Precise for J2EE

User Guide

Version 10.1

IDERA

Precise for J2EE User Guide

Global Headquarters
Brookhollow Central III
2950 North Loop Freeway West, Suite 700
Houston, Texas 77092
Phone: +1-877-693-1886
Fax: +1-650-898-1666

Third-party legal notices

Third-party software may be recommended, distributed, embedded, or bundled with this Precise product. Such third-party software is licensed separately by its copyright holder. All third-party copyrights associated with this product are listed in the accompanying release notes.

Technical support

For technical assistance, visit our customer portal at http://www.idera.com/support/productsupport where you can find an extensive knowledge base, product updates and our online community forums. You can also contact our Customer Support Team via our customer portal, or go to http://www.idera.com/support/productsupport for a list of our support access numbers in your country.

Contents

Precise for J2EE Basics

This section includes the following topics:

- About Precise for J2EE workspaces
- How most workspaces are structured
- About drilling down in context
- Tasks common to most workspaces
- About configuring Precise for J2EE settings

About Precise for J2EE workspaces

The Precise for J2EE user interface is comprised of three workspaces - Dashboard, Activity, and Memory & Statistics. Each workspace has a unique focus, highlighting a specific aspect of the performance of your J2EE applications. You can easily navigate between these workspaces to examine the information necessary to successfully track your system's performance.

The table below describes the three workspaces displayed in Precise for J2EE.

Table 1-1   Precise for J2EE workspaces

Workspace	Description
Dashboard	Lets you quickly visualize the overall health and status of all instrumented application server instances. The Dashboard workspace provides support for detailed views of individual application servers, as well as top-level summary views of multiple application servers. See About the Dashboard workspace.
Activity	Lets you investigate your JVM execution tree in a deep or shallow manner, and to track SQL execution, locks and exceptions in the same manner. See About the Activity workspace.
Memory & Statistics	Lets you examine Application Server metrics. See About the Statistics workspace.

You can easily switch between the different workspaces using the Workspace Selection bar. See Switching to a different workspace.

How most workspaces are structured

Though each workspace is structured differently, most workspaces consist of a Precise bar, workspace selection bar, workspace bar, and two areas - the Main area and the Association area. The Main area (left or top pane) features either a table or a tree view of all monitored J2EE instances, and the Association area (right or bottom pane) includes various workspace specific tabs that display information in various formats, such as tables, graphs, or charts.

The information displayed in one area of the workspace has a direct relation to the information displayed in the other area. When you perform an action on any element in one pane, the information displayed in the other pane changes to reflect this action and to show information for the current selected context.

The figure below shows the common workspace elements.

Figure 1-1   Sample Dashboard Workspace Structure
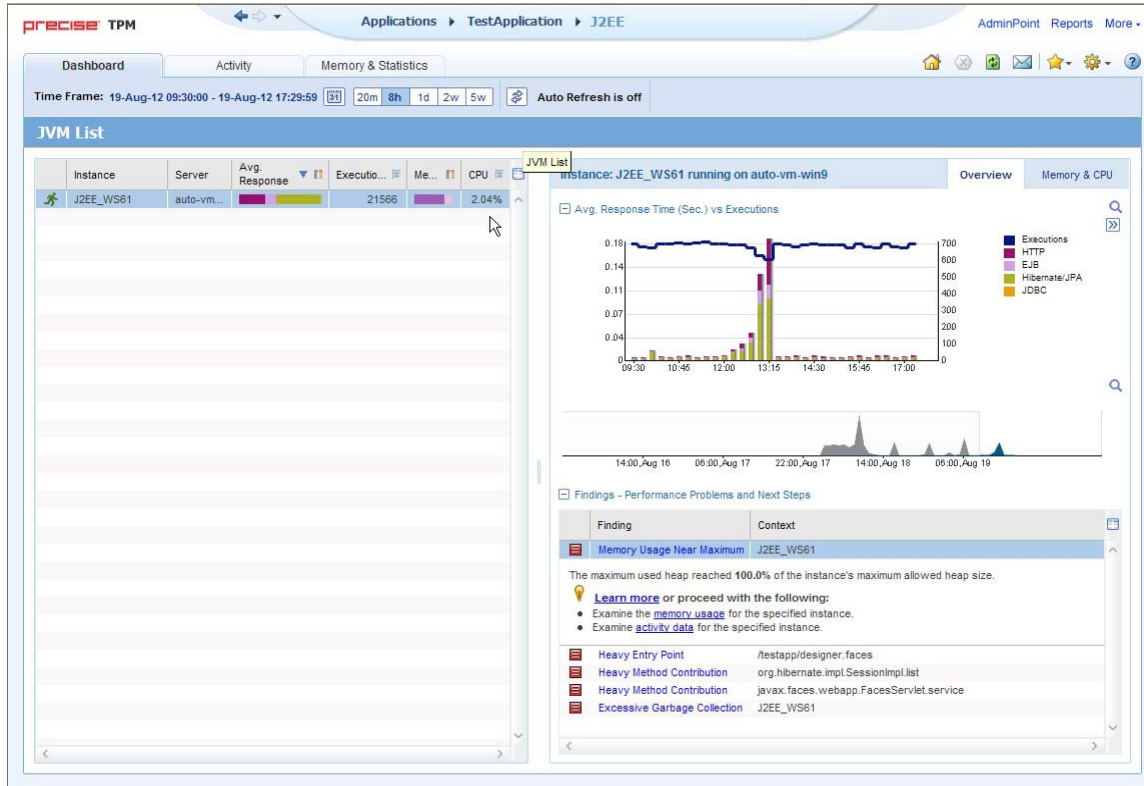
About the Precise bar

The Precise bar enables you to keep track of where you have been and provides various controls.

The following table describes the function of each of the toolbar buttons.

Table 1-2   Precise bar functions

Icon	Name	Description
	Back	During a work session, keeps track of where you have navigated to. The Back button enables you to navigate between previously visited views. The Back control displays your previous view.
	Forward	Enables you to navigate to the next view. This button is only enabled if you clicked Back or if you chose a history option.
AdminPoint	AdminPoint	Opens Precise AdminPoint.
	Home	Navigates to the highest level entity, usually the instance or AppTier (all instances). The time frame settings remain the same.
	Stop	Stops a request for information from the server.
	Refresh	Updates the data currently displayed.
	Favorites	Enables you to add or remove favorites in your Favorites list.

Table 1-2   Precise bar functions

Icon	Name	Description
	Print	Opens the Print Option dialog where you can select the area of the screen to print.
	Mail	Opens the default email program with a link in context to the displayed environment.
	Settings	Opens one of the following setting configuration: ■ Monitor Settings ■ Tree View Settings ■ Display Settings ■ Time Frame Settings ■ Dynamic Types Allocation
	Help	Opens the online help in context.

About the Main area

In most workspaces, the left or top area is the Main area. This area displays an overview of all instances monitored by your Precise product. This information can be displayed in either graph, table, or tree format. The time frame that the information is displayed for can be seen in the workspace toolbar, alongside the selected instance name (and drop-down menu) and auto-refresh status. The times displayed are the local times on the FocalPoint server where the page was generated.

About the Association area

The Association area displays in-depth information for the entity selected in the Main area. Each workspace has analysis tabs specific for that workspace. As you navigate through Precise for J2EE, the analysis tabs change to enable you to view specific information relevant to the selected workspace and entity.

For example, the following analysis tabs are available in the Dashboard workspace:

■ Overview

■ Memory & CPU

About the Findings area

For selected workspaces in Precise products, the association area includes the Findings area, displaying problematic findings for the application. The findings feature is a high level tool, designed to provide the user with an overview of performance issues within the monitored application and enable quick and efficient navigation to the relevant tab for further analysis and handling. The displayed performance findings may indicate performance deteriorations as well as incorrect methods usage. Each finding appears as a row in the displayed table of findings. Hover the mouse indicator over the single-line displayed finding to expand the finding. When expanded, the finding details area provides important guidelines as to what may be the root cause for the reported problem, and what the recommended steps are to resolve this problem.

The table below describes the information that is displayed in the Findings area.

Table 1-3   Information displayed in the Findings area

Column	Description
Severity	The severity of the finding is calculated using a formula. The position of the finding in the list is determined by an internal scoring system that is based on the knowledge of Precise product experts. The severity is indicated by the following colors: ■　　Red: High severity ■　　Orange: Medium severity ■　　Yellow: Low severity ■　　Blue: No severity - the finding is strictly informative By default, findings are displayed according to severity.
Finding	A short name of the Finding.
Context	Entity/Method name (unless specified for the whole instance). The entity/method name is a short name but the long name is displayed in the ToolTip. Some of the findings are identified in specific Methods while others are relevant for the entire instance. In the latter case, a finding is specified as an instance-related finding. **Note:** Rows are aggregated by finding and the URI/method/SQL name and not by ID.
Finding overview	Displays specific details regarding the finding in context. **Note:** This is displayed in the expanded view only.
Learn more (advice)	Provides recommendations for solving the selected finding. For each finding, it lists all relevant pieces of advice and all applicable solutions. You should carefully review all data for the finding and then choose the advice that best suits your needs. **Note:** This is displayed in the expanded view only.
Proceed with the following (bullets)	Provides expert knowledge about the selected finding. The information displayed will direct you if you have difficulties deciding which advice to take or which solution to implement. **Note:** This is displayed in the expanded view only.

Investigating a finding

Perform the following steps to investigate a finding.

To investigate a finding

1   In the Time Frame list, select the period of time you want to analyze.

2   In the All JVMList table, select the instance you want to investigate.

3   In the Finding area, review the top Findings for the selected instance displayed in the Findings table. The findings displayed in this table are sorted by severity.

4   In the Findings table, select the finding you want to analyze further.

5   In the selected finding (the expanded view), read the data displayed for the finding and follow any links provided to view additional information (advice) or next steps (bullets) to resolve the problem.

For more information, see [Examining findings](#).

About drilling down in context

The term 'in-context' means that you can display additional information on a selected item by drilling down to another workspace or view. The filter settings you defined (for example, the selected time frame you chose) and the entity you selected are carried over to the other view or workspace, to allow you to continue analyzing your subject from a different perspective. This concept takes on slightly different meanings depending upon where you are attempting to drill down in context from. In short, the information displayed when drilling down in context is always related to your original selection's settings.

For example, when viewing information on an instance in the Dashboard workspace, you can click a link in the Details area (right pane) to view additional information on the related workspace, in context of your original selection.

Or when viewing a list of alerts for your product in Alerts, you can open your product in context and continue investigating the factors that led the system to issue that alert.

In short, the information displayed when drilling down in context is always related to your original selection's settings.

How a drilldown affects the workspace heading

The workspace heading displays the name of the currently selected entity on this screen. When you drill down to a new entity in the Association area, the Workspace heading changes to reflect the name of the newly selected entity.

About viewing data in the Precise for J2EE workspaces

Precise for J2EE workspaces display information in the following formats:

■ Bar graphs for a selected time frame.

■ Line graphs for a selected time frame.

■ Tables for a selected time frame.

Bar graphs for a selected time frame

The user interface displays data per time slice in bar graphs. Bar graphs summarize one slice of data and generally compare several values of similar type, such as time or throughput. For example, contributors of a higher level metric are displayed as bars in the graph, with their type and percentage work time displayed by putting the mouse over the area of interest (see the "Workspace terminology" figure). Each displayed metric value is an average of observations within the slice.

Line graphs for a selected time frame

The user interface displays overtime data for a defined time frame in line graphs. Line graphs summarize many slices of data. You can change the number of items plotted by selecting or unselecting check boxes located to the right of the graph. Each point in the graph represents the total metric value for a time slice included in the time series.

Tables for a selected time frame

The various Precise for J2EE workspaces display information in tables. The way this information is viewed can be modified according to the user's needs, by clicking on the icon to the right of the column header. The following table describes the displays views available.

Table 1-4   Column display modes in tables

Display Mode (Icon)	Description
Normalized Graph ( )	This mode displays the breakdown bar in full size for each entity. In this mode you can clearly see the relative breakdown per type for each entity in the table.
Stacked Bar Graph ( )	This mode displays a normalized view of the breakdown bar for each entity in relative size to the entity with the highest value. In this mode, you can see the actual breakdown per type (in seconds) for each entity.
Numerical ( )	This mode displays the numerical value in the relevant unit size.
Bar Graph ( )	This mode displays a solid bar graph for each entity in relative size to the entity with the highest value.

Tasks common to most workspaces

The following tasks are commonly performed in most workspaces:

■   [Switching to a different workspace](#)

■   [Selecting a time frame](#)

■   [Selecting an instance](#)

■   [Auto Refresh data](#)

■   [Associating entities with data that meets specific criteria](#)

■   [Focusing on information in overtime graphs](#)

■   [Determining which table columns to display](#)

■   [Adding, viewing, and deleting Favorites](#)

■   [Exporting to Precise Custom Portal](#)

Switching to a different workspace

You can easily switch between the different workspaces using the Workspace Selection bar. When you start your Precise product, usually the Dashboard workspace opens by default.

To select a workspace

1   Click a button on the Workspace Selection bar to display information on the selected entity in a different workspace.

Selecting a time frame

You can configure Precise for J2EE to display transaction performance data for a specific time frame using the predefined time frame options or calendar icons.

Selecting a predefined time frame from the toolbar displays transaction performance data for the selected time period up to the current time. See [Selecting a predefined time frame from the Precise for J2EE toolbar](#).

Selecting the time frame using the calendar icon, you can choose to define a time range independent of the current time, or to define a time range up to the current time. See [Selecting a time frame using the calendar icon](#).

The predefined time frame options are:

■   Last 20 minutes (20m) (default)

■   Last 8 hours (8h)

■   Last 1 day (1d)

■   Last 2weeks (2w)

■   Last 5 weeks (5w)

The time frame selected affects all information displayed in Precise for J2EE. Only data that falls within the selected time frame is shown in these areas.

Selecting a predefined time frame from the Precise for J2EE toolbar

To select a predefined time frame:

•   From the Precise for J2EE toolbar, select one of the predefined time frames.

Selecting a time frame using the calendar icon

To select a time frame

1   Click the calendar icon. In the dialog box that is displayed perform one of the following:

    a   To define a time frame independent from the current time, select the 'Time Range' option and select the Start and End dates and times.

    b   To define a time frame up to the current time, select the 'Last' option and enter the desired time frame.

    c   To use one of the three previously used time frames, select the 'Recently used' option and from the drop down menu select the desired time frame.

    d   To use a previously saved time frame, select **Use a previously saved time frame** and from the drop down menu select the desired time frame.

2   To save your settings for future access, select **Save these definitions for future use as:** and enter a name in the corresponding field.

3   Click **OK**.

Selecting an instance

You can select a specific instance to examine in more detail.

To select an instance

- In the Instance field (located to the right of the Time frame), select the instance you want to examine from the drop-down list.

Auto Refresh data

You can configure Precise for J2EE to automatically refresh collected data displayed on a workspace. By default the option is Auto Refresh is off.

To Auto Refresh data

- Click **Auto Refresh is off/Auto Refresh is on.**

Associating entities with data that meets specific criteria

You can associate the displayed entity with specific data to focus your analysis.

Note: The criteria no longer apply when you drill down to another entity.

To associate entities with data that meets specific criteria

1   Click the arrow located to the left of the Association controls and select **More...**

2   In the Associate With dialog box, in the Entries tab, select the entity you want to associate data with from the Populate table with list.

