Portrait
Foundation

Model and Node Counters Reference Guide

Edition 4.0

04 August 2014

Pitney Bowes
Software

Portrait Foundation
Model and Node Counters Reference Guide

About Portrait Software

Portrait Software is now part of Pitney Bowes Software Inc.

Portrait Software enables organizations to engage with each of their customers as individuals, resulting in improved customer profitability, increased retention, reduced risk, and outstanding customer experiences. This is achieved through a suite of innovative, insight-driven applications which empower organizations to create enduring one-to-one relationships with their customers.

Portrait Software was acquired in July 2010 by Pitney Bowes to build on the broad range of capabilities at Pitney Bowes Software for helping organizations acquire, serve and grow their customer relationships more effectively. The Portrait Customer Interaction Suite combines world leading customer analytics, powerful inbound and outbound campaign management, and best-in-class business process integration to deliver real-time customer interactions that communicate precisely the right message through the right channel, at the right time.

Our 300 + customers include industry-leading organizations in customer-intensive sectors. They include 3, AAA, Bank of Tokyo Mitsubishi, Dell, Fiserv Bank Solutions, Lloyds Banking Group, Merrill Lynch, Nationwide Building Society, RACQ, RAC WA, Telenor, Tesco Bank, T-Mobile, Tryg and US Bank.

Pitney Bowes Software Inc. is a division of Pitney Bowes Inc. (NYSE: PBI).

For more information please visit: http://www.pitneybowes.co.uk/software/

UK

Portrait Software
The Smith Centre
The Fairmile
Henley-on-Thames
Oxfordshire, RG9 6AB, UK

Email: support@portraitsoftware.com
Tel: +44 (0)1491 416778
Fax: +44 (0)1491 416601

America

Portrait Software
125 Summer Street
16th Floor
Boston, MA 02110
USA

Email: support@portraitsoftware.com
Tel: +1 617 457 5200
Fax: +1 617 457 5299

Norway

Portrait Software
Portrait Million Handshakes AS
Maridalsveien. 87
0461 Oslo
Norway

Email: support@portraitsoftware.com
Tel: +47 22 38 91 00
Fax: +47 23 40 94 99

About this document

Purpose of document

This document describes the *Model and Node Counters* facility within Portrait Foundation. This facility enables the capture of information about the execution of Portrait Foundation process models within a live (or test) system, and later to access the captured information through a Profiler application.

Intended audience

The document is intended for anyone needing to set up the facility or make use of the data captured through it.

Related documents

None

Software release

Portrait Foundation 5.0 or later.

Contents

1 Introduction

The Model and Node Counters enable the capture of information about the execution of Portrait Foundation process models within a live (or test) system, and later to access the captured information through a Profiler application. The information capture process has little or no effect on the performance of the runtime system, so it is possible to leave this facility switched on permanently in a production environment.[1]

The counter information is intended for three main purposes:

1. Ongoing health monitoring of a Portrait Foundation implementation. By accumulating information over a period of time (potentially weeks, months or years) it is possible to detect trends in the performance of specific parts of the system.
2. Investigation of performance issues. Configurers, or Developers can make use of the counters to locate poor-performing models and nodes. This may help to identify bottlenecks in the process models – for example calls into an external system may be taking a long time.
3. Analysis of model execution patterns. Configurers can access information about how often each node in each model gets executed, which provides a view of the most common routes through each model. This information can be compared with the expected behaviour of the model, and any anomalies investigated.

Figure 1 shows an example of viewing the summary information for all models that were executed over a two minute period in a test environment.

