Monitoring Ensemble

Version 2017.2
2020-06-26

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

Monitoring Ensemble
Ensemble Version 2017.2 2020-06-26

For Support questions about any InterSystems products, contact:

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

Table of Contents

About This Book

This book, which is primarily for system administrators, explains how to use the Management Portal to monitor the Ensemble environment, your Ensemble productions, and Ensemble production components. It contains the following chapters:

- Concepts
- Monitoring All Namespaces
- Monitoring a Production
- Viewing, Searching, and Managing Messages
- Viewing the Event Log
- Enabling Tracing
- Viewing the Business Rule Log
- Viewing Business Process Instances
- Viewing the Ensemble I/O Archive
- Viewing Messages from Multiple Productions
- Using the Enterprise Message Viewer
- Using the Enterprise Message Bank
- Monitoring Alerts
- Monitoring Activity Volume

For a detailed outline, see the table of contents.

The following books are also relevant to system administrators:

- *Managing Ensemble* explains how to use the Management Portal to manage Ensemble.
- *Caché Monitoring Guide* describes the tools, routines, and third-party interfaces available to monitor Caché. These tools apply to Ensemble as well, at a lower level than the tools described in *Monitoring Ensemble*.
- *Caché System Administration Guide* describes common system administration tasks in a pure Caché environment. There is some overlap between the tools in *Managing Ensemble* and that book.
- *Caché Security Administration Guide* describes authentication, authorization, auditing, managed key encryption, SSL/TLS, and other aspects of Caché security.

For general information, see the *InterSystems Documentation Guide*.

1

Core Concepts

This chapter discusses the concepts that are relevant when you monitor Ensemble. This chapter includes the following sections:

- Productions
- Production States
- Business Hosts
- Messages
- Jobs and Message Queues

1.1 Productions

An Ensemble *production* is a specialized package of software and documentation that integrates multiple, potentially disparate software systems. A production includes elements that communicate with these external systems, as well as elements that perform processing that is internal to the production.

Ensemble permits only one production to be running in a given namespace at any given time.

A running production continues to run even when you close the Management Portal.

1.2 Production States

It is important to be familiar with the acceptable states of a production, as well as the problem states. These states are displayed in the Management Portal:

State	Meaning
Running	When a production has been started and is operating normally, it has a status of Running. This is an acceptable state.
Stopped	A production acquires a status of Stopped when, at the end of the shutdown sequence, all of its queues are free of synchronous messages. This is an acceptable state.

State	Meaning
Suspended	A production acquires the Suspended status if, at the end of the shutdown sequence, some queues still contain synchronous messages, waiting for a response. Depending on how the production has been designed, this may or may not indicate a problem. If you suspect a problem, see "Correcting Production Problem States," later in this book.
Troubled	A production acquires a status of Troubled if Ensemble is stopped but the production did not shut down properly. See "Correcting Production Problem States," later in this book.

1.3 Business Hosts

An Ensemble production includes a number of *business hosts* that communicate with each other (and with external systems). A business host is responsible for some category of work.

There are three distinct types of business host. In order to monitor a production, it is not necessary to know the details of why a given business host was implemented as a certain type. However, it is useful to be aware of the types:

- A *business service* receives input from outside the production, sometimes by means of an inbound *adapter*.
- A *business process* is responsible for communication and logic that is entirely within the production.
- A *business operation* usually sends output from the production, sometimes by means of an outbound adapter.

  Business operations can also be used for communication and logic within a given production.

1.4 Messages

Within a production, all communication is carried out by means of request and response messages between the business hosts. In order to understand a production at a high level, just remember that messages carry all traffic, and that the work of a production is to process messages.

Without exception, the Ensemble message warehouse stores all messages, and this information is available in the Management Portal. This section provides an overview of messages and the information that is available about them:

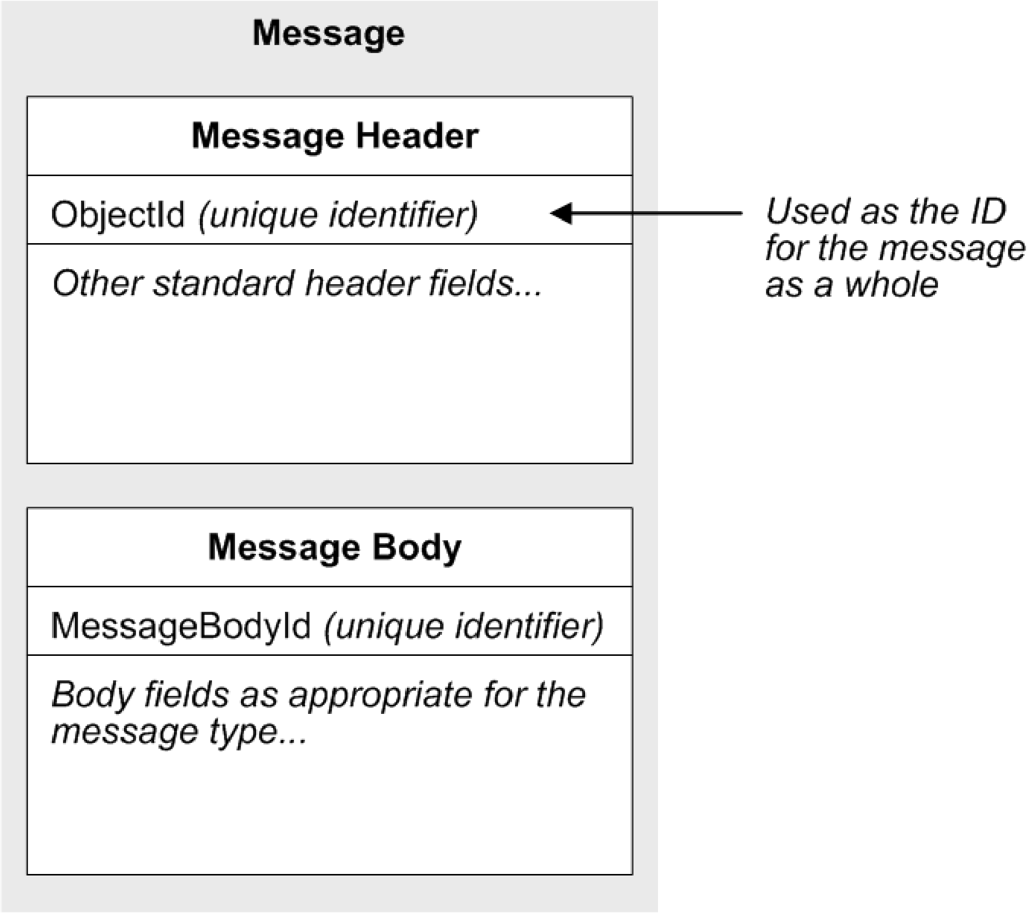
- Message Basics
- Sessions, which correspond to sets of messages related in time
- Message Status
- Message Invocation Style and Time stamps
- Message Priority

1.4.1 Message Basics

Every message is either a request or a response. There may be requests that do not have a corresponding response defined, but every response is (conceptually) paired with a request. Any request may be synchronous (waits for a response) or asynchronous (does not wait), depending on the details of the business host; you cannot configure this.

Each message has a unique numeric identifier or *ID*. This is displayed in many places in the Management Portal, with the caption **ID** or **<ObjectId>**, depending on the location on the page.

A message has a header, whose structure is the same for all messages. The header contains data fields that help route the message through the system. A message also contains a body, which provides different fields depending on the message type.


Message
Message Header
ObjectId (unique identifier)
Other standard header fields...
Used as the ID for the message as a whole
Message Body
MessageBodyId (unique identifier)
Body fields as appropriate for the message type...

Each message uses a specific message body class, chosen by the production developers. The message body class determines whether the message is a request or a response and determines the fields that the message contains. These decisions are not configurable once the production is complete.

For Electronic Document Interchange (EDI) formats, Ensemble provides specialized message body classes that represent the message as a virtual document. In this case, the message body does not contain fields to represent data in the message. Ensemble provides alternative mechanisms for accessing that data. For an introduction, see *Ensemble Virtual Documents*.

The Management Portal displays contents of the message as a whole, treating the message header and the message body as a single unit. The ID of the message header is the ID of the message as a whole. In some cases (for example, if you resend a message), a new header is added (with a new unique ID); as a result, the ID of the resent message is not the same as that of the original message.

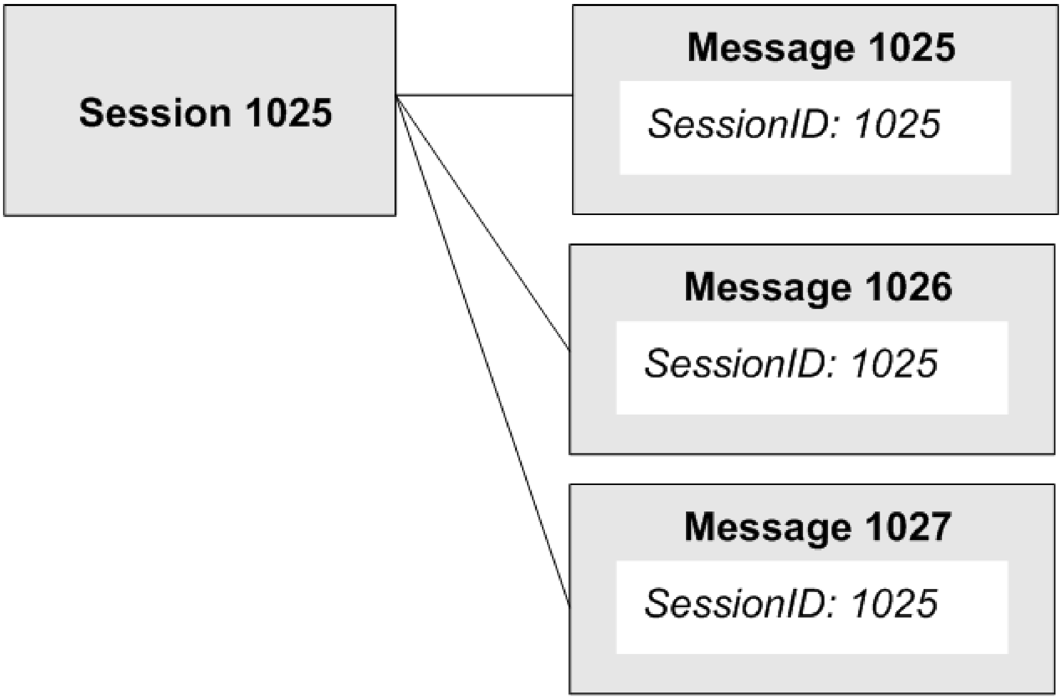
1.4.2 Sessions

Every message is associated with a specific *session*. A session marks the beginning and end of all the activities prompted by a *primary request message* from outside Ensemble. Sessions are useful to you because they give you an easy way to see sets of related messages; the Management Portal provides an option for visually tracing the messages, and you can filter this display by session.

Each session has a unique numeric *SessionID*. All messages associated with a session use the same SessionID. Ensemble assigns these SessionIDs as follows:

1. The primary request message starts the session. The SessionID is the same as the ID of the primary request message.
2. Each additional message that is instantiated during this session has the same SessionID as the primary request, but has a unique message ID.

The following shows an example. Note that (unlike the example) the message IDs within a session are unlikely to be sequential in a production that has many business hosts. When creating a new message, Ensemble always uses the next available integer as the message ID.



1.4.3 Message Status

Each message has a life cycle during which its *status* changes. These statuses are visible on most pages that display messages. The possible status of any message is one of the following:

`Created`

The message is in transit between its sender and the appropriate queue. This is the first stage in the normal life cycle of a message.

`Queued`

The message is on a queue. This is the second stage in the normal life cycle of a message.

`Delivered`

The intended recipient has received the message. This is the third stage in the normal life cycle of a message.

`Completed`

The intended recipient has received the message and has finished processing the message. This is the fourth stage in the normal life cycle of a message.

`Deferred`

This status applies only to response messages.

A business operation can defer a message response for later delivery. The response can be picked up and sent back to the original requester by any business host in the production. Between the time when the business operation defers the response, and when the response is finally sent, the response message has a status of `Deferred`.

The sender of the original message is unaware of the fact that the message was deferred. If the original call was synchronous, the call does not return to the sender until the response is sent.

When the response message is finally sent, it has a status of `Completed`.

`Discarded`

A response message becomes `Discarded` if it reached its destination after the timeout period for the corresponding request expired.

You can also manually mark a message as `Discarded`, which you might do for a suspended message that cannot be delivered for some reason.

Note that a message that is marked as `Discarded` still remains in the permanent store; messages are deleted only when you explicitly delete them.

`Suspended`

The message was suspended by the business operation after failing to reach its external destination or was manually suspended by an administrator. Note that some business operations are designed to set the status of any failed messages to Suspended.

In either case, you can view this message within the Management Portal to determine why it failed and you can resend it if appropriate. For example, if the problem is on the external side of the communication, the external system can be repaired, and then the message can be resent. You could also discard it or even delete it.

`Aborted`

The message was aborted by an administrator.

`Error`

The message encountered an error.

Note that request and response messages have separate statuses. Request-response pairs are not tracked together for various reasons: a request might be repeated several times before it is successfully delivered; some requests have an optional response that can be ignored if it does not arrive; some responses can be deferred for later delivery; some requests are designed to have no response.

1.4.4 Message Invocation Style and Time Stamps

Each message has an *invocation style*, which describes how the message was sent. The business host that sends a message specifies its invocation style:

- `Queue` means the message is created in one job, then placed on a queue, at which time the original job is released. Later, when the message is processed, a different job will be allocated for the task.
- `Inproc` means the message will be formulated, sent, and delivered in the same job in which it was created. The job will not be available again in the sender’s pool until the message is delivered to the target.

Ensemble records the following two time stamps for each message. Note that the invocation style affects the meaning of these time stamps:

- The *message creation time stamp*. For `Queue` messages, this is when Ensemble placed this message on the queue. For `Inproc` messages, this is when Ensemble called the Send method.
- The *message processed time stamp*. Ensemble sets TimeProcessed when the message is taken off of the queue but then resets it to the current time while the message is being processed. Typically, for a completed message, it represents the time that the message processing was completed.

1.4.5 Message Priority

The Management Portal displays the *priority* of the messages in several places. The priority of a message determines how that message is handled relative to other messages in the same message queue. For information about message queues, see the chapter "Monitoring a Production."

The Ensemble messaging engine determines the priority of a message, which is one of the following:

- `HighSync` (1) — Used for ACK messages and alarms for interrupted tasks.
- `Sync` (2) — Used for synchronous messages.
- `SimSync` (4) — Used for an asynchronous call made for a BPL synchronous <call>. This ensures that the request and response are processed before any queued asynchronous message.
- `Async` (6) — Used for other asynchronous messages.

1.5 Jobs and Message Queues

The business hosts process messages by means of jobs. A *job* is a CPU process that hosts the work done by an Ensemble production. This terminology is intended to avoid confusion between CPU processes ("jobs") and business processes ("processes").

In general, a job is either *running* (processing a message) or it is not running. From a low-level, system viewpoint, an Ensemble production consists almost entirely of jobs waiting to wake up to perform work.

A *pool* consists of one or more jobs. Each business host can have its own, private pool of allocated jobs — a pool with a specific size that cannot be exceeded.

When a business host receives a message, the business host assigns the work to a job from its pool. The job then performs the work. If no job is available, the business host waits for a job to become available. After the job completes the work, the job either returns to the pool or starts work on another message.

More formally, each message is assigned to a specific *message queue*, which handles messages in the order that it receives them. Each queue is associated with a single pool, and vice versa.

Unlike other types of business host, a business process can use a public pool (called the *actor pool*), which receives messages from a public queue (called the *actor queue* or Ens.Actor). The actor pool and actor queue are shared by all business processes that do not use private pools and queues.

For further information, see "Pool Size and Actor Pool Size" in *Configuring Ensemble Productions*.

2

Monitoring All Namespaces

This chapter describes how to monitor productions in all namespaces. It contains the following topics:

- General Notes
- Viewing Summaries for Active Productions
- Using the Ensemble System Monitor
- Monitoring Multiple Productions with the Enterprise Monitor to monitor multiple namespaces across multiple instances of Ensemble

2.1 General Notes

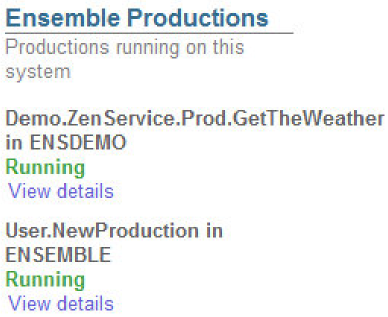
For background information, see the chapter “Concepts.”

For information on starting and stopping productions, see “Managing Ensemble.” Note that for a live, live, deployed production, InterSystems recommends that you use the auto-start option, which is described in that book.

If a production is Suspended or Troubled, see “Correcting Production Problem States,” in the next chapter.

2.2 Viewing Summaries for Active Productions

When you select any option in the **Ensemble** menu, the right side of the page displays summary information about the productions, as follows:


Ensemble Productions
Productions running on this system

Demo.ZenService.Prod.GetTheWeather in ENSDEMO
Running
View details

User.NewProduction in ENSEMBLE
Running
View details

2.3 Using the Ensemble System Monitor

The **Ensemble System Monitor** page provides a high-level view of the state of your system, across all namespaces. (It displays Ensemble information combined with a subset of the information shown on the **System Dashboard** page, which is provided for the users of Caché.)

To access this page in the Management Portal, select **Ensemble** > **Monitor** > **System Monitor**.

This page displays tables of information, described in the following subsections.

Also see “General Notes,” earlier in this chapter.

2.3.1 Ensemble Throughput

This table provides information about the throughput of productions in all namespaces. The table lists the following values:

- **Productions Running** — Number of productions that are currently running.
- **Productions Suspended or Troubled** — Number of productions that are currently suspended or troubled.

  If a production is Suspended or Troubled, see “Correcting Production Problem States,” in the next chapter.
- **Incoming Messages in Last 30 Seconds** — Number of messages received by business services in the last 30 seconds.
- **Last Incoming Message** — Date and time of last message received by any business service.
- **Outgoing Messages in Last 30 Seconds** — Number of messages processed by business operations in the last 30 seconds.
- **Last Outgoing Message** — Date and time of last message processed by any business operation.

2.3.2 Ensemble Jobs

This table provides information about the jobs associated with the currently running productions in all namespaces. The table lists the following values:

- **Total System Processes** — Number of system processes that are currently active, including processes that are not specifically associated with Ensemble productions.

  If you click **Total System Processes** and then click the link at the bottom of the page, Ensemble displays the **Processes** page. For information on this page, see “Controlling Caché Processes” in the chapter “Managing Caché” in the *Caché System Administration Guide*.
- **Active Ensemble Jobs** — Number of Ensemble jobs that are currently active.
- **Visiting Ensemble Jobs** — Number of jobs outside of Ensemble that are currently invoking Ensemble code.
- **Most Active Processes** — Displays a table of the processes that have the highest number of recently executed commands, across all namespaces, including processes that are not specifically associated with Ensemble productions. **PID** is the process ID, and **Commands** is the number of recently executed commands.

Also see “Diagnosing Problems with Jobs” in the next chapter.

2.3.3 System Time

This table provides the same information as in the **System Time** table of the **System Dashboard** page. See “Monitoring System Dashboard Indicators” in the chapter “Monitoring Caché Using the Management Portal” in the *Caché Monitoring Guide*.

If you click a row in this table and then click the link at the bottom of the page, Ensemble displays the **System Dashboard** page.

2.3.4 System Usage

This table provides a subset of the information in the **System Usage** table of the **System Dashboard** page. See "Monitoring System Dashboard Indicators" in the chapter "Monitoring Caché Using the Management Portal" in the *Caché Monitoring Guide*.

If you select a row in this table and then click the link at the bottom of the page, Ensemble displays one of the following pages, as appropriate for the row you selected:

- **Databases**
- **Journals**
- **Locks**

2.3.5 Ensemble Queues

This table provides information about Ensemble queues in all namespaces. The table lists the following values:

- **Active Queues** — Count of currently active Ensemble queues.
- **Most Active Queues** — Displays a table of the queues that have the largest number of unprocessed messages. In this table, **Messages** is the count of messages in the given queue.

Also see "Diagnosing Problems with Queues" in the next chapter.

2.3.6 Errors and Alerts

This table provides information about errors and alerts. The table lists the following values:

- **Serious System Alerts** — Number of serious system-level alerts that have been raised.
- **Ensemble Alerts** — Number of serious Ensemble alerts that have been raised.
- **Ensemble Errors** — Number of application errors that have been logged.

For information on configuring a production to send alerts, see "Configuring Alerts" in *Configuring Ensemble*.

2.3.7 Licensing

This table provides the same information that is shown in the **Licensing** table of the **System Dashboard** page. See "Monitoring System Dashboard Indicators" in the chapter "Monitoring Caché Using the Management Portal" in the *Caché Monitoring Guide*.

If you select a row in this table and then click the link at the bottom of the page, Ensemble displays the **License Usage** page.

2.3.8 Task Manager

This table provides the same information that is shown in the **Task Manager** table of the **System Dashboard** page. See "Monitoring System Dashboard Indicators" in the chapter "Monitoring Caché Using the Management Portal" in the *Caché Monitoring Guide*.

If you select a row in this table and then click the link at the bottom of the page, Ensemble displays the **Upcoming Tasks** page.

2.4 Monitoring Multiple Productions with the Enterprise Monitor

The Enterprise Monitor displays the overall status of multiple running productions. These productions can be running on different namespaces within the same instance of Ensemble or can be running on multiple instances of Ensemble. You can display the Production Monitor or the Ensemble management portal for any of the productions being monitored. The monitored productions can be running in different namespaces on the same Ensemble instance, running on multiple Ensemble instances on the same system, running on multiple systems, or running on any combination of these.

This section contains the following topics:

- Configuring the Enterprise Monitor
- Using the Enterprise Monitor
- Configuring and Using Enterprise Monitor Roles
- Troubleshooting the Enterprise Monitor

2.4.1 Configuring the Enterprise Monitor

The Enterprise Monitor runs in its own namespace with a special production that gets the status of the monitored systems. To configure an Enterprise Monitor, you perform the following steps:

1. Create a namespace for the Enterprise Monitor or choose to use an existing namespace for it. The following steps are done in this namespace.
2. Define credentials that provide access to the systems that you will be monitoring.
3. Configure Enterprise Systems, defining a new connection for each system that you are monitoring. Optionally, specify a queue threshold for each system. For details on configuring Enterprise Systems, see “Identifying Enterprise Systems for Viewing and Monitoring” in *Configuring Ensemble Productions*.
4. Optionally, specify Enterprise Monitor Roles. If the user using the Enterprise Monitor has one of these roles, the user only monitors the configuration items that have one of the specified categories. For details, see “Configuring and Using Enterprise Monitor Roles”
5. Create a production for the special Enterprise Monitor service. The class of this production must derive from the Ens.Enterprise.Production class. You can create the production in Studio or create it using the management portal and then edit in Studio as follows
   a. In Studio, open the class that defines the production running in the Enterprise Monitor namespace.
   b. Replace the class Ens.Production with Ens.Enterprise.Production. For example, if the class definition is

      ```
      Class EMon.EntMonitorProd Extends Ens.Production
      ```

      Edit it so that the class extends Ens.Enterprise.Production so that it appears as:

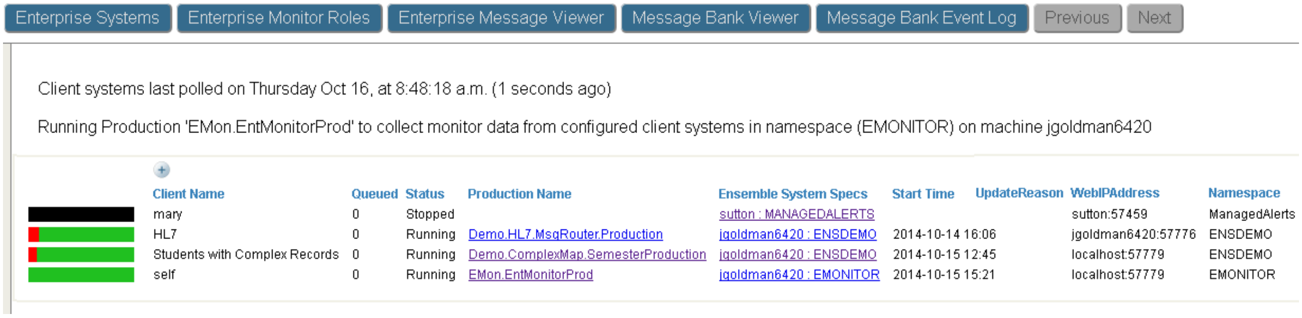
      ```
      Class EMon.EntMonitorProd Extends Ens.Enterprise.Production
      ```

   c. Compile the class.
6. Add the Ens.Enterprise.MonitorService business service to the production and enable it.
7. Start the production.

8. Select **Ensemble**, **Monitor**, and **Enterprise Monitor** to display the Enterprise Monitor. Note that this menu item is only visible if you have configured Enterprise Systems in the current namespace.

2.4.2 Using the Enterprise Monitor

The Enterprise Monitor displays a line for each system being monitored. The following Enterprise Monitor is monitoring four systems:



Client systems last polled on Thursday Oct 16, at 8:48:18 a.m. (1 seconds ago)

Running Production 'EMon.EntMonitorProd' to collect monitor data from configured client systems in namespace (EMONITOR) on machine jgoldman6420

	Client Name	Queued	Status	Production Name	Ensemble System Specs	Start Time	UpdateReason	WebIPAddress	Namespace
⬛	mary	0	Stopped		sutton : MANAGEDALERTS			sutton:57459	ManagedAlerts
🟥🟩	HL7	0	Running	Demo.HL7.MsgRouter.Production	jgoldman6420 : ENSDEMO	2014-10-14 16:06		jgoldman6420:57776	ENSDEMO
🟥🟩	Students with Complex Records	0	Running	Demo.ComplexMap.SemesterProduction	jgoldman6420 : ENSDEMO	2014-10-15 12:45		localhost:57779	ENSDEMO
🟩	self	0	Running	EMon.EntMonitorProd	jgoldman6420 : EMONITOR	2014-10-15 15:21		localhost:57779	EMONITOR

The Enterprise Monitor displays the following information for each system:

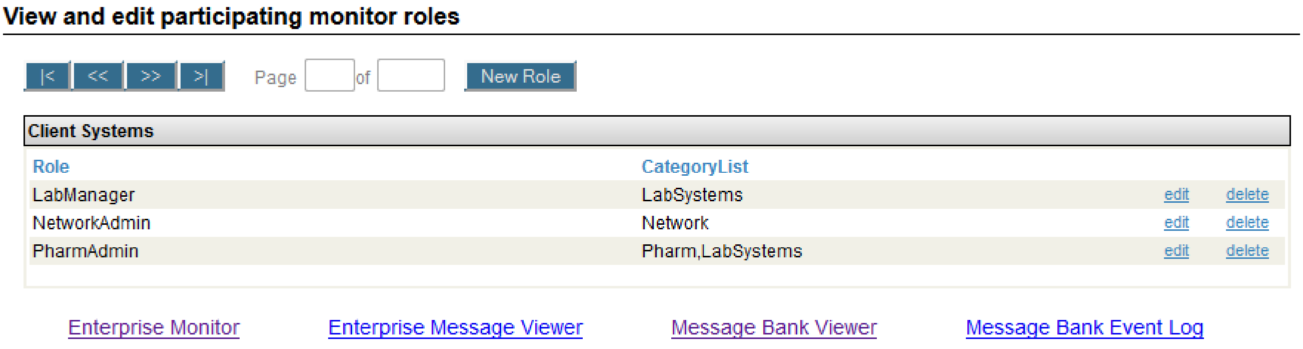
- Bar graph—Indicates the status of the configuration items of the production. The green, red, and yellow indicate the percentage of the items in each state. Green indicates the items that are active and running correctly; yellow indicates the items that are inactive; and red indicates the items that have encountered an error. If you hover over the bar graph, the pop-up text displays the number of items in each state.
- Client Name—Name defined when configuring Enterprise Systems to identify the system in the Enterprise Monitor.
- Queued—Specifies the total number of messages currently waiting in queues. If you have set the queue threshold and the number of messages exceeds the threshold, the number is displayed in red. If the number exceeds 85% of the threshold, it is displayed in yellow. If the number is below 85% of the threshold, it is displayed in green. If no threshold is specified, the queue number is displayed in black.
- Status—Indicates the status of the productions: running, stopped, suspended, or troubled.
- Production Name—Displays the production name. If you click on this link, the Enterprise Monitor opens the Production Configuration page on this system.
- Ensemble System Specs—Displays the system name and namespace of the system. If you click on this link, the Enterprise Monitor opens the Ensemble management portal on the system.
- Start Time—Displays the date and time that the Ensemble instance was started if it is currently running.
- UpdateReason—Specifies the reason that the production configuration was last updated.
- WebIPAddress—Specifies the system name and port number.
- Namespace—Specifies the namespace of the system.

