incomplete entry. • After the update to the write image journal is complete but before the update (s) to the database is complete. In this

Measurement	Description	Measurement Unit	Interpretation
Users updating:	Indicates the number of users performing global updates.	Number	
Write daemon status:	Indicates the status of the write daemon.	Boolean	The value 1 indicates that the write demon is running, and the value 0 indicates that write demon is waiting for wakeup. If this value becomes 2, it indicates that Cache stopped the write daemon to prevent further harm.

4.3.2 Cache Resources Test

A number of areas of the core Caché code require protection from simultaneous access by multiple processes. These areas are dubbed resources. Sometimes, when a resource requested by a Cache process is already in use, then a spin lock is created on the resource - in this case, the process simply waits in a loop (i.e., spins) and repeatedly checks the lock status of the resource, until a lock on the resource is available to it. A few other times, the process, instead of creating a spin lock, might wait for a while to see if the resource is released, and if not, switch to the sleep mode. This process of acquiring a lock on a resource (with or without spinning) is also known as seizing. The **Cache Resource** test auto-discovers the resources on every configured Cache instance, and reports the number of times seizes have occurred on the discovered resources, thereby enabling administrators to determine whether there is a serious contention for resources on the Cache instance. Moreover, the detailed diagnosis of this test additionally reveals the number of seizes per seize state, thus isolating Cache instances where highly expensive seize operations are being performed.

Target of the test : A Cache Database server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every *InstanceName:Resource* pair discovered

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed

2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening
4. **INSTANCEDIRECTORY** - Typically, the Cache console log file will be available in the install directory of a Cache instance. Therefore, in the **INSTANCEDIRECTORY** text box, specify the name of the instance being monitored and the install directory that holds the Cache console log file of that instance, in the following format: *InstanceName:InstallDirectory*. In case you want to monitor the console log files pertaining to multiple Cache instances, then provide a comma-separated list of *InstanceName:InstallDirectory* pairs in the **INSTANCEDIRECTORY** text box. For example: *CACHEWEB:d:\Intersystems\CacheWeb,CACHE2:d:\Intersystems\Cache2*.
5. To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

   The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

   - The eG manager license should allow the detailed diagnosis capability
   - Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Resource seizures:	Indicates the number of times this resource was seized.	Number	Ideally, this value should be low. If the value is high, then, you can use the detailed diagnosis of this measure (if enabled), to know the seize state, and the number of seizes in every state, so that you can analyze the implications of the high seize count effectively. The various seize states are as follows:

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><th>Seize State</th><th>Description</th></tr>
<tr><td>A Seize</td><td>Number of times spun before acquiring lock on the resource</td></tr>
<tr><td>N Seize</td><td>Number of times spun before failing to acquire a lock on the resource</td></tr>
<tr><td>B Seize</td><td>Number of times spun and switched to sleep mode before acquiring a lock on the resource</td></tr>
<tr><td>BusySet</td><td>Other waiters spinning should wait behind this one</td></tr>
</table>
N Seizes are typically expensive on SMP systems.
</td></tr>
</table>

4.3.3 Cache Network Test

This test connects to the Cache database server, executes a query, and determines the availability and responsiveness of the Cache database server.

This test requires JDK 1.5 for execution. However, only JDK 1.3 is bundled with the eG agent. Therefore, to ensure the smooth execution of the test, you will have to do the following before attempting to monitor the Cache database server:

- Install JDK 1.5 on the host of the external agent that will be executing this test.
- Edit the **debugon.bat** and **debugoff.bat** files in the **<EG_AGENT_INSTALL_DIR>\lib** directory.
- These files contain path specifications to the default JRE 1.3.1 (which is in the **<EG_AGENT_INSTALL_DIR>\JRE** directory). Replace these path specifications with that of JRE 1.5 that you just installed on the host. In the extract below, the text in **Bold** indicates where and what changes have to be made to these files:

```
set path=”<JDK1.5_install_
dir>\bin
”;C:\eGurkha\bin;C:\eGurkha\lib;C:\eGurkha\bin\ic;D:\oracle\ora92\bin;C:\Program
Files\Oracle\jre\1.3.1\bin;C:\Program
Files\Oracle\jre\1.1.8\bin;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wb
em;C:\Program Files\Common Files\InterSystems\Cache;C:\Program Files\Microsoft SQL
Server\80\Tools\BINN;C:\Program Files\Microsoft SQL Server\80\Tools\Binn\;%EGURKHA_
PATH%
```

```
net stop eGurkhaAgent
```

```
js -uninstall eGurkhaAgent
```

```
js -install eGurkhaAgent “<JDK1.5_install_dir>\jre\bin\client\jvm.dll” -Xrs -
Djava.class.path=%classpath% -Djava.library.path=C:\eGurkha\lib;C:\eGurkha\bin -start
EgMainAgent -params -manager 192.168.10.12 -port 7077 -dir C:\eGurkha -ssl false -
highSecurity false -path “<JDK1.5_install_dir>\jre\bin”
```

```
exit
```

- Save the changes.
- Run the **debugoff.bat** file to reinstall the agent service to use the new JRE.
- Finally, restart the eG agent that will be executing this test.

This test is disabled by default. To enable the test, go to the **ENABLE / DISABLE TESTS** page using the menu sequence : Agents -> Tests -> Enable/Disable, pick *Cache Database* as the **Component type**, *Performance* as the **Test type**, choose this test from the **DISABLED TESTS** list, and click on the **<<** button to move the test to the **ENABLED TESTS** list. Finally, click the **Update** button.