Figure 1 – Example of model execution summary information

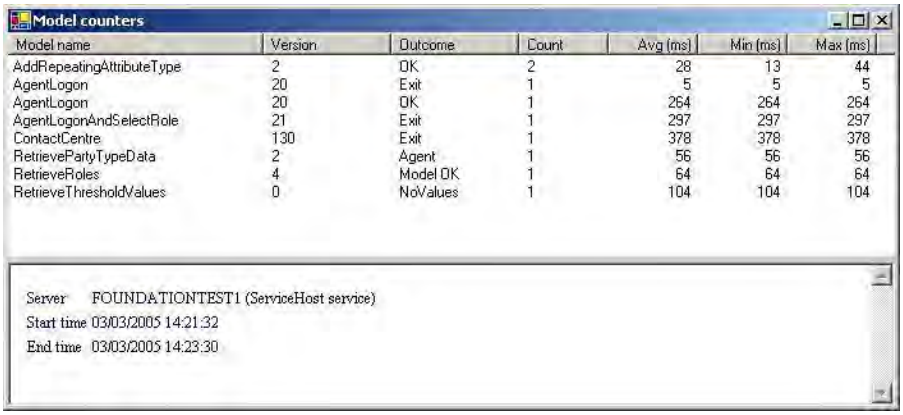
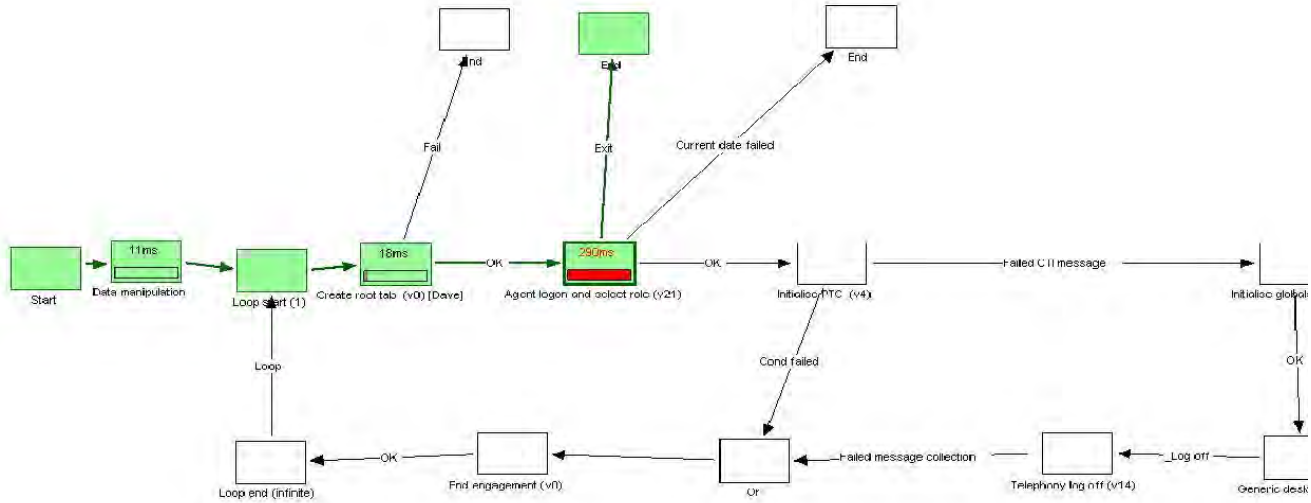


Figure 2 shows an example of a model being viewed in the Profiler, with node timing information overlaid. The green shading indicates those nodes that have been executed at least once. Node times are shown for any node taking more than one millisecond, and if the time is more than a certain threshold, the time is shown in red.

[1] See discussion on data management in section 6

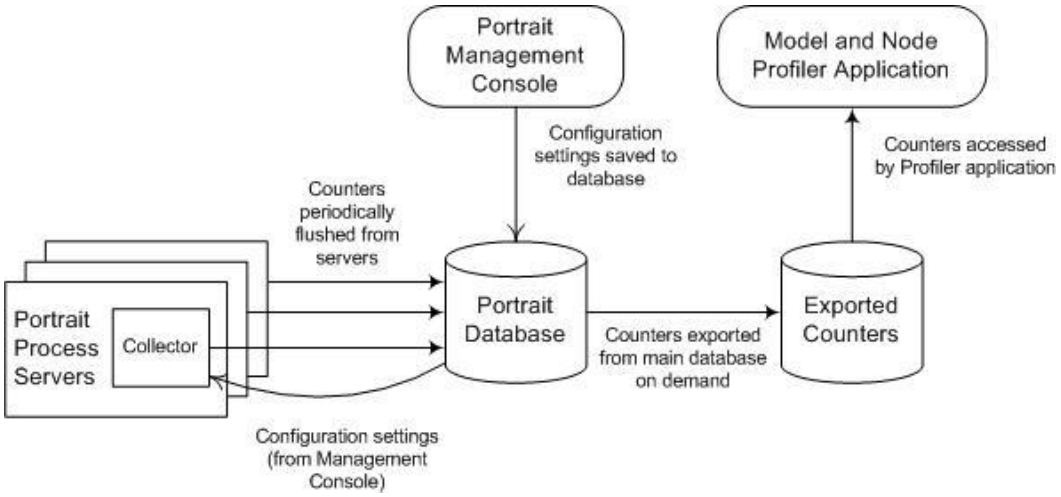
Figure 2 – Example of node timings overlaid on a process model