If you click on an item in the Enterprise Monitor that is not a link, the Enterprise Monitor displays the production monitor for that system. For information on the Production Monitor, see Monitoring a Production.

2.4.3 Configuring and Using Enterprise Monitor Roles

Enterprise Monitor Roles allow you to limit the production components that are visible in the Enterprise Monitor based on the roles of the current user and the categories specified in the production configuration for the component. When a user displays the Enterprise Monitor, it checks if the user has any roles specified in the Enterprise Monitor Roles. If none of the roles match, the Enterprise Monitor displays information about all the components in the productions. If one or more of the roles match, the Enterprise Monitor displays information about components that have one of the specified categories.

To add new roles or edit an existing role, select **Enterprise Monitor Roles** on the Enterprise Monitor. The following illustrates the Enterprise Monitor Roles page:



View and edit participating monitor roles

Client Systems

Role	CategoryList		
LabManager	LabSystems	edit	delete
NetworkAdmin	Network	edit	delete
PharmAdmin	Pharm,LabSystems	edit	delete

Enterprise Monitor Enterprise Message Viewer Message Bank Viewer Message Bank Event Log

To add a new role, select **New Role**. To edit or delete a role, select “edit” or “delete”. When you enter the role and category, the form does not list the existing roles or categories. You must know these and enter them as text. Once you have entered a category, it is available as a checkbox when you add or edit a role.

2.4.4 Troubleshooting the Enterprise Monitor

If the Enterprise Monitor is not working, this troubleshooting list may help you resolve the problem:

- If the Enterprise Monitor cannot get access from any monitored system and displays the message “Not currently collecting monitor data from configured client systems - No Message Bank or Enterprise Production is running in this namespace (EMONITOR) on machine jgoldman6420”:

  Ensure that the production in the namespace is running. You will get this message if it is not running.

- If the Enterprise Monitor appears to be working correctly but it is displaying the message “Not currently collecting monitor data from configured client systems - No Message Bank or Enterprise Production is running in this namespace (EMONITOR) on machine jgoldman6420”:

  The production in the namespace must have a class that extends the Ens.Enterprise.Production class. If you create a new production using the Ensemble portal, it creates a production that extends Ens.Production. To fix this problem, edit the production in Studio and change the class that it extends. Then compile the class and stop and restart the production.

- If the Enterprise Monitor does not display an error message but the clients are not being polled and the data is not being updated:

  Ensure that the production contains the Ens.Enterprise.MonitorService business service and that it is enabled.

3

Monitoring a Production

The Management Portal provides pages to enable you to monitor a single production more closely (in contrast to the previous chapter, which describes how to monitor all namespaces). This chapter describes how to use these pages. It discusses the following topics:

- General Notes
- Using the Production Monitor
- Monitoring Production Queues
- Monitoring Active Jobs
- Using the Production Configuration Page
- Correcting Production Problem States

3.1 General Notes

For background information, see the chapter "Concepts."

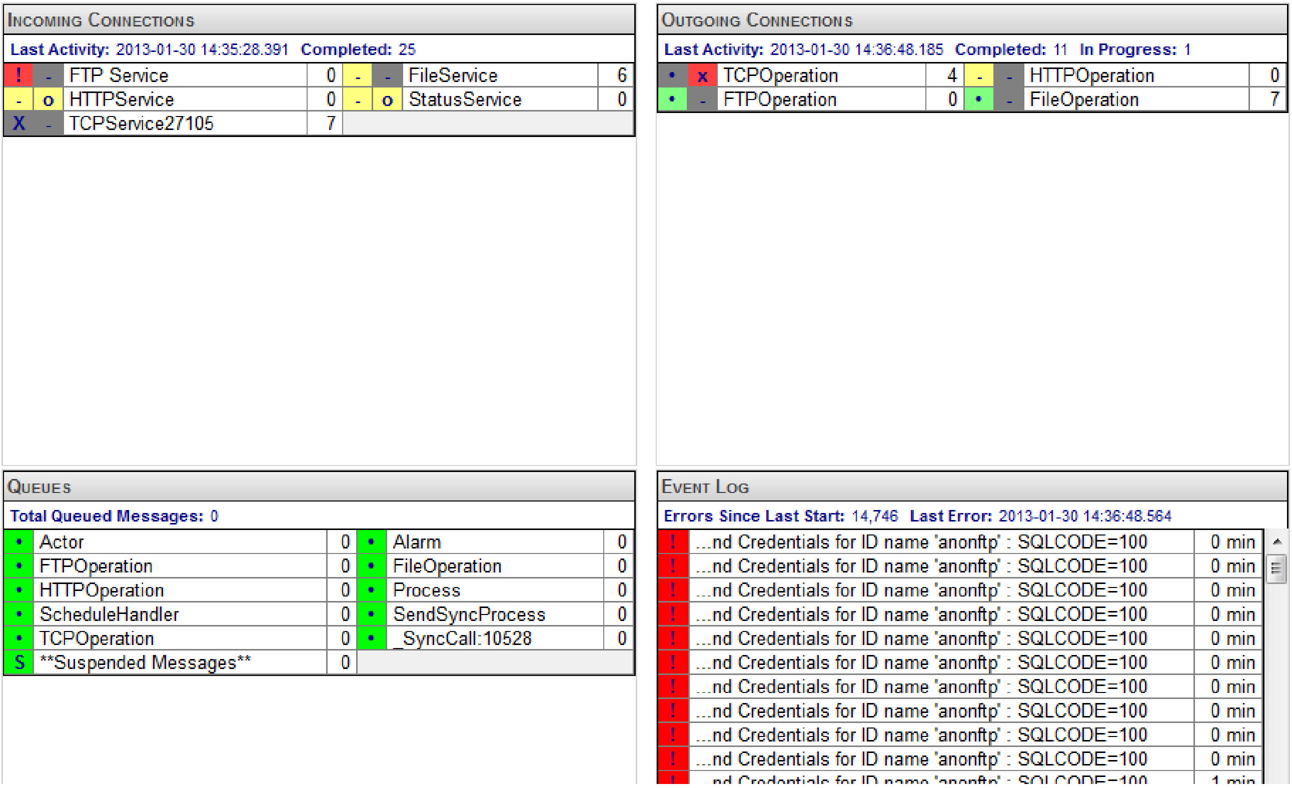
For information on starting and stopping productions, see "Managing Ensemble." Note that for a live, live, deployed production, InterSystems recommends that you use the auto-start option, which is described in that book.

If a production is Suspended or Troubled, see "Correcting Production Problem States."

3.2 Using the Production Monitor

The **Production Monitor** page displays real-time status information about the currently running production in a condensed, one-page format, with links for further details. To display this page in the Management Portal, select **Ensemble** > **Monitor** > **Production Monitor**.

You can use this page to monitor the general health of the production in the selected namespace. The following is a partial example of what this page displays:



Incoming Connections

Last Activity: 2013-01-30 14:35:28.391 **Completed:** 25

!	-	FTP Service	0	-	-	FileService	6
-	o	HTTPService	0	-	o	StatusService	0
X	-	TCPService27105	7				

Outgoing Connections

Last Activity: 2013-01-30 14:36:48.185 **Completed:** 11 **In Progress:** 1

•	x	TCPOperation	4	-	-	HTTPOperation	0
•	-	FTPOperation	0	•	-	FileOperation	7

Queues

Total Queued Messages: 0

•	Actor	0	•	Alarm	0
•	FTPOperation	0	•	FileOperation	0
•	HTTPOperation	0	•	Process	0
•	ScheduleHandler	0	•	SendSyncProcess	0
•	TCPOperation	0	•	_SyncCall:10528	0
S	**Suspended Messages**	0			

Event Log

Errors Since Last Start: 14,746 **Last Error:** 2013-01-30 14:36:48.564

!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	0 min
!	...nd Credentials for ID name 'anonftp' : SQLCODE=100	1 min

- Input Connections
- Output Connections
- Queues
- Event Log
- Activity Graph
- Custom Metrics

By default, this page is automatically updated at frequent intervals. In the left area, you can clear the **Auto update** check box to disable these updates.

The **Production Monitor** page displays real-time information provided by the Monitor Service. The Monitor Service is a business service that is implicitly included in every Ensemble production (not visible as part of its configuration). The Monitor Service continually monitors the activities of Ensemble items while a production is running, and records data about them at frequent intervals.

3.2.1 Input Connections

The **Input Connections** table (upper left) lists all incoming connections from external systems. Each entry indicates following:

1. Business service status
2. Business service connection status
3. Business service name
4. Number of messages processed since the production started

The statuses are indicated by the cell color. The item status and the connection status cells have the following meaning:

- Business Service status (first cell).
  - Green • (dot) — Active and OK.
  - Yellow - (hyphen) — Currently inactive, but otherwise OK.
  - Red ! (exclamation mark) — Error.
  - Gray X (letter X) — Disabled.
- Connection status (second cell). The connection status is meaningful for TCP, HTTP, FTP, and ODBC connections.
  - Green + (plus sign) — Connected.
  - Yellow o (letter o) — Listening.
  - Red x (letter x) — Disconnected.
  - Gray - (hyphen) — Not applicable, disabled, unscheduled, or not connecting.

If you hover over the name of the service, the hover text provides additional information. If you select on the name of the service, the left area is updated with details and also displays the following associated links:

- **Event Log** — Click to view the Event Log entries for the selected configuration item. For information, see "Viewing the Event Log."
- **Queue Contents** — Click to view the production queues. For information, see "Monitoring Production Queues," later in this chapter.

3.2.2 Output Connections

The **Output Connections** table (upper right) lists all outgoing connections to external systems. Each entry indicates following:

1. Business operation status
2. Business operation connection status
3. Business operation name
4. Number of messages processed since the production started

The statuses are indicated by the cell color. The item status and the connection status cells have the following meaning:

- Business Operation status (first cell).
  - Green • (dot) — Active and OK.
  - Yellow - (hyphen) — Currently inactive, but otherwise OK.
  - Red ! (exclamation mark) — Error.
  - Gray X (letter X) — Disabled.
  - Gray • (dot) — Retry. The business operation connection failed and the operation is retrying the connection.
- Connection status (second cell). The connection status is meaningful for TCP, HTTP, FTP, and ODBC connections.
  - Green + (plus sign) — Connected.
  - Yellow o (letter o) — Listening.
  - Red x (letter x) — Disconnected.
  - Gray - (hyphen) — Not applicable, disabled, unscheduled, or not connecting.

If you select the name of the operation, the left area is updated with details and the same links as for the **Input Connections** table.

3.2.3 Queues

The **Queues** table (lower left) lists the status of Ensemble internal message queues and how many messages are currently waiting in each queue.

This table uses the same icons and color-coding as the **Input Connections** table. If you click an item in this table, the left area is updated with details and the **Queue Contents** link.

3.2.4 Event Log

The **Event Log** (lower right) summarizes recent entries in the Event Log.

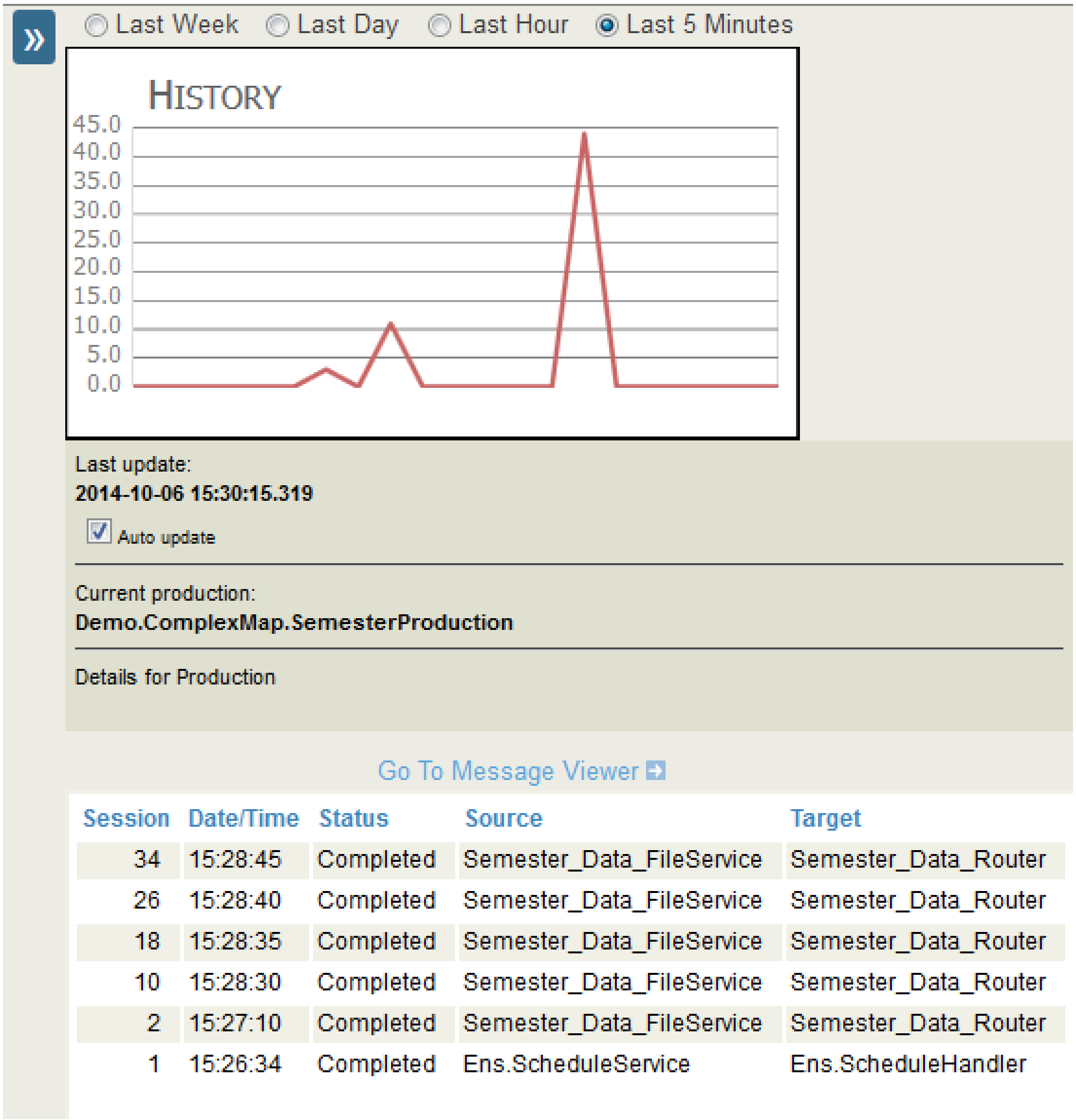
Each entry provides an icon and color to indicate the item’s status, as follows:

- Red ! — Error.
- Orange W — Warning.
- Yellow A — Alert.

If you select an item in this table, the left area is updated to show details of that Event Log entry. It also displays the **Event Log** link, which you can use to see the entire Event Log.

3.2.5 Activity Graph

The activity graph shows the message activity for the production or for a selected incoming or outgoing connection. The graph can show the message activity over a time period ranging from the previous 7 days to the previous 5 minutes. The following displays the activity graph or history of the production monitor:



○ Last Week ○ Last Day ○ Last Hour ● Last 5 Minutes

Last update:
2014-10-06 15:30:15.319

☑ Auto update

Current production:
Demo.ComplexMap.SemesterProduction

Details for Production

Session	Date/Time	Status	Source	Target
34	15:28:45	Completed	Semester_Data_FileService	Semester_Data_Router
26	15:28:40	Completed	Semester_Data_FileService	Semester_Data_Router
18	15:28:35	Completed	Semester_Data_FileService	Semester_Data_Router
10	15:28:30	Completed	Semester_Data_FileService	Semester_Data_Router
2	15:27:10	Completed	Semester_Data_FileService	Semester_Data_Router
1	15:26:34	Completed	Ens.ScheduleService	Ens.ScheduleHandler

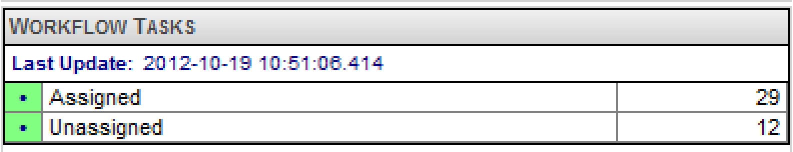
You can specify the following for the activity graph:

- Component to monitor—when you start the Production Monitor, the Activity Graph shows the messages for all incoming and outgoing connections. If you select an incoming or an outgoing connect on the Production Monitor, the Activity Graph shows the activity for the selected component only. If you want to return to the activity of the entire production, select on the currently selected connection to deselect it.
- **Auto update**—if this check box is selected, the Production Monitor regularly updates the Activity Graph.
- Time period to display—select one of the following:
  - **Last week**—display the activity for the previous 7 days. The vertical axis specifies the number of messages per hour.
  - **Last day**—display the activity for the previous 24 hours. The vertical axis specifies the number of messages per 15-minute interval.
  - **Last hour**—display the activity for the previous 60 minutes. The vertical axis specifies the number of messages per minute.

- **Last 5 minutes**—display the activity for the previous 5 minutes. The vertical axis specifies the number of messages per 15-second interval.

3.2.6 Custom Metrics

The bottom area of the page might display one or more tables of custom metrics added by your Ensemble developers. For example:



WORKFLOW TASKS

Last Update: 2012-10-19 10:51:06.414

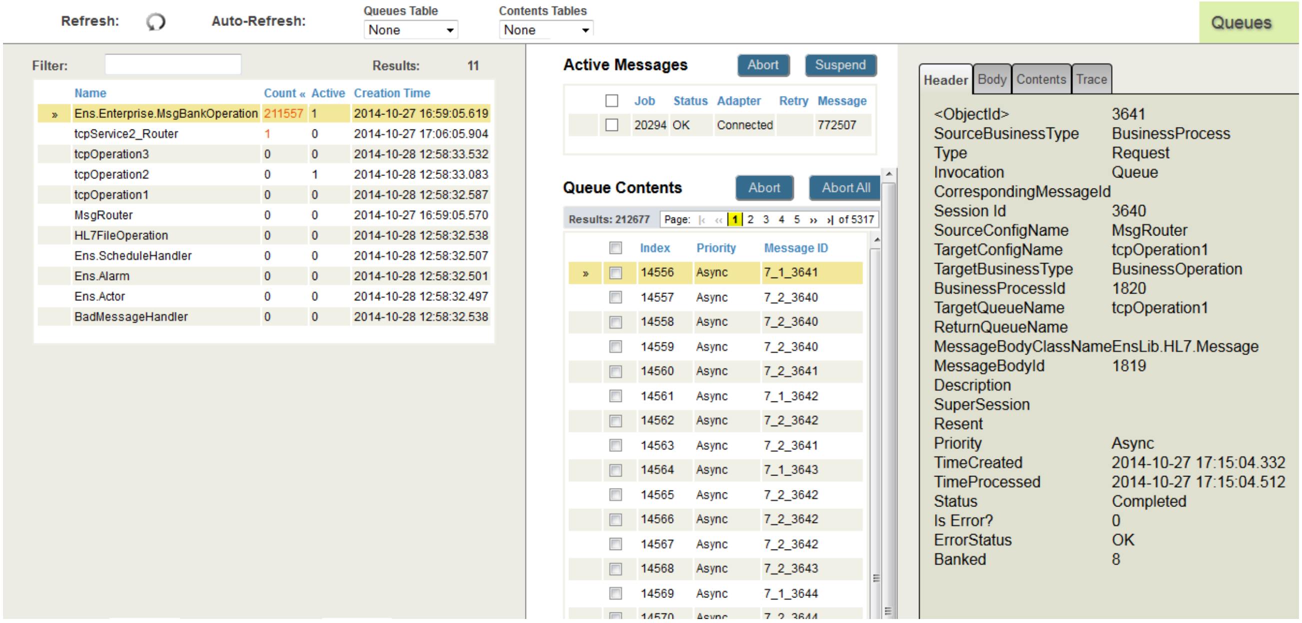
•	Assigned	29
•	Unassigned	12

See “Adding Business Metrics to the Production Monitor,” in *Developing Ensemble Productions*.

3.3 Monitoring Production Queues

The **Queues** page shows the current state of all the message queues being used by the running Ensemble production in the selected namespace.

To display this page in the Management Portal, select **Ensemble** > **Monitor** > **Queues**.



Queues Table: None

Contents Tables: None

Filter:

Results: 11

	Name	Count «	Active	Creation Time
»	Ens.Enterprise.MsgBankOperation	211557	1	2014-10-27 16:59:05.619
	tcpService2_Router	1	0	2014-10-27 17:06:05.904
	tcpOperation3	0	0	2014-10-28 12:58:33.532
	tcpOperation2	0	1	2014-10-28 12:58:33.083
	tcpOperation1	0	0	2014-10-28 12:58:32.587
	MsgRouter	0	0	2014-10-27 16:59:05.570
	HL7FileOperation	0	0	2014-10-28 12:58:32.538
	Ens.ScheduleHandler	0	0	2014-10-28 12:58:32.507
	Ens.Alarm	0	0	2014-10-28 12:58:32.501
	Ens.Actor	0	0	2014-10-28 12:58:32.497
	BadMessageHandler	0	0	2014-10-28 12:58:32.538

Active Messages

☐	Job	Status	Adapter	Retry	Message
☐	20294	OK	Connected		772507

Queue Contents

Results: 212677 Page: |‹ ‹‹ 1 2 3 4 5 ›› ›| of 5317

	☐	Index	Priority	Message ID
»	☐	14556	Async	7_1_3641
	☐	14557	Async	7_2_3640
	☐	14558	Async	7_2_3640
	☐	14559	Async	7_2_3640
	☐	14560	Async	7_2_3641
	☐	14561	Async	7_1_3642
	☐	14562	Async	7_2_3642
	☐	14563	Async	7_2_3641
	☐	14564	Async	7_1_3643
	☐	14565	Async	7_2_3642
	☐	14566	Async	7_2_3642
	☐	14567	Async	7_2_3642
	☐	14568	Async	7_2_3643
	☐	14569	Async	7_1_3644
	☐	14570	Async	7_2_3644

<ObjectId>	3641
SourceBusinessType	BusinessProcess
Type	Request
Invocation	Queue
CorrespondingMessageId	
Session Id	3640
SourceConfigName	MsgRouter
TargetConfigName	tcpOperation1
TargetBusinessType	BusinessOperation
BusinessProcessId	1820
TargetQueueName	tcpOperation1
ReturnQueueName	
MessageBodyClassName	EnsLib.HL7.Message
MessageBodyId	1819
Description	
SuperSession	
Resent	
Priority	Async
TimeCreated	2014-10-27 17:15:04.332
TimeProcessed	2014-10-27 17:15:04.512
Status	Completed
Is Error?	0
ErrorStatus	OK
Banked	8

The table on this page has one row for each queue. The columns in this table are as follows:

- **Name** — The name of the configuration item that has the queue. It may be different from the host class name.
- **Count** — How many messages are on the queue. This value is a snapshot and may change when you refresh the page.
- **Active** — The number of active messages.
- **Creation Time** — The date and time when the queue was first created.

To see the contents of any given queue, select the row for that queue. The active messages and queue contents for that queue are displayed. If you select an entry in the queue contents or active messages, information about the message is displayed.

You can refresh the list of queues and contents by clicking the refresh arrow. You can also specify the time period to automatically refresh the list of queues and the active messages and queue contents tables.

The **Active Messages** table is displayed when there are active messages in the selected queue. It has one row for each active message, which identifies the message and its state. If you select one or more messages by checking the check box, you can abort or select the selected messages.

In the **Active Messages** table, you can select a message row to view the details of the selected message. The details are displayed to the right in the **Header**, **Body**, **Contents**, and **Trace** tabs. These tabs are the same as in the **Message Viewer** page; see "Viewing, Searching, and Managing Messages,"

The **Queue Contents** table on this page is displayed if there are messages in the selected queue. It has one row for each message in the given queue. The columns in this table are as follows:

- **Index** — This integer value starts at 1 for the first message placed on the queue after the production starts, and increments by 1 for each successive message. A message has the same **Index** value for the entire time it is on the queue. **Index** values are never reused.
- **Priority** — The priority of the message. See "Message Priority" in the first chapter.
- **MessageId** — The object identifier for the message.

In the **Queue Content** table, you can perform the following tasks:

- Select a message row to view the details of the selected message. The details are displayed to the right in the **Header**, **Body**, **Contents**, and **Trace** tabs. These tabs are the same as in the **Message Viewer** page; see "Viewing, Searching, and Managing Messages,"
- Select messages by checking the check box for the messages.
- Click **Abort** to abandon any ongoing attempts to send one or more messages selected with the check box. Click **OK** to verify the operation.
- Click **Abort All** to abandon ongoing attempts to send all the messages in the queue. You must then click **OK** to verify the operation.
- Select a page number to view that page in the list. Selecting **|<** displays the first page, **<<** displays the previous page, **>>** displays the next page, and **>|** displays the last page.

3.3.1 Diagnosing Problems with Queues

By looking at queues and jobs, you can often quickly spot a problem in the system.

When there is buildup on a queue, it usually means something needs to be repaired. Usually the most important information about queues is the destination, or "target," of any message that has been too long on a queue. In general, when a queued message is not being sent, it is because it cannot get to its target. If you can find out what is causing a problem with the target, when you solve that problem, the queue buildup will generally disappear. For example:

- For a business service or business operation, if a queue is *suddenly* longer, this generally means that there is a problem communicating with an external system. An external connection may be down, or there may be a peak-hour effect that is affecting throughput on your external connections.
- For a business service or business operation, if a queue is *consistently* long, this generally means that there is a consistent delay in sending messages. You should probably examine the external connection to see if there a performance problem that you can solve. If that is not possible, you could increase the appropriate pool size (unless you need to ensure first-in-first-out processing).