3   In the Sort entries by list, determine which criteria you want the information to be sorted by and in which order.

4   From the Display top list, select the number of rows to display.

5   On the Criteria tab, do the following for each entity you want to associate data with:

From the left drop-down list	Select an entity.
From the middle drop-down list	Select an operator, such as, Like, <>, Not Like, In, Not In.

In the text box	Type the criteria (case-sensitive for the selected entity.
	If you select the operator Like or Not Like, you can use the % wildcard character to represent 0 or more characters, and the _ wildcard character to represent exactly one character. If you select the operator In or Not In, type a comma to separate values.

6   Click **OK**.

Focusing on information in overtime graphs

Some entities display an overtime graph. The overtime graph displays activity statistics over a specified time period. Depending on the number of points displayed in the graph, you may need to zoom in or out. The text displayed on the x-axis varies according to the time frame. If there is a year or day change, x-axis labels will display accordingly.

Use the vertical or horizontal scroll bars (if displayed) to view additional information on the graph.

Click the double-arrow (>>) icon to either hide or show the overtime graph legend.

Use the small zoom (spyglass) icon, displayed on the upper right of a time range and above the overtime graph legend, to select a desired time frame that you wish to focus. To select an area on the graph, first click and drag the mouse pointer and then click the spyglass icon.

Determining which table columns to display

Tables are used to display information about a set of related entities in either the left or right panes. It is possible to determine which columns to display in the Association area tables.

To determine which columns to display in a table

1   Click the Table icon on the upper right-hand side of a table and select **Column Chooser**.

2   In the Column Chooser dialog box, click the arrows to move the names of the columns that you want to display to the Visible box and the ones that you do not want to display to the Invisible box.

3   Click **OK**.

Sending an email message

You can send an email message to one or more recipients from the Precise toolbar. The default subject for the message will be "Link to a Precise environment".

The email will include a link to the Precise product in the current context (time frame and selected entries).

To send an email message

1   Click the email icon on the Precise toolbar. The default email program opens.

2   Fill in the required fields and click **Send**.

Adding, viewing, and deleting Favorites

The Favorites function enables you to save a specific location in your environment and to retrieve the same location later without having to navigate to it. For this purpose, a friendly user interface similar to the familiar Favorites option in the Internet Explorer has been implemented.

About the Favorites function

The new Favorites function includes the following features:

■   Relative Time Frame - Saving a relative time frame instead of static date. For example, saving the last 7 days will always display the last 7 days, depending on the day entered.

- One click to specific location - Once you open Precise by launching a saved Favorite item, you will not have to enter a login credential nor click the login button.

- IE Favorites support - Adding a new Favorite item in Precise will also add it to the IE Favorites menu.

- Auto Complete - The Favorites dialog includes a new combo box which supports AutoComplete.

- Auto Naming - The Favorites dialog generates item names based on the current location.

UI description

An Add/Delete Favorites option under the Favorites menu allows you to save the current location or delete an existing one.

To add a new Favorite location

1   On the Add/Delete Favorites dialog box, enter the name of the new Favorites entry.

2   Click **Add**. The dialog box is closed and the new Favorite is added to the list.

To view a Favorites location

1   On the Precise bar, click **Favorites**.

2   Select the Favorites location you want to view.

To delete an existing Favorite location

1   On the Add/Delete Favorites dialog box, select the Favorite location to be deleted.

2   Click **Delete**. The dialog box closes and the selected Favorite is deleted from the list.

Note: The favorite address is displayed in the Address field and cannot be edited.

Exporting to Precise Custom Portal

The Export to Precise Custom Portal Portlet feature enables you to export the view of the chosen table or graph and generate a portlet with that view in Precise Custom Portal, so that it will provide you with another way of monitoring your environment.

Prerequisites

To be able to use this feature, you need to have the following rights in Precise:

- View permissions to all AppTiers in the environment

If you do not have sufficient rights, you will get an error message when trying to execute this feature.

Exporting the information

You can either export a table view or a graph view.

Note: The name field has the following restrictions: maximum 100 characters.

To export a table view

1   Click the Column Chooser icon.

2   Select **Export to Precise Custom Portal portlet**.

3   Insert a name in the name field that clearly describes the table view.

4   Click **OK**.

To export a graph view

1   Click the Export icon.

2   Select **Export to ASD**.

3   Insert a name in the name field that clearly describes the graph view.

4   Click **OK**.

About configuring Precise for J2EE settings

The Settings menu on the Precise bar allows you to configure the following:

■   Monitor Settings

■   Tree View Setting (Activity workspace only)

■   Display Settings

■   Time Frame Settings

■   Dynamic Types Allocation

To configure any of the above-named settings

1   On the Precise bar, click **Settings**.

2   Select the setting type (from the menu) you want to configure.

3   On the appropriate settings dialog box, enter your configuration choices. See Configuring Precise for J2EE settings.

Launching Precise for J2EE from Precise StartPoint

The Precise user interface is a server-side Java application that listens on a network port for HTML page requests.

The syntax of the Precise URL address is: `http://<server machine>:<port>/`

where *<server machine>* refers to the Precise FocalPoint server and *<port>* refers to the port number that the FocalPoint communicates on. By default, the port number is 20790.

Note: The Precise FocalPoint and Precise for J2EE FocalPoint must be started before they can be accessed. See the *Precise Administration Guide* to learn how to start Precise and Precise for components.

To access the Precise for J2EE user interface using Precise StartPoint

1   Type the URL address of the StartPoint user interface into the Address bar of your browser and press **Enter**. The Precise login page opens. The login page provides secure access to Precise and to your specific product.

2   Specify your authorized role name and password. By default, both the role name and password are `admin`. For more information on role names, see the *Precise Administration Guide*.

3   Click **Login**. The StartPoint page opens. This is the Precise home page.

4   On the StartPoint page, click the Experts TPM icon on the top right side and select J2EE.

5   Precise for J2EE opens to the Dashboard workspace.

Getting an overview of your environment

This section includes the following topics:

- [About the Dashboard workspace](#)

- [How the Dashboard workspace is structured](#)

About the Dashboard workspace

The Precise for J2EE Dashboard workspace provides a comprehensive overview of the health and status of all instrumented application server instances, based on the data collected according to the application instrumentation. The information displayed in this workspace pinpoints performance trends and issues in your application and offers clear navigational recommendations throughout Precise for J2EE for further analysis and handling.

See [About the Activity workspace](#) and [About the Statistics workspace](#).

How the Dashboard workspace is structured

The Dashboard workspace is divided into two (left and right) panes. The Main area (left pane) displays a list of all the instances monitored by Precise for J2EE and compares them in terms of performance and resource usage. The Association area (right pane) contains two tabs, each displaying overtime graphs of the instances' behavior. The Association area also features the Findings table, providing specific details regarding performance issues and advice for further analysis.

The figure below shows a typical Dashboard workspace.

Figure 2-1   Typical Dashboard workspace
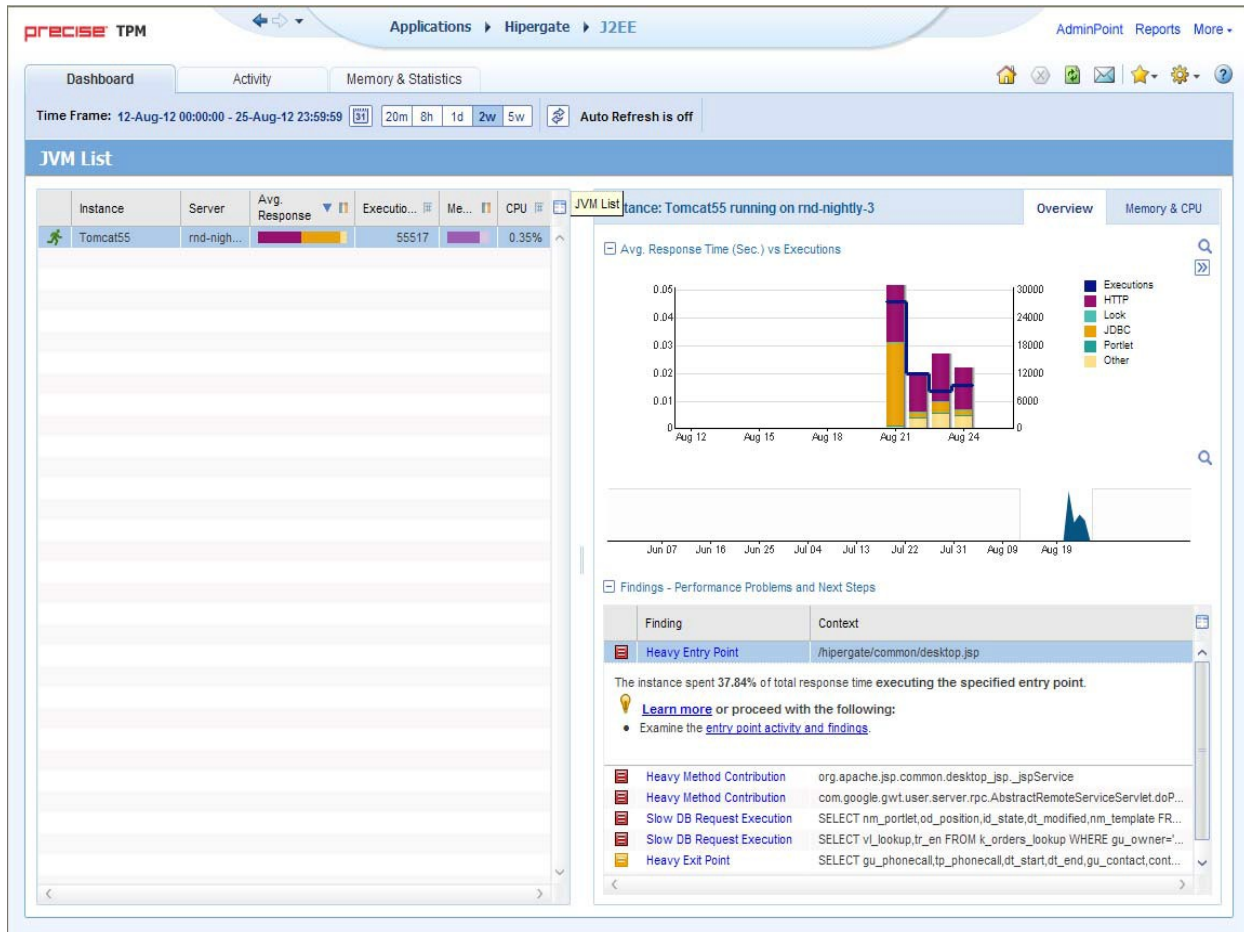
About the Main area in the Dashboard workspace

The Instance (JVM) table in the Main area (left pane) lists all instances monitored by Precise for J2EE. Each row corresponds to an instance. The All row summarizes the activities of all monitored instances.

The following table describes the information displayed for the different instances (JVMs) in the environment, and is sorted alphabetically by instance by default.

For more information regarding viewing the displayed data, see Tables for a selected time frame.

Table 2-1   Instance table

Column	Description
Status Icon	Indicates one of four possible instance availability states: Green man running - available and running Red man standing - available but not running Question mark - not available Blank mark - not monitored
Instance	Name of the instance (JVM).
Server	Name of the server on which the instance runs.
Avg. Response Time (Sec)	Average time it took to process a service request for a specific instance.

Table 2-1   Instance table

Column	Description
Executions	Total number of executions during the selected time frame.
Memory	Average amount of memory used by all the executions of the instance during the selected time frame.
CPU	Average percentage of CPU time used by the instances during the selected time frame.
Availability	Average percentage of time that the instance was available within the selected time frame.
SLA Compliance	Indicates the extent to which HTTP entry point service requests complied with the instance's predefined SLA thresholds.

About the Association area in the Dashboard workspace

The Association area in the Dashboard workspace contains two tabs, providing a performance trends overview and resource consumption information for the selected instance.

The following tabs appear in the Dashboard workspace:

■   About the Overview tab

■   About the Memory & CPU tab

The overtime graphs display instance or entity statistics for the selected time frame. Depending on the number of points displayed in the graph you may want to zoom in or out.

About the Overview tab

The Overview tab displays overtime graphs and a findings table, providing the user with an overview of the performance of all instances, or a selected instance, in the selected time frame.

The information displayed in the Overview tab includes:

■   Avg. Response Time (Sec) vs Executions - an overtime graph displaying a breakdown, by type, of the average response time for the instance, relative to the number of executions.

■   Overtime trends - an overtime graph displaying the performance trends for the instance. The performance trends for the instance in the selected time frame are shown in color, and the performance trends for the instance in the previous and (if applicable) following time frame are shown in grayscale. This view provides the user with a greater overtime perspective of the instance's performance patterns and behavior.

■   Findings table - a table listing the findings detected in the selected time frame, displayed in order of severity. For more information, see About the Findings area.

About the Memory & CPU tab

The Memory& CPU tab displays overtime graphs, providing the user with an overview of the selected instance's resource consumption in the selected time frame.

Note: Information displayed in this tab can only be displayed for a specific instance. Selecting "All" in the instances table in the Main area will display an empty graph area and a message prompting you to select a specific instance.

The information displayed in the Memory & CPU tab includes:

■   Heap Usage - an overtime graph displaying the breakdown of available and used heap. The overtime graph also shows JVM start and restart occurrences.

■   CPU Consumption - an overtime graph displayed the CPU consumption rate. The overtime graph also shows when the JVM starts running or is refreshed.

Examining performance over time

This section includes the following topics:

About the Activity workspace

The Precise for J2EE Activity workspace displays an execution tree of all monitored instances and various analysis tabs, enabling comprehensive and effective drilling down in the monitored instances and their invoked methods to locate specific performance issues and their underlying causes.

How the Activity workspace is structured

The Activity workspace is divided up into two main areas. The Main area (left pane) features an execution tree of all monitored instances. The Association area (right pane) features analysis tabs that provide specific information about the node selected in the execution tree in the left pane. The workspace heading displays the instance name, the server name, and what field the tree is sorted by.