2 Solution Overview

Figure 3 shows an overview of the components that are involved in the solution and how the data flows between them.

Figure 3 – Components of the Model and Node Counter facility



The components shown here are as follows:

- **Portrait Foundation Management Console** – The *Performance Auditing* property page in the Management Console enables the configuration of various parameters that determine if and how the Model and Node Counters are collected within a Portrait Foundation system.
- **Collectors on Portrait Foundation Process Servers** – This component collects information about the execution of each model (and each node within those models) that are executed by the Process Server. This information is accumulated in memory within the Collector until the end of a configurable time interval. After that time interval, all counters are written to the database and the in-memory counters are reset, ready for the next time period. The set of counters written together are known as a *group*.
- **Portrait Foundation Database** – The database is used for two purposes in the scenario described here. First, the configuration settings from the Management Console are stored here. (These settings are always stored in the primary, operational database). Secondly, the counter information captured by the Collectors on each of the Portrait Process Servers is written to a Portrait Foundation database. (This information can be written to either the operational database or the "transient" database, depending on an installation-time option).
- **Exported Counters** – This is another database, which holds a set of counters that have been exported from the main Portrait Foundation database, along with any required configuration data. Exporting data from the Portrait Foundation database is a manual task that can be carried out as and when required. (Note that the counter data is copied, not moved from the Portrait Foundation database).
- **Model and Node Profiler Application** – The Profiler enables a user to browse through the counter information in the export database and to examine in detail the performance characteristics of each model and each node executed. The data can be filtered by time or by the server that generated it.

  It is preferable to run the Profiler on a machine with either the Configuration Suite or the Portrait Process Server installed. It is, however, possible to run the Profiler on a non-Portrait Foundation machine, but this will result in

reduced functionality. In particular, it will not be possible to view models graphically or expand model configurations[2] on this set-up.

[2] Section 7.6 discusses the need to expand model configurations.

3 Using the Counters – Overview

There are a number of steps that must be carried out in order to enable the Model and Node Counters to be captured by the runtime system and subsequently to be analysed.

In Portrait Foundation version 2.x the counters are *disabled* by default and must be enabled before the first use. The first three steps below are one-off activities that must be completed before first use. The remaining steps are regular activities that need to be carried out each time the counters are analysed.

The following steps need to be carried out once, ideally at the time when a counter-enabled version of Portrait Foundation is installed.

1. A suitable version of Portrait Foundation must be installed in the runtime environment to enable the counters to be captured and the Profiler application must be installed on at least one machine to enable the captured counters to be viewed. Check with Portrait Support for availability of the counters in different Portrait Foundation releases. Section 4 provides more information about installation.
2. Enable the counters through the *Performance auditing* properties page in the Portrait Management Console. If this page is not available it means that your version of Portrait Foundation does not include the counter functionality. The settings on this page are described in detail in Section 5. After enabling the counters, data will start to be written periodically to the database.
3. Set up a database job to purge counters that are accumulated over time in the primary or transient database. See Section 6 for details of purging old data.

The remaining steps need to be carried out on a regular basis, each time the counters need to be analysed.

4. Copy the counters from the live database into an "extract" database for analysis. This should be done after the data has been collected for the required amount of time, perhaps a day. If your support team has requested it, zip up the extract database and send it to them. See Section 6 for details of extracting counters.
5. If analysing the data yourself, start the Model And Node Profiler application and specify the details of your extract database. Use the Profiler to browse through the captured model and node counters and to view models graphically, with overlain node information. See Section 7 for details on using the Profiler. Note that the first time that the viewer is used with any given database it is necessary to expand the model configurations before viewing the counters, as discussed in section 7.6.
6. If required, ad hoc reports can be generated from the counter data in the extract database. This may be useful for investigating trends in data over time, or for other purposes. Section 8 discusses queries and reporting.