See “Pool Size and Actor Pool Size” in *Configuring Ensemble Productions*.

- For a business process that uses a private pool, if a queue is consistently long, you could increase the appropriate pool size (unless you need to ensure first-in-first-out processing).
- If the actor queue is suddenly longer, a business process may have experienced an error that has caused it to become “stuck” in some way.
- If the actor queue is consistently long, the actor pool for the production may need to be larger.
- If many queues have a consistently large buildup, there may be a general capacity issue on the host computer, the Ensemble production (in its role as a CSP application) may need more resources, or the underlying Caché installation may need to be tuned. For suggestions, see the *Caché System Administration Guide* and *Using Caché Server Pages (CSP)*. In general, however, you will be able to keep queues moving with the simpler adjustments listed in this topic.

3.4 Monitoring Active Jobs

The **Currently Active Jobs** page shows the currently active jobs for the production in the selected namespace.

To display this page in the Management Portal, select **Ensemble** > **Monitor** > **Jobs**.

The table on this page has one row for each active job. The columns in this table are as follows:

- **Job** — Internal numeric identifier of the job.
- **Configuration Name** — Configuration name of the business host for which this job was started.

  Each time a business service, business process, or business operation needs to do work, it starts a system job in which to complete its tasks. This job comes either from a private pool of jobs belonging to the business service, business process, or business operation, or (in the case of a business operation) it may come from the public actor pool for the production. When the task is done, the job returns itself to the pool of jobs from which it came.

  A production might need to start and stop several different jobs to complete a single request. The details depend (in part) on whether requests are made synchronously or asynchronously. For example, if a job is required to wait, the job returns itself to its pool during the wait time to free up that resource.
- **Mode** — Either `Background` or `Foreground`.
- **Status** — Typically, this is `running` or `dequeuing`.
- **Detail** — Any additional detail that is available for the job.
- **Active Message** — ID of the message currently being processed, if any.
- **State** — Typically, this is `active`.

3.4.1 Diagnosing Problems with Jobs

By looking at jobs and queues, you can often quickly spot a problem in the system.

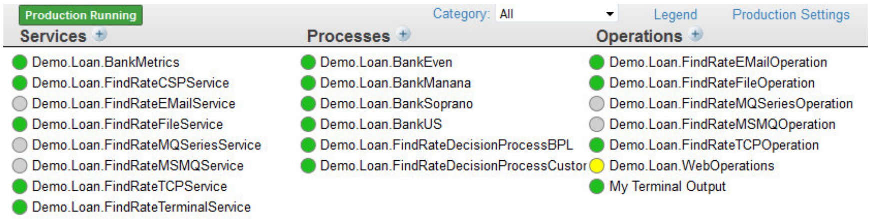
Most jobs spend most of their time in a `dequeuing` state while they wait for messages. During shutdown they should become quiescent. If the job does not become quiescent during shutdown, that likely indicates a problem. If the job is constantly in a `running` state, that also indicates a problem, unless you expect the component to be doing a lot of processing (and it is actually completing this processing).

Jobs that are marked as `dead` are jobs that have been terminated for some reason and Ensemble has detected that the job is no longer present on the system. This is normally an indication of a serious problem and should not occur. Also, if Ensemble detects a dead job, it writes an error to the Event Log.

3.5 Using the Production Configuration Page

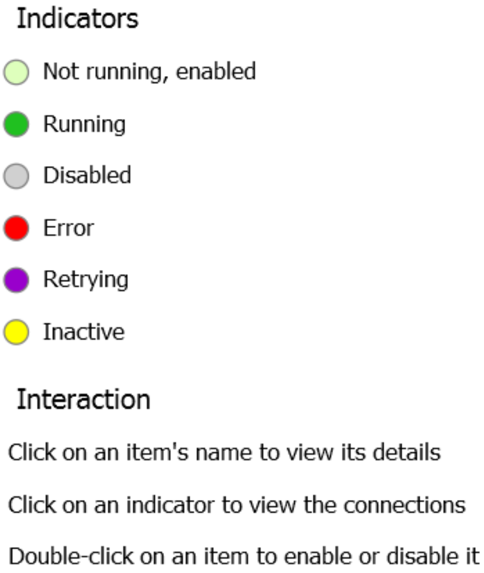
Ensemble provides another way to view a production, the **Production Configuration** page. To access this page, select **Ensemble** > **Configure** > **Production**.

This page displays the business hosts in the production, with useful color coding as in the following example:



Services	Processes	Operations
🟢 Demo.Loan.BankMetrics	🟢 Demo.Loan.BankEven	🟢 Demo.Loan.FindRateEMailOperation
🟢 Demo.Loan.FindRateCSPService	🟢 Demo.Loan.BankManana	🟢 Demo.Loan.FindRateFileOperation
⚪ Demo.Loan.FindRateEMailService	🟢 Demo.Loan.BankSoprano	⚪ Demo.Loan.FindRateMQSeriesOperation
🟢 Demo.Loan.FindRateFileService	🟢 Demo.Loan.BankUS	⚪ Demo.Loan.FindRateMSMQOperation
⚪ Demo.Loan.FindRateMQSeriesService	🟢 Demo.Loan.FindRateDecisionProcessBPL	🟢 Demo.Loan.FindRateTCPOperation
⚪ Demo.Loan.FindRateMSMQService	🟢 Demo.Loan.FindRateDecisionProcessCustor	🟡 Demo.Loan.WebOperations
🟢 Demo.Loan.FindRateTCPService		🟢 My Terminal Output
🟢 Demo.Loan.FindRateTerminalService		

This page displays a circular status indicator next to each business host. If you click **Legend** to see the meaning of this indicator, Ensemble displays the following:



Indicators

- ○ Not running, enabled
- 🟢 Running
- ⚪ Disabled
- 🔴 Error
- 🟣 Retrying
- 🟡 Inactive

Interaction

Click on an item's name to view its details

Click on an indicator to view the connections

Double-click on an item to enable or disable it

Note that the primary purpose of this page is for configuring productions as described in *Configuring Ensemble Productions*.

3.6 Correcting Production Problem States

If a production is Suspended or Troubled, read this section.

If the state of a production is Running, then a production has been started and is operating normally. This is an acceptable state.

If the state of a production is Stopped, it is not running and all of its queues are free of synchronous messages. This is also an acceptable state.

In some cases (usually during development), you might see the **Update** button on this page for a production that is Running. Click this, and Ensemble updates the production to resolve the discrepancy. For an explanation, see “The Update Button” in *Configuring Ensemble Productions*.

3.6.1 Recovering a Suspended Production

A production acquires the Suspended status when, at the end of the shutdown sequence, some queues still contain synchronous messages.

You can start the Suspended production again to permit these messages to be processed. However, if the underlying problem is not resolved, you might acquire more synchronous messages in the queue without processing the previous messages.

Therefore, if a live, deployed Ensemble production goes into a Suspended state, contact the InterSystems Worldwide Response Center (WRC) for assistance.

If a production becomes Suspended during development, see "Correcting Production Problem States" in *Developing Ensemble Productions*. In this case, you can use a procedure that discards the messages.

3.6.2 Recovering a Troubled Production

A production acquires a status of Troubled if Ensemble is stopped but the production did not shut down properly. This can happen if you restarted Ensemble or rebooted the machine without first stopping the production. In this case, click the **Recover** button.

4

Viewing, Searching, and Managing Messages

All communication within a production is accomplished with *messages*, and the Management Portal provides many tools for viewing and working with messages. This chapter includes the following sections:

- Browsing the Messages
- Filtering the Messages
- Filtering with Basic Criteria
- Filtering with Extended Criteria
- Viewing the Message Details
- Tracing the Path of Related Messages
- Resending Messages
- Viewing the SQL Query Used by the Message Viewer
- Managing Suspended Messages

Also see the chapter "Viewing Messages from Other Productions."

For background information on messages, see the chapter "Concepts."

4.1 Browsing the Messages

You can view information about the messages that your Ensemble production has sent or queued. To access the **Message Viewer** page, do either of the following in the Management Portal:

- Click **Messages** on the Ensemble **View** menu.
- Click **Go to Message Viewer** at the top of the **Messages** tab on the **Production Configuration** page.

The middle area lists the messages. To refresh this area, click the **Search** button. You can use the bottom left area to filter the list of messages; for information, see the following section. The right area displays details; see "Viewing Message Details" and "Tracing the Path of Related Messages."

4.1.1 Available Information

The top area of this page displays the following information for each message:

- **ID** — The ID of the message. See "Message Basics," in the first chapter.
- **Time Created** — The message creation time stamp. See "Invocation Style and Message Time Stamps," in the first chapter.
- **Session** — The ID of the session associated with this message. See "Sessions," in the first chapter.

  You can select the **Session** number in any row of the table to see a visual trace of the message object through the production.
- **Status** — Indicates the status of the message. See "Message Status," in the first chapter.
- **Error** — Provides a quick overview of the results returned by the message to the business host that sent it.

  `OK` means normal behavior; `Retry` means the message experienced a failure but the business host that sent it is configured to retry the failed message. `Error` means an error was reported somewhere in the activity. `Inactive` means that the business host that sent the message has been idle for longer than its Inactivity Timeout setting, and may require diagnostic action.
- **Source** — The business host that sent the message.
- **Target** — The business host to which the message was sent.

The **Session** column also uses color as follows:

Background Color	Indication
Red	The message encountered an error.
Green	The message marks the start of a session.
Silver	The message arrived after a timeout expired and is marked as discarded.
Orange	The message is suspended.
White and pale tan, in alternating rows	These messages are OK or are queued.

If you are using the Message Bank Viewer, there is an additional option to specify the Message Bank client to search. See Using the Enterprise Message Bank

4.1.2 Paging Through the Messages

Typically there are multiple pages of messages. To see all the messages, you have the following options:

- You can display the next page of messages. To do so, click **Next**.
- You can display more messages. To do so, select a larger value for **Page Size** and click **Search** again.

  The default is 100 messages.
- You can display the previous page of messages. To do so, click **Previous**.
- You can change how the messages are sorted. To do so, click a different value for **Sort Order**.

Also use **Time Format** to specify whether to show only the time or the complete time with the date. The default is `Time Only`.

The read-only **Page** field indicates which page of the list is being displayed.

4.2 Filtering the Messages

To find a specific message more easily, you can filter the messages shown in the **Message Viewer** page. The basic process is as follows:

1. Access this page as described in previously.
2. Specify the filter criteria. In general, you can do this in two different ways:
   - Use the Basic Criteria and Extended Criteria areas to specify filter criteria, as described in the following sections.
   - Use the **Saved Searches** area to retrieve a previously saved set of filter criteria. To do so, select a value from the drop-down list and then click the check mark.
3. Click **Search**. The page is redisplayed with a list of messages that match your filter criteria. If the search has not yet completed, you can interrupt it by clicking **Cancel**.

   Or click **Reset** to restore the default criteria.
4. If more matches are found than can be displayed, the **Next** button is active, and you can use that. Or, to display more data, select a larger value for **Page Size** and click **Search** again. Or adjust your filter criteria to narrow the search.
5. Optionally click **Save** or **Save As** to save the search criteria for later reuse. Ensemble then displays a field in which you provide a name for the search criteria. Enter a value and click the check mark.

   This operation overwrites any previously saved criteria with the same name.

   To delete a saved search, click its name in the **Saved Searches** list and then click the red X.

4.3 Filtering with Basic Criteria

To filter the messages shown in the **Message Viewer** page, specify some or all of the following fields in the **Basic Criteria** area:

- **Status** — Select a value from the drop-down list. See “Message Status,” in the first chapter.
- **Type** — Select a value from the drop-down list: Session Start, Request, Response, or All (the default).
- **Start Time** — Enter the earliest desired message creation time stamp. See “Invocation Style and Message Time Stamps,” in the first chapter.
- **Start Id** — Enter the lowest desired message ID.
- **End Time** — Enter the latest desired message creation time stamp.
- **End Id** — Enter the highest desired message ID.
- **Source** — The business host that sent the message. Choose from the list.
- **Target** — The business host that is the target of the message. Choose from the list.

If you are using the Message Bank Viewer, there is an additional filter that restricts the search to a single Message Bank client. See Using the Enterprise Message Bank.

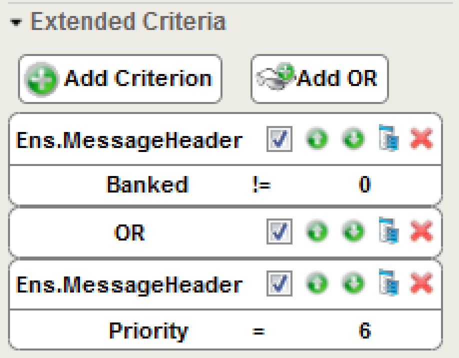
4.4 Filtering with Extended Criteria

The **Extended Criteria** area enables you to filter the displayed messages by extremely specific criteria. An advanced filter consists of one or more conditions, combined with the logical operators AND and OR. Each condition can use any information contained in the messages, comparison operators from a rich set, and your arbitrary expressions. Only messages that meet all the combined conditions are displayed.

To use this area, click the triangle next to **Extended Criteria**. Then do either of the following:

- To add a criterion, click **Add Criterion**. See the first subsection for details.
- To add an OR, click **Add OR**. By default, the criteria are combined with AND. Use this option to combine adjacent criteria instead with OR. See the subsection “How Criteria Are Combined.”

After you add these items, the **Extended Criteria** list displays your selections. For example:

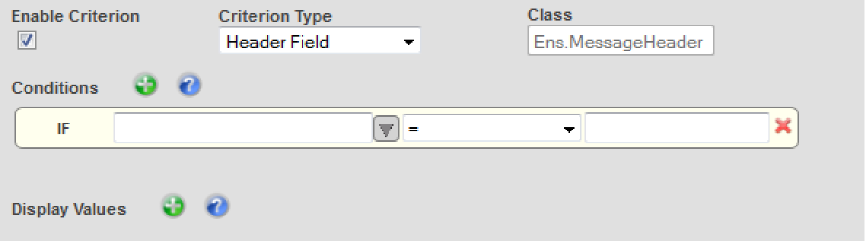


Ens.MessageHeader ☑
Banked != 0
OR ☑
Ens.MessageHeader ☑
Priority = 6

When you are satisfied with your selections, click **Search**. The Message Viewer page displays the list of messages that match all your filter criteria.

4.4.1 Adding a Criterion

To add a criterion, click **Add Criterion**. Ensemble displays a wizard as follows:



Enable Criterion ☑
Criterion Type: Header Field
Class: Ens.MessageHeader
IF =

Specify the following values:

- **Enable Criterion** — Select to enable this search.
- **Criterion Type** — Choose a value from the list. See the next subsection.
- **Class** — Click a class name from the list. See the “Class” subsection.
- **Conditions** — Allows you to specify fields and values for your logical statement. See the “Conditions” subsection.
- **Display Values** — Allows you to specify additional values to display in the table. Your chosen values are displayed on the right side of the table.

Click **OK** to save this criterion and add it to the **Extended Criteria** list.

4.4.2 Criterion Type

For **Criterion Type**, select a value from the drop-down list, if applicable. The following table lists the choices and how they affect your subsequent choices in the **Class** and **Conditions** fields.

Type	Class and Conditions refer to...
Body Property	Properties of a standard Ensemble message body object.
Header Field	Fields in a standard Ensemble message header object.
OR	(used to logical OR two filter terms)
SearchTable Field	Entries in a search table class that you have defined in this Ensemble namespace. A search table class is a specialized tool that you create to work with virtual documents.
VDoc Segment Field	Fields in a virtual document message segment. Identify the standard and the segment of interest. Ensemble then prompts you to choose from a list of fields in that segment.
VDoc Property Path	Fields in a virtual document message segment. Identify the standard and then enter a virtual property path that identifies a message segment and field that is valid for that standard.

Note: For background information about the VDoc fields in the **Extended Criteria** interface, see *Ensemble Virtual Documents*. You do not need to use these fields unless your production routes some type of virtual document.

4.4.3 Class

For **Class**, select a value from the drop-down list, if applicable. Ensemble lists all the classes appropriate for the selected **Criterion Type**. For example:

Type	Class Name
Body Property	Choose from all the message classes in this namespace.
Header Field	—
OR	—
SearchTable Field	Choose from all the search table classes in this namespace.
VDoc Segment Field	Choose from all the virtual document classes in this namespace.
VDoc Property Path	Choose from all the virtual document classes in this namespace.

4.4.4 Filter Conditions

For **Conditions**, specify fields and values for your logical statement, from left to right, as follows:

1. For the first cell, select a value from the drop-down list, which includes all choices appropriate for this context. For further instructions, see the first table below.
2. For the second cell, select a comparison operator from the drop-down list. See the second table below.
3. In the third cell, type the literal string that you intend to match using the selected operator.

Do not use double quotes around the string.

Choices in the **Conditions** panel vary according to your choice of **Type**. The following table describes the choices.

Type	Conditions
Body Property	Choose from all the properties in the **Class Name** message class.
Header Field	—
OR	—
SearchTable Field	Choose from all the search table entries defined in the **Class Name** search table class.
VDoc Segment Field	Select a value for **Segment Type** and then select a value for **Field Name**. (Or type values, if you know the applicable values.) For HL7, you can type a numeric references if you prefer them to names, for example [5], [18.1], or 2.3.1:[3().1]. You may edit out the category reference and colon prefix, but keep the square brackets and their contents intact. Square brackets differ from curly brackets in that square brackets enclose a *segment*:*field* combination that does not require you to identify its containing message structure. In the example above, Ensemble matches any message structure in the `2.3.1` schema category that contains a `PID` segment with a `PatientAddress().city` field .
VDoc Property Path	Select a value for **DocType** and then select a value for **Property Path**. (Or type values, if you know the applicable values.) Instead of selecting options to fill the left-hand **Conditions** field, you can type a virtual property path into the field, as long as you are careful to use the correct syntax. Curly bracket syntax requires a specific message structure, such as ADT_A06, to be identified.

The comparison operator between the two values in a **Conditions** statement can be any one of the following.

Operator	The condition is true when the value at left is...
=	Equal to the value on the right.
!=	*Not* equal to the value on the right.
>	Greater than the value to the right of the operator.
>=	Greater than or equal to the value to the right.
<	Less than the value to the right.
<=	Less than or equal to the value to the right. If a condition >, >=, <, or <= involves strings, they are sorted alphabetically to determine the result. Symbols and numbers sort *before* alphabetic characters.

Operator	The condition is true when the value at left is...
Contains	A string that contains the substring to the right. The Contains operator is case-sensitive (except possibly within search table fields). If the value at left is `Hollywood, California` and the value at right is `od, Ca`, there is a match, but a value of `Wood` does not match. The Contains operator might or might not be case-sensitive in search table fields, depending on the implementation of a particular search table class.
DoesNotContain	A string that does *not* contain the substring at right.
DoesNotMatch	A string that does *not* match the pattern in the string specified to the right, which uses syntax suitable for the ? pattern matching operator in ObjectScript. For details, see “Pattern Matching” in the chapter “Operators and Expressions” of *Using Caché ObjectScript*.
In	Identical to one of the items in the comma-delimited string at right.
NotIn	Identical to *none* of the items in the comma-delimited string at right.
StartsWith	A string that starts with the substring at right.
DoesNotStartWith	A string that does *not* start with the substring at right.
Like	A string that matches the pattern in the substring specified to the right, according to the rules for the LIKE predicate in SQL. Matching for the Like and NotLike condition may be summarized as follows: The character “_” matches any single character, and the character “%” matches any sequence of zero or more characters. Thus, if the value at left contains the pattern `%Com_` and the selected operator is Like, values of `TransCom1` and `UltraCom2` match, but values of `UltraCom17` and `Foxcom8` do not match.
Matches	A string that matches the pattern in the string specified to the right, which uses syntax suitable for the ? pattern matching operator in ObjectScript. For details, see “Pattern Matching” in the chapter “Operators and Expressions” of *Using Caché ObjectScript*.
NotLike	A string that does *not* match the pattern in the substring specified to the right, according to the rules for the LIKE predicate in SQL.
InFile	Found in the text file whose full pathname is specified to the right.
NotInFile	*Not* found in the text file whose full pathname is specified to the right.

Important: When Ensemble indexes virtual documents (thus adding to the search tables), it replaces any vertical bar (|) with a plus sign (+). Take this into consideration when you use the search table to search for content. For example, to search for a message that contains `my|string`, use `my+string` as the search criterion.

4.4.5 Rearranging and Modifying Criteria

If you have multiple items in the **Extended Criteria** section, you can click the up-arrow and down-arrow icons to adjust their order.

To edit an item, click the edit button for that item.

To delete an item, click **X**.

4.4.6 How Criteria Are Combined

If **Extended Criteria** displays contains multiple criteria, they are implicitly joined by AND. For example, suppose that you have three statements visible:

```
Logical Statement 1
Logical Statement 2
Logical Statement 3
```

In this case, your filter actually works like this:

```
Logical Statement 1
AND
Logical Statement 2
AND
Logical Statement 3
```

To modify this logic, use **Add OR** and reposition the OR as needed. Suppose you added an OR row and a fourth logical statement to the list shown above. The **Extended Criteria** panel now looks like this:

```
Logical Statement 1
Logical Statement 2
Logical Statement 3
OR
Logical Statement 4
```

And the resulting logic is now:

```
Logical Statement 1
AND
Logical Statement 2
AND
Logical Statement 3
OR
Logical Statement 4
```

The operator AND binds more tightly than OR, so the effect of the above sequence is actually:

```
(1 AND 2 AND 3) OR 4
```

4.5 Viewing Message Details

For each message, Ensemble provides details about how the message was created and sent. You can access the relevant page from multiple places. For example:

1. Access the **Message Viewer** page as described in "Browsing the Messages."
2. Click a message.

The system displays the following tabs in the right pane:

- **Header** — Displays the fields in the message header.
- **Body** — Displays the fields in the message body.
- **Contents** — Displays contents of the message body in an appropriate format.
- **Trace** — Displays a small visual trace of the message and related messages through the production. To see the larger version (the **Visual Trace** page), click **View Full Trace**.

The following subsections describe the **Header**, **Body**, and **Contents** tabs.

The **Trace** tab displays a subset of the data and options that you see on the larger **Visual Trace** page. The **Visual Trace** page is described in the next section.

4.5.1 Message Header Fields

The **Header** tab displays the standard fields in any Ensemble message header:

- **<ObjectId>** — The ID of the message header (and also the message ID; see “Message Basics,” in the first chapter).
- **TargetConfigName** — The name of the business host that is intended to receive the message.
- **Type** — The message type, Request or Response.
- **Invocation** — Indicates how the message was sent. See “Invocation Style and Message Time Stamps,” in the first chapter.
- **CorrespondingMessageId** — For a request message, this field contains the message ID of the corresponding response (if any) or it is blank. For a response message, this field contains the message ID of the corresponding request.
- **SessionId** — The ID of the session associated with this message. See “Sessions,” in the first chapter.
- **SourceConfigName** — The business host that sent the message.
- **SourceBusinessType** — BusinessService, BusinessProcess, BusinessOperation, or Unknown.
- **TargetBusinessType** — BusinessService, BusinessProcess, BusinessOperation, or Unknown.
- **BusinessProcessId** — Every business process that gets executed has an instance and this is the object ID of that instance. If the message is a request, this field identifies the business process in whose context the message was created (sender). If the message is a response, this field identifies the business process to which it is being returned (receiver). This field is empty in various circumstances, for example if an error occurred.
- **TargetQueueName** — The destination “address” for the message, this indicates where it is going:
  - If this is a name, it identifies a public queue, such as Ens.Actor.
  - If this is a number, it identifies the job ID associated with the private queue of a business host.
- **ReturnQueueName** — The return “address” for the message, this indicates where it came from:
  - If this is a name, it identifies a public queue, such as Ens.Actor.
  - If this is a number, it identifies the job ID associated with the private queue of a business host.

  The **ReturnQueueId** value is listed even if there is no response expected or needed for a particular request message type.
- **MessageBodyClassName** — The class name for the message body.
- **MessageBodyId** — The ID for the message body. This field matches the **<ObjectId>** field in the **Message Body** table.
- **Description** — A text description of the message. The Ensemble Business Process Language (BPL) provides text in this field automatically, based on the type of BPL activity that generated the message.
- **SuperSession** — The ID for messages sent via HTTP from one production to another. For details, see “SendSuperSession” in *Using HTTP Adapters with Ensemble*.
- **Resent** — Indicates whether this is a resent message.
- **Priority** — The priority of the message relative to others in the queue, as assigned by the Ensemble messaging engine. See “Message Priority,” in the first chapter.
- **TimeCreated** — The message creation time stamp. See “Invocation Style and Message Time Stamps,” in the first chapter.

- **TimeProcessed** — The message usage time stamp. Ensemble sets this field when the message is taken off of the queue but then resets it to the current time while the message is being processed. Typically, for a completed message, it represents the time that the message processing was completed.
- **Status** — Indicates the status of the message. See "Message Status," in the first chapter.
- **IsError?** — The value 1 means that the message encountered an error. The value 0 means the message did not encounter any errors.
- **ErrorStatus** — If **IsError?** is 1, then this is the text associated with the error. When **IsError?** is 0, **ErrorStatus** is the string "OK".
- **Banked** — Indicates whether or not this message is part of a message bank.

4.5.2 Message Body Fields

The **Body** tab displays the message body information. Fields include:

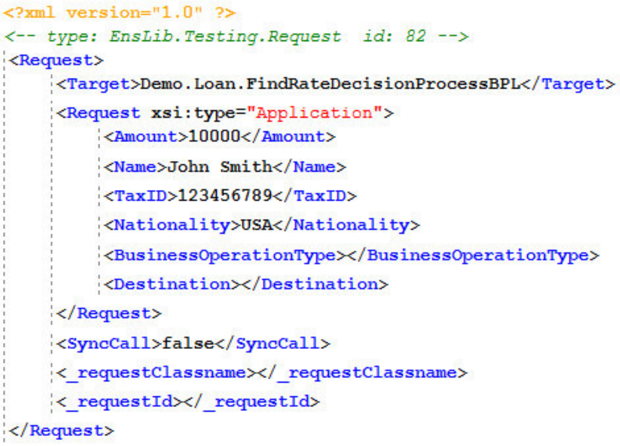
- The message body class name above the list of fields.
- If the message body is a standard Ensemble message body object, a table displays the following information:
  - **<ObjectId>** — The object identifier for the message body. This field matches the **MessageBodyId** field in the **Header** tab.
  - The name and value of each property in the **Message Type** class.

  If the message body is any other type, there are no additional fields in the display.

4.5.3 Message Contents

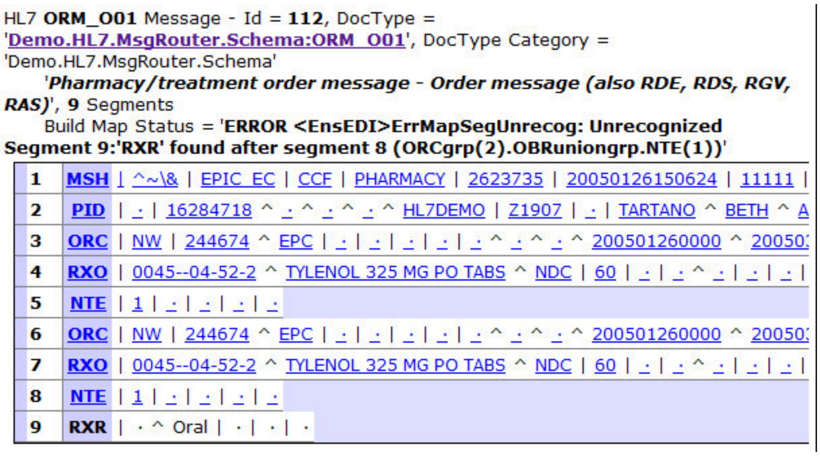
The **Contents** tab displays formatted contents of the message body.