Figure 3-1 Sample Activity Workspace
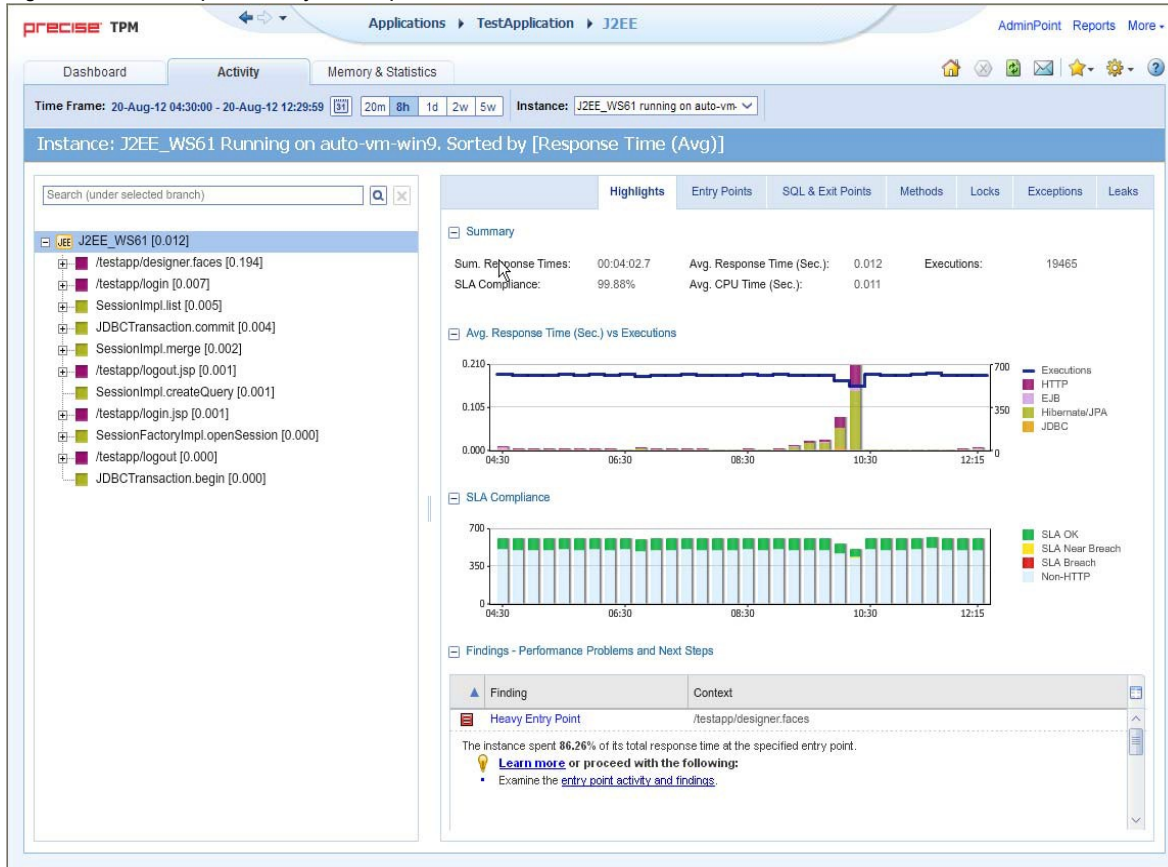
About the Main area

The Main area displays an execution tree of the monitored instance, broken down into the following node types:

■   All instances (default)

■   Instance

■   URI/iView

■   Method (all types)

■   SQL (other exit points are methods)

Selecting a node displays information in context for that node and it's underlying call path in the analysis tabs in the Association area. See About the Association area.

Viewing information in the Main area

By default, the execution tree in the Main area is launched displaying the top instance level (either a selected instance or all monitored instances) and the entry point level only. Navigate through the execution tree by clicking the '**+**' sign next to the node name. This displays deeper layers of the tree, enabling you to view invoked methods and SQLs while maintaining the execution hierarchy.

Note: The degree to which you can drill down in an execution tree is defined by the filtering settings for your application. For more information, see *Configuring Precise for J2EE* in the *Precise Administration Guide.*

The following table describes the information displayed in the execution tree.

Table 3-1   Information displayed in the execution tree

Field	Description
Icon	Displays an icon reflecting the node type
Name	Displays the name of the selected node. Depending on the tree view settings selected, the name will be displayed either its short name, or its full execution context.
Selected "Sorted by" value	Displays an aggregated value of the selected "sort by" option for the node's entire underlying call tree. For example, the JDBC contribution to JVM response time will be an aggregation of all JDBC methods times of that JVM.

To change the current tree view settings, see Configuring Tree View Settings.

Hovering over a node displays a ToolTip. The following table describes the information displayed in the ToolTip:

Table 3-2   Information displayed in a node's ToolTip

Field	Description
Type	Displays the node's method type.
Context	Displays the full execution context of the selected node
Selected "sorted by" type and value	Displays the field by which the tree view is sorted by and the value for the selected node and its underlying call tree.

The table below describes the "sort by" options available for the execution tree.

Table 3-3   Sort options for the execution tree

Sorted by	Description
Response Time (Avg)	Displays the average response time for each node and its underlying call path. The average response time is displayed in seconds.
Response Time (Sum)	Displays the total response time for each node and its underlying call path. The summed response time is displayed in hours: minutes: seconds: milliseconds.
Work Time (Avg)	Displays the average work time in seconds for each node. The average work time is displayed in seconds.
Work Time (Sum)	Displays the summed work time in seconds for each node. The summed work time is displayed in hours: minutes: seconds: milliseconds.
Executions	Displays the number of executions within each node.

All of the sort options can be displayed in ascending or descending order. You can also select the maximum number of executions to display under each node, the maximum number of results to display, and whether to display the method's long or short name.

See Configuring Tree View Settings.

When you right-click on an node in the execution tree, a popup menu is displayed. The following table describes the right-click menu options.

Searching within the execution tree

Above the execution tree in the Main area there is a search bar.

To search for a specific node

1   Enter the desired value in the search bar.

2   Click the search icon to the right of the search field. The heaviest results for the search value are emphasized in bold within the execution tree and their relevant call paths are opened.

Note: If the 'Show short name only' option is selected in the tree view settings dialog and the user enters the full name in the search value, the search will not return any results.

About the Association area

The Association area displays a variety of analysis tabs, corresponding to the selected node in the execution tree. See About the Activity workspace analysis tabs.

About the Activity workspace analysis tabs

The Activity workspace analysis tabs display in-depth information about and in context of the node selected in the execution tree, shown within the tree in bold for referencing. While navigating through the analysis tabs, you can select different hyperlinked entities associated with the originally selected node. Selecting a hyperlinked node in an analysis tab will select that node within the execution tree, and all information displayed in the analysis tabs will be refreshed to correspond to the newly selected node.

The following table describes the analysis tabs available in the Association area of the Activity workspace and the node types for which each tab is displayed. As you navigate through the execution tree, the available analysis tabs change according to the selected node.

Table 3-4   Analysis tabs available in the Precise for J2EE Activity workspace

Tab Name	Description	Displayed For	For more information
Highlights	Displays a comprehensive performance overview of the node selected in the execution tree. This view reveals performance problems of the entire call tree invoked under the selected node.	■ All entities	See The Highlights tab.
Load Balance	Displays information for the selected node throughout the monitored application.	■ All entities **Note:** This tab is only displayed when "All Instances" is selected.	See About the Load Balance tab.
Impact	Displays information regarding the selected node's impact on the application. It displays general performance data for the selected node in the selected time frame and specific information regarding its contribution to all entry points and methods by which it is directly invoked. This view provides the user with a comprehensive overview of the specific method's performance throughout the application.	■ Methods ■ SQLs ■ Exit Points	See About the Impact tab.
Entry Points	Displays and compares performance data of the entry points in the selected time frame. This view enables efficient identification of problematic entry points.	■ Instances/All Instances	See About the Entry Points tab.
SQL & Exit Points	Displays and compares performance data of the SQL and exit points in the selected time frame. This view enables efficient identification of problematic SQLs and exit points.	■ All entities except SQLs.	See About the SQL & Exit Points tab.

Table 3-4   Analysis tabs available in the Precise for J2EE Activity workspace

Tab Name	Description	Displayed For	For more information
Methods	Displays information regarding all the methods and URI/iView invoked in the call tree under the selected node. The view provides a deeper look into a problematic node by showing comprehensive data for all invoked methods.	■ All entities except SQLs	See About the Methods tab.
Locks	Displays all methods in the call tree under the selected node that were locked at any point. This view provides an overview of all locked methods and enables the user to see locking trends.	■ All entities except SQLs	See About the Locks tab.
Exceptions	Displays data for exceptions thrown from the call tree under the selected node. This view pinpoints methods with a high exceptions rate.	■ All entities except SQLs	See About the Exceptions tab.
Leaks	Displays the object allocation data for the call tree under the selected node.	Instances/All Instances	See About the Leaks tab.

The Highlights tab

The Highlights tab appears when any node type within the execution tree is selected, and displays a comprehensive performance overview of the selected node. The highlights tab provides information regarding performance trends for the selection's underlying call path and displays performance findings to facilitate focused navigation to the root cause of a performance issue.

The Highlights tab includes the following components:

- Summary area

- Avg. Response Time (Sec) vs Executions overtime graph

- SLA Compliance overtime breakdown graph

- Findings table

- Avg. JDBC Time (Sec) overtime graph (for SQL only)

- SQL Text (for SQL only)

- Destinations table (for SQL only)

- Findings table

Summary area

The summary area displays the following information for the selected node. The value displayed is the aggregated value for the selected node's underlying call path (unless otherwise stated) in the selected time frame:

- SLA Compliance - displays the relative percentage of HTTP entry point service requests that approached or breached the defined SLA thresholds.

- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.

- Executions - displays the number of executions within the selected node.

- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.

- Avg. CPU Time (Sec) - displays the average time that CPU was consumed for the selected node and its underlying call path.

- Method Type (when a method is selected only) - displays the method type.

- Impact on Entry Point (when a method or an SQL is selected only) - displays the percent value of the selected node's summed response time out of its entry point's summed response time.

- Max. Response Time (Sec) (when an SQL is selected only) - displays the peak response time detected in the selected time frame.

Avg. Response Time (Sec) vs Executions overtime graph

The Avg. Response Time (Sec) vs Executions overtime graph displays a bar for each time slice in the selected time frame that the selected node (or part of its underlying call path) was active. These bars display a breakdown of the invoked method types according to each method type's average response time. The average response time for the entire time slice is compared to the number of executions in the time slice, displayed in linear format.

To the right of the table there is a legend detailing the method types displayed in the graph. Hovering over any point in the graph displays a ToolTip detailing the date and time of the selected time slice bar, the number of executions, and the average response times per method (in seconds) for that time slice.

SLA Compliance overtime breakdown graph

The SLA Compliance overtime graph displays a bar for each time slice in the selected time frame that the selected node (or part of its underlying call path) was active. These bars display a breakdown of the SLA compliance of the invoked methods as follows:

Note: SLA compliance is calculated for HTTP entry points only.

- Green - the invoked method's SLA is below the defined SLA threshold.

- Yellow - the invoked method's SLA is approaching the defined SLA threshold.

- Red - the invoked method's SLA exceeded the defined SLA threshold.

- Blue - top level methods that do not have SLA thresholds defined for them (for example, EJB, JDBC, and so on). These executions will appear as "Non-HTTP" executions.

  Note: Information for non-HTTP methods will not appear in the legend and in the graphs, but will appear in the ToolTip.

Avg. JDBC Time (Sec) overtime graph (for SQL only)

The Avg. JDBC Time overtime graph displays a bar for each time slice in the time frame that the SQL was active. The average JDBC time (comprised of the JDBC time only as it is the lowest point in the execution tree and has no underlying branches) is compared to the number of executions in the time slice, displayed in linear format.

SQL Text (for SQL only)

This area displays the entire name of the selected SQL.

Destinations table (for SQL only)

Displays a table at all SQL destinations with details such as:

- DBMS

- DB Server

- DB Name

- Connection string details

Findings table

The Findings table shows findings for the selected entity and its underlying call tree. Context names that are too long to display are automatically shortened using ellipses.

For more information regarding findings, see "How to identify performance problems" on page 49

About the Load Balance tab

The Load Balance tab displays the selected entity and compares its performance in all the contexts it was invoked by in the selected time frame. If "All instances" is selected in the execution tree, it will show the instances table (similar to the view provided in the Dashboard workspace).

Note: This tab is only displayed when "All Instances" is selected as the top level of the execution tree. When a specific instance is selected as the top level of the execution tree, this tab is not displayed.

The Load Balance tab includes the following:

- Load Balancing between JVMs table
- Avg. Response Time (Sec) overtime graph
- Executions overtime graph

Load Balancing between JVMs table

The Load Balancing between JVMs table displays the following information for all instances:

- Instance - displays the instance name. If you select the highlighted instance, the execution tree will update itself to display the specific instance's execution tree.

  Note: Once a specific instance is selected, the Load Balance tab will no longer appear.

- Server - displays name of the server on which the instance is running.
- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.
- Executions - displays the number of executions within the selected node.
- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.

Avg. Response Time (Sec) overtime graph

The Response Time (Avg) overtime graph displays and compares the average response time of the selected instance in the selected time frame against the average response time frame for the AppTier. If the average response time is identical, the overtime graph will display information for the selected JVM on top of the AppTier average.

Executions overtime graph

The Executions overtime graph displays and compares the average number of executions for the selected instance in the selected time frame against the average number of executions for the AppTier. If the average number of executions is identical, the overtime graph will display information for the selected JVM.

About the Impact tab

The Impact tab displays information regarding the selected method's impact on the application. It displays general performance data in context for the selected node and specific information regarding its contribution to all entry points and methods in the application by which it is directly invoked. This provides the user with a comprehensive overview of the specific method's performance throughout the application.

The Impact tab displays the following information:

- Summary area

- Impact on All Entry Points table

- Impact on All Direct Callers table

Summary area

The summary area displays the following information for the selected entity. The value displayed is the aggregated value for the selected entity's underlying call tree in the selected time frame:

- Method (only when a method is displayed, not SQL) - displays the method type.

- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.

- Executions - displays the number of executions within the selected node.

- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.

- Max. Response Time (Sec) - displays the peak response time detected in the selected time frame.

- Min. Response Time (Sec) - displays the lowest response time detected in the selected time frame.

- Avg. Work Time (Sec) (not displayed for SQLs) - displays the average actual work time of the selected node without the underlying call path.

- Max. Work Time (Sec) (not displayed for SQLs) - displays the maximum work time.

- Avg. CPU Time (Sec) - displays the average time that CPU was consumed for the selected node and its underlying call path. (Not displayed for SQLs).

- Max. CPU Time (Sec) (only when a method is displayed, not SQL) - displays the peak CPU time detected for the selected node.

Impact on All Entry Points table

The Impact on All Entry Points table displays the following information for every entry point that the selected method was invoked by in the selected time frame:

- Entry Point - displays the name of the entry point from which the entity was invoked.

  Note: Selecting the hyperlinked entry point name selects the entry point in the execution tree and displays the highlights tab for the entry point.

- Work Impact on Entry Point - displays the percent value of the selected node's summed response time out of its entry point's summed response time.

- Method Avg. Response Time (Sec) - displays the method's average response time.

- Executions by Entry Point - displays the number of times the selected method was invoked by the specific entry point.

- Sum. Method Response Times - displays the method's total response time.

- Sum. Entry Point Response Times - displays the entry point's total response time.

Impact on All Direct Callers table

The Impact on All Direct Callers table displays the following information for every method that directly invoked the selected method:

- Caller - displays the name of the method that directly invoked the selected method.

Note: Selecting the hyperlinked direct caller name selects the method in the execution tree and displays the Highlights tabs for the method.

- Work Impact on Caller - displays the percentage of the selected method's total response time out of the direct caller method's total response time.

- Method Avg. Response Time (Sec) - displays the method's average response time.

- Executions by Caller - displays the number of times the selected method was executed by the specific direct caller method.

- Sum. Method Response Times - displays the method's total response time.

- Sum. Caller Response Times - displays the direct caller method's total response time.

About the Entry Points tab

The Entry Points tab displays the top level URI/iView or methods (EJB or others) that are monitored for this instance, comparing performance data of the detected entry points within the monitored application in the selected time frame. This view enables efficient identification of problematic entry points and, in the event of many entry points from the same instance showing distress, problematic instances as well.

The Entry Points tab includes the following components:

- Entry Points table

- Avg. Response Time (Sec) vs. Executions overtime bar graph

Note: The SLA Compliance data can be switched to display as a percentage in the table, or in a ToolTip.

Entry Points table

The Entry Points table displays the following information for the top 30 entry points in the monitored instance:

- Icon - displays the entry point type.

- Name - displays the entry point name.

Note: Selecting the hyperlinked entry point name selects the entry point in the execution tree and displays the Highlights tab for the entry point.

- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.

- Executions - displays the number of executions within the selected node.

- SLA Compliance - displays the relative percentage of HTTP entry point service requests that approached or breached the defined SLA thresholds.

- Type - displays the method type name.

- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.