4 Installation Considerations

The counter functionality affects two different installation processes – the runtime servers and the machine used to view the results.

4.1 Process Server installation

The System Setup tool allows administrators to choose which database to store the Model and Node Counters in.

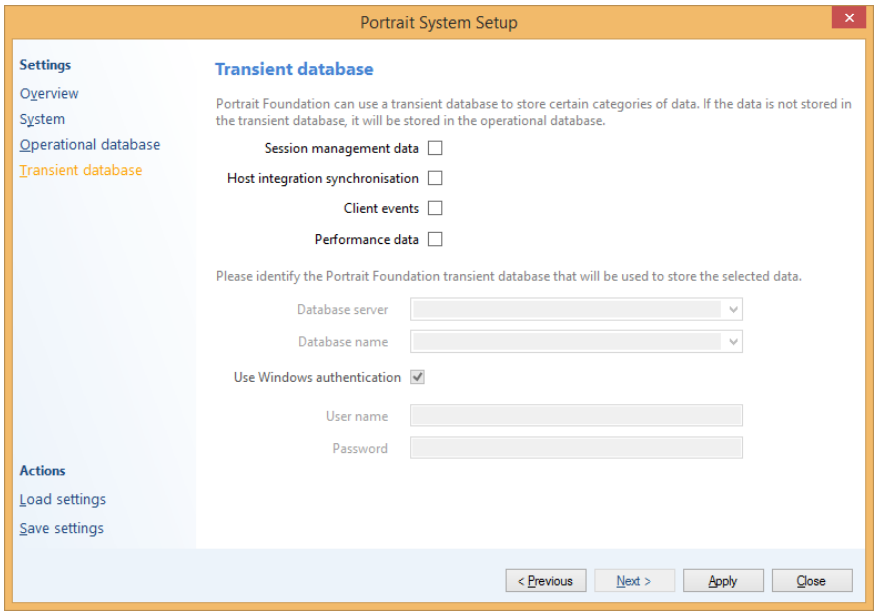


Figure 4 - Transient database configuration during System Setup

The transient database is optional, and many Portrait Foundation implementations do not use it. Its purpose is to offload some of the processing overhead from the primary database in areas that do not require access to the main "business" data.

For implementations already using a transient database, the recommendation is to direct the Performance Counter information to that database. However, if a transient database is not being used, it is not usually necessary to introduce it just for the Performance Counters. In this situation the Performance Counters can make use of the primary database. See notes in Section 6 for more information about database management.

4.2 Profiler application installation

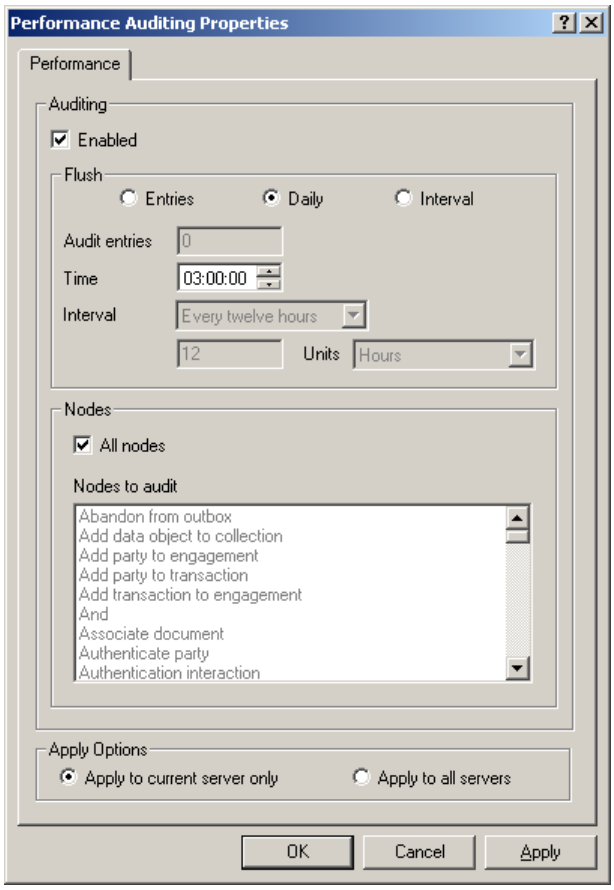
The Model and Node Profiler tools is installed by default using a "Typical" Core Software install and is included as part of the Client Tools feature.