The standard Ensemble message body appears in colorized XML format, as shown in the following example:



```
<?xml version="1.0" ?>
<-- type: EnsLib.Testing.Request  id: 82 -->
<Request>
    <Target>Demo.Loan.FindRateDecisionProcessBPL</Target>
    <Request xsi:type="Application">
        <Amount>10000</Amount>
        <Name>John Smith</Name>
        <TaxID>123456789</TaxID>
        <Nationality>USA</Nationality>
        <BusinessOperationType></BusinessOperationType>
        <Destination></Destination>
    </Request>
    <SyncCall>false</SyncCall>
    <_requestClassname></_requestClassname>
    <_requestId></_requestId>
</Request>
```

A virtual document, such as an HL7 message, appears in segments, with one segment per line, as shown in the following figure:



HL7 **ORM_O01** Message - Id = **112**, DocType = '[Demo.HL7.MsgRouter.Schema:ORM_O01]', DocType Category = 'Demo.HL7.MsgRouter.Schema'

'***Pharmacy/treatment order message - Order message (also RDE, RDS, RGV, RAS)***', **9** Segments

Build Map Status = '**ERROR <EnsEDI>ErrMapSegUnrecog: Unrecognized Segment 9:'RXR' found after segment 8 (ORCgrp(2).OBRuniongrp.NTE(1))**'

<table>
<tr><td>1</td><td>MSH</td><td>| ^~\& | EPIC_EC | CCF | PHARMACY | 2623735 | 20050126150624 | 11111 |</td></tr>
<tr><td>2</td><td>PID</td><td>| · | 16284718 ^ · ^ · ^ · ^ HL7DEMO | Z1907 | · | TARTANO ^ BETH ^ A</td></tr>
<tr><td>3</td><td>ORC</td><td>| NW | 244674 ^ EPC | · | · | · | · | · ^ · ^ · ^ 200501260000 ^ 20050</td></tr>
<tr><td>4</td><td>RXO</td><td>| 0045--04-52-2 ^ TYLENOL 325 MG PO TABS ^ NDC | 60 | · | · ^ · | · | · |</td></tr>
<tr><td>5</td><td>NTE</td><td>| 1 | · | · | · | ·</td></tr>
<tr><td>6</td><td>ORC</td><td>| NW | 244674 ^ EPC | · | · | · | · | · ^ · ^ · ^ 200501260000 ^ 20050</td></tr>
<tr><td>7</td><td>RXO</td><td>| 0045--04-52-2 ^ TYLENOL 325 MG PO TABS ^ NDC | 60 | · | · ^ · | · | · |</td></tr>
<tr><td>8</td><td>NTE</td><td>| 1 | · | · | · | ·</td></tr>
<tr><td>9</td><td>RXR</td><td>| · ^ Oral | · | · | ·</td></tr>
</table>

For example, for HL7 messages, each line displays the following, from left to right:

- The segment number on a white background: 1, 2, 3, etc.
- The segment name on a shaded background: MSH, PID, etc. If you hover the cursor over a segment name in the shaded column, a tooltip describes the purpose of the message segment.
- The field contents and separator characters as provided in the message. If you hover the cursor over a field in this display, a tooltip provides syntax information that a developer can use to access fields within the message body from Ensemble data transformations and routing rules.

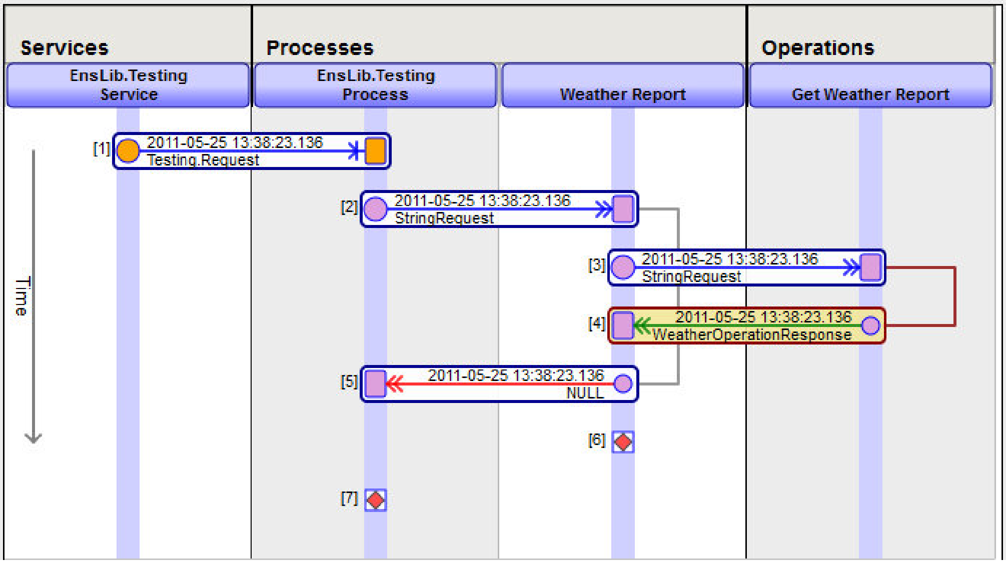
To view all the contents of large virtual document, it may be necessary to drag the bottom scroll bar far to the right. To view the message in a wider display, click the **View Full Contents** link or the **View Raw Contents** link. **View Full Contents** displays the message formatted by fields and **View Raw Contents** displays the unprocessed message contents, which can easily be copied and pasted into a text editor. For background information, see *Ensemble Virtual Documents*. Or see the book for a specific kind of virtual document.

4.6 Tracing the Path of Related Messages

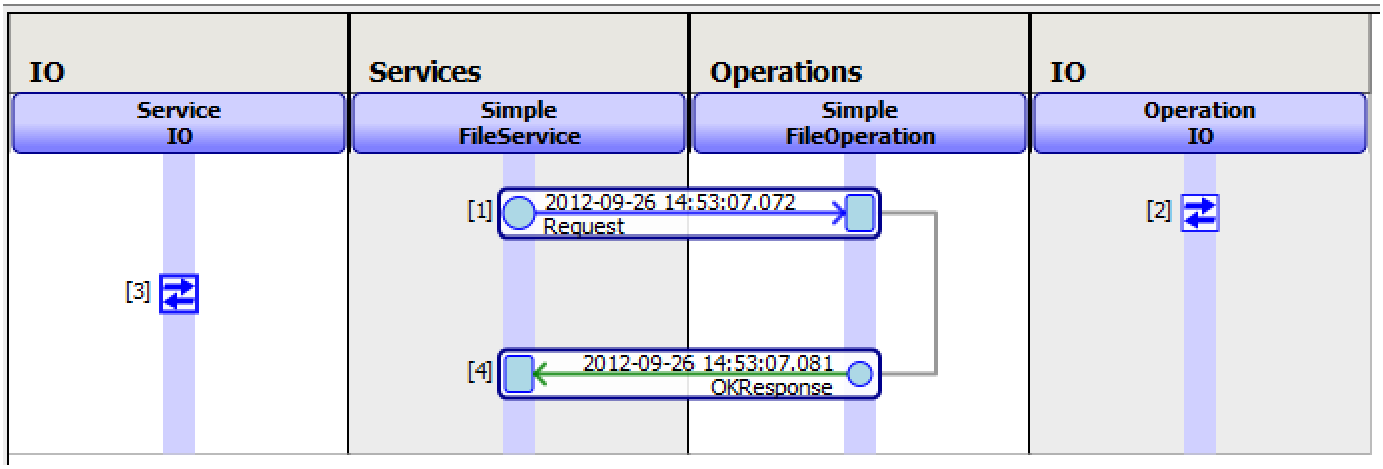
The **Visual Trace** page enables you to visually trace the path of a set of related messages between business hosts. You can access this tool from multiple places. For example:

1. Access the **Message Viewer** page as described in "Browsing the Messages."
2. Click a message.
3. Click the **Trace** tab, which displays a small version of the trace.
4. Click **View Full Trace**.

The left area of the **Visual Trace** page displays a visual representation of message activity, with one column for each business host that handles the message. The business hosts are grouped into business services, business processes, and business operations.


Services
Processes
Operations
EnsLib.Testing Service
EnsLib.Testing Process
Weather Report
Get Weather Report
Time
[1] 2011-05-25 13:38:23.136 Testing.Request
[2] 2011-05-25 13:38:23.136 StringRequest
[3] 2011-05-25 13:38:23.136 StringRequest
[4] 2011-05-25 13:38:23.136 WeatherOperationResponse
[5] 2011-05-25 13:38:23.136 NULL
[6]
[7]

If you enable the Archive IO setting for one or more business service and business operations, the **Visual Trace** also displays the input and output data in addition to the Ensemble messages. For example:



Note: When you view calls with this tool, synchronous calls from long-running business processes are portrayed as if they were asynchronous. This does not change the fact that the calls are actually synchronous. It is a side effect of the internal tracking mechanism that Ensemble uses to free system resources while it is waiting for a synchronous call to return.

When a message has traveled from one item to the next, an arrow connects the two items within a rounded box:

- The source item is marked with a circle; this is the item that sends the message.
- The target item is marked with a rounded rectangle. The arrow points to this item, away from the source item.

In both cases, you can read the name of the business host at the top of the column.

Each of these rounded boxes corresponds to a message and displays information as follows:

- A number in square brackets outside the box on left is the message identifier.
- A date and time within the box above the arrow indicates the message creation time stamp. See “Invocation Style and Message Time Stamps,” in the first chapter.

- The arrow is displayed in color. Normal request message arrows are blue; responses are green. If a message encountered an error, its arrow is red.
- The text below the arrow is the message name.

Additional lines indicate when a given business host receives and later sends a message.

If there are many messages shown in the full trace, it can be useful to limit the messages displayed by using a filter. This is also useful if you are trying to find messages that are related to an ACK or Archive IO message. The **Apply Filter** drop-down allows you to restrict messages in the following ways:

- To filter for the matching request or reply for a specific message—select the message and then select **Corresponding** from the **Apply Filter** drop-down menu. This selection limits the messages displayed to the following:
  - All preceding messages with the target set to the source of the selected message.
  - If a request message is selected, the corresponding response message.
  - If a response message is selected, the corresponding request message.
  - If an ACK or IOLog message is selected, the corresponding request or reply.
- To filter for all messages with a specific component as either the source or target of the message—select the column for the component and then select **Host** from the **Apply Filter** drop-down menu. This selection limits the messages to those having either a source or target of the selected component.
- To filter for all messages having the same source and target as a specific message—select the message and then select **Host** from the **Apply Filter** drop-down menu. This selection limits the messages to those that have both the same source and same target as the selected message.

When you apply a filter, the message trace displays the filter it is applying. For example:

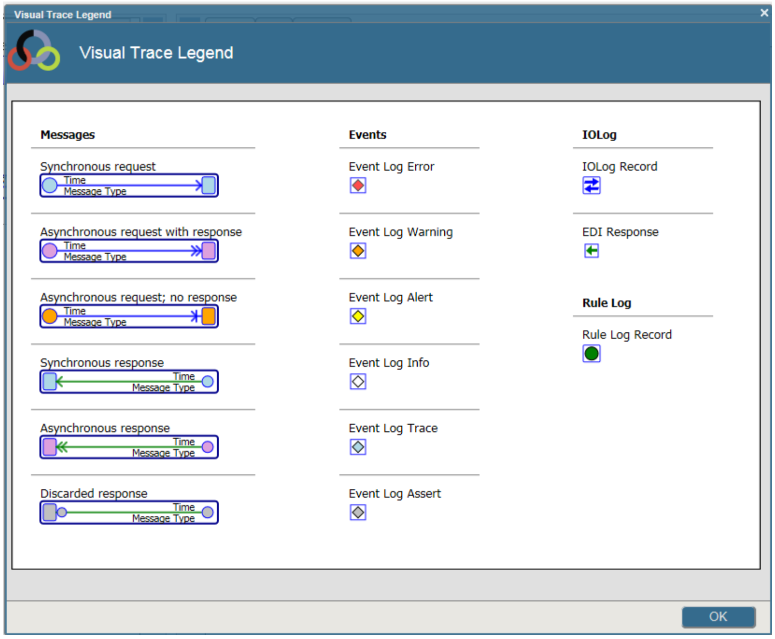
Filter = SourceHost:MsgRouter250, TargetHost:TCPOp001

Once you have applied a filter, the **Apply Filter** label changes to **Reapply Filter**. If you change the selection of the value of the drop-down menu, you must select **Reapply Filter** to change filters.

If there are more messages than will display on a page, you can use the **Items per page** drop-down to control the number of items displayed. You can either use the **Go to items** drop-down or the **Previous Page** and **Next Page** links to scroll through the pages.

The right area displays details for the selected message in the trace. The **Header**, **Body**, and **Contents** tabs display the same information as the **Message Viewer** page; see "Viewing the Message Details."

If you click **Legend**, Ensemble displays a popup guide with additional information, as follows:



If a message is sent out from an outbound HTTP adapter to another Ensemble namespace, the incoming message is assigned a new SessionID. If you want to associate the related messages across namespaces, you can use the **SendSuperSession** setting. If this setting is specified on an outbound HTTP adapter, the adapter sets the SuperSession property in the HTTP header. This header property is preserved through the incoming HTTP adapter and preserved throughout the Ensemble production. For details, see “SendSuperSession” in *Using HTTP Adapters with Ensemble*.

4.7 Resending Messages

Sometimes it is useful to resend a message, particularly if the message delivery failed. (You would first correct any underlying problem, of course.) To resend messages, do the following:

1. Access the **Message Viewer** page as described in “Browsing the Messages.”
2. Select the messages by selecting the check boxes in the left column. Or filter the display appropriately and then click the check box at the top of the left column.

   If you need to edit messages before resending them, then select a single message. You cannot use the **Edit and Resend** option (shown later) if you select multiple messages on this page.
3. Click **Resend Messages**.
4. If you clicked the check box at the top of the left column and if there are multiple pages of selected messages, the system then displays the following message:

   More messages match your search criteria than appear here. If you want to resend all the messages that match your criteria, including those not shown on this page, click OK. To resend only your selected messages, click Cancel.

   Now do one of the following:

   - Click **OK** to continue with all the selected messages.
   - Click **Cancel** to continue with only the selected messages that were visible on the first page.

   In either case, you can later cancel the action.
5. The system then displays details for the selected messages. The table includes the following information:

- **Session** — The session to which each message belongs. Click this to view the visual trace of the primary message object through the production. See "Sessions," in the first chapter.
- **Header** — The ID for the message header (also the message ID). Click this to view the visual trace of this specific message. See "Message Basics," in the first chapter.
- **Msg Body** — The ID for the message body. Click this to view the message contents.
- **Created** — The message creation time stamp. See "Invocation Style and Message Time Stamps," in the first chapter.
- **Source** — The business host that sent the message.
- **Target** — The business host that was intended to receive the message. This field also indicates if the production target is not running; note that you cannot resend a message if the intended target is not running. Click this to see the contents of the target's message queue.

   If you selected more than 1000 messages, only the first 1000 are shown, but the page indicates the total number that you selected.

6. Optionally select a new target business host. To do so, select a value for **New target**.
7. Optionally select **Resubmit at head of queue**.

   If you do so, Ensemble places the resent message at the front of its target queue. This helps to preserve FIFO (first in, first out) processing when the order of messages is important.

8. Click one of the following:
   - **Cancel** — To cancel this action.
   - **Resend** — To resend the message or messages as specified.
   - **Edit and Resend** — To edit the message and resend it as specified. See the subsection for details.

When you resend multiple messages, Ensemble resends them in order by age, starting with the oldest messages.

When you resend the messages, the page is redisplayed with an additional **Resend Status** column. Any status other than **OK** indicates that the resend operation failed. A resent message retains the same **Session** identifier and transmits the same message body, but acquires a new message header (with a new, unique ID) to mark its additional trip through the production. The visual trace includes both the original message transmission and any resend operations that involve the same message. The description in the message header contains text indicating this message has been resent; the description includes the original along with any subsequent header object identifiers.

4.7.1 Resend Editor

When you select one message to resend from the Ensemble Message Viewer page, you have the option to edit the body of the message before resending it.

1. Click **Edit then Resend** to display the **Ensemble Resend Editor** page.
2. Use the entry fields to update the message body data. The fields vary depending on the message. If the message has no properties, none are displayed.

   If you are editing a virtual document message, you can edit data in the message content and also edit object properties in the box below the content box.

3. Click **Resend** to send a new copy of the message header with your edited message body to the target.
4. After a successful resend, the page refreshes with text indicating the new **Header** and **Msg Body** identifiers. Click **Trace** to see the visual trace of the resent message.

4.8 Viewing the SQL Query Used by the Message Viewer

For debugging purposes, you might want to view the SQL query currently used by the Message Viewer. To do so:

1. Open the Terminal, change to your working namespace, and enter the following command:

   ```
   Set ^Ens.Debug("UtilEnsMessages","sql")=1
   ```

   This command sets a code in a utility debugging global.

2. Access the **Message Viewer** page as described in "Browsing the Messages."

   Notice that this page now includes the **Show Query** button; this button is shown only if the previously mentioned global is set.

3. Click **Show Query**.

4.9 Managing Suspended Messages

As noted earlier, some business operations might set the status of a failed message to suspended. Alternatively, you can manually suspend a message.

Ensemble automatically places all `Suspended` messages on a special queue, shown in the **Suspended Messages** page of the Management Portal. You can use this page to determine why the message failed, fix the problem, and then resend the message. For example, if the problem is that an external destination went out of service, you can make a change to reestablish communication with that server. Then you can resubmit the suspended messages to the external server from this page. Or you can discard or delete the message.

To manage suspended messages, do the following:

1. Click **Ensemble**.
2. Click **View**.
3. Click **Suspended Messages**.
4. If any messages in the currently running production are suspended, the system lists them in a table with the following information:

   - **ID** — The ID of the message. See "Message Basics," in the first chapter.
   - **Time Created** — The message creation time stamp. See "Invocation Style and Message Time Stamps," in the first chapter.
   - **Session** — The ID of the session associated with this message. See "Sessions," in the first chapter.
   - **Error?** — A quick overview of the results returned by the message to the business host that sent it.

     `OK` means normal behavior; `Retry` means the message experienced a failure but the business host that sent it is configured to retry the failed message. `Error` means an error was reported somewhere in the activity. `Inactive` means that the business host that sent the message has been idle for longer than its Inactivity Timeout setting, and may require diagnostic action.

   - **Source Configuration Name** — The business host that sent the message.

5. Select the messages by selecting the check boxes in the left column.

6. Then use any of the following buttons:

   - **Resubmit** — Click this to resubmit the messages. As message similar to the following displays for each successfully resubmitted message:

     ```
     Resubmit suspended message ID '7' completed.
     ```

     Any message that you resubmit retains the same **Session** identifier and transmits the same **Header** object. The description in the message header contains text indicating this is a `Resubmitted` message.

   - **Edit & Resubmit** — Click this to edit a single message before resubmitting; this command is only valid if you select exactly one message. See the following subsection for details.

   - **Discard** — Click this to remove the message from list on this page. The message is still accessible from the **Ensemble Message Viewer**, and its status is now `Discarded`.

   - **Delete** — Click this to remove all record of the message from the Ensemble database.

WARNING! You cannot undo the **Discard** or **Delete** operations.

Any message that you edit and resubmit retains the same **Session** identifier and includes the same **Header** object, but it contains a new **Msg Body** object with a new identifier. The description in the original message header contains text indicating that this message has been resubmitted and includes the original **Msg Body** object identifier.

4.9.1 Resend Editor for Resubmitting Messages

When you select one message to resubmit from the **Suspended Messages** page, you can edit the body of the message before resubmitting it.

1. Click **Edit & Resubmit** to display the **Ensemble Resend Editor** page.
2. Use the entry fields to update the message body data. The fields vary depending on the method signature of the message. If the method has no properties, none are displayed.

   If you are editing a virtual document message (HL7, X12, or other), you can edit data in the message content and also edit object properties in the box below the content box.
3. Click **Resubmit** to resubmit the original message header with your edited message body contents to the target.
4. After a successful resubmit, the page refreshes with text indicating the **Header** and **Msg Body** identifiers. Click **Trace** to see the visual trace of the resubmitted message.

5

Viewing the Event Log

This chapter describes the purpose of the Event Log and explains how to use it. This chapter includes the following sections:

- Introduction to the Event Log
- Introduction to the Event Log Page
- Entering Search and Purge Criteria
- Viewing Event Log Entries
- Viewing Event Details

5.1 Introduction to the Event Log

The Event Log is a table that records events that have occurred in the production running in a given namespace. The primary purpose of the Event Log is to provide diagnostic information that would be useful in case of a problem while the production is running. It includes the following items:

- System-generated Event Log entries. These entries are generated for events such as production startup and are not discussed in detail in this book.

  Note that these events are not the same as *system events*, which are generated and handled internally by Ensemble. System events, for example, include putting background processes to sleep and later waking them. The Ensemble Event Log does not record system events.

- Event log entries generated by the business host classes used in the production. For information, see "Generating Event Log Entries" in the chapter "Programming in Ensemble" in *Developing Ensemble Productions*.

  For a typical production, this is the most common kind of entry in the Event Log.

- Alerts. An *alert* sends notifications to applicable users while an Ensemble production is running, in the event that an alert event occurs. The intention is to alert a system administrator or service technician to the presence of a problem. Alerts may be delivered via email, text pager, or another mechanism. All alerts are recorded in the Event Log.

  For information on configuring a production to send alerts, see "Configuring Alerts" in *Configuring Ensemble*. That chapter also provides information on settings that specify the conditions under which certain events cause alerts.

- Trace messages, discussed later in this book.

Viewing the Event Log is a way to “take the pulse” of a production by scanning the informational text messages that it produces while it runs. Event log entries are stored persistently in the Ensemble database and may be purged according to age, as they accumulate.

5.2 Introduction to the Event Log Page

To view the **Event Log** page from the Management Portal, click **Ensemble** > **View** > **Event Log**.

This page is divided into the following three panes where you can perform the indicated functions:

Left	Middle	Right
Enter search and purge criteria	View Event Log entries	View event details

To expand and collapse the right and left panes, use the double arrow icons.

The **Event Log** page has the following commands:

- **Search** — Select to sort and filter the list of event entries using the criteria shown in the left pane. See “Search Events By” for details.
- **Cancel** — Select to cancel the current search.
- **Reset** — Select to reset the Event Log search criteria to the default values of the quick search fields and selected event types. See “Quick Search” and “Event Types” for details.
- **Previous** — Select to show the previous page of results based on the **Page Size**.
- **Next** — Select to show the next page of results based on the **Page Size**.
- **Export** — Select to export the selected entries to a text, tab-delimited (.csv), HTML, or XML file. This exported file is useful troubleshooting problems for developers or the InterSystems Worldwide Resource Center. You can use any application to examine the exported event file, but the exported file is not intended to be imported into Ensemble. To specify the format of the exported file, select the **All Files (*)** file type option and then enter `.txt`, `.csv`, `.html`, or `.xml` file type explicitly as part of the file name.

5.3 Entering Search and Purge Criteria

Use the left pane to enter search and purge criteria to filter the list of events.

There are three types of search filters:

- Quick Search
- Event Types
- Search Events By

Or remove entries from the Event Log as described in the following section:

- Purge Event Log

5.3.1 Quick Search

Enter the following values to filter the event list:

- **Sort Order** — Select to list either the oldest or the newest entries first. The default is **Newest First**.
- **Page Size** — The maximum number of Event Log entries to display in the middle panel as a result of the search. To see additional entries, click **Previous** and **Next**. The default is `500`.
- **Page** — (Read-only) Indicates which page of the list is displayed.
- **Time Format** — Select to show the time only or the time with the date. The default is **Complete** (time with date).
- **Auto-refresh** — Select a time interval to refresh the list or select **None**. The default is **None** (no auto-refresh).

As you enter values in these fields the middle pane display updates to reflect your entries.

5.3.2 Event Types

Select or clear the following check boxes to filter events as you determine necessary:

- **Assert**
- **Error**
- **Warning**
- **Info**
- **Trace**
- **Alert**

The default list shows events of all types; each event type is selected.

5.3.3 Search Events By

If you do not see the message you want to view in the Event Log page, you can filter the list of entries. To do so, enter values in one or more of the following fields:

- **Start Time** — Enter the lower limit of a range of **Time Logged** values.
- **Start ID** — Enter the lower limit of a range of **ID** values.
- **End Time** — Enter the upper limit of a range of **Time Logged** values.
- **End ID** — Enter the upper limit of a range of **ID** values.
- **Source Config Item** — Select a configuration item or type the name of one.
- **Source Class** — Enter a value in this field to list all the events logged by a specific host class.
- **Session ID** — Find all the Event Log entries associated with a particular session.
- **Source Method** — Enter a value in this field to list all the events logged by a specific method.
- **Job** — Enter a value in this field to find events hosted by a specific system job.
- **Text** — Enter a value in this field to list all the events whose text contains this string.

Note: Most of these fields support the use of the SQL Like wildcard character (`%`).

Once you enter new search criteria, click **Search** in the ribbon bar to refresh the list accordingly.

5.3.4 Purge Event Log

You can purge outdated records from the Ensemble Event Log by entering the number of days to keep the entries and then clicking **Purge**.

The displayed fields aid you in purging Event Log entries as follows:

- **Current Count** — Read-only field displaying the total number of Event Log entries that are now in the persistent store for this production. Use the **Current Count** to decide whether or not it is worthwhile to purge the Event Log at this time.
- **Do Not Purge Most Recent** — Parameter for the purge operation. It tells Ensemble how many days' worth of Event Log entries to keep. The default value is `7`, which keeps entries for the last seven days. If you want to purge all entries in the log, enter `0` in the **Days** field.

  The count of days includes *today*, so keeping messages for 1 day keeps the messages generated on the current day, according to local server time.

When you click **Purge**, Ensemble immediately starts to purge the Event Log according to the parameters you have entered.

CAUTION: You cannot undo the **Purge** operation.

Note: The portal provides another page where you can purge Event Log entries along with other management data. See "Purging Data" in *Managing Ensemble*.

5.4 Viewing Event Log Entries

Each time an event of interest occurs in the life cycle of a production, Ensemble writes an entry to the Event Log stating the details of what happened. You can view this log on the **Event Log** page of the Management Portal. The list displays the following information for each Event Log entry:

- **Type** — Indicates the type of entry: Alert, Assert, Error, Info, Trace, or Warning. The column color also indicates the event type as follows:

Event type	Column coloring
Alert	Yellow background with bold red text
Assert	Silver background with bold red text
Error	Pink background with bold red text
Info (production start)	Green background with bold green text
Info (production stop)	Green background with bold green text
Trace	Light blue background with bold blue text
Warning	Orange background with bold red text
Info (all others)	Default row color

- **ID** — The unique identifier for the message that comprises this Event Log entry.
- **Time Logged** — The date and time when this entry was logged.
- **Session** — The ID of the session associated with this message. See "Sessions," in the first chapter.