- Avg. Work Time (Sec) - displays the average actual work time of the selected node without the underlying call path.

- Sum. Work Times - displays the total work time for the selected node without the underlying call path.

- Max. Response Time (Sec) - displays the peak response time detected in the selected time frame.

- Min. Response Time (Sec) - displays the lowest response time detected in the selected time frame.

- Avg. CPU Time (Sec) - displays the average time that CPU was consumed for the selected node and its underlying call path.

- Sum. CPU Time (Sec) - displays the total CPU time.

- Max. CPU Time (Sec) - displays the peak CPU time detected for the selected node.

- CPU Work Impact % - displays the percent value of the selected node's CPU time out of its entry point's CPU time.

- Avg. CPU Work Time (Sec) - displays the average actual work time that CPU was consumed for the selected node without its underlying call path.

- Sum. CPU Work Times - displays the total CPU work time for the selected node without its underlying call path.

Avg. Response Time (Sec) vs. Executions overtime bar graph

The Avg. Response Time (Sec) vs. Executions overtime bar graph displays the relationship over time between the average response time of the entry point selected in the table above and the number of times the entry point was executed in the same time frame. Hovering over any point in the bar graph will display a ToolTip with the date and time of the time slice, the average response time, and the number of executions.

About the SQL & Exit Points tab

The SQL & Exit Points tab displays the following information for all SQLs and exit points in the call tree beneath the selected entity.

- Destinations table

- Heaviest Exit Points Invoked Directly and Indirectly table

- Avg. Response Time (Sec) vs. Executions overtime bar graph

The exit points tab can provide information for the following:

- SQL statements

- Exit points methods (each has its relevant type)

- Custom exit points - custom exit points connect Precise for J2EE to technologies within your environment that Precise cannot directly monitor. Examples are the Documentum Content Server, the Mainframe, etc. Connecting to these technologies, Precise for J2EE can provide you with indepth information regarding performance bottlenecks seen within your Precise application and pinpoint the root cause from within the external technology. By default, this feature is not configured in Precise for J2EE. For more information regarding enabling and configuring custom exit points, contact Customer Support.

Heaviest Exit Points Invoked Directly and Indirectly table

Note: When All Instances is the selected top level, this table will be called, Heaviest Exit Points by Work Time - ones that have at least 1% impact on the total.

The Heaviest Exit Points Invoked Directly and Indirectly table displays the following information for the selected entity:

- Icon - indicates the method type or designated entity it represents.

- Name - displays the exit point name. If the exit point is an SQL, the short SQL text is displayed. If the exit point is a method, the method's short name is displayed.

- Work Impact - displays the percentage of work time of the selected exit point in relation to the total work time of the invoking method.

- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.

- Executions - displays the number of executions within the selected node.

- Type - displays the method type name.

- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.

- Max. Response Time (Sec) - displays the peak response time detected in the selected time frame.

- Min. Response Time (Sec) - displays the lowest response time detected in the selected time frame.

Avg. Response Time (Sec) vs. Executions overtime bar graph

The Avg. Response Time (Sec) vs. Executions overtime bar graph displays the relationship over time between the average response time of the exit point selected in the table above and the number of times the exit point was executed in the same time frame. Hovering over any point in the bar graph will display a ToolTip with the date and time of the time slice, the average response time, and the number of executions.

Destinations table

The Destinations table displays all destinations for the selected entity:

- Name - displays the destination name that can be either the Web service request name, 'DB Name' on 'Server Name' for SQLs, and so on.

- Type - displays the exit point destination type such as, Web service, DB type, and so on.

- Work Impact - displays the percentage of work time of the selected exit point in relation to the total work time of the invoking method

- Executions - displays the number of executions within the selected node.

About the Methods tab

The All Methods tab contains both URI/iView and methods from the entire call tree beneath the current context - instance. All data is aggregated from the underlying call tree.

The Methods tab includes the following components:

- Methods Invoked Directly and Indirectly

- Avg. Response Time (Sec) vs. Executions for selected method overtime bar graph

Methods Invoked Directly and Indirectly

On top appears the comparison table called "All Methods Invoked Directly and Indirectly".

Note: This will be Heaviest methods by work time when all instances in the top level in the executions table, with 1% impact min.

- Icon - displays the entry point type.

- Name - displays the entry point name.

    Note: Selecting the hyperlinked entry point name selects the entry point in the execution tree and displays the Highlights tab for the entry point.

- Work Impact - displays the percentage of work time of the selected exit point in relation to the total work time of the invoking method

- Avg. Response Time (Sec) - displays the average response time for the selected node. The average response time is displayed in seconds.

- Executions - displays the number of executions within the selected node.

- Type - displays the method type name.

- Sum. Response Times - displays the total response time for the selected node. The summed response time is displayed in hours: minutes: seconds: milliseconds.Avg. Work Time (Sec) - displays the average actual work time of the selected node without its underlying call path.

- Avg. Work Time (Sec) - displays the average actual work time of the selected node without the underlying call path.

- Sum. Work Times - displays the total work time for the selected node without its underlying call path.

- Max. Response Time (Sec) - displays the peak response time detected in the selected time frame.

- Min. Response Time (Sec) - displays the lowest response time detected in the selected time frame.

- Avg. CPU Time (Sec) - displays the average time that CPU was consumed for the selected node and its underlying call path.

- Sum. CPU Times - displays the total CPU time.

- Max. CPU Time (Sec) - displays the peak CPU time detected for the selected node.

- CPU Work Impact % - displays the percent value of the selected node's CPU time out of its entry point's CPU time.

- Avg. CPU Work Time (Sec) - displays the average actual work time that CPU was consumed for the selected node without its underling call path.

- Sum. CPU Work Times - displays the total CPU work time for the selected node without its underlying call path.

Avg. Response Time (Sec) vs. Executions for selected method overtime bar graph

The Avg. Response Time (Sec) vs. Executions overtime bar graph displays the relationship over time between the average response time of the exit point selected in the table above and the number of times the exit point was executed in the selected time frame. Hovering at any point in the bar graph will display a ToolTip with the date and time of the time slice, the average response time, and the number of executions.

About the Locks tab

The Locks tab displays locked methods from the entire call tree beneath the selected node. All data is aggregated from the underlying call tree.

The Locks tab also displays the top level URI or iView, or methods (such as: URI, EJB, or others) that are monitored for this instance.

Note: The locking methods themselves are not displayed on the tree.

The "lock" identifier is calculated at runtime. It is based on the object name that is being locked (synchronized). It is possible to use the same object more than once during the same method. In that case, Precise for J2EE adds an index for each of the lock events, in order to enable the developer to distinguish between the two lock events.

For example:

```
Public void foo(){
   //run some code...
   Synchronized (myLockObject){// will be labeled as myLockObject+0
      //run some synchronized code...
   )
   //run some more code...
   // and then we need to have another synchronized block, locking the same
    object as before
   Synchronized (myLockObject){// will be labeled as myLockObject+1
      // run some more synchronized code...
   )
```

The Locks tab includes the following components:

- Locked Methods under the Selected Entity
- Methods Simultaneously Locked with the Method Selected Above
- Avg. Lock Time (Sec) of the Methods Selected in the Tables Above

Locked Methods under the Selected Entity

- Icon - indicates the method type.
- Name - displays the short name of the actual method that participated in a lock in the selected time frame. The long method name is shown in the ToolTip.

  Note: Clicking on a method name selects that method within the execution tree and refreshes the Activity workspace data for the selected method. The Activity workspace will open the Locks tab for the selected method.

- Lock Impact- displays the relative percentage of the selected entity's lock time, out of the instance's summed response time.
- Avg. Lock Time (Sec) - displays the average time the selected entity was locked.
- Lock Acquisitions - displays the number of locks by the selected entity in the selected time frame.
- Sum. Lock Times - displays the total time that the selected entity was locked in the selected time frame.

Methods Simultaneously Locked with the Method Selected Above

When a method is selected in the Locked Methods under the Selected Entity table, this table displays all actual methods (and not a method in their call path) that participated in a lock with the method selected above.

- Icon - indicates the method type.
- Name - displays the short name of the actual method that participated in a lock in the selected time frame. The long method name is shown in the ToolTip.

  Note: Clicking on a method name selects that method within the execution tree and refreshes the Activity workspace data for the selected method. The Activity workspace will open the Locks tab for the selected method.

- Lock Impact- displays the relative percentage of the selected entity lock time, out of the instance's summed response time.
- Avg. Lock Time (Sec) - displays the average time the selected entity was locked.
- Lock Acquisitions - displays the number of locks by the selected entity in the selected time frame.
- Sum. Lock Times - displays the total time that the selected node was locked in the selected time frame.

Avg. Lock Time (Sec) of the Methods Selected in the Tables Above

The Avg. Lock Time (Sec) of the Methods Selected in the Tables Above overtime bar graph shows an overtime view of the periods where the methods selected in the tables above acquired locks. This view enables you to understand the mutual locking relationship of the two methods - are they always locked at the same times (meaning, they lock each other only), is one locking (indicated by a short lock time) and other locked (indicated by a long lock time), and so on. Hovering at any point in the bar graph will display a ToolTip with the date and time of the time slice, the average lock time, and the number of locks.

About the Exceptions tab

The Exceptions tab displays data for exceptions thrown from the entire call tree beneath the current context - instance. All data is aggregated from the underlying call tree.

The Exceptions tab includes the following components:

- Exceptions Thrown Directly and Indirectly

- Locations for Selected Exception

- Stack Traces

Exceptions Thrown Directly and Indirectly

On top appears a comparison table which summarizes the number of exceptions.

Note: Exceptions monitoring is turned off by default. To turn on exceptions monitoring, go to **Settings>Monitor Settings**. See "Configuring Monitor Settings" on page 41.

All columns can be sorted. The default sort is by Count column.

- Thrown Class - displays the short name of the class. The long name, including packages, appears in the ToolTip.

- Count - the number of times the exceptions class was thrown.

- Last Detected On - displays the exact date and time (to the second) that the thrown exceptions classes were detected.

Locations for Selected Exception

- Location - displays the short location name of the exception. The long name appears in the ToolTip.

- Count - displays the number of times the exception was thrown from the specific location.

- Message - displays the message that was set in the thrown class.

- Exception Cause - displays the exception cause (when available).

- Monitored Caller - displays the direct calling method where the exception was thrown.

Stack Traces

On the bottom appears the title "Exception Stack Traces". This table displays information where each row is a specific occurrence of the exception selected on top. There is one row per stack trace on this table.

Catches exceptions during a time slice, based on the instrumentation configuration (switched off by default) and filtering criteria.

See *Precise Administration Guide*, Appendix A, regarding configuring exception seeker options.

About the Leaks tab

The Leaks tab displays object allocation data from the entire call tree beneath the current context - instance. All data is aggregated from the underlying call tree.

Note: The Leaks tab is only relevant for instance or All Instances levels.

The Leaks tab includes the following components:

- Leak Candidates Allocated Directly and Indirectly

- The table below describes the columns in the Leak Candidates Allocated Directly and Indirectly table.

- Stack Trace Details

Leak Candidates Allocated Directly and Indirectly

- Allocation Method - the method name where the object was allocated.

- Object Type - the object, collection, array, or string buffer. It contains values such as: "Integer", "String", or "MyObject"

- Growth Rate - the rate that the number of objects grows over time. This is displayed in percent for growth rate per minute.

Live Objects Count

On top appears an objects comparison table.

All columns can be sorted. The default sort is by Growth Rate (%) column.

Note: Leaks monitoring is turned off by default. To turn on Leaks monitoring, go to the Settings menu. See "Configuring Monitor Settings" on page 41.

The Leaks tab displays details about the top 64 allocation sites with the largest total live allocated objects in the selected time frame. Each allocating method displays the collection's stack trace in its ToolTip. The stack trace displays the executions leading backwards from the allocation site to the application's initial execution.

For the top n allocating methods, the chart shows the approximate number of live objects in the collection, or its growth rate (according to the sorting column). Negative trends can be identified using these charts by detecting memory leaking locations.

Note: Aggregation in the table is according to: Allocating Method name, the Object Type, and the full path.

The table below describes the columns in the Leak Candidates Allocated Directly and Indirectly table.

Table 3-5 Columns in the JVM Leak Candidates Allocated Directly and Indirectly table

Item	Description
Chart Legend Color	Color icon showing the color representing the row in the graph.
Allocated Method	The method name where the object was allocated.
Object Type	The Object, Collection, Array, or String Buffer. It contains values such as: "Integer", "String", or "MyObject".
Growth Rate (%)	Rate of growth (in percents) of the number of objects over time (objects/minutes). This column is default for sorting.
Live Objects Count	Number of live instances of an object. Live Objects Count can be displayed as a "waterfall" bar, or in numbers.

Stack Trace Details

Stack Trace details appear in the expanded view that appears when hovering over each row's complete path (Stack Trace).

How the identify performance problems in the Tree View

You can identify a performance problem in Tree View by doing one or more of the following:

- Identifying slow Service Requests

- Identifying slow Methods

- Identifying the most invoked Service Request or Method

- Searching for a specific Service Request or Method

Note: The performance attribute by which the Tree View is sorted is shown on the workspace heading as well as on the ToolTip displayed when you place the cursor over a tree node. The number in brackets is the value of the entity according to its sort attribute.

Identifying slow Service Requests

To identify slow Service Requests

1   In the Time Frame list, select the period of time you want to analyze.

2   In the Instance list, select the J2EE instance you want to analyze.

3   Sort the tree by the performance attribute you want to explore (for example, Response Time (Avg) or Response Time (Sum). The default is Response Time (Avg), but you can change it using the Tree View Settings.

4   In the Tree pane, select the Service Request you want to analyze. Focus on the Service Request that has a performance counter (for example, Response Time) that is higher than it should be.

5   In the Details pane, look at both the Response Time graph and the Executions graph. Select a problematic time period to investigate and hover over it with the mouse. Additionally, analyze the details that are displayed in the Overview.

6   On the displayed ToolTip, look for the time period.

7   To view the problem in more detail, in the Time Frame list, select a narrower time frame which includes the time period that is displayed on the ToolTip.

8   To investigate the root cause of the slow performance, expand the Service Request tree node to see its invoked methods. Continue with the Identifying slow Methods procedure.

Identifying slow Methods

To identify slow Methods

1   In the Time Frame list, select the period of time you want to analyze.

2   In the Instance list, select the instance you want to analyze.

3   Sort the tree according to the performance parameter you want to investigate.

4   Select the Method you want to analyze.

5   Continue expanding additional tree levels until you can identify the slow Methods.

Identifying the most invoked Service Request or Method

To identify the most invoked Service Request or Method

1   In the Time Frame list, select the period of time you want to analyze.

2   In the Instance list, select the instance you want to analyze.

3   On the Precise bar, go to **Settings>Tree View Settings**.

4   In the Tree View Settings dialog box, in the Sort by drop-down list, select Executions. Click A to display in Ascending order or D to display in Descending order.

5   Select the Method you want to analyze. Methods with the highest number of executions are most frequently invoked.

Searching for a specific Service Request or Method

To search for a specific Service Request or Method

1 In the Time Frame list, select the period of time you want to analyze.

2 In the Instance list, select the instance you want to analyze.

3 In the Tree pane, right-click the Service Request, Method, or instance node to search under.

4 From the popup menu, select the **Search** option.

5 In the Search dialog box, specify the search parameters by inserting the string to search for and the number of results to display.