Target of the test : A Cache Database server

Agent deploying the test : An external agent; if you are running this test using the external agent on the eG manager box, then make sure that this external agent is able to communicate with the port

on which the target Intersystems Cache database server is listening. Alternatively, you can deploy the external agent that will be running this test on a host that can access the port on which the target server is listening.

Outputs of the test : One set of results for the Cache database server being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the Cache database server
3. **PORT** – The port on which the server is listening
4. **DB** - The database to which the test should connect. The default value is **%SYS**.
5. **USER**- The user name to be used by the test for connecting to the specified **DB**. The default value is **_SYSTEM**.
6. **PASSWORD** - Specify the password of the **USER**.
7. **CONFIRM PASSWORD** - Confirm the password by retyping it in this space.
8. **QUERY** - The query to be executed on the **DB** to ascertain the availability and responsiveness of the server

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Availability:	Whether the database server is available or not	Number	While the value 0 indicates that the server is not running, 100 indicates that the server is alive and well. Availability problems may be caused by a misconfiguration/malfunctioning of the database instance, or because the instance is using an invalid user account.
Response time:	Indicates the time taken by the server to respond to requests	Secs	A sudden increase in response time is indicative of a bottleneck at the database server.

5

Chapter 5: Troubleshooting

If all the SNMP-based tests pertaining to the **Cache** database server are in an **UNKNOWN** state, then do the following:

- Check whether the eG agent is up and running
- Ensure that there are no issues in the agent-manager communication
- Check whether the Windows SNMP service has been installed and started on the Cache host
- Verify whether the Cache Monitoring Service has been enabled (refer to Section **Chapter 2** for details)
- Verify whether the Cache SNMP Agent has been set to start automatically at Cache startup (refer to Section **Chapter 2** for more details)
- Check whether the critical SNMP base classes such as **%Monitor.System.Freespace**, **%Monitor.System.Database**, **%Monitor.System.Processes**, and **%Monitor.System.SystemMetrics** were activated from the Cache Terminal.
- If one/more SNMP-based tests pertaining to the Cache database server are not reporting measures, then go to the command prompt of the eG agent host, and execute the following command from the **<EG_AGENT_INSTALL_DIR>\bin** directory, to see if the desired statistics are being retrieved from the SNMP MIB of the Cache database server:

  snmpwalk nfq - O 161 <SNMP_ community_ string_ configured_ for_ the_ tests> <IP/hostname_of_the_CacheDB> <OID_of_the_measure>

  For instance, to view the value of a particular measure reported by the **Cache Buffer** test for a Cache database server with IP 192.16.10.51 (community string: **public**), your command would be:

  snmpwalk nfq -O 161 public 192.168.10.51 .1.3.6.1.4.1.16563.1.1.7.1.2

- Given below are the details of OIDs that are associated with the tests mapped to the Cache database server model. You will find that most tests are associated with a wide range of OIDs, one for every measure reported by the test. To check whether a test is reporting measures or not, it would suffice to run the above-mentioned command using any one of the OIDs in the given range, and not each one of them.

- For the **Cache Buffer** test, use the OIDs in the range, .1.3.6.1.4.1.16563.1.1.7.1.2 to .1.3.6.1.4.1.16563.1.1.7.1.17
- For the **Cache Database** test, use any of the following OIDs:

  .1.3.6.1.4.1.16563.1.1.3.1.5

  .1.3.6.1.4.1.16563.1.1.3.1.6

  .1.3.6.1.4.1.16563.1.1.3.1.7
- For the **Cache Performance** test, use the OIDs in the range .1.3.6.1.4.1.16563.1.1.2.1.2 to .1.3.6.1.4.1.16563.1.1.2.1.18
- For the **Cache Resource** test, use the OID, .1.3.6.1.4.1.16563.1.1.6.1.3
- For the **Cache Systems** test, use the OIDs in the range .1.3.6.1.4.1.16563.1.1.1.1.6 to .1.3.6.1.4.1.16563.1.1.1.1.11
- For the **Ecp Application Server** test, use the OIDs in the range, .1.3.6.1.4.1.16563.1.1.4.1.1 to .1.3.6.1.4.1.16563.1.1.4.1.3
- For the **Ecp Data Server test**, use the OIDs in the range, .1.3.6.1.4.1.16563.1.1.5.1.1 to .1.3.6.1.4.1.16563.1.1.5.1.5

➢ Similarly, if one/more tests using the **cstat** utility are not reporting measures, then issue the **cstat** command from the command prompt to see if it returns a valid output. For that, follow the steps given below:

- Go to the command prompt on the Cache host.
- Switch to the **<CACHE_INSTALL_DIR>.**
- Issue the command: **cstat –s <CACHE_INSTALL_DIR>\mgr -e2 –m-1 –n3 –j5 –g1 –m3 –L1 –u-1 –v1 –p-1 –c-1 –q1 –w2 –S –1**

For example, if the Cache instance being monitor is installed in **c:\CacheSys**, then the command would be:

cstat –s c:\CacheSys\mgr -e2 –m-1 –n3 –j5 –g1 –m3 –L1 –u-1 –v1 –p-1 –c-1 –q1 –w2 –S –1

6

Chapter 6: Conclusion

This document has described in detail the monitoring paradigm used and the measurement capabilities of the eG Enterprise suite of products with respect to **Intersystems Cache Database** server. For details of how to administer and use the eG Enterprise suite of products, refer to the user manuals.

We will be adding new measurement capabilities into the future versions of the eG Enterprise suite. If you can identify new capabilities that you would like us to incorporate in the eG Enterprise suite of products, please contact support@eginnovations.com. We look forward to your support and cooperation. Any feedback regarding this manual or any other aspects of the eG Enterprise suite can be forwarded to feedback@eginnovations.com.