You can click **Session** link to see a visual trace of the session that contained this event.

- **Job** — The system job that hosted the event.
- **Source** — The configuration item (service, process, or operation) that sent the message.
- **Text** — The text string associated with the Event Log entry.

5.5 Viewing Event Details

You can select a log entry to view the details of that particular event. Select a row in the middle pane and the expanded right pane displays the following informational fields:

ID

Unique identifier for the message that comprises this Event Log entry.

Type

Indicates the type of entry: Alert, Assert, Error, Info, Trace, or Warning. The type contains the same coloring as the list entry.

Text

Text string associated with the Event Log entry.

Logged

Date and time when this entry was logged.

Source

Configuration item (service, process, or operation) that sent the message.

Session

The ID of the session associated with this message. See "Sessions," in the first chapter.

If this event has a session ID, you can click **Trace** at the top of the right pane to see a visual trace of the session that contained this event.

Job

System job that hosted the event.

Class

Business host class that logged the event.

Method

Method of the business host class that was running when the event was logged.

Trace

(none)

Stack

List of instructions leading up to the error.

6

Enabling Tracing

This chapter describes how to enable tracing, view trace messages, and log trace messages. It contains the following sections:

6.1 About Tracing

Tracing is a tool for use primarily during development. Trace elements enable you to see the behavior of various elements in a production, for the purpose of debugging or diagnosis. You typically disable tracing before a production goes live.

The Ensemble trace mechanism works as follows:

- As part of the development process, Ensemble developers add trace elements to the appropriate areas of your code. These trace elements (potentially) write trace messages at runtime. See "Adding Trace Elements" in the chapter "Programming in Ensemble" in *Developing Ensemble Productions*.

  Note that these are messages only in a general sense; trace messages are simply strings and are unrelated to Ens.Message and its subclasses.

- As part of the configuration process, do the following:
  - Configure the production to enable tracing. This step means that, at runtime, the trace elements are executed (rather than being ignored).
  - Optionally enable logging for the trace messages. This step writes trace messages to the Event Log.
  - Optionally configure the applicable business hosts to run in the foreground so that you can see trace messages in the Terminal while the production is running. See the last section in this chapter.

You typically disable tracing before a production goes live.

6.2 Enabling Tracing

By default, all user trace elements are enabled. You can also enable tracing of various system events.

To do so, set values for some or all of the following nodes of the `^Ens.Debug` global:

Node	Purpose
`^Ens.Debug("TraceCat")`	Controls tracing overall as follows: • If this node is not set, only user trace elements are enabled. In this case, you can enable specific kinds of system tracing by setting subnodes, as described in the rest of this table. • If the value of this node is 0, no tracing is enabled. • If the value of this node is 1, all tracing is enabled, apart from any explicitly disabled kinds of tracing.
`^Ens.Debug("TraceCat","bproc")`	Enables or disables system traces from business processes. For this node and all the rest of the nodes in this table, if the node value is 1, the specified traces are enabled. If the node value is 0, these traces are disabled. This node is ignored if the parent node value is 0.
`^Ens.Debug("TraceCat","connwait")`	Enables or disables system traces from adapters waiting to connect.
`^Ens.Debug("TraceCat","exterr")`	Enables or disables system traces showing errors from external systems.
`^Ens.Debug("TraceCat","file")`	Enables or disables system traces from file read or write operations.
`^Ens.Debug("TraceCat","ontask")`	Enables or disables system traces from business host framework events.
`^Ens.Debug("TraceCat","parse")`	Enables or disables system traces from HL7 and other virtual document parsers.
`^Ens.Debug("TraceCat","protocol")`	Enables or disables system traces of sequence numbers from the MSH segment in HL7 messages.
`^Ens.Debug("TraceCat","queue")`	Enables or disables system traces related to message queue management.
`^Ens.Debug("TraceCat","system")`	Enables or disables general system trace elements.
`^Ens.Debug("TraceCat","timing")`	Enables or disables system traces providing information about duration of calls.
`^Ens.Debug("TraceCat","transform")`	Enables or disables system traces about DTL data transformations, apart from errors.
`^Ens.Debug("TraceCat","user")`	Enables or disables user traces.

Node	Purpose
`^Ens.Debug("TraceCat","xform")`	Enables or disables system traces about errors in DTL data transformations.

For example, to enable tracing related to message queue management, enter the following command in the Terminal, in the appropriate namespace:

```
set ^Ens.Debug("TraceCat","queue")=1
```

Also see “Enabling %ETN Logging” in the chapter “Testing and Debugging” in *Developing Ensemble Productions*.

6.3 Enabling Logging for Trace Messages

Ensemble can also log trace messages (that is, write them to the Event Log). To enable or disable logging of trace messages, use the following settings:

- For any business host, use the Log Trace Events setting. When this setting is selected, Ensemble logs all the enabled trace messages for this business host log.
- For the production, use the Log General Trace Events setting. When this setting is selected, Ensemble logs all enabled trace messages from production elements that are *not* business hosts.

There is no overlap or interaction between these settings; Log General Trace Events does not override or provide a default value for Log Trace Events.

See “Settings in All Productions” in *Configuring Ensemble Productions*.

6.4 Seeing Trace Messages in the Terminal

To see the trace messages in the Terminal, do the following:

1. If you are using Windows Vista or Windows 7, enable the Interactive Services Detection Service, as follows:
   a. On the Windows Start menu, go to **Administrative Tools > Services**.
   b. Scroll to **Interactive ServicesDetection**.
   c. Right click and select **Start**.
2. Enable the Foreground setting for the business host or business hosts in which you are interested.

   When you run the production, Ensemble opens a Terminal window for each foreground business host. This Terminal window shows all enabled trace messages for that business host. It also shows all log items and alerts.
3. On Windows Vista or Windows 7, the Interactive Services Detection Service displays a dialog box to indicate that a program is attempting to display a message. Click **View the Message**. The Interactive Services Detection Service then displays a window that contains one or more Terminal windows.

7

Viewing the Business Rule Log

The Business Rule Log is a persistent record of business rules that have been executed, their respective results, and reasons for the result.

To access this page from the Management Portal, click **Ensemble** > **View** > **Business Rule Log**. Or click **Rule Log** in the ribbon bar of the **Business Process List** page.

This chapter describes this page and how to use it. It contains the following sections:

- Introduction
- Entering Search and Purge Criteria
- Viewing the Executed Rule List
- Viewing Rule Execution Details

7.1 Introduction

The **Business Rule Log** page is divided into the following three panes where you can perform the indicated functions:

Left	Middle	Right
Enter search and purge criteria	View the executed rule list	View rule execution details

You can expand and collapse the right and left panes as desired using the double arrow icons.

There are four commands in the ribbon bar of the Rule Log page:

- **Search** — Click to sort and filter the list of rule log entries using the criteria shown in the left pane. See the Search Rules By section for details.
- **Reset** — Click to reset the rule log search criteria to the default values of the quick search fields. See the Quick Search section for details.
- **Previous** — Click to show the previous page of results based on the **Page Size**.
- **Next** — Click to show the next page of results based on the **Page Size**.

7.2 Entering Search and Purge Criteria

The left pane permits you to enter search and purge criteria to filter the list of rules.

There are two types of search:

- Quick Search
- Search Rules By

You can also remove entries from the rule log as described in the following section:

- Purge Rule Log

7.2.1 Quick Search

Enter the following values to filter the executed rule list:

- **Sort Order** — Select to list either the oldest or the newest entries first. Default is **Newest First**.
- **Page Size** — The maximum number of rule log entries to display in the middle panel as a result of the search. If more entries exist, you can click **Previous** and **Next** to page through the results. Default is `500`.
- **Auto-refresh** — Select a time interval to refresh the list or select not to have the list automatically refreshed. Default is **None** (no auto-refresh).
- **Page** — This is a read-only field showing what page of the list is being displayed.
- **Time Format** — Select to show the time only or the time with the date. Default is **Complete** (time with date).
- **Errors** — Select this check box if you only want to see rule executions that had errors. Default is to show all executions (check box is cleared).

As you enter values in these fields the middle pane display updates to reflect your entries.

7.2.2 Search Rules By

You can filter what entries display in the list by entering values in one or more of the following fields:

- **Start Time** — Enter the lower limit of a range of **Time Executed** values.
- **End Time** — Enter the upper limit of a range of **Time Executed** values.
- **Rule Name** — Choose a rule name as defined in the Ensemble Rule Editor. The filter finds all of the occasions when this rule has been invoked by business processes.
- **Session Id** — Find all the rule log entries associated with a particular session.

Once you enter new search criteria, click **Search** in the ribbon bar to refresh the list accordingly.

7.2.3 Purge Rule Log

You can purge the rule log by entering the number of days to keep the entries and then clicking **Purge**. The **Current Count** is a read-only field displaying the number of entries in the rule log. If you want to purge all entries in the log, enter `0` in the **Days** field.

7.3 Viewing the Executed Rule List

Each time a business process executes or “fires” a rule, Ensemble writes an entry to the Business Rule Log stating the details of what happened. You can view this log on the **Business Rule Log** page of the Management Portal. The list displays the following information for each business rule log entry:

- **Session** — The unique identifier for the session that is (or was) associated with this rule. A session marks the beginning and end of all the activities prompted by a primary request message from outside Ensemble.
- **Time Executed** — The date and time when this rule was last invoked.
- **Rule Name** — The name assigned to the rule in the Ensemble Rule Editor.
- **Error** — The value 1 means that the rule encountered an error. The value 0 means that the rule did not encounter any errors.
- **Return value** — The value returned by the rules engine for this rule.

You can also perform the following actions on a selected rule:

- Click the **Session** to navigate to the **Visual Trace** display for the session that contained this particular execution of the business rule.
- Click the **Rule Name** to navigate to the **Ensemble Rule Editor** page for this business rule definition class.

7.4 Viewing Rule Execution Details

You can select a log entry to view the details of that particular execution of the rule. Select a row in the middle pane and the expanded right pane displays the following informational fields:

Execution ID

Unique identifier for this rule execution.

Session ID

Unique identifier for the session that is (or was) associated with this execution of this rule. A session marks the beginning and end of all the activities prompted by a primary request message from outside Ensemble.

Time Executed

Date and time when this rule was executed.

Rule Name

Name of the rule definition class that was executed.

Rule Set

Name of the rule set that was executed.

Reason

Specific rule name that cause the rules engine to generate the result. If a business rule is empty or undefined, the reason is **Rule Missing**.

Error?

Displays `Yes` or `No` depending on whether the execution of the rule resulted in an error.

Error Message

If the rules engine returns an error, then this is the text associated with the error.

Return Value

Value returned by the rules engine for this rule.

Activity Name

Name assigned to the <rule> activity in the BPL code.

Effective Begin Date/Time

Effective begin date and time of the executed rule set.

Effective End Date/Time

Effective end date and time of the executed rule set.

These informational values correspond to properties of the Ens.Rule.Log class, which you can view in the *Class Reference*.

From the right pane, you can perform the following actions on the selected rule:

- Click **Trace** to see a visual trace of the session that contained this execution of the business rule.
- Click **Rule** to navigate to the **Ensemble Rule Editor** page for this business rule definition class. See *Developing Business Rules*.

8

Viewing Business Process Instances

This chapter describes how to view and monitor business processes. It contains the following topics:

- Introduction
- Filtering the List of Process Instances
- Viewing Summary Information for Business Process Instances
- Purging the Business Process Log

8.1 Introduction

The **Business Process List** page displays any current instances of a business process in the currently running production. If a business process has completed its work, there is no entry for it on this page.

To access this page, click **Ensemble** > **View** > **Business Process Instances**.

On this page:

- The left area permits you to enter search criteria to filter the list of instances.
- The middle area displays lists the instances.
- The right area displays details.

The following topics provide details.

8.2 Filtering the List of Process Instances

The left area of the **Business Process List** page provides the following options that you can use to filter the list of business process instances displayed on this page:

- **Sort Order** — Select a value from the drop-down list: **Oldest First** or **Newest First**.
- **Page Size** — Select the number of instances to display on a given page of results.
- **Time Format** — Select a value from the drop-down list: **Time Only** or **Complete** (time with date).
- **Auto-Refresh** — Specify how often Ensemble should automatically refresh this page.

- **Time Created: Start** — Enter the lower limit of a range of **TimeCreated** values.
- **Time Created: End** — Enter the upper limit of a range of **TimeCreated** values.
- **Session Id** — Enter the ID of a session.
- **Primary Request** — Enter the message ID number of the request that caused this business process to be instantiated.
- **Configuration Name** — Enter the configuration name of a business process to find all instances of that item.

After you have edited these fields, you can click one of the commands at the top of the page:

- Click **Search** to sort the list of entries in the top display using the criteria shown in the bottom display.
- Click **Reset** to redisplay the entries in their default order and return the fields in the bottom display to their default values.

Ensemble then redisplays the page.

8.3 Viewing Summary Information for Business Process Instances

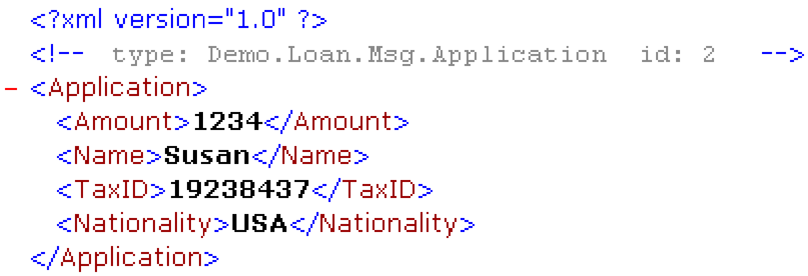
The middle area of the **Business Process List** page displays the following information for each business process instance:

- **ID** — The unique identifier for the instantiated business process.
- **IsCompleted** — The value 1 means that the primary request that initiated this business process has been completed. The value means that the primary request has not yet been completed.
- **Configuration Name** — The configured name of the business process host class.

  When this is underlined, it means that the host class is a BPL business process. You can click on the underlined name to display the BPL diagram for the host class.
- **SessionId** — The ID of the session associated with this business process. See "Sessions," in the first chapter.

  One or more business processes may be instantiated within Ensemble during the session, in order to fulfill the primary request. All of these business processes share the same **SessionId**, but business process has a different **ID** value.
- **PrimaryRequest** — The message ID number of the request that caused this business process to be instantiated. The **PrimaryRequest** number is distinct from the object **ID** number of the business process. The **PrimaryRequest** may or may not be the same as the **SessionId**. If the numbers are different, it means that the request message that started the session triggered subsequent requests within Ensemble, and one of these later messages is the one that actually instantiated the business process.

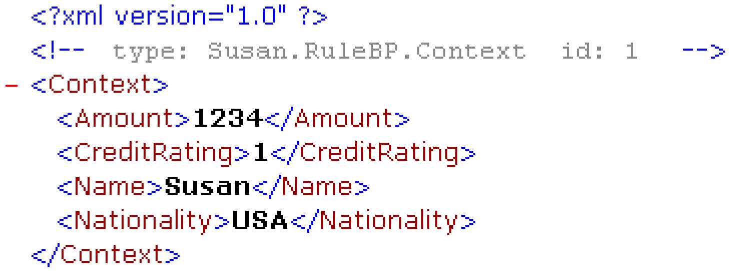
  When the **PrimaryRequest** number is underlined, it means that the primary request message can be displayed as a set of properties in XML format. You can click on the underlined **PrimaryRequest** number to display the message properties in colorized XML format in the right area of the page. For example:



```
<?xml version="1.0" ?>
<!--  type: Demo.Loan.Msg.Application  id: 2   -->
- <Application>
    <Amount>1234</Amount>
    <Name>Susan</Name>
    <TaxID>19238437</TaxID>
    <Nationality>USA</Nationality>
  </Application>
```

- **TimeCreated** — The date and time when this business process was instantiated.
- **TimeCompleted** — The date and time when this business process completed the primary request that instantiated it. If this request has not been completed, this field is blank.
- **ContextId** — The unique identifier for the general-purpose, persistent variable *context*, which is defined using the <context> and <property> elements in BPL to hold persistent properties for this business process instance. This column is available for BPL business processes only.

  The **ContextId** is underlined to indicate that you can click on it to display the *context* properties in colorized XML format in the right area of the page. For example:

  ```
  <?xml version="1.0" ?>
  <!--  type: Susan.RuleBP.Context  id: 1   -->
  - <Context>
      <Amount>1234</Amount>
      <CreditRating>1</CreditRating>
      <Name>Susan</Name>
      <Nationality>USA</Nationality>
    </Context>
  ```

- **RuleLog** — Click this command to display the business rule log for this business process instance.

In each row, background color indicates the status of the business process instance:

- Gray — Completed.
- White — These items are in progress.

8.4 Purging the Business Process Log

You can purge outdated records from the Ensemble business process archives by clicking the **Purge** command at the bottom left of the **Business Process List** page.

The fields in this dialog allow to you purge business process instance data as follows:

- **Current Count** — The number in this column reflects the total number of instances that are now in the persistent store for this production. Use the **Current Count** to decide whether or not it is worthwhile to purge these records at this time.
- **Do not purge most recent** — Specifies the number of days' worth of records to keep. The number can be 0 (zero), which keeps nothing and deletes all business process instance records that exist at the time of the purge operation. The default value for **Do not purge most recent** is 7, which keeps records for the last seven days.

  The count of days includes *today*, so keeping messages for 1 day keeps the messages generated on the current day, according to local server time.

To purge the data, click **Purge**. Ensemble immediately starts to purge instances according to the parameters you have entered in the dialog box.

CAUTION: You cannot undo the **Purge** operation.

Note: For information on how to purge business process instances along with other management data, see "Purging Production Data" in *Managing Ensemble*.

9

Viewing the Ensemble I/O Archive

If you enable the Archive IO setting for one or more business service and business operations, Ensemble archives the input and output data in addition to the Ensemble messages. The input and output data is shown in the Visual Trace window; see “Tracing the Path of Related Messages,” earlier in this book.

You can also display the I/O archive by executing an SQL query as follows:

1. In the Management Portal, click **System Explorer**, **SQL**, **Execute SQL Statements**, and then click **OK**.
2. Choose the namespace where you want to execute the query from the list on the left.
3. In the **SQL Query** box, type:

   ```
   SELECT * FROM Ens_Util.IOLog
   ```

   Where the table name identifies Ens.Util.IOLog or one of its subclasses. The choice of subclass depends on the type of data. Options include Ens.Util.IOLogFile, Ens.Util.IOLogObj, Ens.Util.IOLogStream and Ens.Util.IOLogXMLObj.
4. Click **Execute Query**. The results display in the bottom half of the page.
5. If there are no results, or if the results do not match your expectations, check to see that you have enabled the Archive IO setting for the business service or business operation that you want to investigate. See *Configuring Ensemble Productions*.

10

Viewing Messages from Multiple Productions

In some cases an overall business process spans multiple productions. Although a namespace can have only a single production, you can have multiple productions running using any of the following configurations or any combination of them:

- Multiple namespaces running on a single instance of Ensemble, each with its own production.
- Multiple instances of Ensemble running on a single system.
- Multiple systems each running an instance of Ensemble.

Note: If you are running and monitoring multiple productions, we recommend that you define a separate namespace for each production, even if they are running on separate Ensemble instances or on separate systems.

There are two Ensemble features that allow you to search for messages from multiple productions:

- Enterprise Message Viewer. See Using the Enterprise Message Viewer
- Enterprise Message Bank. See Using the Enterprise Message Bank.

The Enterprise Message Viewer allows you to search for and view messages from multiple productions, without the need to store the messages centrally. The Enterprise Message Bank stores messages from multiple productions into a central repository, where they can be stored, analyzed and processed in other ways. This reduces the need to store data on the individual Ensemble systems.

The Enterprise Message Viewer allows you to search for messages across multiple productions. The Enterprise Message Viewer sends your query to the Ensemble namespaces running the business productions, and the productions return the messages that match the query to the Enterprise Message Viewer. You do not have to modify the production to access its messages in an Enterprise Message Viewer.

The Enterprise Message Bank allows you to store messages from one or more productions into a central repository, where they can be stored, analyzed and processed in other ways. The productions send their messages to the Enterprise Message Bank. To enable this, you must add a Message Bank Operation to each production and configure it to connect to the Message Bank. The Message Bank operation forwards the messages to the Enterprise Message Bank. The Enterprise Message Bank creates a separate copy of each message in a special Message Bank production. The Enterprise Message Bank is a central repository and can require substantial resources to store and retain messages. Since you can retain messages in the Enterprise Message bank, you may be able to purge them in the originating productions.

The Enterprise Message Viewer and the Enterprise Message Bank have the following differences:

- The Enterprise Message Viewer can only query messages which are currently accessible in the namespace message store. If the message has been purged, it is not accessible to the Enterprise Message Viewer. The Enterprise Message

Bank contains an independent copy of each message, and the Enterprise Message Bank has retention policies that are independent of the policies on the systems running the business productions. Consequently, even if the original message is purged, the copy can still be available on the Enterprise Message Bank. Maintaining the copies of the messages consumes storage resources.

- The Enterprise Message Viewer displays basic information about the message as well as the values in columns specified in the query. You cannot view the complete message or resend a message from the Enterprise Message Viewer. In order to access the message, you can follow a link in the Enterprise Message Viewer and open a Message viewer on the Ensemble instance running the production. In contrast, the Enterprise Message Bank provides access to the complete message. It is possible to resend a message directly from the Enterprise Message Bank. You cannot resend a message from the Enterprise Message Viewer but can resend a message after following the link to the system running the production.

On the Enterprise Message Viewer system, you must identify and specify credentials for each production that you will be querying for messages. For each production, you provide a name, Web IP address (including port number), namespace, SOAP credentials, and SSL configuration. For details, see Identifying Enterprise Systems for Viewing and Monitoring. This information enables the Enterprise Message Viewer to find and access the namespace running the production. On the Enterprise Message Bank, you should provide the same information. Although the Enterprise Message Bank can receive messages without knowing the credentials, it cannot resend messages without them.

The following chapters describe Using the Enterprise Message Viewer and Using the Enterprise Message Bank.

11

Using the Enterprise Message Viewer

This chapter describes using the Enterprise Message Viewer. It discusses the following topics:

- Specifying a Query in the Enterprise Message Viewer
- Working with the Search Results in the Enterprise Message Viewer

The Enterprise Message Viewer is a special Ensemble production that runs in a namespace. In this namespace, you should identify the systems that you intend to search for messages. To identify a system, you provide a name, Web IP address (including port number), namespace, SOAP credentials, and SSL configuration. For details, see Identifying Enterprise Systems for Viewing and Monitoring. In addition, you must define on the Enterprise Message Viewer system all classes that you will be using in search criteria. These classes are the ones that define the body parts and search tables in the production messages.

Once you have identified one or more systems, you can access the Enterprise Message Viewer by clicking **Ensemble**, **View**, and **Enterprise Messages**. You can also access the Enterprise Message Viewer by clicking **Ensemble**, **Configure**, and **Enterprise Systems** and then clicking the **Enterprise Message Viewer** link.

For an overview of the Enterprise Message Viewer and the Enterprise Message Bank, see Viewing Messages from Other Namespaces.

11.1 Specifying a Query in the Enterprise Message Viewer

To specify a query in the Enterprise Message Search, you can enter any of the following:

- **Sort Order**—specifies whether the messages should be stored newest or oldest first.
- **Page Size** and number—specifies the number of messages to display on a page and the page to display.
- **Time Format**—specifies whether the time or time and date are displayed.
- **Basic Criteria**—provides the basic selection criteria.
  - **Status**—specifies whether all messages are to be selected or only messages with the specified status, such as suspended messages.
  - **Type**—specifies whether session start messages, request messages, response messages, or all messages are to be selected.
  - **Start Time** and **End Time**—specifies the range of date-times of the messages to be selected.
  - **Start ID** and **End ID**—specifies the range of message IDs that are to be selected. Since each system numbers messages independently, these criteria are typically only useful when you are searching for messages from a single system.

  - **Source**—specifies the name of the Ensemble component that originates the message. Typically, this is a business service or business process. By default, messages from all sources are included in the selection.
  - **Target**—specifies the name of the Ensemble component that is the target of the message. Typically, this is a business process or business operation. By default, messages to all targets are included in the selection.
  - **Maximum Rows**—specifies the maximum number of messages that each system should return to the Enterprise Message Viewer.
  - **Query Timeout**—specifies the number of seconds to wait for the systems to respond to the query. A value of zero specifies that there is no timeout period. The query timeout can be specified to 1/100th of a second precision.

- **Enterprise Clients**—specifies which systems should be queried. Check the box for each system that you want to query. The systems listed are the ones specified in Identifying Enterprise Systems for Viewing and Monitoring.
- **Extended Criteria**—allows you to specify queries, which can be combined with the AND and OR operators. The query form specifies:
  - **Enable Criterion**—specifies whether this element of the query is active.
  - **Criterion Type**—specifies whether this element of the query is on the **HeaderField**, **BodyProperty**, **SearchTableField**, **VDocSegmentField**, **VDocPropertyPath**, or an **OR** operation combining the previous element of the query with the next element of the query.
  - **Class**—is dependent on the criteria type and the types defined on the Enterprise Message Viewer system. Note that you do not have access to classes defined on the systems running the productions that are not also defined on the Enterprise Message Viewer system.
  - **Conditions**—specify one or more conditions of the query. The fields of the conditions form are determined by the criterion type.
  - **Display Values**—specify one or more fields to display in the results but that are not part of the selection criteria. Note that systems running a version of Ensemble previous to version 2012.2 will not return display values in the search results. These older versions return only the fields referenced in the query.
- **Saved Searches**—allows you to save the current search or retrieve a saved search.

11.2 Working with the Search Results in the Enterprise Message Viewer

The Enterprise Message Viewer displays a page of the search results. You can page through the results with the **Next** and **Previous** buttons. You can examine the information about the message that was included in the search results or you can click the system name in the first column, which is a link to the Message Viewer on the system that runs the business production. You may be prompted for username and password to log onto the system. The link opens the Message Viewer on the system but does not select the individual message that you were examining on the Enterprise Message Viewer. You can use the message ID to search for the message.

12

Using the Enterprise Message Bank

This chapter describes how to use the Enterprise Message Bank. The Enterprise Message Bank is an optional remote archiving facility where you can collect messages, Event Log items, and search table entries from *multiple* Ensemble client productions. It consists of the following components:

- The Message Bank server, which is a specialized Ensemble production consisting exclusively of a Message Bank service that receives submissions from any number of client productions.
- A client operation (the Message Bank operation) that you add to an Ensemble production and configure with the address of a Message Bank server.

To access the Message Bank pages from the Message Bank server or from any configured client, click **Ensemble**, **Configure**, and **Message Bank Link**. To query and view the Message Bank server, click **Ensemble**, **Configure**, and **Enterprise Systems** and then click the link to **Message Bank Viewer** or **Message Bank Event Log**.

The Message Bank Viewer page is similar to the **Message Viewer** page. For general information on using this page, see "Browsing the Messages," earlier in this book.

On this page, when you resend messages, optionally select a value for **Target Client**. By default, the message is sent to its original client, but you can send it to a different client.