6 Click **OK**. The results are highlighted in bold font and you can decide to focus on one of them by selecting it.

4

Examining statistical information

This section includes the following topics:

- About the Statistics workspace
- How the Statistics workspace is structured

About the Statistics workspace

The Precise for J2EE Collector collects performance metrics provided by WebSphere, WebLogic, and other application servers. These application server metrics are displayed in the Statistics workspace.

Note: The information collected can be modified in the Monitor Settings dialog box. For more information, see "Configuring Statistics Settings" on page 45.

The application servers collect the metrics and provide an interface for the Precise for J2EE Collector to obtain them. Depending on the application server, the Precise for J2EE Collector may log into the administrative server of the monitored JVM to gather the metrics or obtain the metrics in other ways. These metrics provide an important source of performance data for Database Connection Pools, EJB Pools and Caches, Sessions, and Threads. Precise for J2EE identifies a set of similar metrics using a Group Name. For example, the Database Connection Pool group includes many metrics describing a Database Connection Pool such as maximum capacity, active connections, and so on.

The Statistics workspace enables you to provide answers to such questions as, "Does the JVM spend too much time in the garbage collection process?"

Note: Precise for J2EE collects general metrics such as garbage collection, memory and CPU for all monitored JVMs. Additional set of metrics - JMX metrics are collected for some Application Servers like WebLogic, WebSphere, Tomcat etc. See support matrix for more information. Enabling JMX metrics collection might require execution of additional action items.

How the Statistics workspace is structured

The Statistics workspace displays information on a selected entity and its associated entities. For example, it is possible to associate with all counters that are related to a specific instance, by selecting the Counters entity from the Association controls.

About the Main area in the Statistics workspace

The Main area includes two view options - Memory and GC and CPU Consumption - each containing overtime graphs displaying different aspects of the selected entity's performance. Select the desired view from the drop-down list underneath the workspace heading.

About the Memory and GC view

The Memory and GC view includes the following overtime graphs:

- Avg. Heap Usage - this bar graph displays the breakdown between free and used heap in the selected time frame, and any JVM starts and restarts that occur. Hovering over a breakdown bar will display a ToolTip with the following information for that time slice: used heap, free heap, total allocation heap, max allowed heap, max used heap, and min used heap.

- GC Time % - this bar graph displays the relative time spent in garbage collection out of the total response time in the selected time frame, and any JVM starts and restarts that occur.

- Maximum Used Heap - this line graph displays and compares the maximum allowed and maximum used heap size in the selected time frame.

- Minimum Used Heap - this line graph displays the minimum used heap size in the selected time frame. Hovering at any point in the graph displays the time slice date and time and the numerical value of the minimum used heap size.

About the CPU Consumption view

The CPU Consumption view includes the following overtime graph:

- CPU Consumption - this bar graph displays the CPU consumption percentage rate in the selected time frame, and any JVM starts and restarts that occur.

About the Association area in the Statistics workspace

The Association area provides corresponding information for the entities associated with the selected entity (displayed in the Main area). You can view information for one type of entity at a time, such as instances only, by selecting an item from the Association controls. The selection you make is reflected in the Association area only; the Main area remains unchanged.

From the Association area, you can also drill down to another entity by clicking a row in the table. A drilldown affects the whole workspace. When you drill down to another entity, the Workspace heading displays the new selection; the Main area displays an overtime line graph about the newly selected entity, and the Association area displays the entities associated with the selected entity.

For example, it is possible to associate to all instances that are related to a specific server or all counters that have been collected for a specific server, by selecting the appropriate row from the Association controls.

If you want detailed information on a specific server counter, in the Association area, click the row of the counter that you want to view detailed information for. The Workspace heading indicates the newly selected entity, and the Main area displays an overtime graph for the counter you drilled down to. There is no Association area data for a counter.

About configuring Precise for J2EE settings

This section includes the following topics:

Configuring Precise for J2EE settings

The Precise for J2EE user can define how Precise for J2EE collects data and how the collected data is displayed.

The Settings menu on the Precise bar allows you to configure the following:

■   Monitor Settings

■   Tree View Setting (Activity workspace only)

■   Display Settings

■   Time Frame Settings

■   Dynamic Types Allocation

Configuring Monitor Settings

The Monitor Settings dialog box enables you to configure and modify the data aggregation and collection definitions for each Precise for J2EE Collector, monitored Java Virtual Machine (JVM) and cluster.

To view and modify the Precise for J2EE monitor settings

1   On the Precise bar, go to **Settings>Monitor Settings**. The Monitor Settings dialog box appears.

2   From the drop-down list in "Step 1: Select JVM / Cluster", select the cluster (clusters appear in bold) or JVM (instance and server name). Clicking **Affected Instances** opens the Affected Instances dialog box, displaying the instances connected to the selected instance/cluster that will be affected by any instrumentation change.

- To view and modify configuration in the Production monitoring mode, see "About Monitoring Configuration" on page 42.

3 To enable statistics collection, mark the "Collect Statistics" check box.

4 To modify the statistics collections settings, click **Statistics Configuration**. The Statistics Configuration dialog appears. See "Configuring Statistics Settings" on page 45.

5 To copy the current instrumentation to additional JVMs or clusters, click **Populate Configuration**. The Populate Configuration dialog appears. See "Enabling Populate Configuration" on page 46.

6 To enable one or more of the following monitoring options, mark the relevant check boxes:

- Monitor Leaks
- Monitor Exceptions
- Enable Application Awareness for the DB Tier

7 Click **OK**.

About Monitoring Configuration

You can use the Monitoring Configuration dialog box to view and modify the status of all classes and methods in the selected JVM or cluster. To open this dialog box, go to **Settings>Monitor Settings>Monitoring Configuration**.

About the JVM Loaded Classes table

The JVM Loaded Classes table in the Monitoring Configuration dialog box displays all classes and methods detected in the selected JVM in alphabetical order. To ensure that you are viewing the most updated list, check the "Last Updated" date and time on the top right of the window, and if needed, click the refresh icon. If the "JVM is stopped - last saved snapshot" text appears, the JVM must be restarted in order to refresh the list.

The following table describes the information displayed in the "JVM Loaded Classes" table:

Table 5-1 Monitoring Configuration icons

Column	Description
Class	Displays the loaded classes in the JVM.
Method	Displays the methods included in the class.
Status	Displays the monitored status of the selected entity. For more information, see "About Status values" on page 43.
Type	Displays the type name of the selected entity. For more information, see "About Type values" on page 43.
Remarks	Displays relevant remarks regarding the selected entity. JDBC methods will not appear in the loaded methods list. Methods invoking JDBC calls will appear with an indicator in the Notes column as "Calls JDBC methods".

Note: When there are no pending changes, the list in the table displays "actual" monitoring status of classes and methods in the JVM. If pending changes exist, the list in the table displays "estimated" status of classes and methods in the JVM.

About defining the information displayed in the JVM Loaded Classes table

In the Monitoring Configuration dialog box, you can define and filter the information displayed by selecting one of the pre-defined view options and by using the search fields.

The pre-defined view options are:

- All Classes and Methods - shows all loaded default classes and associated methods (no filtering by status)
- Configured (Monitored & Blocked) - shows all statuses except for <blank> that are retrieved (for example: classes and methods including monitored, monitored (partial), pending, etc)
- Not Configured - shows only <blank> and pending statuses that can be retrieved
- Pending - shows only pending statuses

In addition to selecting a view option, the user can apply search criteria by either Package/Class pattern and/or Method pattern.

Click the **Search** (spyglass) icon to search and select an existing pattern name, or begin typing a pattern name into the appropriate field. Precise for J2EE has a unique auto-complete feature which displays a list of all pattern names with letters that match the typed letters in the field. The table displays results based on the selected viewing option and the search criteria.

Click the **Clear** (X) icon to clear any entered search criteria, or simply delete the pattern name field content.

About Status values

Status can have any of the values shown below in the table.

Table 5-2   Available Status values

Name	Description
Monitored	For a class, it means that all of its loaded methods are instrumented.
	For a method, it means that the method is instrumented.
Monitored (partial)	For a class, it means that only some of its loaded methods are instrumented.
	For a method, it means that only some of the signatures are instrumented.
Excluded	For a class, it means that all of its loaded methods are ignored.
	For a method, it means that it is ignored.
Excluded (partial)	For a class, it means that all of its loaded methods are either ignored (exists at least once) or not instrumented (exists at least once).
<Blank>	For a class, it means that all of its loaded methods are not instrumented (but not ignored).
	For a method, it means that it is not instrumented (not ignored).
Pending	Relevant for classes or methods after a change was performed and the JVM was not restarted. The Class will have a pending status if it has at least one pending method.

About Type values

The type is used to categorize classes and methods as meaningful parts of the monitored application. Predefined types can contain HTTP, EJB and others. User defined types can be added when defining classes/methods to be monitored. (If no type is defined, the Custom default type is used.) The types are the foundation for displaying response time breakdown into its components in the Precise for J2EE User Interface (for example, how much time is spent in EJBs or in JDBC methods).

The type of methods is either defined by the user, or by the product's predefined definitions.

The type of classes can be defined by the user or by the product's predefined definitions, or (when no specific definition exists) as a combination of all of its methods' types.

About modifying the Monitoring Configuration

The Monitoring Configuration dialog box enables the user to easily view and edit the monitoring status of the loaded classes or methods. It covers a broad spectrum of use cases, serving both novice and experienced users.

Note: A Pending Changes message will appear at the top of the screen after the user has made any changes to instrumentation definitions, and restart has not yet been performed.

To monitor one or more loaded classes or methods

1   In the table, click to select one or more loaded classes or methods for monitoring.

2   Click **Monitor**.

3   Click **Save** to save your changes, or **Cancel** to close the screen without saving changes.

   Note: Restart JVM after saving your changes.

The user can stop monitoring a selected row in the table according to class or method status.

To stop monitoring a loaded class (classes) or methods

1   In the table, click to select one or more loaded classes or methods to stop monitoring.

2   Click **Stop Monitor**.

3   Click **Save** to save your changes, or **Cancel** to close the screen without saving changes.

   Note: Restart JVM after saving your changes.

The user can exclude a selected row in the table according to class or method status.

To exclude (ignore) a loaded class (classes) or methods

1   In the table, click to select one or more loaded classes or methods to exclude.

2   Click **Exclude**.

3   Click **Save** to save your changes, or **Cancel** to close the screen without saving changes.

   Note: Restart JVM after saving your changes.

4   Click **Manage Patterns** to edit or view the matching classes or methods.

5   After selecting a search pattern, click **Monitor Search Results** to monitor the search criteria pattern. After clicking the monitor search result, assign a type to the pattern in the opened pop-up message.
     Use **Monitor Search Results** to create and monitor patterns.

About the Patterns Manager

Use the Patterns Manager dialog to manage monitoring patterns to view or edit types and matching classes.

Note: User-defined classes can be edited (to add, edit, or delete) types, classes, and methods.

At the top of the screen, the selected JVM name and its "Last JVM start time" (date and time) are displayed.

The Monitoring Patterns table displays configured patterns (classes and methods, and their assigned types).

Click **Add Pattern** (above the table) to create a new pattern name to the table.

Click on the **+** icon in the table to open and view all members of the selected class.

To add a class or method pattern

1   For a new class pattern, click **Add Pattern**.

2   Begin typing the class name. Pattern Manager will automatically display currently detected classes starting with the same letters. Select the appropriate class name from the list.

3   For a selected class pattern, click the blank field in the Method Pattern column for the selected class row.

4   Begin typing the method name. Pattern Manager will automatically display currently detected methods starting with the same letters. Select the appropriate method name from the list.

About Patterns Manager loaded classes and methods results

The table (lower pane) displays the results (actual instrumentation status) for the selected classes or methods that match the selected definition row (upper pane).

The JVM "Loaded Classes and Methods" displays (in parenthesis) the total number of classes that were loaded. Also displayed is the date and time when the classes were "Last updated".

Click the Refresh icon to refresh the screen and update all viewed data.

Click on the **+** icon in the table to open and view all members of a selected class.

The table displays the following results:

■   Class (the class name)

■   Method (the method name)

■   Status (See "About Status values" on page 43).

■   Type (the assigned type, "custom" by default)

■   Notes (additional information for a specific class)

Configuring Statistics Settings

The Statistics Settings dialog enables you to modify the Application Server Metrics data collection options for the selected JVM or cluster. You can view data for all the selected metrics in the Association area of the Memory and Statistics workspace.

Warning: Application server overhead will increase when the application server collects metrics and when the Precise for J2EE collector gathers them. Refer to the application server documentation for further details.

To configure Statistics Settings

1   On the Precise bar, go to **Settings>Monitor Settings>Statistics Configuration** (the link will only be active if the "Collect Statistics" check box is marked). The Statistics Configuration dialog box will appear, displaying a partial list of all metrics detected in the selected JVM or cluster. The two lists, Available Metrics and Selected Metrics, respectively show which metrics have not yet been selected to appear in the Memory and Statistics workspace and which metrics have already been selected.

2   Use the left and right arrows to move metrics between the two lists. By default, the Available Metrics list only displays a partial group of all detected metrics. To display all detected metrics, click **Update Metrics**. A message dialog box will appear. If you have made changes to the metrics, click **Yes** to save those changes before the metric retrieval process begins. Clicking **No** will begin the retrieval process without saving any changes made.

3   To search for a specific group or metrics, enter a JVM property in the field below the Available Metrics heading and click **Filter**.

4   Mark the "passwordRequired" check box.

5   Click **OK**.

About information displayed in the metrics lists

The table below describes the information displayed in the metrics lists.

Table 5-3 Metrics list fields

Option	Description
Group	Category of statistics in which the metric will be displayed in the Memory and Statistics workspace.
Metric	Name of metric as it will be displayed in the Memory and Statistics workspace.

Enabling Populate Configuration

The Populate Configuration dialog box enables you to choose clusters or JVMs and copy the configuration from the selected JVM or cluster in the Monitor Settings dialog box to these clusters or JVMs.

To copy configuration to one or more JVMs or clusters

1 On the Precise bar, go to **Settings>Monitor Settings>Populate Configuration**. The Populate Configuration dialog box will appear, displaying the JVMs and clusters in the Available JVMs/Clusters and Selected JVMs/Clusters lists. The dialog heading will feature the name and server name of the source JVM or cluster.

2 Using the left and right arrows, move JVMs and clusters between the two lists until all JVMs and clusters that you wish to take on the instrumentation appear in the Selected JVMs/Clusters list.

Note: To search for a specific JVM or cluster, enter the JVM or cluster name (or part of it) in the field above the Available JVMs/Clusters list and click **Filter**.

3 Click **OK**.

Enabling monitoring for leaks and/or exceptions

Mark the appropriate option(s) to monitor for leaks and exceptions.

Configuring Dynamic Types Allocation

The Dynamic Types Allocation dialog box enables you to customise the appearance of user-defined dynamic types in Precise for J2EE.

In this dialog, you can designate a color to a dynamic type which will represent the specified type in the various breakdown bars throughout Precise for J2EE. You can also define that information regarding specific types will appear in the information ToolTip in these graphs.

Up to 5 dynamic types can be assigned a color or ToolTip, and up to 5 dynamic types can be assigned only a ToolTip.

Note: To define dynamic types, see "About the Patterns Manager" on page 44.

To configure the dynamic types allocation

1 In any Precise for J2EE workspace, go to **Settings>Dynamic Types Allocation**.The Dynamic Types Allocation dialog will appear.

2 From the No Color and No ToolTip list on the left, select one or more types (up to 5).

3 Click the upper right arrow to move your selections to the Color and ToolTip list. Types in this list will appear as the specified color and information about them will appear in the information ToolTip.