To improve efficiency, you can restrict the search to a single client by using the **Message Bank Client** filter. This allows Ensemble to optimize searches with filters using message create times by using the system ID index.

The Message Bank Event Log page is similar to the **Event Log** page. For general information on using this page, see "Viewing the Event Log," earlier in this book.

For an introduction and information on defining the Enterprise Message Bank, which is a specialized Ensemble production, see *Developing Ensemble Productions*. For information on configuring it, see *Configuring Ensemble*.

For an overview of the Enterprise Message Viewer and the Enterprise Message Bank, see Viewing Messages from Other Namespaces.

13

Monitoring Alerts

Alerts are automatic notifications triggered by specified events or thresholds being exceeded. This chapter describes how to configure and monitor alerts and contains the following topics:

- Introduction to Alerts
- Choosing How to Handle Alerts
- Calibrating Alert Sensitivity
- Configuring Alert Monitoring
  - Configuring Simple Notifications
  - Configuring Alert Routing
  - Configuring Alert Management
    - Configuring Alert Management in Production Settings
    - Defining Alert Groups
    - Adding and Configuring the Alert Manager
    - Adding the Alert Manager and Defining Its Rule
    - Adding the Notification Manager and Defining Its Data Transformation
    - Adding and Configuring Notification Operations
    - Adding the Optional Alert Monitor and Defining Its Rule

- Monitoring, Tracking, and Resolving Managed Alerts
  - Handling and Updating Managed Alerts
  - Reporting Managed Alert History and Status

- Managed Alerts Walkthrough

13.1 Introduction to Alerts

Alerts provide a way for Ensemble to automatically notify users about a serious problem or condition that requires a quick resolution to ensure that the production continues operating normally. When properly configured Ensemble generates alerts for potentially serious problems and should not generate any alerts caused by normal variations in the production performance.

The other monitoring features described in this document require the user to actively check for a problem. Typically, users only check for a problem after it has had a noticeable impact on the production's performance. Once you have configured alerts to notify users automatically, alerts may make it possible to resolve issues before they have a serious impact on performance and become critical problems.

Ensemble can automatically send an alert message to users when specified thresholds are exceeded, when specified events occur, or when user code explicitly generates alerts. You can process alerts in a number of ways:

- Alerts are only written to a log file and there is no automatic notification.
- A simple alert notification system, where all alerts get sent to a list of users.
- Routing alert notification where selected alerts are sent to different users depending on the kind of alert and the component that generated it.
- An alert management framework that notifies users about alerts, escalates unresolved alerts, and documents the current state and history of actions taken to resolve the alert.

In configuring any alert notification system, it is important to calibrate the level that triggers the alert and ensure that the users being notified understand the alert and know how to respond. If the trigger level is set too high, problems may already have a significant impact on performance before Ensemble notifies users. But, if the trigger level is set too low, Ensemble sends out many notifications during the normal operation of the production and users tend to ignore these notifications and may not respond to the few critical ones among them.

Alerts are messages of type Ens.AlertRequest that can be generated by any business service, process, or operation in a production. Ensemble always stores alerts messages in the log. If there is a production component named Ens.Alert, Ensemble sends all alert messages to it. Ens.Alert typically is an operation, such as EnsLib.Email.AlertOperation, a routing process, or the Alert Manager, which has the class Ens.Alerting.AlertManager.

13.2 Choosing How to Handle Alerts

To choose how to handle alerts in your production, you should first answer the following questions:

- Do you want alert notifications to be automatically sent to users or do you want no automatic notification and require the user to view the alerts in the log?
- Do you want to send all alerts to the same group of users by the same transmission mechanism or do you want to select alerts to go to different sets of users possibly by different mechanisms and possibly not send some alerts to any users?
- Do you want to assign an owner to an alert, track whether the issue has been dealt with and the alert closed, possibly escalate and reassign the alert, and be able to view the alert status and history including how long it took to resolve the issue?

In all cases, you should define what conditions generate alerts as described in Calibrating Alert Sensitivity. Once you have done that:

1. If you don't want automatic notification, you are done. Your production should not have a component named Ens.Alert. Ensemble will write alerts to the log file but will not do anything else with them.

2. If you want to send all alerts to a single list of users by the same mechanism but do not want to track and manage alerts, add an operation with a class such as EnsLib.Email.AlertOperation and name the component Ens.Alert. Ensemble sends all alerts to the operation and it notifies the configured users. See Configuring Simple Notifications.
3. If you want the capability to route alerts to a set of users and other alerts to other users or to none but do not want to track and manage alerts, add a routing engine to your production and name it Ens.Alert. You will also need to add alert operations, such as EnsLib.Email.AlertOperation. See Configuring Alert Routing.
4. Finally, if you want to manage your alerts so that you can assign alerts to users, track the status of alerts, and manage the alerts, add an Alert Manager, Notification Manager, one or more Alert operations, and, optionally, an Alert Monitor to your production. See Configuring Alert Management.

13.3 Calibrating Alert Sensitivity

Ideally you would like to generate alerts only for conditions that require investigation and resolution and not generate alerts when the production is behaving normally, but in order to generate alerts for all serious conditions, you typically will also get some alerts from the normal variation of production execution. For example, you could set the inactivity timeout for a business service at 300 seconds. During peak hours, this business service may get many requests during a 300 second interval and, if no requests come in during that interval, it probably indicates a problem that should generate an alert. However during off-peak hours, there may be no requests for longer periods with all systems operating correctly. To set the system to catch the important alerts during the peak period, you will generate false positive alerts during the off-peak period. If you are using a router or the Alert Manager, you can suppress notifications for these off-peak alerts.

Each business service, process, and operation in a production can generate alerts when it encounters an error or exceeds a limit. You calibrate alert sensitivity by configuring the alert and error settings for each business service, process, and operation. After you configure these production settings, you should monitor the production during normal operation. If there are large numbers of false positive alerts or missed alerts for serious conditions, you should adjust the settings.

The following settings specify the conditions under which Ensemble generates alerts:

Alert On Error

Check this option if you want the component to generate an alert when it encounters an error. Other production settings control what conditions are considered errors. These settings are described later in this section.

Do not check **Alert On Error** on any component that is involved in delivering or processing alerts.

Alert Grace Retry Period

This is a time in seconds that the component will retry sending its output before issuing an alert. This setting is most commonly used for business operations. If the component is retrying and eventually succeeds within this time there will be no alert. Setting this to a value such as 60 seconds will suppress alerts on transient issues such as a dropped network connection that fixes itself, but won’t wait too long before alerting you to a real problem.

Inactivity Timeout

This is a time in seconds that the component will wait for a request before issuing an alert. This is typically used for business services. If you set it to a value such as 300 seconds for a busy system you will be alerted fairly quickly if you stop receiving messages. For quieter systems you might want a longer interval. This value applies all day, so you might get false positives on off-peak times, when traffic is much lower than normal, but you can filter these in your alert handler if necessary.

Queue Wait Alert

This is a time in seconds that a message can stay on a queue before it generates an alert. Setting this to a value such as 300 seconds would be a good starting point. For critical systems, 5 minutes might be too long, but for other less critical systems a longer interval may be appropriate.

Queue Count

Specifies the number of items on a queue. When the queue size reaches this number Ensemble generates an alert. Setting this to a value such as 10 causes an alert when a queue starts to build on a critical item indicating some delay. But some queues may get this large during normal operation and if a component receives a nightly batch of work, setting this parameter may cause unnecessary alerts.

Alert On Bad Message

In routers that provide validation, such as the HL7 router, this setting specifies that any message that fails the specified validation generates an alert. This alert is in addition to sending the original message to the bad message handler.

There are many settings that control what conditions Ensemble treats as errors. If Alert On Error is checked, these settings control what conditions generate alerts. The following are some commonly used settings that control error conditions:

Stay Connected

If this is a positive integer, the connection is dropped and an error is issued after that number of seconds with no activity. Setting this to -1 means never disconnect and never cause an error on a disconnect from the other end. A large value such as 99999 means the connection will be dropped after an extremely long period of inactivity, but has the side effect of not causing an error if the other end drops the connection.

Failure Timeout

Specifies the number of seconds to retry sending a message before failing and considering it an error. Setting this property to –1 instructs Ensemble to continue to retry sending the message indefinitely and to not fail. Typically, this is the setting to use for critical messages that must be processed in FIFO order, such as HL7 messages. If Failure Timeout is set to a positive integer, the operation fails after this number of seconds, discards the current message and attempts to process the next message.

Replycode Action

This setting for HL7 determines how the operation responds when the target application returns an error reply code. This is a critical setting that determines the operation behavior for a failure. The values are fairly complex but hover text help is available by clicking on the ‘Replycode Action’ label in config settings tab. Typically an operation should retry indefinitely if there is an error sending the message. To ensure indefinite retries after all NACKs set FailureTimeout to -1 and ReplyCodeAction to :?R=RF and :?E=RF. If the target application rejects the message (AR) retrying will probably have the same results so you might want to do something different. However applications can’t always be relied on to use the ACK codes correctly and each application should be considered individually. One option is to disable the business operation after a rejected message but that stops all traffic. Suspending rejected messages until the problem can be fixed will allow other messages to flow but will break the FIFO ordering.

13.4 Configuring Alert Monitoring

Alerts are messages generated by production components. Ensemble automatically writes the alerts to a log file and sends then to the production component named Ens.Alert. If your production does not have a component named Ens.Alert, then

Ensemble writes alerts to the log file but does not send them to any component. The component named Ens.Alert can be of any class. The most frequently used classes for Ens.Alert are:

- An operation class that sends the alert via email or another mechanism. See Configuring Simple Notifications.
- A router class that can direct selected alerts to one or more operations. See Configuring Alert Routing.
- The Alert Manager class that provides a mechanism to document and track the progress in handling the problem indicated by the alert. See Configuring Alert Management.

13.4.1 Configuring Simple Notifications

If you can handle all the alerts via the same output mechanism and all alerts are to be sent to the same list of users, you can use an operation class as the component named Ens.Alert. To send alerts by email, use the EnsLib.EMail.AlertOperation class. To use another mechanism, you must develop a new operation. See Using a Simple Outbound Adapter Alert Processor for details.

If you are using the EnsLib.EMail.AlertOperation class, you should specify the following configuration settings:

- **SMTP Server**—specifies the address of the email server.
- **SMTP Port**—specifies the port that the SMTP server uses.
- **Credentials**—specifies the Ensemble credentials that provide access to the SMTP server.
- **Recipients**—specifies the list of email addresses where the alerts are to be sent. You can separate the email addresses with a comma or semicolon.
- **Cc**—specifies the list of email addresses that should receive a copy of the message.
- **UseDetailedSettings**—if checked, the email subject is “Ensemble Alert from configuration item *component-name* on system *Ensemble-Instance-Name* and the message body includes the following information in the email message:
  - Alert Text
  - Alert Time
  - Production
  - Source
  - Session
  - System
  - Instance
  - Node

  If **UseDetailedSettings** is not checked, the email message contains only the alert text and the email subject is “Ensemble Alert from configuration item *Ensemble-Instance-Name*:*component-name*, The default is to not check this setting for compatibility with previous versions.
- **SubjectPrefix**— if **UseDetailedSettings** is checked, this setting specifies text that is prepended to the email subject.
- **IncludeNodeinSubject**—if **UseDetailedSettings** is checked, this setting specifies that the server address should be included in the email subject.
- **IncludeManagedAlertHistory**—this setting is not used when configuring simple notifications.
- **SystemName**— if **UseDetailedSettings** is checked, this setting specifies text to be used in place of the *Ensemble-Instance-Name*.

13.4.2 Configuring Alert Routing

If you need to contact users via multiple output mechanisms or you need to send alerts selectively to specified users, add a business process named Ens.Alert with a EnsLib.MsgRouter.RoutingEngine class. In this case, your production must also contain a business operation for each output mechanism, and the alert processor forwards messages to those business operations.

The business process would examine the messages and forward them to different business operations, depending on the alert contents and any logic that you include.

Your logic may need to consider the following factors:

- Different requirements for various users
- Different requirements depending on the time of day
- The organization’s problem-solving policies and procedures

The EnsLib.MsgRouter.RoutingEngine class provides the setting **Business Rule Name**. If you specify this setting as the name of a routing rule set, this business host uses the logic in that rule set to forward all the messages that it receives. To use the routing rule to specify the addresses to send the alert, you can add a transformation that sets the AlertDestination property of the Ens.AlertRequest class. For an example, see the Demo.HL7.MsgRouter.Production in the ENSDEMO namespace. In this example, the routing rule sends all alerts to the email operation and uses a transform with a lookup to specify the alert destinations based on the component that created the alert. The addresses specified in the AlertDestination property are added to any addresses specified in the **Recipients** setting.

13.4.3 Configuring Alert Management

Managed alerts are persistent messages that provide a record of what problems occurred in a production, who responded to the problems, what they did to resolve the problems, and how much time it took to resolve the problems. Alert Management can notify key personnel of alerts that are not resolved promptly.

Alert management provides alert routing capability plus the tools needed to track and resolve alerts. Alert management allows you to assign an alert to a specific user, track whether the alert has been resolved or escalated, and report the time that it took to resolve the alert. Alert management can be added to a production using the Ensemble management portal including the rule and transformation editors without writing custom code. For specialized requirements, it is possible to add custom code in the alert management production components. See Adding Custom Code to Alert Management for more information.

The Alert Management framework consists of the following business services, processes, and operations:

- Alert Manager
  - Typically is named Ens.Alert.
  - Receives alerts generated by business services, processes, and operations in the production.
  - Converts them to managed alerts and assigns an owner using a business rule.
  - Sends the managed alerts to the Notification Manager.
- Notification Manager
  - Receives managed alerts from Alert Manager and Alert Monitor.
  - Sends them to alert operation so that the specified users are notified using a transformation.
- Alert operation

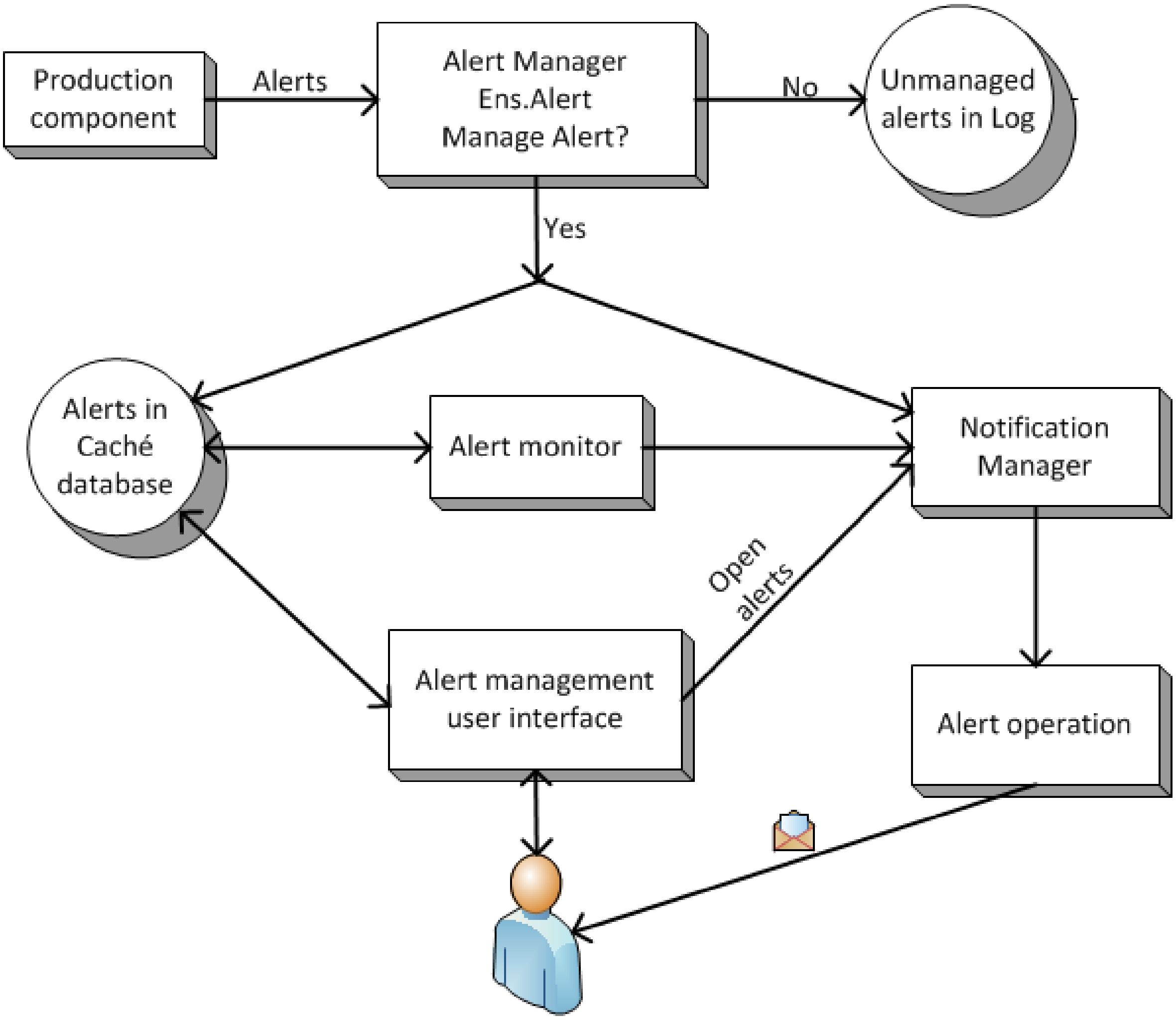
  - Sends alert notifications to users using email or another mechanism.

- Alert Monitor
  - Queries database for managed alerts that have passed the next action deadline without being updated or closed.
  - Sends a reminder notification, updates the NextMonitorTime, and escalates the alert using a business rule.

- **Managed Alerts** page. To reach this page, select **Ensemble** > **Monitor** > **My Managed Alerts.**
  - Display and update open alerts.
  - Tabs allow you to see your alerts, unassigned alerts, and all alerts. Alerts are organized by the alert's next action time.

- **Managed Alert Viewer** page. To reach this page, select **Ensemble** > **View** > **Managed Alerts**.
  - Display-only access to managed alerts.
  - Can search for alerts based on owner, open status, time, escalation level, source, and alert group.

The following figure illustrates how the components in the alert management framework are connected.



The following sections describe how to add and configure the alert management components:

- Configuring Alert Management in Production Settings

- Defining Alert Groups
- Adding the Alert Manager and Defining Its Rule
- Adding the Notification Manager and Defining Its Data Transformation
- Adding and Configuring Notification Operations
- Adding the Optional Alert Monitor and Defining Its Rule

13.4.3.1 Configuring Alert Management in Production Settings

The production settings **Alerting Control** group is used only for alert management. If you are not using alert management, you can leave these fields blank. If you are using alert management, set these fields as follows:

- **Alert Notification Manager**—Enter the name that of the Notification Manager.
- **Alert Notification Operation**—Enter the name of the email alert operation.
- **Alert Notification Recipients**—Enter valid email addresses, separated by commas. By default, the alert management sends all alert notifications to this email address list. If you use the Alert Notification Manager rule to specify destinations for the messages, this field is not used.
- **Alert Action Window**—Enter a number of minutes. This is the default number of minutes that a user has to resolve and close an alert before the next reminder message is sent. If you include the Alert Monitor in your production, by default it sends a reminder after the specified number of minutes have expired and resets the notification time for the managed alert by adding the specified number of minutes to the current time.

If you are sending alert notifications to different distribution lists based on the component that generated the alert, it is useful to specify what alert groups each component belongs to.

13.4.3.2 Defining Alert Groups

If you are sending alert notifications to different distribution lists based on the component that generated the alert, it is useful to specify what alert groups each component belongs to. For large productions with many components, it is more practical to select a subset of alerts based on alert groups than on the individual component names. You specify the alert groups for a component as a string containing a list of alert groups separated by commas. You specify a component's alert groups in the AlertGroups property. Once you have defined an alert group for one component, you can select it using the check box for another component.

13.4.3.3 Adding the Alert Manager and Defining Its Rule

The alert processor must be a business process named Ens.Alert. Typically, if you are using the Alert Manager, you add a business process named Ens.Alert with the class Ens.Alerting.AlertManager. If you are adding alert management to a production that already has a router alert processor, you could keep the router named Ens.Alert and have it send some or all of the alerts to the Alert Manager business process.

If you do not define a rule for the Alert Manager, it promotes all alerts to managed alerts, leaves alerts unassigned, sets the deadline based on the number of minutes specified in the **Alert Action Window**, and sends all managed alerts to the Notification Manager.

To use a rule for the Alert Manager, create a rule and specify the **Creation Rule for Managed Alert** type in the rule wizard. This creates a rule with a Ens.Alertiing.Context.CreateAlert context. The alert context provides access to:

- AlertRequest—the incoming alert. You only have read access to the alert. If the rule makes any changes to the alert, they are ignored.
- AlertGroups—the alert groups configured for the component that originated the alert.
- BusinessPartner—the business partner configured for the component that originated the alert.

- Owner—if the rule sets this property, the Alert Manager assigns the managed alert to the specified user.

The rule can suppress promoting the alert to a Managed Alert by returning 0 or can promote the alert to a Managed Alert by returning 1.

The rule can check whether the alert is a repeat occurrence of a previous alert that is represented by a currently open managed alert. To do this, the rule uses the **Ens.Alerting.Rule.FunctionSet.IsRecentManagedAlert()** function. The **IsRecentManagedAlert()** function tests if there is a recent open managed alert that is from the same component and has the same text as the specified alert. You can optionally specify that the function adds a reoccurs action to the existing managed alert.

After you have defined the rule, you specify the rule name in the Alert Manager configuration as the **CreateManagedAlertRule** property. If you need to customize the Alert Manager in ways that the rule does not allow, you can implement a subclass of Ens.Alerting.AlertManager and override the **OnProcessAlertRequest()** method. See Adding Custom Code to Alert Management.

13.4.3.4 Adding the Notification Manager and Defining Its Data Transformation

To add the Notification Manager, add a business process with class Ens.Alerting.NotificationManager. If you do not define a rule, the Notification Manager sends all managed alert messages and reminders to the email addresses set in the **Alert Notification List** using the operation set in the **Alert Notification Operation**.

To use a data transformation with the Notification Manager, create a data transformation following these instructions:

- Specify the Source Class as Ens.Alerting.NotificationRequest.
- Specify the Target Class as Ens.Alerting.Context.Notify.
- On the **Transform** tab of the data transformation editor, set the **Create** property to **existing**.

Since the Notification Manager sets the target NotificationRequest before executing the data transformation, you do not need to copy any of the alert information from the source to the target. You should set the following in the target:

- target.Notify—set to 1 to send a notification and set to 0 to suppress the notification.
- target.Targets—add an element to the collection for each different alert operation that you are using. If you are only using one alert operation, there should only be one element in Targets. Each element contains:
  - TargetConfigName—specifies the name of the alert operation.
  - AlertDestinations—is a collection where each element specifies a single destination address for the alert operation. The format of the address depends on the requirements of the alert operation. Operations with the class EnsLib.EMail.AlertOperation take email addresses, but an operation that sends text messages could take a phone number.

Your rule should not modify target.NotificationRequest. If you add any destinations to the NotificationRequest, they are ignored.

Once you have defined the data transformation, on the production configuration page, select the Notification Manager and set the **NotificationTransform** to the transformation that you have defined.

If you need to customize the Notification Manager in ways that the data transformation does not allow, you can implement a subclass of Ens.Alerting.NotificationManager and override the **OnProcessNotificationRequest()** method. See Adding Custom Code to Alert Management.

13.4.3.5 Adding and Configuring Notification Operations

To use the email alert operation provided with Ensemble, add an operation with the class EnsLib.EMail.AlertOperation, and enter the following settings in the **Basic Settings** and the **Additional Settings** group:

- **SMTP Server**—Specify the name of the server.
- **SMTP Port**—Update the port if the server does not use the standard port.
- **Credentials**—You have to create credentials using the **Credentials** page to specify a username and password that can use the SMTP server to send email. To access this page, select **Ensemble** > **Configure** > **Credentials**. Then specify the credentials in this field.
- **Recipient** and **Cc** fields—You can leave these fields blank. If you enter email addresses here, all alerts that are sent through this operation will always be send to these addresses in addition to any addresses specified by the production-wide settings or by the Notification Manager.
- **From**—Enter a valid email address.
- **IncludeDetails**—This field is used only for messages of type Ens.AlertRequest, which are unmanaged alerts. For unmanaged alerts checking this setting includes additional information in the email. This setting is ignored for managed alerts.
- **SubjectPreface**—Specify text that is prefixed to the beginning of subject lines on the alert email message. By default the subject line is “Ensemble ManagedAlert from configuration item '*ComponentName*’ on system '*EnsembleProcessName*'.
- **IncludeNodeInSubject**—If checked, the node name (computer name) is included in the subject line.
- **IncludeManagedAlertHistory**—Controls whether the managed alert update history is included in the email message and whether they are list oldest or newest update first.
- **SystemName**—Replaces the process name in the subject line.
- **IncludeUTCTimes** checkbox—if checked, the email message includes the UTC time as well as the local time.

13.4.3.6 Adding the Optional Alert Monitor and Defining Its Rule

The Alert Monitor is optional. It controls whether overdue alerts are escalated and whether reminder notices are to be sent. To add an alert monitor, add a Business service with class Ens.Alerting.AlertMonitor. If you do not include the alert monitor, there are no automatic reminders and no automatic escalations.

If you do not define a rule, the alert monitor sends out reminder notices when a managed alert is still open and its NextMonitorTime has passed. It resets the NextMonitorTime by incrementing it with the number of minutes specified in the **Alert Action Window** production setting. It does not escalate alerts.

To use a rule for the Alert Monitor, create a rule and specify the **Overdue Rule for Managed Alert** type in the rule wizard. This creates a rule with a Ens.Alertiing.Context.OverdueAlert context. The alert context provides access to:

- ManagedAlert—provides read access to the managed alert.
- CurrentTime—provides the current time in UTC format.
- NewNextActionTime—allows you to set the NextActionTime to a UTC format date-time value.
- NewEscalationLevel—allows you to set the escalation level of the managed alert.