4 Click the lower right arrow to move your selections to the No Color and ToolTip list. Types in this list will appear as part of the general "custom" type in the breakdown bar but will be mentioned by name in the information ToolTips.

5   To remove a selected type from either of the right lists, select the item(s) and click the appropriate left arrow.

6   Click **OK**.

Configuring Tree View Settings

The Tree View settings dialog box enables you to modify how information is displayed in the Tree view.

Note: When you click OK in this dialog, the current workspace is automatically refreshed.

To configure the tree view settings

1   In any Precise for J2EE workspace, go to **Settings>Tree View Settings**. The Tree View Settings dialog will appear.

2   In the "Sort by" drop-down list, select a field to sort the displayed data by. The default value is Avg. Response Time (Sec).

3   Click A to sort in ascending order, or D to sort in descending order.

4   From the drop-down list, select the maximum number of search results to be returned by a search within the execution tree. The default value is five.

       Note: Changing this setting will automatically update the Search dialog box.

5   Mark the "Show short name only" check box to display the method name in the execution tree by a short name.

       Note: The full method name will always be displayed in the information ToolTip that appears when hovering over any entity in the execution tree.

6   Click **OK**.

Configuring Display Settings

The Display Settings dialog box enables you to define Precise for J2EE to retain the selected view or tab, if applicable, when you drill down on an entity or select a different entity. This is sometimes referred to as sticky tabs. For example, if you have chosen the Locks view in the Activity workspace, this will become the default view for all entities, where applicable

To define Precise for J2EE to retain the selected view or

1   In any Precise for J2EE workspace, go to **Settings>Display Settings**. The Display Settings dialog will appear.

2   Mark the "Maintain the selected tab or view when switching entities" check box.

3   From the drop-down list, select the maximum number of executions to be displayed under each node. The default value is 30.

4   Click **OK**.

Configuring Time Frame Settings

The Time Frame Settings dialog box enables you to modify the default settings for both the time frame for displayed data and the auto-refresh mechanism.

To modify the default settings for the time frame and auto-refresh mechanism

1   In any Precise for J2EE workspace, go to **Settings>Time Frame Settings**. The Time Frame Settings dialog will appear.

2   Modify the "By default, show performance information for the last" value by selecting the desired unit of time from the drop-down list and entering in the desired value for the selected unit. This will be the default time frame for data displayed in Precise for J2EE.

3   Modify the default frequency of the auto-refresh mechanism by entering the desired value (in minutes) in the "Auto-Refresh data every" field. When auto-refresh is enabled, data will be refreshed according to the updated value in this field.

4   Click **OK**.

Examining findings

This section includes the following topics:

- How to identify performance problems
- About J2EE findings

How to identify performance problems

In most cases, a Precise for J2EE workspace displays information in the context of a specific instance and time frame. However, if you want to view findings for another time frame or instance, you can change these settings using the respective dropdown lists.

See "About the Findings area" on page 11.

How to investigate a finding

When investigating finding, it is recommended to investigate findings based on their severity.

To investigate a finding

1 In the All J2EE Instances table, select the instance you want to investigate. In the Finding Details area, the top findings for a selected instance are displayed in the Findings table of a workspace.

2 Identify the findings with the highest severity rankings (red, orange, and yellow icons where red is the highest severity and yellow the lowest) in the Findings table.

3 In the Findings table, select a finding to analyze further the problem.

4 In the selected finding (the expanded view), read the data displayed for the finding and follow any links provided to view additional information (advice) or next steps (bullets) to perform the recommendation(s) that best suit(s) your needs.

About J2EE findings

Several workspace findings exist in the Findings table to help the user.

The following is a list of current Precise for J2EE findings for an instance/method:

- Heavy Entry Point
- Frequent SLA Breaches
- Heavy Method Contribution
- Excessive CPU Usage
- Excessive Garbage Collection Time
- Memory Usage Near Maximum
- High Exceptions Rate
- Excessive Lock Time

- Slow DB Request Execution

- Slow Web service Execution

- Heavy Exit Point

- Significant JDBC Activity

- Significant External Activity

- Tuning Opportunities Detected

- Locks Detected

- Exceptions Detected

- Unbalanced Activity

- Impact on Multiple Entry Point

Heavy Entry Point

Heavy entry points may indicate a potential performance irregularity in the entry point's underlying call path. When viewing information for "All Instances" in the execution tree, a high entry point finding across multiple instances will clarify if the irregularity is a result of the entry point or if it is a result of a specific instance.

Note: The finding is based on the total response time values, whereas by default, the execution tree is displayed according to the average response time. As a result, the specified heavy entry point is not necessarily found towards the top of the execution tree.

Working with the finding

- Select the link in the expanded finding area. This will refresh the execution tree and analysis tabs to focus on the entry point, which will now be the selected node in the execution tree. In the new Highlights tab, examine the entry point's performance overtime activity and follow the specific findings to easily navigate to and analyze the root cause of the performance issue.

Frequent SLA Breaches

SLA thresholds are defined to help the user pinpoint HTTP entry points experiencing performance issues according to specific criteria. Frequent SLA breaches and near breaches can be caused by an underlying performance issue.

Working with the finding

To effectively locate the root cause of the performance finding, perform one of the following:

- Click on the entry point's link, and then look at the overtime SLA behavior to locate and zoom in to a specific (problematic) time frame. View the findings for that time frame and drill down until you locate the root cause.

- Select the root level of the execution tree and select the Entry Points tab. A high rate of SLA breaches across the application could result from overall resource exhaustion. Open the Memory and Statistics workspace to learn more about the environment performance issues, like high memory usage, CPU usage and so on.

- Go to **AdminPoint>Settings>SLA** to view the threshold definitions. (When you have too many SLA breaches, it may be a result of thresholds that are not defined appropriately for your application).

Heavy Method Contribution

A high work time for a specific method (reflecting the method's work time only, without the underlying call path), can indicate a performance issue within the context of that method.

In the same way, a high work time for a specific occurrence of a method invoked multiple times in the execution tree can indicate a performance issue within the context of that specific occurrence.

Working with the finding

■ Examine the heaviest occurrences further by following the featured link.
The execution tree opens to the method's heaviest call path, facilitating effective navigation to the root cause. Examine the information displayed and look at the overtime graph and findings to drill down further to find the root cause of the performance issue.

Note: By default, information is displayed for the heaviest method's call path. To investigate the other call paths, select them from the execution tree. (They are highlighted in bold).

Excessive CPU Usage

High CPU usage may indicate that the JVM is experiencing a lack of resources. This influences the performance of applications running on the JVM.

High CPU consumption can be directly related to your JVM's activity. However, it can also result from a general lack of free CPU resources on that server.

Before working with the finding, verify that the high CPU consumption is not caused by the above independent scenario.

Working with the finding

■ Examine the CPU consumption rate by following the featured link.
The Memory and Statistics workspace appears. If you notice a consistently high CPU rate or an increase in the CPU usage overtime, open the Activity tab, observe the performance trends and findings that can explain the high CPU usage, and drill down to locate the root cause of the performance issue. If you notice a peak in the CPU consumption, zoom in to the time frame in which the peak occurred before opening the Activity tab.

Note: To obtain a clearer view of your application's overall CPU consumption trends, it is recommended to increase the time frame of the displayed information.

Excessive Garbage Collection Time

The application spends too much time performing Garbage Collection activities on the specified JVM.

This can result from:

■ Incorrect object allocation and management like performing too many object allocations, performing too large allocations, keeping references to an object after it's no longer needed, and so on. Specifically, objects that have many references from the code and are freed only after numerous Garbage Collection cycles take more time to clean, and therefore impede the application's performance.

■ The application code directly invokes Garbage Collection - it is highly recommended not to interfere with the independent Garbage Collection operation. Therefore, manual executions by application code should be eliminated.

Working with the finding

- Examine the Garbage Collection time percentage trends by following the featured link.
  The Memory and Statistics workspace appears.

  Note: To obtain a clearer view of your application's overall Garbage Collection times trends, it is recommended to increase the time frame of the displayed information.

Memory Usage Near Maximum

Each JVM has a defined amount of maximum allowed heap size. Reaching the maximum allowed heap size can cause the JVM and the application to crash. This finding is triggered when the maximum used heap is getting dangerously close to that limit.

Working with the finding

- Examine the memory usage trends by following the featured link.
  The Memory and Statistics workspace appears. Specifically focus on the overtime graph of the maximum used heap compared to the maximum allowed heap size.

- If the maximum and/or minimum heap usage graphs indicate rising trends, consider switching on the Leak Seeker through the Settings screen.

  Note: This is recommended in Non-production mode only.

Note: To obtain a clearer view of your application's overall memory usage trends, it is recommended to increase the time frame of the displayed information.

High Exceptions Rate

Exceptions handling has a high price in terms of processing time. It is also takes time away from a performing application's business logic. Therefore, a high exceptions rate can significantly influence application performance.

A high exceptions rate can also indicate that the application code is using exceptions as part of its natural flow, and not only as a means to handle errors and unexpected situations.

Working with the finding

- Examine the exceptions information further by following the featured link.
  The Exceptions tab opens, featuring all exceptions types and their total occurrences in the code. Examine the exceptions and their stack traces to understand where you need to focus the investigation.

- Follow the Monitored Caller links of relevant exceptions to locate the closest monitored methods inside the execution tree. You may continue further, or forward this information for next-level handling.

Excessive Lock Time

A significant percentage of the selected entity's total response time is spent waiting for lock acquisitions.

Waiting for locks means that the online activity (or batch process) is on hold, and is waiting for another activity or process to release the lock. Typically, eliminating locks can improve the performance of your transactions.

A possible solution is to consider tuning your transaction performance to reduce the time spent waiting for locks.

Working with the finding

■   Examine the lock acquisitions further by following the featured link.
    The Locks tab opens, featuring all methods participating in a lock. Examine which methods are competing against each other to acquire locks. Consider whether the lock is indeed protecting a shared resource and whether access to the resource can be made more efficient, reducing the transaction's running time.

Slow DB Request Execution

A significant percentage of the selected entity's total response time is spent waiting for a specific DB request execution.

A possible solution is to tune your DB requests to optimize transaction performance.

Working with the finding

■   Click on the queries to find their occurrences in the call tree, find the target DBs, and inspect their behavior over time.

■   Examine the DB request further by following the featured link.
    The SQL and Exit Points tab opens, displaying details for SQL statements that the selected entity or its underlying call tree is waiting for, according to their contribution to the overall performance. From this view of the overall external activity:

    ■   Follow one of the slowest DB requests to drill-down within the call tree and check the load balancing information further by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.

    ■   If the database instance is monitored by Precise, follow the "Analyze" link in the Highlights tab to drill down to the respective expert view (for example, Precise for Oracle), and see how it could be more efficiently tuned.

■   Consider parallelizing the external activity to run while processing is being done on the JVM side.

Slow Web service Execution

A significant percentage of the selected entity's total response time is spent waiting for a specific external Web service execution.

A possible solution is to tune the external Web service to optimize transaction performance.

Working with the finding

■   Follow the featured link and examine which call paths invoke the external web service and which call paths are the heaviest. From this view of the overall external activity:

    ■   Follow one of the web service requests to drill-down within the call tree and check the load balancing information further by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.

    ■   If the Web service is running on a J2EE or .NET instance which is monitored by Precise, follow the "Analyze" link in the Highlights tab to drill down to the respective expert view (for example, Precise for Oracle), and see how it could be more efficiently tuned.

■   Consider parallelizing the external activity to run while processing is being done on the JVM side.

■   Check the load balancing information of the Web service by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.

Heavy Exit Point

A significant percentage of the selected entity's total response time is spent waiting for external activity (for example, SAP RFCs, Tuxedo service requests, RMI calls, and so on).

A possible solution is to consider tuning your transaction performance and the time spent executing external activity.

Working with the finding

- Examine the external activity further by following the featured link.
  The SQL and Exit Points tab opens, displaying details for exit points that the selected entity or its underlying call tree is waiting for, according to their contribution to the overall performance. DB activity is featured in this list as SQL statements.

  - Click on a link to examine the call paths that invoke the external activity, and focus on the heaviest.

  - If the exit point is monitored by Precise, follow the "Analyze" link in the Highlights tab to drill down to the respective expert view (for example, Precise for Oracle), and see how it could be more efficiently tuned.

- To understand the influence the exit point has on the entire application, open the Impact tab to view the impact of the exit point on all entry points and call paths in the application.

- Check the load balancing information of the exit points by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.

- Consider parallelizing the external activity to run while processing is being done on the JVM side.

Significant JDBC Activity

A significant percentage of the selected entity's total response time is spent waiting for DB activity invoked by a JDBC request.

This can result from a specific heavy statement or a collection of many, relatively short statements, being executed.

A possible solution is to consider tuning your transaction performance to reduce the time spent executing DB statements.

Working with the finding

- Examine the external activity further by following the featured link.
  The SQL and Exit Points tab opens, displaying details for exit points that the selected entity or its underlying call tree is waiting for, according to their contribution to the overall performance. DB activity is featured in this list as SQL statements. From this view of the overall external activity:
  - Consider unifying queries to eliminate communication and query overheads.
  - Follow one of the heaviest SQL statements' links to drill-down within the call tree to locate the target DB and perform one of the following:
    - Check the overtime activity graph and summary area in the highlights tab.
    - Check the load balancing information further by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.
- Consider parallelizing the queries to run while processing is being done on the JVM side.
- To understand the influence the exit point has on the entire application, open the Impact tab to view the impact of the exit point on all entry points and call paths in the application.

Significant External Activity

A significant percentage of the selected entity's total response time is spent waiting for external activity (for example, SAP RFCs, Tuxedo service requests, RMI calls, and so on).

This can either indicate that a specific external activity is experiencing performance issues, or a large number of external calls, each of them relatively short, producing a high total response time.

Working with the finding

- Examine the external activity further by following the featured link.
  The SQL and Exit Points tab opens, displaying details for exit points that the selected entity or its underlying call tree is waiting for, according to their contribution to the overall performance.
  From this view of the overall external activity:
  - Consider unifying requests to eliminate communication overheads.
  - Consider using an internal cache mechanism for reducing external calls, in case the same calls are being invoked repeatedly.

- Follow one of the heaviest exit points' links to drill-down within the call tree, and perform one of the following:
  - Check the overtime activity graph and summary area in the Highlights tab.
  - Check the load balancing information further by opening the Load Balance tab. Examine whether it is relatively heavier when being called from specific JVMs.
  - To understand the influence the exit point has on the entire application, open the Impact tab to view the impact of the exit point on all entry points and call paths in the application.
  - If the exit point is monitored by Precise, follow the "Analyze" link in the Highlights tab to drill down to the respective expert view (for example, Precise for Oracle), and see how it could be more efficiently tuned.

- Consider parallelizing the external activity to run while processing is being done on the JVM side.

Tuning Opportunities Detected

The selected method showed a high work time. This may indicate that the selected method is the last branch of the call tree, or that visibility to the selected method's underlying call path can be enhanced by adding instrumentation.

Working with the finding

- If the information regarding the method and its performance metrics is sufficient, forward this information over to the Java expert or developer for next-level handling using either the email or print option.

- If more accurate pinpointing is needed, increase the level of visibility by adding instrumentation for all methods in the selected method's call tree by updating the instrumentation definitions. Following a JVM restart, you will see a detailed breakdown of the work time of the selected method and its call tree, enabling easy identification of the specific problematic method. For more information, see the *About instrumenting all calls from a method* section in the *Precise Administration Guide*.

Locks Detected

While executing the selected context and its underlying call tree, time was spent waiting for lock acquisitions.

While waiting for lock acquisition, the online activity (or batch process) is put on hold until another activity or process acquires the lock. Therefore, eliminating locks can improve the performance of your transactions.

Working with the finding

- Examine the lock details by following the featured link.
  The Locks tab opens, displaying all methods involved in a locking activity. Examine which methods are competing against each other for acquiring locks.

- To examine methods with considerable lock time, follow the lock's link to locate it within the execution tree. Consider whether the lock is indeed protecting a shared resource and whether access to the resource can be made more efficient, thus reducing the transactions running time.

Exceptions Detected

Exceptions were thrown in the selected context or its underlying call tree.

This may cause users (or batch activities) to experience performance or availability issues.

Working with the finding

- View the full details of the thrown exceptions by following the featured link.
  The Exceptions tab opens, displaying all thrown exceptions, their frequency, and the full stack trace. Use this information to determine whether this behavior is normal or needs further attention, or forward a link to the displayed tab to a Java expert or developer for next level handling.

Unbalanced Activity

The selected entity was invoked in multiple JVMs and encountered significantly different response times across the JVMs.

A transaction executed on a cluster of JVMs may encounter different service levels per execution, depending on the activity of the specific cluster's JVMs executing the transaction.

Working with the finding

- Examine the affected paths and transactions by following the featured link.
  The Load Balancing tab opens, displaying an overview of the selected entity's behavior across the different JVMs, and providing the ability to select the specific JVM and examine its relative performance against the application's average.

- Select a JVM row and check its overtime response time and number of executions.

  - In case of a high or growing number of executions over time, the difference in response time may be the result of load balancing issues. The affected JVMs may be getting more requests than their counterparts, and may not be able to cope with the increasing load. Check your load balancing component, as well as the scalability of the JVM's software.

  - When there are no load balancing issues, but the JVM still has a high response time, this may be a result of a resources shortage on one or more of the JVMs. Go to the Memory & Statistics workspace and examine the behavior of the affected JVMs, to determine whether a resource issue is affecting the performance of the application running on it. Typically, in case of resource shortage, more than one entry point will be affected.

  - When there are no load balancing issues and there is no resource shortage, select the slower JVMs from the Load Balancing tab, and drill down to examine the behavior of the entity on the specific JVM.

Impact on Multiple Entry Point

The selected method/SQL is invoked from multiple call paths and therefore affects the performance of the instance (JVM) in a cumulative manner.

Therefore, improving the performance of the selected method/SQL will impact more than its current call path, and may improve the performance of additional paths and transactions containing the method.

Working with the finding

- Examine the affected paths and transactions by following the featured link.

The Impact tab opens, displaying a list of entry points and call paths affected by the selected method. Tuning methods/SQLs executions with notably high impact rates will positively affect the overall performance.

■   Take note of high variations between method response times when called from different paths. Such variations may indicate a dependence of the method's performance on context. Therefore, drill down to the problematic context(s) for further tuning.

Glossary

abandonment rate	In Web, a counter that keeps track of the percentage of users that abandon the loading of the Web page before it completes downloading.
action	An operation that Alerts FocalPoint automatically performs when detecting a warning or critical status for a specific metric. According to the defined action, Alerts FocalPoint opens a message box, sends an email or SNMP trap, or executes a program.
action item (manual)	Some installation steps cannot be executed automatically by Precise Installer. The administrator is required to execute them manually. Action Items are presented (when necessary) at end of installation or product update.
AdminPoint	The central administration console of Precise that facilitates the maintenance, configuration, and management of all installed Precise components, such as monitoring the status of all Precise agents and PMDB processes, getting license information, starting and stopping the agents, getting log data on agents and events, changing PMDB settings, and installing patches. See also PMDB. See also agent.
advice	In Oracle, an algorithm that is designed to recommend on gathering statistics, creating new indexes, change existing indexes, and add or delete hints to make Oracle's Optimizer choose a better access path and make the statement perform better. Can be activated from any DML (Data Manipulation Language) statement.
agent	A program that runs on a server machine to collect, process, or communicate performance information. The Precise installation consists of multiple agents.
alert	The state of an Alerts metric that has reached a near-critical or critical status. An alert is issued by Alerts, triggering an action and informing of a problem that has occurred or is likely to occur within the area sampled by the specific metric. See also action.
Alerts	These product provide alerts to problematic conditions before they turn into performance problems, based on predefined metrics and thresholds. Alerts can automatically perform an action, such as displaying a pop-up message, sending an email message or SNMP trap, or running a program.
Alerts FocalPoint	An agent that receives data from the InformPoint agents, stores it, and performs any action that has been user-defined for that specific alert, such as displaying a pop-up message, sending an email message or SNMP trap, or running a program. See also InformPoint.
alert type	In Alerts, the status of all metrics belonging to a metric group or a monitored instance, indicating the current performance level through colors. See also alert. See also metric.
application server metric	In J2EE, a metrics that is provided by the application server or by customer code. This can include metrics published by the Java Management Extension (JMX) APIs or vendor-specific APIs, such as IBM's Performance Management Interface (PMI).
AppTier	The abbreviation for an application tier in a Precise environment. An AppTier contains one or more instances of the same technology and purpose. Application tiers do not necessarily correspond to distinct physical servers: in many cases, the tiers are logical, with one server running multiple AppTiers or one AppTier spans a cluster of servers. A Precise environment may contain multiple AppTiers on the same technology. For example, you may group J2EE instances (JVMs) into a J2EE Presentation AppTier and a J2EE Business Logic AppTier. Segmenting application response time into the contribution of individual application tiers is helpful in identifying the source of performance problems. Analyzing the performance and behavior of each tier separately is crucial for isolating the root causes of performance problems.
cabinet	In Oracle, SQL Server, DB2, and Sybase, the highest logical level in the SQL workspace hierarchy. A cabinet contains folders and, within folders, statements.

call path	A subset of a stack trace including only those methods that have been instrumented.
client cache	In Web, a counter that keeps track of the percentage of requests taken from the client cache (http status 304).
client-side collection	See Web client browser-side.
cluster	A group of servers or instances that are configured to be treated as a single entity. In Oracle, all Oracle instances of the same Oracle database (RAC configuration) are treated as a cluster and presented in the Precise for Oracle UI as a Database. In J2EE, a group of JVMs that serve the same application can be defined as Cluster (even if they are not defined as a cluster in the application server configuration). All JVMs of the same cluster share the same configuration.
collapsed access plan	In SQL Server and Sybase, the access plan of a unique group of statements or batches that belong to the same collapsed statement or batch but have different access plans. This can differ due to the constants in the text of the original (not collapsed) statements.
	See also explained statement.
	See also collapsed statement.
collapsed statement	In Oracle, SQL Server, Sybase, and DB2, a statement whose constants are replaced with parameters. Each collapsed statement can have several access plans, according to the occurrences of its statements.
	See also collapsed access plan.
Collector	The program that runs on the monitored server to collect performance information.
	Some technologies allow a single Collector (agent) to serve multiple instances running on the same server. Other technologies require a dedicated Collector per monitored instance.
	In most cases, it must be installed on the monitored server. Collectors for SAP, SQL Server, and Sybase may reside on a remote server.
	See also agent.
collector slice size	In J2EE, the smallest unit of data presentation and storage. It can be configured as 30 seconds or five minutes. If SmartLink is configured, relevant Web and Microsoft .NET Collectors will report to SmartLink using the same collector slice size. Regardless of the defined collector slice size, the most granular slice size presented on both the Precise for Web UI and the Precise for Microsoft .NET UI will be 15 minute intervals.
completion rate	In Web, a counter that keeps track of the percentage of users that completed the loading of a Web page.
count	The number of occurrences observed during a measurement interval.
CPU time	The average amount of time consumed by the operating system actively processing instructions on behalf of a running activity.
critical status	In Alerts, the status represented by a red bullet indicating that the value of the sampled metric has exceeded the near-critical and critical threshold values.
	See also metric.
Cross-AppTiers	In Insight, Report Manager, and Alerts, a perspective that provides high-level information about the performance of all AppTiers in an environment, including operating system data. It provides a holistic view of the entire environment and helps understand how the AppTiers interact.
	See also AppTier.
current data	In SmartLink, the end-to-end activity, detected by Precise over the last five minutes, including activities that are still in progress. In all databases products, a workspace that presents Current Sessions and the SQL statements that were executed including statements that are still in progress.
custom	Represents executions of a type not specifically defined.
	See also work time.
customized metric	In Alerts, a user-defined metric that measures site-specific parameters.
drilldown	Within Precise products and Insight SmartLink, the filtering of analyzed data by clicking a specific entity. Then, additional information about the selected entity is presented, plus a breakdown of its activity by another entity. For example, drilldown on a transaction and get an overtime graph of the selected transaction, plus a list of top users that executed the selected transaction.
entry point	Usually a top-level HTTP or EJB request. An entry point can originate when a user clicks in a browser or an E-commerce server invokes a remote EJB. J2EE makes a distinction between service request executions and other executions. The first HTTP or EJB execution within a call path is designated as an entry point.

environment	The highest Precise logical group. It may contain multiple AppTiers of various technologies that serve an application together. For example, a Payroll Production environment may contain all Web servers, application servers, transaction managers, databases, and servers that serve this application. Alternatively, it may contain any set of instances that form together an administrative group. Since Precise version 8.5, an instance can belong to more than one environment. Let's say a single Oracle database serves two different applications plus the DBA wants to associate this database with a group of other databases under his responsibility regardless of served applications. In this case the Oracle database will be associated to three different Precise environments.
	See also AppTier.
	See also instance.
ERP Extension	In Oracle, DB2, and SQL Server, an extension to the Collector that provides detailed information on the activities and resource consumption of packaged application components. It correlates the database information, and the packaged application information and lets you see users, transactions, reports, and other elements of ERP applications, such as Oracle Applications, SAP, PeopleSoft, and Siebel.
event	In Alerts, the occurrence of a sampling or progress. A sampling occurrence occurs every time a metric samples. A progress occurrence occurs when a metric's progress status is changed or when the investigated status reaches the end of the given investigation time. In AdminPoint, all occurrences reported by Precise agents, including informational events, warnings, and errors related to one of the agents. All events are shown in the Events view.
	See also metric.
	See also progress.
execution	An execution of a J2EE entity (a Servlet, EJB, SQL Statement, method, and so on). When mentioned in plural (Executions), means the amount of times that the entity was executed.
execution context	The context within which a method is invoked. For example, if Method A is invoked both from Method B and from Method C, there will be two different execution contexts for the performance metrics collected for Method A, one for when it is invoked from Method B and one for when it is invoked from Method C.
	See also call path and instrumentation context.
executions	In Oracle, SQL Server, DB2, and Sybase, the number of times a SQL statement was executed during the selected time frame. In SmartLink and SAP, the number of times a transaction was executed during the selected time frame.
explained statement	In Oracle, SQL Server, DB2, and Sybase, a statement whose access path (chosen by the RDBMS Optimizer) is clarified and translated into a visual display. Explained results include information on the objects referenced by the statement and the operations performed on these objects.
extended collection	In Oracle, a function that proactively specifies a future period during which Oracle activity data is collected and organized for subsequent analysis. Extended collections are an easy means to view collected information, assess application resource consumption, and identify bottlenecks that are inhibiting application performance and end-user productivity.
Federated Precise	Federated Precise is a version of Precise (starting with version 8.5) that can manage multiple Precise installations within unified StartPoint and AdminPoint screens, displaying and managing all environments, instances, and installations.
findings	A ranked list of top performance problems in the selected context. They are presented in Oracle, SQL Server, .NET, and J2EE. Each finding provides: a short explanation of the problem, detailed background information, and most important: one or more links to further investigate the problem and possibly get more granular findings on the selected context.
first byte time	In Web, a counter that keeps track of the time that it takes from the moment a new Web page is called until the first byte arrives back from the Web server.
FocalPoint	An agent that communicates with the Listeners installed on the monitored servers, receives data from the Collectors, periodically processes and stores this data in the PMDB, and serves UI requests.
folder	In SQL Server, Oracle, DB2, and Sybase, the intermediate logical level in the SQL workspace hierarchy. Folders are grouped into cabinets and contain SQL statements.
	See also cabinet.
Framework Installer	The application that facilitates the installation of Precise framework components. It can be invoked from the installation DVD to install a new Precise deployment. It can also be invoked from an existing Precise deployment to install an additional framework node and attach it to the originating Precise deployment.

See also Framework node.

Framework node

A set of FocalPoint agents that are installed together and manages a set of monitored instances. The performance data of these instances will be loaded into a dedicated PMDB. A single Precise deployment may contain multiple framework nodes (using a separate PMDB for each node). An environment cannot span over multiple framework nodes.

garbage collection

An automatic process in the Java runtime environment that periodically reclaims memory used by objects that are no longer referenced. The process can impact an application's performance while memory is being reclaimed. Java programmers may initiate garbage collection explicitly.

grouper

In Web, the identifier that is used to group other identifiers, such as sites or URLs.

hint

In Oracle, an instruction directed at the Oracle Optimizer that includes considerations for an execution plan. The Oracle Optimizer will build an execution plan based on the hint, ignoring its own set of considerations.

hour group

A unit that reflects the type and level of activity within the system at different times. By defining the times of the day that are peak and off-peak, or day and night, the performance analysis can be focused on those particular times of the day. If, for example, most performance problems occur within nighttime and weekend batches, it can be useful to focus only on them.

InformPoint

In Alerts, an agent that retrieves performance data from all installed Precise products, analyzes it, and sends an alert to the Alerts FocalPoint when the predefined thresholds are exceeded.

See also agent.

Insight

The Precise product family that facilitates the process of monitoring and correlating system performance. It consists of Insight.

See also Report Manager.

See also FocalPoint.

Insight FocalPoint

An agent that receives performance information from Insight Savvies, which monitor the environment. Insight FocalPoint then correlates, processes, and stores this information in a centralized location. The Insight performance history is stored in the PMDB.

See also Savvy.

See also PMDB.

instance

A monitored object of a specific technology. The following list specifies what constitutes an instance for the various supported technologies: J2EE - a Java Virtual Machine (JVM - a logical name set by the user), Microsoft .NET - a Common Language Runtime (CLR - a logical name set by the user), Oracle Applications - an Oracle Applications Form server, Oracle - an Oracle instance, SAP - a SAP system, SQL Server - an SQL instance, Sybase - a Sybase instance, Tuxedo - a Tuxedo domain, Web - a Web server, WebSphere MQ - an IBM WebSphere Queue Manager, DB2 - a DB2 non-partitioned database or a DB2 database partition. During installation, the instance is associated with one AppTier and environment. An Instance can be moved to a different environment or associated with multiple environments without re-installation nor losing historical data.

See also environment.

instance statistics

See execution context.

instance/database changes

In SQL Server and Oracle, the part of the Collect Schema Changes process that captures instance definition changes and database option changes and saves them in the PMDB.

instrumentation

The process of inserting fault-tolerant recording hooks in Java byte code, .NET MSIL, HTML pages, or other monitored components, resulting in the capture of performance metrics. In J2EE, a mechanism that enables collecting performance information when an application is executed. The process involves inserting special fault-tolerant recording hooks into application class objects. In WEB, the insertion of recording hooks into HTML pages can either be in memory (Dynamic Instrumentation) or file-based (Static Instrumentation).

instrumentation context

See execution context.

internal execution

The process of invoking a request from an HTTP request (Servlet or JSP) or an EJB. J2EE displays an internal execution like any other execution.