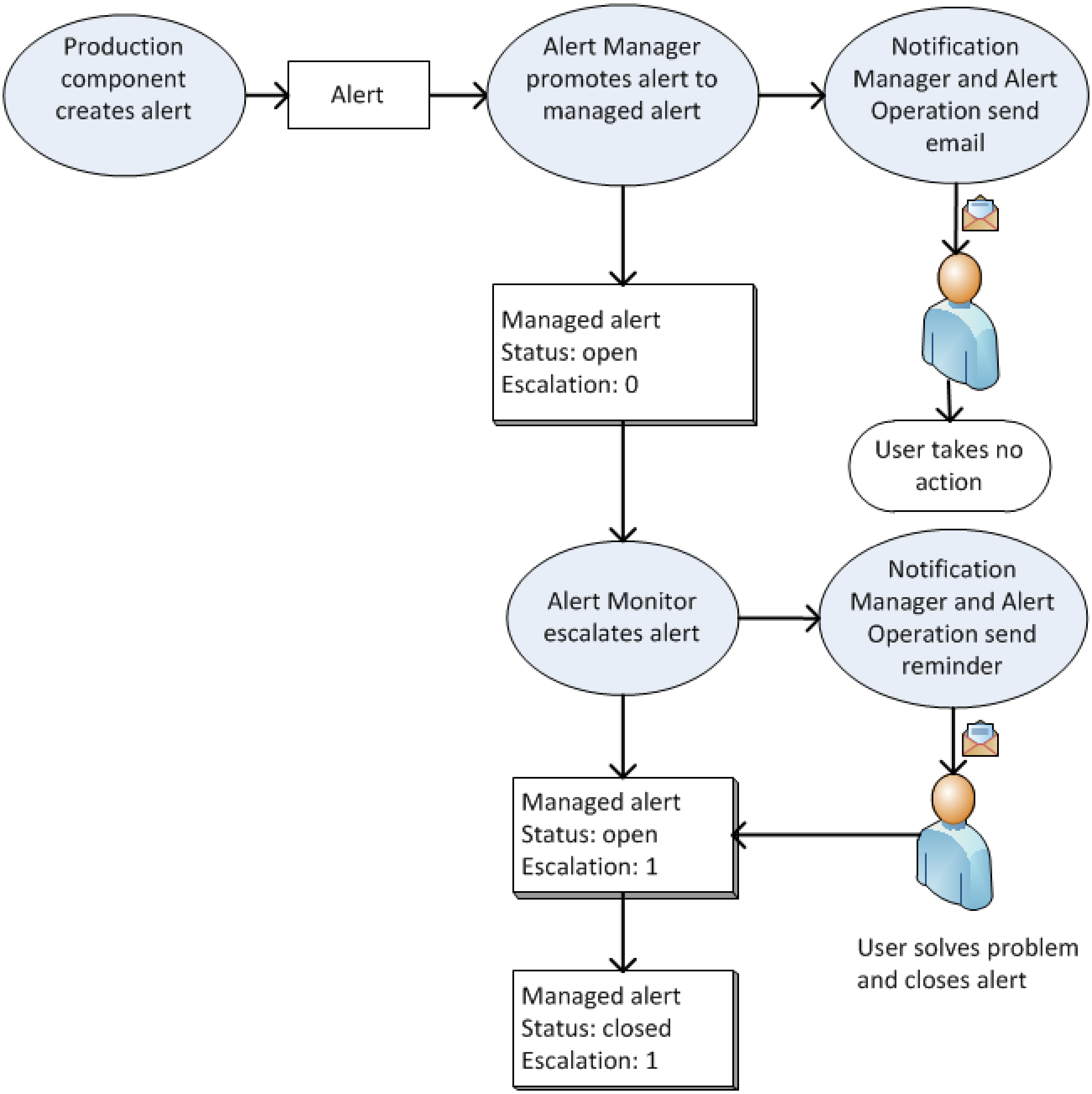
If the rule returns 1, the Alert Monitor sends the managed alert to the Notification Monitor. If the rule returns 0, no reminder message is sent.

If you set the New Next Action Time, the Alert Monitor sets the NextMonitorTime to the same time. If you do not set the New Next Action Time, the Alert Monitor takes its default action, which is to set the NextMonitorTime to the current time plus the number of minutes specified in the **Alert Action Window** production setting.

If you need to customize the Alert Monitor in ways that the rule does not allow, you can implement a subclass of Ens.Alerting.AlertMonitor and override the **OnProcessOverdueAlert()** method. See Adding Custom Code to Alert Management.

13.5 Monitoring, Tracking, and Resolving Managed Alerts

Managed alerts are a special kind of persistent message. The Alert Manager generates a managed alert when it receives an alert that meets the specified conditions. The following figure illustrates the life cycle of a managed alert: from the component creating the alert to a user resolving the problem indicated by the alert and closing the managed alert.


Production component creates alert
Alert
Alert Manager promotes alert to managed alert
Notification Manager and Alert Operation send email
Managed alert
Status: open
Escalation: 0
User takes no action
Alert Monitor escalates alert
Notification Manager and Alert Operation send reminder
Managed alert
Status: open
Escalation: 1
User solves problem and closes alert
Managed alert
Status: closed
Escalation: 1

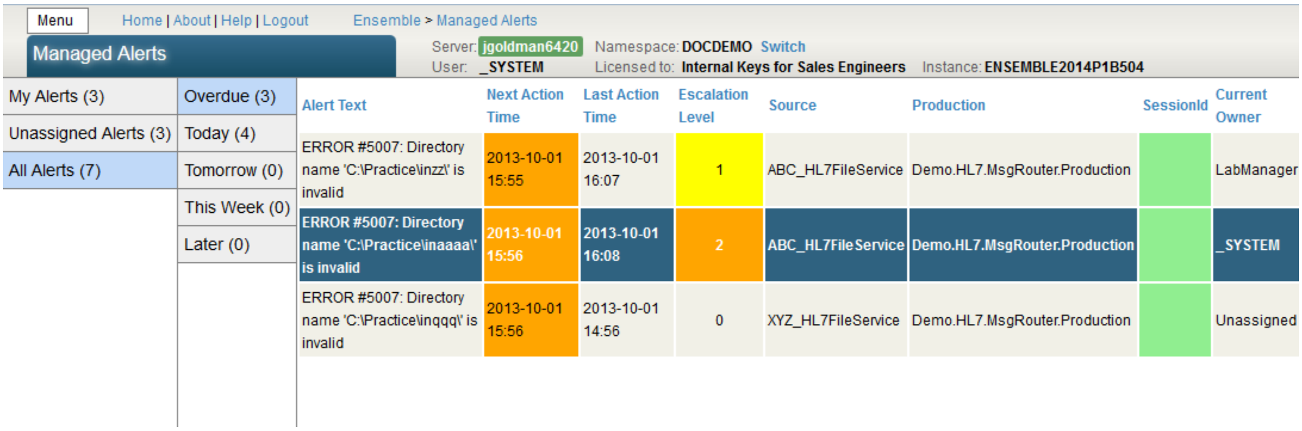
This figure illustrates the life cycle of a typical managed alert:

1. A business service, process, or operation encounters a specified condition and generates an alert.
2. The alert is sent to the Alert Manager, which is a business process named Ens.Alert. The alert manager promotes the alert to a managed alert, assigns it to a user, and sends it to the notification manager.

3. The Notification Manager determines what group of users to contact and how to contact them. It sends the Managed Alert to an alert operation, which sends notifications to a group.
4. The owner of the managed alert does not solve the problem or update the managed alert.
5. The alert monitor queries for alerts that are still open when the alert's NextMonitorTime has been reached. It finds the managed alert. It escalates the alert and sends it to the notification manager.
6. The Notification Manager determines what group of users to contact and how to contact them. Since the alert is escalated it sends it to a different distribution list. It sends the Managed Alert to an alert operation, which sends email to a group.
7. In this case the user is able to solve the problem that caused the initial alert. The user updates the managed alert to close the alert.
8. The managed alert is now inactive but contains the alert history and remains available for reports and analysis.

13.5.1 Acting on Alerts by Viewing My Managed Alerts

After you have received an email or other message indicating that there is a managed alert that requires your action, you can view the open managed alerts that are assigned to you or that are unassigned by selecting **Ensemble**, **Monitor**, and **My Managed Alerts**.



Server: jgoldman6420 Namespace: DOCDEMO
User: _SYSTEM Licensed to: Internal Keys for Sales Engineers Instance: ENSEMBLE2014P1B504

Alert Text	Next Action Time	Last Action Time	Escalation Level	Source	Production	SessionId	Current Owner
ERROR #5007: Directory name 'C:\Practice\inzz\' is invalid	2013-10-01 15:55	2013-10-01 16:07	1	ABC_HL7FileService	Demo.HL7.MsgRouter.Production		LabManager
ERROR #5007: Directory name 'C:\Practice\inaaaa\' is invalid	2013-10-01 15:56	2013-10-01 16:08	2	ABC_HL7FileService	Demo.HL7.MsgRouter.Production		_SYSTEM
ERROR #5007: Directory name 'C:\Practice\inqqq\' is invalid	2013-10-01 15:56	2013-10-01 14:56	0	XYZ_HL7FileService	Demo.HL7.MsgRouter.Production		Unassigned

You can view **My Alerts**, **Unassigned Alerts**, or **All Alerts**. You can select alerts that are:

- Overdue—the Next Action Time has past.
- Today—the Next Action Time is today.
- Tomorrow—the Next Action Time is tomorrow.
- This Week—the Next Action Time is in the next 7 days.
- Later—the Next Action Time is after the next 7 days.

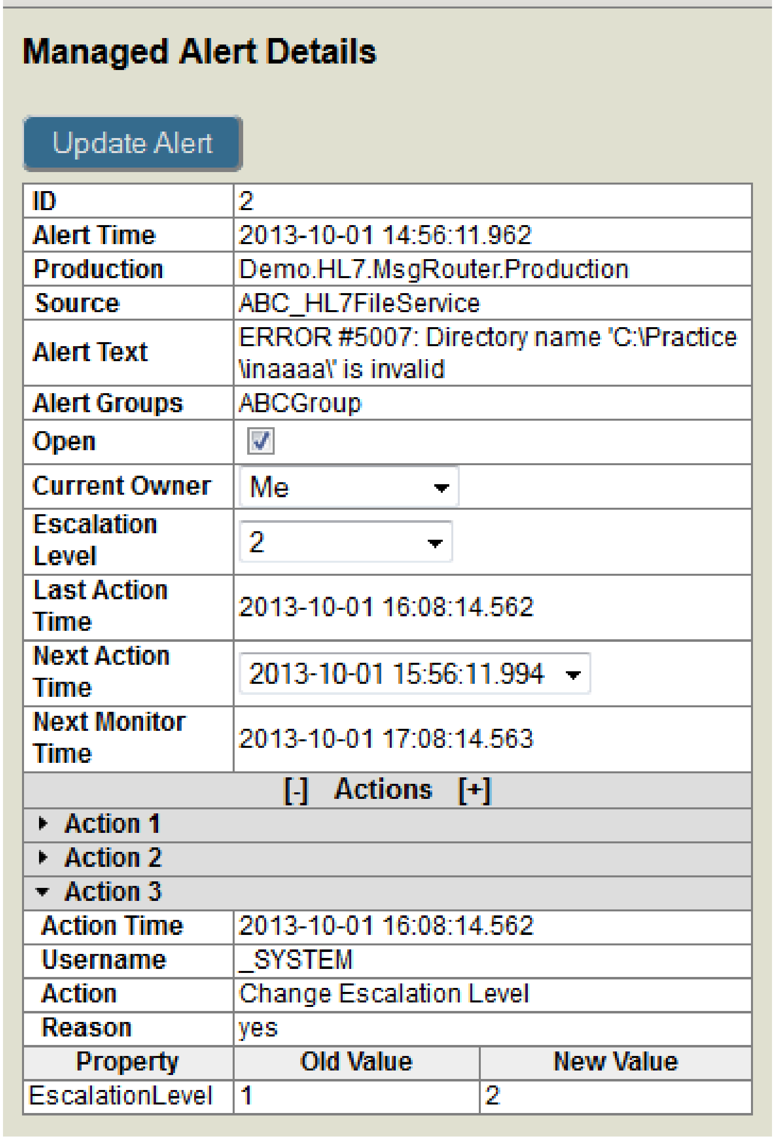
The table displays the following information about the alerts:

- Alert Text—the text defined for the alert.
- Next Action Time—the deadline for resolving the issue causing the alert. If the alert is not closed by this deadline, the alert monitor will evaluate the alert and may escalate or reassign the alert.
- Last Action Time—the previous time the managed alert was updated.
- Escalation Level—Alerts by default are created with a 0 Escalation Level. When the alert is escalated, the Escalation Level field is colored to indicate the escalation level:
    - 0: No color

  - 1: Yellow
  - 2: Orange
  - 3: Red

- Source—name of the component causing the alert.
- Production—name of the production.
- SessionId—not used.
- Current Owner—current owner or shown as “Unassigned”.

To display the alert details and to update the alert, select the alert in the list. The **Managed Alert Details** form is displayed.



ID	2
Alert Time	2013-10-01 14:56:11.962
Production	Demo.HL7.MsgRouter.Production
Source	ABC_HL7FileService
Alert Text	ERROR #5007: Directory name 'C:\Practice \inaaaa\' is invalid
Alert Groups	ABCGroup
Open	☑
Current Owner	Me
Escalation Level	2
Last Action Time	2013-10-01 16:08:14.562
Next Action Time	2013-10-01 15:56:11.994
Next Monitor Time	2013-10-01 17:08:14.563
[-] Actions [+]	
Action 1	
Action 2	
Action 3	
Action Time	2013-10-01 16:08:14.562
Username	_SYSTEM
Action	Change Escalation Level
Reason	yes

Property	Old Value	New Value
EscalationLevel	1	2

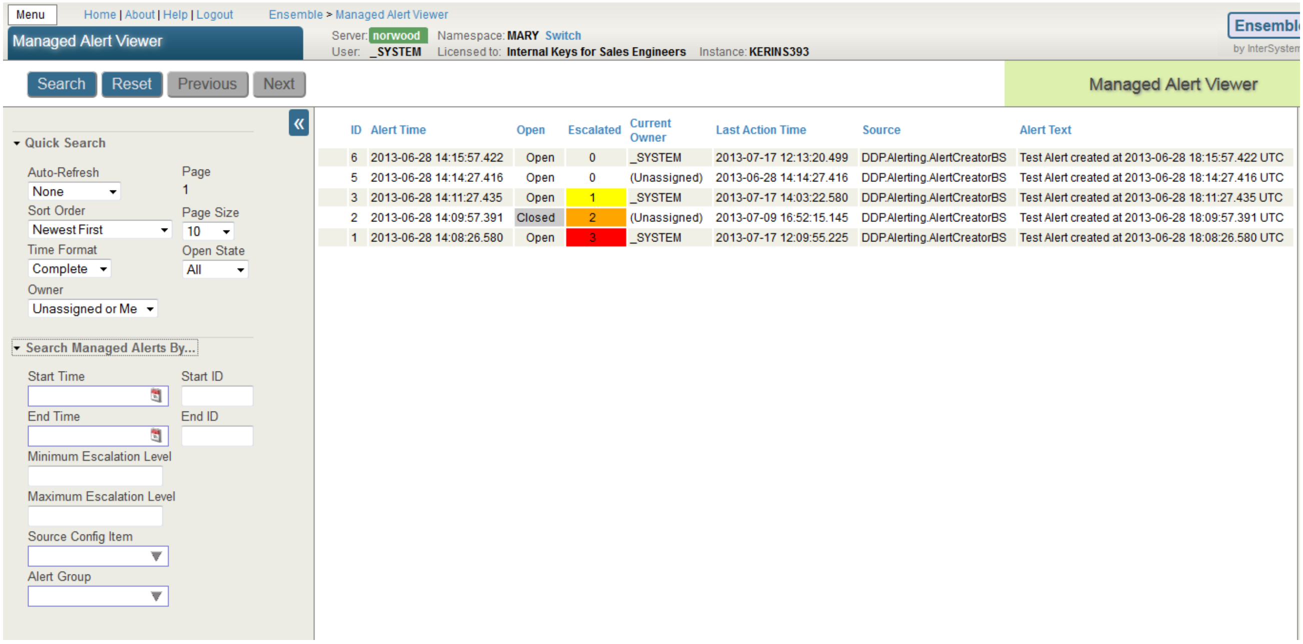
The **Managed Alert Details** form displays the following fields. You can update the **Open**, **Current Owner**, **Escalation Level**, and **Next Action Time** fields.

- **Update Alert** button and alert update reason comment field—after you update a field in the details form, the comment field is displayed. You must enter a comment and then click the **Update Alert** button to update the managed alert.

- **ID**, **Alert Time**, **Production**, **Source**, and **Alert Text**—display the message ID, time of the initial alert, name of the production, name of the component causing the alert, and the text message defined for the alert.
- **Alert Groups**—displays the alert groups associated with the alert.
- **Open** check box—to close the alert, clear the check box, enter the reason you are closing the alert in the comment field, and click the **Update Alert** button.
- **Current Owner**—you can set the current owner to yourself, to one of the listed users, or to unassigned. The user list includes all users who have any of the following roles: `%EnsRole_Administrator`, `%EnsRole_AlertAdministrator` and `%EnsRole_AlertOperator`.
- **Escalation Level**—you can keep the escalation level unchanged, de-escalate the alert (which decrements the alert level), or escalate the alert (which increments the alert level).
- **Last Action Time**—the previous time the alert was updated.
- **Next Action Time**—the deadline for the alert to be updated or closed. You can leave the deadline unchanged, make the deadline earlier, or postpone the deadline by making it later. You can update the Next Action Time relative to its current value or relative to the current time.
- **Next Monitor Time**—the next time the alert monitor will check the managed alert and possibly escalate it or send a reminder notification..
- **Actions**—displays the history of updates to the managed alert. You can show or hide the details of each action by clicking on the triangle for that action. Clicking on the + (plus sign) expands all of the actions and clicking on the – (minus sign) hides the details of all actions. The actions can include a reoccurs action, which indicated that an identical alert was generated by the same component within a specified time period. The **IsRecentManagedAlert()** function adds this action to a managed alert.

13.5.2 Viewing Managed Alerts

The **Managed Alert Viewer** allows you to search and view all managed alerts that are stored in the database, including closed alerts and alerts assigned to other users, but you cannot update alerts from the **Managed Alert Viewer**. To access the **Managed Alert Viewer**, select **Ensemble**, **View**, and **Managed Alerts**.



Server: norwood Namespace: MARY Switch
User: _SYSTEM Licensed to: Internal Keys for Sales Engineers Instance: KERINS393

Auto-Refresh: None
Page: 1
Sort Order: Newest First
Page Size: 10
Time Format: Complete
Open State: All
Owner: Unassigned or Me

Start Time:
Start ID:
End Time:
End ID:
Minimum Escalation Level:
Maximum Escalation Level:
Source Config Item:
Alert Group:

ID	Alert Time	Open	Escalated	Current Owner	Last Action Time	Source	Alert Text
6	2013-06-28 14:15:57.422	Open	0	_SYSTEM	2013-07-17 12:13:20.499	DDP.Alerting.AlertCreatorBS	Test Alert created at 2013-06-28 18:15:57.422 UTC
5	2013-06-28 14:14:27.416	Open	0	(Unassigned)	2013-06-28 14:14:27.416	DDP.Alerting.AlertCreatorBS	Test Alert created at 2013-06-28 18:14:27.416 UTC
3	2013-06-28 14:11:27.435	Open	1	_SYSTEM	2013-07-17 14:03:22.580	DDP.Alerting.AlertCreatorBS	Test Alert created at 2013-06-28 18:11:27.435 UTC
2	2013-06-28 14:09:57.391	Closed	2	(Unassigned)	2013-07-09 16:52:15.145	DDP.Alerting.AlertCreatorBS	Test Alert created at 2013-06-28 18:09:57.391 UTC
1	2013-06-28 14:08:26.580	Open	3	_SYSTEM	2013-07-17 12:09:55.225	DDP.Alerting.AlertCreatorBS	Test Alert created at 2013-06-28 18:08:26.580 UTC

You can specify the search criteria in the left pane:

- **Quick Search**
    - **Auto-Refresh**—specifies whether Ensemble should refresh the list at the selected interval.
    - **Sort Order**—specifies whether Ensemble sorts the list based on the initial alert time or the escalation level.
    - **Page Size**—specifies the number of alerts to display on a page.
    - **Time Format**—specifies whether Ensemble displays the full date and time or just the time.
    - **Open State**—specifies whether Ensemble searches for only open alerts, only closed alerts, or all alerts.
    - **Owner**—specifies whether Ensemble searches for only alerts owned by the current user, only unassigned alerts, or all alerts.
- **Search Managed Alerts By ...**
    - **Start Time** and **End Time**—specify the range of the date and time of the initial alert.
    - **Start ID** and **End ID**—specify the range of the alert ID.
    - **Minimum Escalation Level** and **Maximum Escalation Level**—specify the range of escalation levels.
    - **Source Config Item**—specifies the component that sent the initial alert.
    - **Alert Group**—specifies the alert group.

If you select an alert, the alert details are displayed in a panel to the right. The alert details panel displays the same information as described in the previous section, but you cannot update any of the values. Instead of displaying the alert state as a check box, the panel displays an open alert as `1` and a closed alert as `0`.

13.6 Managed Alerts Walkthrough

There is no sample in ENSDEMO that demonstrates alert management. You can start with any of the example productions and follow the instructions in this section to demonstrate alert management. As an example, this walkthrough starts with the Demo.HL7.MsgRouter.Production production. This starting point uses a routing alert processor, but you can follow the same steps with a production that does not have any alert code. In order to complete this walkthrough you must have access to an SMTP server to send email.

Note: Because this is a walkthrough of a complex feature, it covers the important steps in the procedure but does not explicitly describe every user action. For example, it assumes that the readers know that they must enable each component in the production and respond to dialog windows.

13.6.1 Open a Sample Production and Delete Any Alert Processor

Open the Demo.HL7.MsgRouter.Production production.

Note: If you want to run the example production as it is to demonstrate a routing alert processor, you need to update some settings. If you are going to replace these components with the alert management components, you can skip this step. To run any alert processor that uses email notification, you need access to an SMTP server and you need to update the following settings:

- EMailAlertOperation
  - **SMTP Server**—specify the name of the server.
  - **SMTP Port**—update the port if the server does not use the standard port.
  - **Credentials**—you have to create credentials using the **Ensemble** > **Configure** > **Credentials** page to specify a username and password that can use the SMTP server to send email.
  - **Recipient**, **Cc**, and **From** fields—specify valid email addresses.
- AlertTable data lookup table—specify valid email addresses.

You will be modifying this production. If you want to retain the original sample, you should export the production and then import it into another namespace. To export a production, select **Production Settings**, **Actions** tab, and **Export** button. To import a production, select **System Explorer**, **Classes**, and the **Import** button. If you want to preserve your work you should not work in the ENSDEMO namespace, as it is cleared when you upgrade Ensemble with a new or maintenance release.

Select the Ens.Alert routing business process. Delete it by clicking **Delete** on the **Actions** tab. You will be adding an Alert Manager and naming it Ens.Alert in the next step.

You can skip deleting the EMailAlertOperation since you would be adding the same component in the next step.

Before you continue on the walkthrough, you should make the following preparations:

- Ensure you have access to an SMTP server. You'll need to know the name and port of the server and have a username and password that provides access to the server.
- Define credentials in Ensemble with the username and password to access the SMTP server. To get to the **Credentials** page, select **Ensemble** > **Configure** > **Credentials**, then specify an ID to identify the credentials, a username and a password.
- Define a user to handle alerts. You can use a user that you've already created on your system or define a new user. To define a new user, select **System Administration**, **Security**, and **Users**. Then click the **Create New User** button. In the walkthrough, we are using the username LabManager, but you can use any name. Give the user the following roles: `%EnsRole_AlertAdministrator` or `%EnsRole_AlertOperator`. The `%EnsRole_AlertAdministrator` role allows a user to update any alert, including an alert that is assigned to another user. The `%EnsRole_AlertOperator` role allows a user to update any alert assigned to that user and to update unassigned alerts.

13.6.2 Adding the Alert Manager, Notification Manager, Alert Monitor, and Alert Operation

Add the following components to your production:

- Business process that must be named Ens.Alert with class Ens.Alerting.AlertManager.
- Business process named NotifyMan with class Ens.Alerting.NotificationManager.
- Business operation named EMailAlertOperation with class EnsLib.EMail.AlertOperation. You can skip this step if the production already has this business operation.
- Business service named AlertMon with class Ens.Alerting.AlertMonitor.

All component names in this walkthrough are arbitrary except Ens.Alert, which is a required name.

13.6.3 Configuring the Production

Select **Production Settings** and the **Settings** tab. The following settings are in the **Alerting Control** group:

- **Alert Notification Manager**—Enter the name that you specified for the Notification Manager. For this walkthrough it is NotifyMan.
- **Alert Notification Operation**—Enter the name you specified for the email alert operation. In this walkthrough it is EMailAlertOperation.
- **Alert Notification Recipients**—Enter valid email addresses, separated by commas. By default, the alert management sends all alert notifications to this email address list.
- **Alert Action Window**—Enter a number of minutes. This is the default number of minutes that a user has to resolve and close an alert before the next reminder message is sent. For this walkthrough, you can leave this setting at the default 60 minutes.

For this and the other settings in this walkthrough, click **Apply**.

Select the EMailAlertOperation operation, and enter the following settings in the **Basic Settings** and the **Additional Settings** group:

- **SMTP Server**—Specify the name of the server.
- **SMTP Port**—Update the port if the server does not use the standard port.
- **Credentials**—Enter the ID of the credentials that you defined in preparatory steps.
- **Recipient** and **Cc** fields—You can leave these fields blank. If you enter email addresses here, all alerts that are sent through this operation will always be sent to these addresses in addition to any addresses specified by the production-wide settings. Ensure that any email addresses are valid.
- **From**—Enter a valid email address.
- **IncludeManagedAlertHistory**—Set to Oldest First.

You can leave the **Alerting Control** settings at their default values. You should ensure that the **Alert On Error** check box is not checked on the EMailAlertOperation operation or any of the alerting business processes or business operations.

If you are sending alert notifications to different distribution lists based on the component that generated the alert, it is useful to specify what alert groups each component belongs to. For large productions with many components, it is more practical to select a subset of alerts based on alert groups than on the individual component names. For this walkthrough, assign the following alert groups to the specified components:

Component	Alert Groups
ABC_HL7FileService	ABCGroup
XYZ_HL7FileService	XYZGroup
Regular_FileOperation	ABCGroup,XYZGroup
Extra_Observations	NotImportantAlertGroup

13.6.4 Starting the Production and Managing Alerts

You have completed adding and configuring the Managed Alert service, processes, and operation. You have not yet defined a rule or transformation so your alert management components provide their default behavior. This is:

- Alert Manager promotes all alerts to managed alerts, leaves alerts unassigned, sets the deadline based on the number of minutes specified in the **Alert Action Window**, and sends all managed alerts to the Notification Manager.
- Notification Manager sends all managed alert messages and reminders to the email addresses set in the **Alert Notification Recipients** using the operation set in the **Alert Notification Operation**. Both of these fields are in the **Production Settings**.
- Alert Monitor queries for open managed alerts that have passed their deadline, resets the next notification time by adding the number of minutes specified in the **Alert Action Window**, sends a reminder notification, but does not escalate the alert.
- Alert operation sends the email message to the distribution list and includes the managed alert history in the message.

The **Alert Groups** settings are not used with the default behavior. You will use them when you add rules and the transformation.

Start the production if it is not already running. In order to use the alert management system, you need to first generate alerts. One easy way to do this is to modify a file service **File Path** to point to a nonexistent directory. For example:

1. Modify the ABC_HL7FileService **File Path** to specify an nonexistent directory and click **Apply**. The component should turn red.
2. Repeat with XYZ_HL7FileService.
3. The production should have sent email to the addresses set in **Alert Notification Recipients**. Check and see if the email has been delivered to this account.
4. Select **Ensemble**, **Monitor**, and **My Managed Alerts**. The two managed alerts should be displayed after you select the **Unassigned** and **Today** tabs. Select an alert. You can now update by reassigning the alert to yourself or another user, by making the next action time earlier or later, by escalating the alert, or by closing the alert. Once you have updated a field, you must enter a reason before you click the **Update** button.
5. If you do not close the alerts, the Alert Monitor will send a reminder message and update the NextMonitorTime when the current NextMonitorTime is reached.
6. You can also go to the Managed Alert Viewer by selecting **Ensemble**, **View**, and **Managed Alerts**. This page allows you to query for alerts including alerts that are closed.