JRE (Java Runtime Environment)

As the runtime part of the Java software, the combination of the components that enable the execution of a Java program: a Java virtual machine, the core class libraries, and the files that form the Java platform.

JSP (Java Server Page)

An HTML page with special tags for Java scripting. An application server processes the tags and generates a Servlet.

JVM (Java Virtual Machine)	An instance of a JRE that executes Java programs. A server-side Java application server is itself a Java program that runs inside a JVM. Servlets, JSPs and EJBs are Java programs (applications) that run within the application server's JVM. J2EE monitors the JVM running the application server and the server-side Java applications within the application server.
JVM Heap Memory	The amount of real computer memory that is allocated to the JVM for executing Java programs.
key metric	In Alerts, a parameter that monitors a very important performance aspect. The status of the Oracle instance (up/down), for example, is crucial for system performance. If the instance is down, this is the first problem that needs to be solved. Marking a metric as key metric ensures that a critical alert raised for this metric receives top priority by the person that is responsible to handle alerts at any time. It can also help determine which alert to handle first in case of multiple alerts. In Alerts, key metrics are always displayed at the top of a metric table when it is sorted by alerts so that they get immediate attention.
Listener	The agent that facilitates the communication between the various Precise agents across different servers must be installed on every server where Collectors or FocalPoint agents are installed. The Listener allows communication with all other agents installed on the monitored server, while only the Precise Listener port is known by other servers.
login name	In SQL Server and Sybase, the session identifier that represents the credential used to connect to the database. When an ERP extension is installed, the user name of the packaged application's client overrides the login name. For example, when SAP extension is installed, the SAP user name overrides the login name.
machine	A session identifier. A machine as sampled, for example, by the SQL Server Collector is the identification of the machine where the client process executes. Machine is also sampled by Oracle, DB2 and Sybase. In Insight terminology Machine is called Client machine.
Main framework node	A main framework node is the single point for login and also serves as the Precise FocalPoint for the entire deployment.
major collection count	Number of estimated major garbage collection events that occurred during the last J2EE collector slice. A major garbage collection can stop the application while JVM heap memory is being reclaimed. See also garbage collection.
major garbage collection time	Percentage of time spent by the JVM executing major garbage collection events during the displayed interval. See also garbage collection.
MBean	A Java object that represents a manageable resource. In J2EE, MBeans, or Managed Beans, are used for application server metrics.
memory logger interval	The interval at which the J2EE Collector gathers JVM heap memory data. All snapshots of the memory logger's data collected according to the memory logger interval are summarized with counters in the current aggregation interval. The memory logger interval's time span is typically a small fraction of the aggregation interval.
metric	In Alerts, a query that helps measure performance in the environment. Three types of metrics are available: System metrics relate to the internal resources, operations, and objects of the monitored infrastructure; application metrics reflect the way the applications perform; user-defined metrics can be customized to specifically relate to a site. When a metric's value exceeds one of the defined thresholds, its status changes to near-critical (yellow bullet) or critical (red bullet). See also key metric.
metric set	In Alerts, a unit that groups metrics that measure related performance aspects. The following metric sets exist: Status: includes metrics that alert to functional problems related to the instance. Performance: includes metrics that alert to performance problems related to the instance. Load: includes metrics that alert to instance-related load problems that may later cause errors or crashes. Service: includes metrics that alert to instance-related SLA breaches. The metrics in this set are sampled by Insight. Performance Trending: includes metrics that alert to potential future performance problems. The metrics in this set are sampled by Report Manager. Load Trending: includes metrics that alert to potential future load problems. The metrics in this set are sampled by Report Manager. Customized: includes user-defined metrics.

Precise Status: includes metrics that alert to the near-critical or critical status of the installed Precise environment.

module	A session identifier. In Oracle it contains the value of MODULE column in V$SESSION table. In DB2 it contains the command name of non-SQL statements or the package name for SQL statements.
near-critical status	In Alerts, a status indicating that the value of the sampled metric has exceeded the defined near-critical threshold. A near-critical status is indicated by a yellow bullet.
network time	In Web, a counter that keeps track of the time spent on network activity from the server side perspective. This includes the time to read the request from the network and the network time to send the response back to the client.
node	See Framework node. Note: when creating a new installation from DVD, it would be called "Framework." In the UI screens (columns in tables, choosing a system for a new environment etc), we would call them "Nodes." For example: choose a node for the new environment. When adding a new system within AdminPoint, it would be called a "Framework node".
packaged application	An application that is created and/or maintained by a third party and is not custom-built to one's specific needs. The following packaged applications have special treatment by Precise: SAP, Oracle Applications, PeopleSoft, and Siebel. They are harder to modify because the application code is either not available or hard to understand. See also ERP Extension.
page size	In Web, a counter that keeps track of the amount of data loaded from the server to display the page.
page views	In Web, a counter that keeps track of the number of Web pages viewed at a specific Web site during a selected time period.
parent metric	In Alerts, a joining of several child metrics. Each time a parent metric samples, it gathers data from a set of child metrics and presents it as a single metric query. The individual child metric values are displayed on the Thresholds tab of the Properties dialog box in Alerts. See also submetric.
performance counter	In SQL Server, a Windows performance counter as reported by the operating system. In Oracle, an instance level statistic as gathered from some Oracle V$ views. In Sybase, a sysmonitors counter collected by Sybase. In Web, an operating system or a Web server performance counter.
PMDB	The Precise data warehouse of performance and availability data. It can be hosted on an Oracle or SQL Server database.
portal server	An application server for Web-based applications that commonly provide personalization, single sign-on, and content aggregation from different sources and that host the presentation layer of Information Systems. Aggregation is the action of integrating content from different sources within a Web page. A portal may have sophisticated personalization features to provide customized content to users. Portal pages may have a different set of portlets creating content for different users.
portlet	A Java-based Web component, managed by a portlet container, that processes requests and generates dynamic content. Portlets are used by portals as pluggable user interface components that provide a presentation layer to Information Systems.
Precise deployment	An independent Precise system. It contains and manages various agent types and provides centralized monitoring and administration. A Precise deployment may contain multiple framework nodes. One of them is defined as the main framework node and it manages all other nodes of the Precise deployment.
Precise FocalPoint	See Precise FocalPoint agent.
Precise FocalPoint agent	An agent that manages all agents in a single Precise deployment. Additional product FocalPoint agents manage specific technologies and resources.
Precise Proxy agent	When multiple Framework Nodes are managed by a single Precise deployment, the Precise FocalPoint agent manages all of the agents of the main Framework node, while every other framework node has a Precise Proxy agent to manage all its agents.
Precise_root	This is the term used in a path for the Precise installation directory. The terms *<i3_root>* or i3 root can appear in text too.
program	A session identifier in Oracle, DB2, SQL Server, Sybase, Insight, and Alerts. A program as sampled, for example, by the SQL Server Collector, is the name of the executable that connects to the database. Applications that do not set the application name have N/A as program. When a SAP extension is installed,

the SAP transaction overrides the program, and it may change during the application's lifetime. When a Siebel extension is installed, the Siebel views override the program, and it may change during the application's lifetime. In Insight, program is the name of an executable as recognized by the operating system. If an executable is invoked from a script (a batch or a shell), the script is displayed as the command entity. In Alerts, program is part of the customized metrics definition and is the name of the executable or stored procedure executed in the database that runs when the metric is sampled. In action definitions, program is the name of the executable that will run if the metric exceeds its threshold.

progress
In Alerts, the management state of a metric for which an alert has been raised. The following statuses exist:

Open: An alert is raised.

Investigated: The alert is taken care of.

Closed: The problem has been solved.

Proxy FocalPoint
See Precise Proxy agent

recommended index
In SQL Server, a function that uses the Microsoft Index Tuning Wizard to recommend on adding indexes or statistics for a selected statement, batch, or table. Based on the results of this function, the Optimizer will choose a better access plan and make the respective statement or batch perform better. For statements and batches, recommendations are based on the content of the statement or batch. For tables, recommendations are based on all the statements that are stored in the PMDB, are executed during the selected time period, and have an average duration time greater than the value defined in the registry (where 0 is the default).

See also advice.

related SQL
In Oracle, a generated statement that uses alternative syntax to access the database in different ways and returns the same output as the original statement.

relative frequency
Number of contributor executions per entry point. For example, if a service request calls three methods each time it is invoked, the Relative Frequency for the method is three executions per service request execution. Similarly, if a service request calls one method every other time it is invoked (half of the time), the Relative Frequency for the method is 0.5 executions per service request execution.

rendering time
In Web, a counter that keeps track of the time that it takes for a Web page to be loaded from the moment the first byte arrives until the Web page is fully loaded or the user interrupts or abandons the loading process.

report
A collection of queries, programming code, and layout settings that Report Manager executes to generate graphical results like tables and charts.

Report Manager
Uses historical information to identify problematic conditions, track long-term performance, volume trends and patterns, view availability problems over long periods of time and on different levels, compare the performance of similar systems, correlate between performance metrics of different products, assist in capacity planning, and generate demand-driven, user-defined reports.

Report Manager FocalPoint
An agent that examines the PMDB tables to produce scheduled performance reports.

See also PMDB.

report parameter
In Report Manager, a keyword used in a report. Its value is set during the report execution. A parameter's value can be updated either permanently or for the current execution only.

report property
In Report Manager, the attributes that define a specific report, consisting of report parameters and scheduling information.

See also report parameter.

request
In Web, a counter that keeps track of the number of HTTP requests sent for a viewed entity. **request**

error
In Web, a counter that keeps track of the percentage of requests completed with an HTTP error.

response time
In Web and SmartLink, the time elapsed from when a request is received by the server to the time a response is sent back to the computer that made that request. The response time is measured on the server side. In J2EE, the time it takes an execution to complete execution. In other words, response time is the average length of time between the start time and end time of a Java method execution. For example, the SQL response time is the time it takes the JDBC method executing the SQL statement to be completed. The response time includes CPU and wait time. The response times are reported in the interval in which they complete execution. Though a contributor's average response time may be very small, the contributor may be called many times. As a result, a contributor's overall contribution to performance may be large even though its average response time is low.

sampling	In Alerts, the process during which a metric queries a Precise product for a specific instance, retrieves values, and calculates the metric's alert level.
sampling base	In Alerts, the start time of a scheduled sampling process (by default Sunday, 00:00 AM).
sampling period	In specific Alerts metrics, the time frame during which statistical data is returned from other Precise products. Such a metric is for example Oracle's General Behavior metric, which returns database behavior for a certain period of time.
sampling rate	In Alerts, the frequency of a metric's regular sampling schedule. The sampling rate is measured in minutes.
Savvy	An Insight agent that collects AppTier-specific performance data.
scalability	A system's ability to withstand load. For example, positive scalability means that the system continues to function properly even when it is called upon to service a larger number of users.
schema changes	In Oracle and SQL Server, a process that captures schema changes and saves them in the PMDB. In SQL Server, instance configuration parameters and database option changes are also captured.
Script Installer	A component of Precise Agent Installer that enables adding a small JavaScript script, also known as a Web browser-side agent, to the Web pages of your Web site.
server	Also called a host machine or server machine. The combination of a computer and associated operating system software that is accessed by multiple clients to perform requested functions and typically returns results to the clients.
server-side collection	In Web, the collecting of performance data from the Web server instance through the Web server agent.
service + network time	In Web, a counter that keeps track of the service and network time, including the total amount of time the request took to reach the server.
Service Level Agreement	See SLA (Service Level Agreement).
servlet	A class that is loaded only once and for which the application server uses multithreading to process requests. The servlet generates an HTML page that is sent back to the Web browser.
size	In Web, a counter that keeps track of the bytes sent and received.
SLA (Service Level Agreement)	A formal definition of an information system's performance goals. Within Precise, an SLA consists of clauses corresponding to various system activities. Once a system's SLA is defined, its SLA compliance can be analyzed, and breaches can be isolated to identify their causes.
SmartLink	In Insight, a component that provides transactive, correlated information across AppTiers from the user's perspective. Insight SmartLink functions in PeopleSoft and Web applications environments.
SmarTune	In Oracle, SQL Server, and .NET, an algorithm that analyzes performance metrics, identifies and ranks potential problem areas, and provides advice for correcting the problems.
socket	An I/O abstraction layer that enables processes to communicate with each other, regardless of whether they are on the same server. Sockets are bi-directional FIFO pipes that look like regular file I/O to the developer with the abstraction layer handling all of the low-level communication details.
StartPoint	The opening page of Precise. It provides a quick overview of the environment status and links to launch any of the Precise products.
Statistics workspace	In Oracle, DB2, SQL Server, Sybase, J2EE, and Web this workspace presents instance level statistics such as hit ratios, utilization, JMX metrics, logging, and more.
submetric	Also called child metric. In Alerts, the subquery of a parent metric. A child metric gathers its own data and combines it with the data gathered by other child metrics to form the result of the parent metric. Each child metric has its own thresholds and may be enabled or disabled individually. See also parent metric.
summary table	A container that stores the information collected by the Precise agents and loads it into the PMDB. Summary tables store the same data at different levels of granularity: time slice, hourly, daily, weekly, and monthly. By storing data in multiple summary tables, Precise can present a detailed view and progressively higher-level views of the same data. Summary tables are particularly useful for data aging. A data purging policy can be implemented for each summary table so that detailed data is retained for short-term historical analyses while more summarized data is used for long-term analyses and trending.
technology	A technology identifies the monitored object. For example, Oracle, SQL Server, Sybase, and DB2 are different technologies, while all Web servers (such as: Apache, IIS, and WebSphere) are defined as a single

Web technology. A single monitored object can be monitored by two different technology's Collectors. For example, WebLogic server can be monitored by both a Web Collector and a J2EE Collector.

throughput   The number of average completions per second that are observed during an interval.

time slice   A unit used to break up long sessions into smaller time periods. The length of a time slice is preset and cannot be changed. It represents the maximum time that passes before the data collected can be displayed. For example, if the length of a time slice is 15 minutes, the collection is only updated at 15-minute intervals.

The length of a time slice is different for each technology: In Oracle, SQL Server, Sybase, DB2, Web, .NET, MQ, Tuxedo, Network, and OS, a time slice is 15 minutes. In J2EE and SmartLink, a time slice is either 30 seconds or 5 minutes. In Precise for SAP, a time slice is 5 minutes.

See also collector slice size.

URI (Uniform Resource Identifier)   The relative path to a resource after the location (network node) is found.

URL mapping   In Web, a function that defines rules that map URLs (Uniform Resource Locators) with dynamic parameters originating from a specific domain to a format that identifies the Web pages and prevents them from having different URLs.

Web client browser-side   In Web, the gathering of performance data from the browser running on the desktop of the Web application's user, through the static or dynamic instrumentation of Web pages.

Web server   A program that receives client requests for Web pages, retrieves the static pages and/or issues a request for dynamic page creation to an application server, and sends the pages to the client. In Web, a computer that delivers (serves up) Web pages. Every Web server has an IP address and possibly a domain name.

work time   The time spent in the specific entity excluding time spent by called (other) entities. In J2EE, work time of a method is calculated as the response time of that method minus the response time of the methods it invoked.

work type   In Web, a counter that keeps track of the type of the Web instance, such as Web, PeopleSoft, SAP, or Siebel.

workspace   A display unit in Precise products. All workspaces display data from different perspectives. For example, in Oracle, SQL Server, and Sybase, the Current workspace shows information on the sessions currently active in an application, and the Objects workspace displays information on Oracle or SQL Server database objects that can be used to understand relationships and associations between database schema objects.