Once you have completed exploring the managed alert user interface, ensure that all managed alerts are closed. If there are any open managed alerts and the Alert Monitor is enabled, it will continue sending reminder messages.

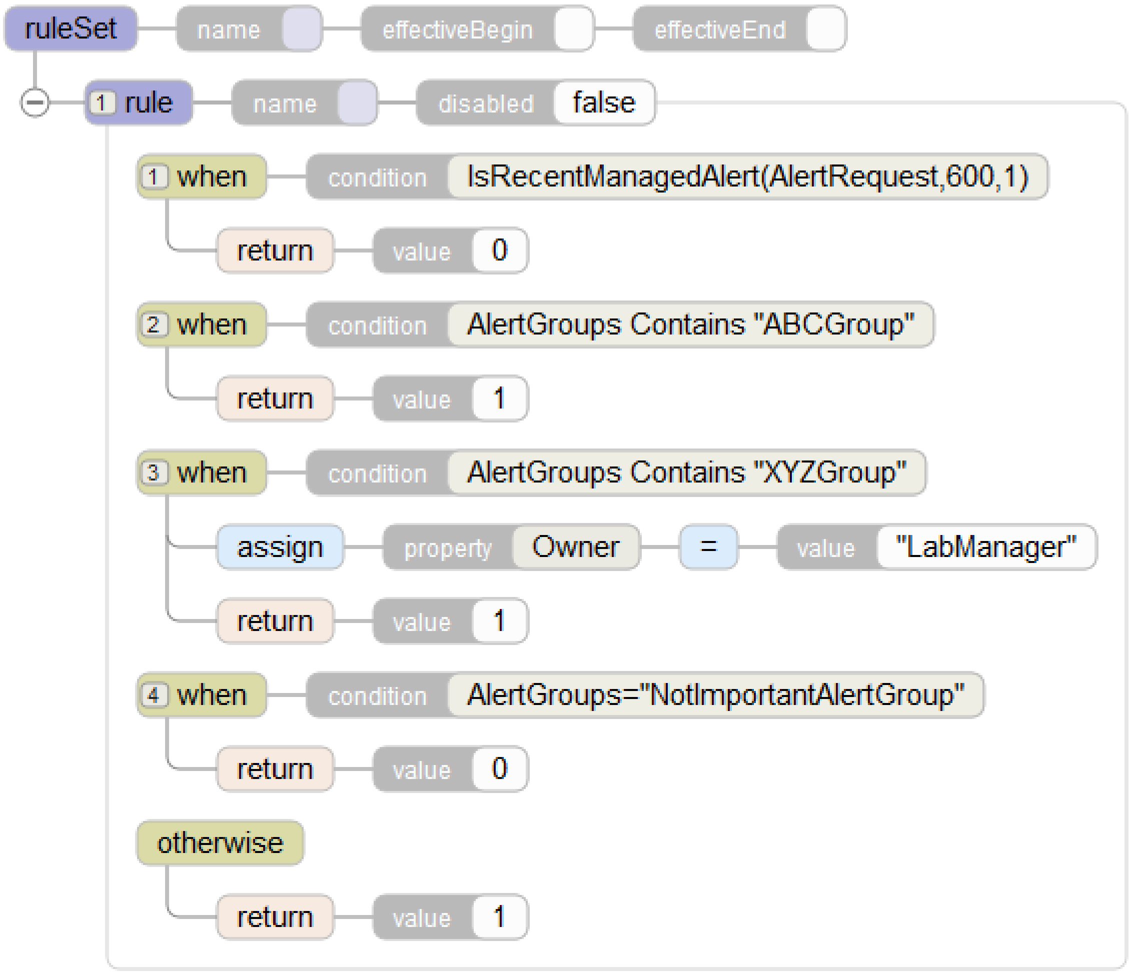
13.6.5 Customizing Alert Management with Rules and Transformations

In this section you customize the Alert Manager and Alert Monitor by defining rules and customize the Notification Manager by defining a transformation.

13.6.5.1 Create an Alert Manager Rule

The Alert Manager rule controls whether a managed alert is created for an alert and can set some properties, such as the alert owner. To create a rule for the Alert Manager, follow these steps:

1. Select **Ensemble**, **List**, and **Business Rules**.
2. Click **New** to create a new rule for the Alert Manager.
   - Enter the package name for the production. In this walkthrough, enter Demo.HL7.MsgRouter.
   - Name the rule. For this walkthrough, name it AlertManCreationRule.
   - Select the type **Creation Rule for Managed Alerts**. This sets the context to Ens.Alerting.Context.CreateAlert.
3. Use the rule editor to enter the following rule. Replace LabManager with a Caché username on your system.



- ruleSet — name — effectiveBegin — effectiveEnd
  - 1 rule — name — disabled: false
    - 1 when — condition: IsRecentManagedAlert(AlertRequest,600,1)
      - return — value: 0
    - 2 when — condition: AlertGroups Contains "ABCGroup"
      - return — value: 1
    - 3 when — condition: AlertGroups Contains "XYZGroup"
      - assign — property: Owner = value: "LabManager"
      - return — value: 1
    - 4 when — condition: AlertGroups="NotImportantAlertGroup"
      - return — value: 0
    - otherwise
      - return — value: 1

4. Save the rule.
5. In the production configuration page, select Ens.Alert and set the **CreateManagedAlertRule** property to Demo.HL7.MsgRouter.AlertManCreationRule.

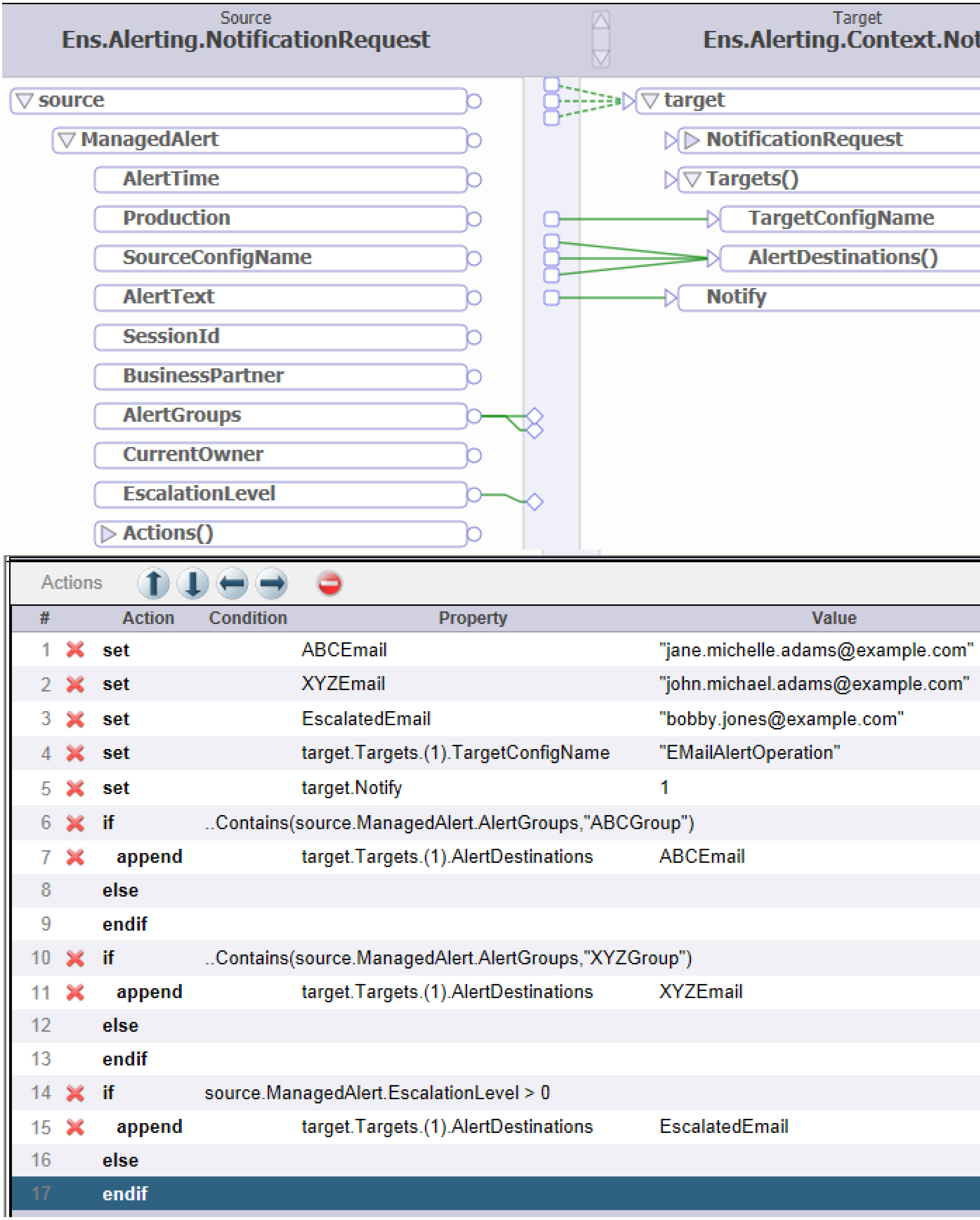
Once you have added this rule, the alert management has the following behavior:

- If the new alert has a corresponding open managed alert that was caused by the same production component, has the same alert text, and was created within the previous 600 seconds, the Alert Manager adds a reoccurs action to the existing managed alert and does not create a new managed alert.
- Alerts in the group ABCGroup create unassigned managed alerts.
- Alerts in the group XYZGroup create managed alerts assigned to the specified user.
- Alerts in the NotImportantAlertGroup do not create a managed alert and are only written to the log. To test this, create an error in the Extra_Observations operation, which is in this alert group. To do this, change the operation’s **File Path** to a nonexistent folder and place a file whose name starts with “ABC” and that contains an ORU_R01 message in the ABC_HL7FileService input folder, C:\Practice\in. This should create an alert that is written to the log file but that does not create a managed alert.

13.6.5.2 Creating a Notification Manager Transformation

The Notification Manager data transformation controls the operations that the targets for a notification and the destination addresses for the message. To create a transformation for the Notification Manager, follow these steps:

1. Select **Ensemble**, **List**, and **Data Transformations** .
2. Click **New** to create a new Data Transformation for the Notification Manager.
   - Enter the package name for the production. In this walkthrough, enter Demo.HL7.MsgRouter.
   - Name the data transformation. For this walkthrough, name it NotifyManTransform.
   - Specify the Source Class as Ens.Alerting.NotificationRequest.
   - Specify the Target Class as Ens.Alerting.Context.Notify.
3. On the **Transform** tab. set the **Create** property to **existing**.
4. Add the actions shown in the following illustration.



#	Action	Condition	Property	Value
1	set		ABCEmail	"jane.michelle.adams@example.com"
2	set		XYZEmail	"john.michael.adams@example.com"
3	set		EscalatedEmail	"bobby.jones@example.com"
4	set		target.Targets.(1).TargetConfigName	"EMailAlertOperation"
5	set		target.Notify	1
6	if	..Contains(source.ManagedAlert.AlertGroups,"ABCGroup")		
7	append		target.Targets.(1).AlertDestinations	ABCEmail
8	else			
9	endif			
10	if	..Contains(source.ManagedAlert.AlertGroups,"XYZGroup")		
11	append		target.Targets.(1).AlertDestinations	XYZEmail
12	else			
13	endif			
14	if	source.ManagedAlert.EscalationLevel > 0		
15	append		target.Targets.(1).AlertDestinations	EscalatedEmail
16	else			
17	endif			

5. Save and compile the transformation.

6. On the production configuration page, select NotifyMan and set the **NotificationTransform** to the transformation Demo.HL7.MsgRouter.NotifyManTransform.

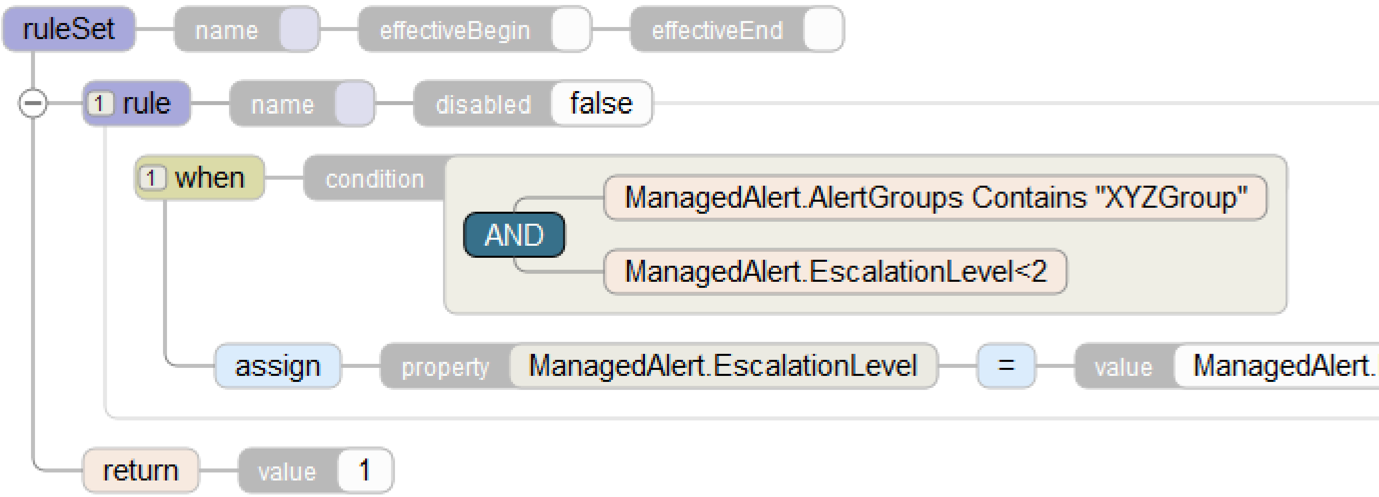
Once you have added this data transformation, the production has the following behavior:

- Managed alerts with “ABCGroup” in AlertGroups are sent to the email address in ABCEmail.
- Managed alerts with “XYZGroup” in AlertGroups are sent to the email address in XYZEmail.
- Managed alerts that have an EscalationLevel 1 or greater are sent to the email address in EscalatedEmail.

13.6.5.3 Creating an Alert Monitor Rule

The Alert Monitor rule controls whether overdue alerts are escalated and whether reminder notices are to be sent. To create a rule for the alert monitor, follow these steps:

1. Select **Ensemble**, **List**, and **Business Rules**.
2. Click **New** to create a new rule for the Alert Monitor.
    - Enter the package name for the production. In this walkthrough, enter Demo.HL7.MsgRouter.
    - Name the rule. For this walkthrough, name it AlertMonitorRule.
    - Select the type **Overdue Rule for Managed Alert**. This sets the context to Ens.Alerting.Context.OverdueAlert.
3. Use the rule editor to enter the following rule:



4. Save the rule.
5. In the production configuration page, select AlertMon and set the **OverdueAlertRule** property to Demo.HL7.MsgRouter.AlertMonitorRule.

Once you have added this rule to the production, it has the following behavior:

- Any managed alert in “XYZGroup” with an escalation level of 0 or 1 will have its escalation level incremented when it passes through the alert monitor.
- Since the rule does not assign a new value to NewNextActionTime, the Alert Monitor has its default behavior, which is to set the NextNotificationTime to the current time plus the number of minutes specified in the **AlertActionWindow** property.

14

Monitoring Activity Volume

The Activity Volume Statistics and Monitoring package provides short-term monitoring of system performance and long-term reporting on message traffic.

This chapter contains:

14.1 Activity Monitoring Overview

Activity monitoring can be useful for tasks such as:

- Monitoring the system health—Using the dashboard provides a quick window into your Ensemble system performance. If the message duration or queue size is growing, it may indicate a performance issue.
- Trouble shooting problems—Using the dashboard can help diagnose a current or past problem. You can use it to determine if a specific configuration component was the primary cause of a performance bottleneck.
- Tracking performance and activity growth to aid in capacity planning—By reviewing long-term changes in message volume you may be able to estimate future growth. You can plan for increased capacity before encountering a significant performance issue.

The Activity Monitoring package provides:

- A centralized store for message statistics.
- A data model that makes it easy to analyze and report on the statistics using SQL or DeepSee.
- DeepSee dashboard showing message current message rates and response times for each interface.
- Variable granularity for long term and short term statistics.
- Long term storage of message statistics available for historic reporting.

- Custom statistic collection using application specific metrics.

This package stores summary statistics that contain information such as the number of messages that pass through a configuration component and the average time to process the message. This summary information is stored in a compact, efficient manner and can be maintained over very long periods of time without requiring extensive amounts of storage.

The monitor provided with this feature allows you to display the current data over several different time periods. But the statistics stored in the database provide a richer set of data. You can use the analysis and reporting tools of your choice to analyze long-term trends or to compare the volume trends during peak traffic times. This capability allows you to analyze and troubleshoot problems with overloaded components and to track long-term load changes so that you can provide additional resources before problems develop.

All classes that inherit from Ens.BusinessService, Ens.BusinessProcess, or Ens.BusinessOperation can use the built-in activity monitoring. In addition, you can use custom code to include custom data in your activity monitoring.

The Activity Volume Statistics and Monitoring package allows you to monitor multiple namespaces running on a single instance of Ensemble and collect the statistics from these namespaces in a single database.

Note: The DeepSee dashboard user interface supports the ability to display statistics from multiple instances of Ensemble, but the mechanism to gather statistics from multiple instances is experimental and should not be used in production environments.

14.2 Enabling Activity Monitoring

You can monitor activity for a single namespace or for multiple namespaces that are running on the same instance of Ensemble. To enable activity monitoring, follow this procedure:

1. For each namespace that you want to monitor:
   a. Add the `Ens.Activity.Operation.Local` business operation to the production in the namespace.
   b. Configure the operation so that it specifies the namespace in which you will be collecting the statistics.
   c. Enable the operation.
   d. To enable statistics collection for all configuration items in the production, call the **EnableStatsForProduction()** method or, to enable statistics collection for individual configuration items, call the **EnableStatsForConfig()** method. For example, to enable statistics collection for all production configuration items in the ENSDEMO namespace, enter the following in the Terminal:

```
zn "ENSDEMO"
do ##class(Ens.Util.Statistics).EnableStatsForProduction()
```

2. If your business service calls **SendRequest** methods directly, you must add the statistics recording APIs as described in Writing Custom Code to Record Activity. If your business service uses **OnProcessInput**, you can skip this step.
3. Create or select an existing Ensemble namespace to use for data collecting. If you are only collecting activity data from a single namespace, you can select that namespace to collect the data or you can create a new one. If you are collecting data from multiple namespaces, we recommend that you create a new namespace and only use it for collecting the statistics.
4. Make it possible to use DeepSee to access to the data by enabling DeepSee in the namespace's default web application. To do this:
   a. Select **System Administration** > **Security** > **Applications** > **Web Applications**.

b. Select the default web application for the namespace. For example, if the namespace is WATCHACTIVITY, the default web application is typically /csp/watchactivity.

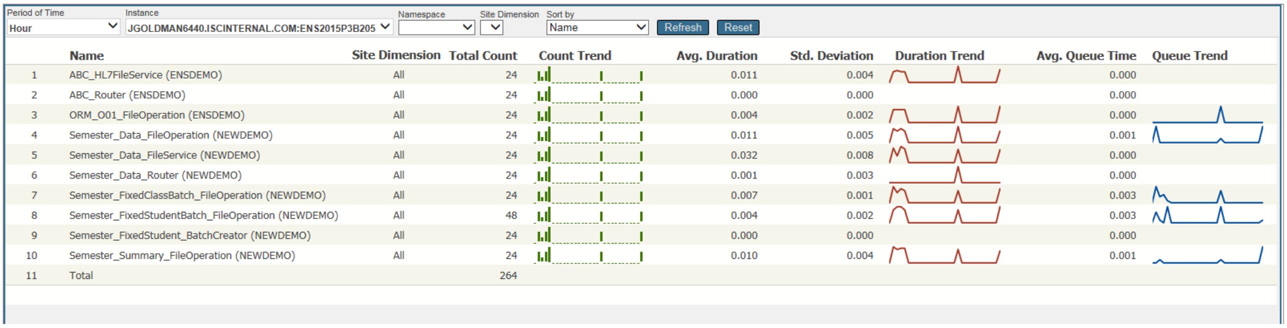
c. Select the **DeepSee** check box.

d. Select **Save**.

14.3 Using the Activity Monitor Dashboard

The Activity Monitor Dashboard is a DeepSee dashboard that displays the activity statistics. To go to the dashboard, ensure that you are in the namespace being used to collect statistics and select **Ensemble** > **Monitor** > **Activity Volume and Duration**. The dashboard is defined in the class Ens.DeepSee.ActivityVolumeAndDurationDashboard .

The dashboard displays current activity information for each production configuration item.



Period of Time: Hour
Instance: JGOLDMAN6440.ISCINTERNAL.COM:ENS2015P3B205
Namespace:
Site Dimension:
Sort by: Name

	Name	Site Dimension	Total Count	Count Trend	Avg. Duration	Std. Deviation	Duration Trend	Avg. Queue Time	Queue Trend
1	ABC_HL7FileService (ENSDEMO)	All	24		0.011	0.004		0.000	
2	ABC_Router (ENSDEMO)	All	24		0.000	0.000		0.000	
3	ORM_O01_FileOperation (ENSDEMO)	All	24		0.004	0.002		0.000	
4	Semester_Data_FileOperation (NEWDEMO)	All	24		0.011	0.005		0.001	
5	Semester_Data_FileService (NEWDEMO)	All	24		0.032	0.008		0.000	
6	Semester_Data_Router (NEWDEMO)	All	24		0.001	0.003		0.000	
7	Semester_FixedClassBatch_FileOperation (NEWDEMO)	All	24		0.007	0.001		0.003	
8	Semester_FixedStudentBatch_FileOperation (NEWDEMO)	All	48		0.004	0.002		0.003	
9	Semester_FixedStudent_BatchCreator (NEWDEMO)	All	24		0.000	0.000		0.000	
10	Semester_Summary_FileOperation (NEWDEMO)	All	24		0.010	0.004		0.001	
11	Total		264						

You can select the statistics to display by time period, instance, namespace, and site dimension:

- Period of Time—After selecting the time period, select the check mark to set it. The options are:
  - Minute—displays the activity in the previous minute.
  - Hour—displays the activity in the previous hour.
  - Day—displays the activity in the previous day.
  - Week—displays the activity in the previous week.
  - Month—displays the activity in the previous month.
  - Year—display the activity in the previous year.
  - All—displays all the stored statistics.
- Instance—You can select the instance of Ensemble that you want to view the activity. If you select the instance, then the instance name is not included in the name column. This reduces the width of the table and helps it fit on the screen. After selecting the instance, select the check mark to set it.

  Note: The DeepSee dashboard user interface supports the ability to display statistics from multiple instances of Ensemble, but the mechanism to gather statistics from multiple instances is experimental and should not be used in production environments.

- Namespace—You can select the namespace that you want to view the activity. If you filter on a namespace, then the namespace does not appear in the name column, reducing the width of the display.
- Site Dimension—Custom property set by code. See the **RecordStats()** method or to the **SetStatsUserDimension()** method for information on setting the site dimension.

You can sort the statistics by: Name, Total Count, Avg. Duration, or Avg. Queue Time.

The dashboard automatically refreshes every 60 seconds. You can also refresh it by selecting **Refresh**. The **Reset** button resets the selection fields to their initial default values.

For each configuration item reporting statistics, the dashboard displays the following information:

- Name—The configuration item name in the production. The instance and namespace are included in parentheses. If you have filtered activity based on instance or namespace, that item is omitted.
- Site Dimension—Identifying information that can be included by custom code. If you filter by the site dimension, this column displays the site dimension value. Otherwise it displays “All”.
- Total Count—Total number of messages during the specified time period.
- Count Trend—Graphic representation of the count within the specified time period. For example, if the time period is a week, the graph shows the count for each day in the week.
- Average Duration—Average time to process the message in the component.
- Standard Deviation—Standard deviation on the times needed to process the message in the component.
- Duration Trend—Graphic representation of the average duration within the specified time period.
- Average Queue Time—Average time message remained on the queue during the specified time period.
- Queue Trend—Graphic representation of the average queue wait time within the specified time period.

Although this dashboard can only display activity for the previous minute, hour, day, week, month, or year, you can design your own dashboard with more flexibility. For example, you could create a dashboard that displays the activity from 9AM to 6PM for a specified day.

14.4 Writing Custom Code to Record Activity

In addition to using the built-in statistics mechanism, you can write custom code to:

- Fill in the site dimension field of the recorded statistics. This allows you to provide additional information in the statistics.
- Explicitly record custom statistics to be stored in the database. In this case, you don’t activate statistics for the component. You are using the mechanism that aggregates the data and transfers the statistics data from the temporary storage to the permanently stored database.

To specify the site dimension to be recorded with the statistics, use the **SetStatsUserDimension()** method. For example, the following code first checks that statistics are enabled and then it sets the site dimension to “CriticalAction”.

```
 If ##class(Ens.Util.Statistics).StatsStarted(..%ConfigName) {
  Do ##class(Ens.Util.Statistics).SetStatsUserDimension(..%ConfigName,"CriticalAction")
 }
```

The **RecordStats()** method writes the specified statistics data to the temporary storage. The data will be aggregated to the three tables. Consider the following example:

```
Do ##class(Ens.Util.Statistics).RecordStats(0,"IncomingMsgSrvc","ActiveMsgs",1,4087,35)
```

The parameters have the following meaning:

- 0—specifies the unknown host type.

- "IncomingMsgSrvc"—used as the configuration item name. This does not have to match the component’s configuration item name.
- ”ActiveMsgs”—used for the site dimension.
- 1—specifies to release temporary memory after writing the statistics.
- 4087—specifies that 4087 messages were processed.
- 35—specifies a total duration for the messages of 35 seconds.

14.5 Accessing the Activity Monitor Database

The activity statistics are stored in these three tables. The three tables contain the data about the same activity, but using a different time period to aggregate the data. The three tables are:

- Ens_Activity_Data.Seconds—aggregates activity over 10-second intervals.
- Ens_Activity_Data.Hours—aggregates activity over 1-hour intervals.
- Ens_Activity_Data.Days—aggregates activity over 1-day intervals.

This redundancy in storage provides the flexibility to minimize the long-term storage required for the statistics without losing the ability to examine historic data. For example, you can use the Ens_Activity_Data.Seconds table to closely examine activity over the previous two days but purge data after two days to minimize storage. The Ens_Activity_Data.Hours and Ens_Activity_Data.Days tables store less data and can be purged much less frequently. You can use the Ens_Activity_Data.Hours table to examine how activity changes over the course of a day. For example, you could use it to generate a report of the peak activity periods during each day of the week and the impact it has on delays and queue sizes.

14.6 Purging the Activity Monitor Database

Although the tables storing the activity statistics data are much smaller than the total size of the corresponding messages, you should purge the activity statistics tables on a regular basis. The PurgeActivityData task purges the specified activity table. You specify the amount of data to keep by specifying a number and a time unit. For example, you could retain 7 days of data for the Seconds table, 12 months of data for the Hours table, and 3 years of data for the Days table.

For details on how to create a task, see “Using the Task Manager” in *Managing Caché*.