After installation, the Profiler is available through **All Programs** \ **Portrait Foundation** \ ***<Foundation_System>*** \ **System Tools** on the Start menu.

5 Management Console settings

There are a number of parameters that control the way in which the Model an Node Counters are captured in the Portrait Foundation operational environment. These parameters are set through the Performance Auditing section of the Portrait Foundation Management Console, which is available for servers configured as CRM Servers.

Figure 5 – Management Console properties page for Counters



The *Enabled* checkbox control whether or not counters will be captured in the runtime environment. When *Enabled* is checked, counters will be captured based on the settings in the *Flush* and *Nodes* sections of the properties page.

The *Flush* section controls how long the server accumulates counters in memory before flushing them to the database. Three options are available:

- **Entries** – This limits the number of counters that will be accumulated in memory at any time. When the server reaches this number of entries it will flush all entries to the database, clear down the counters and start recording the next set. The *Entries* option can be used to limit the amount of memory taken by the counter capture component.
- **Daily** – The counters are flushed to the database once per day, at a time specified by the *Time* field. This is likely to be the setting used in most live environments.
- **Interval** – The counters are flushed to the database at an interval specified by *Interval* combo box. Possible intervals include: Every half hour, Every hour, Every six hours, and so on. Portrait Foundation will ensure that the counters are flushed at times that make sense for the interval – for example, if "Every six hours" is selected, the counters will be flushed at midnight, 6am, noon and 6pm. In an environment with multiple Process Servers (CRM Servers), selecting one of these pre-defined intervals will ensure that all servers flush their counters to the database at approximately the same time,

making it easier to correlate results across servers. A further option is to select *Custom* from the *Interval* combo box, in which case number and a unit can be specified – for example 30 seconds, 3 minutes, or 5 hours. When a custom interval is defined, Portrait Foundation will not attempt to flush on any particular time boundary so it is more difficult to correlate results from multiple servers.

The *Nodes* section of the properties page controls which nodes are counted by the runtime collection process. By default all nodes are counted, meaning that a full set of information is available to any tools or queries needing to process the counters later. However, it is possible to configure the system to just record details of a specified set of node types – for example Data access, Scripting and Get work. This will reduce the amount of memory used by the runtime collection component, reduce the number of records written to the database and reduce the processing overhead on the server. However, less information will be available when viewing the results later.

The operational characteristics of the counter collection mechanism can be tuned by adjusting the values of the *Flush* and *Nodes* settings. The exact values to use will depend on the type of investigation that is being carried out. Some likely scenarios are:

- Production environment, with no particular problem suspected. Flush daily, preferably during a quiet period such as 3am. Capture information for all nodes. Review the results regularly and look for any degradation in performance over time (weeks or months).
- Production environment, with some performance issues observed during peak hours. Flush hourly, or perhaps more often, so that the profile can be compared at different times of the day. Consider capturing data for only a subset of the nodes to reduce the volume of data collected in the database. Control nodes (Start, End, Branch, And, Or, etc) are good candidates for filtering out.
- Performance testing environment. The objective here is to investigate any models that appear to run slowly and also see whether there is any change in performance characteristics over the duration of a load test. The flush interval should be quite short – perhaps 10 minutes for a 2-hour load test – to enable trends to be analysed. Consider capturing data for only a subset of the nodes to reduce the volume of data collected in the database.
- Unit testing environment. The objective here is for the business users (Configurers) who built the models to be able to validate the performance and execution profile of those models. The users will manually execute a pre-defined test script and then investigate the captured counters. This will not normally be done under load. The flush interval should be set to more than the duration of the test – perhaps 30 minutes – so that all counters are included in a single group. All nodes should be recorded because the user will require maximum information when analysing the model and node executions.

Note that when a Portrait Process Server is shut down, it will attempt to flush any current counters to the database regardless of the specified purge interval.

6 Database management

Database Administrators working with the Model and Node Counters facility should be aware of the following.

1. The counters collected by the runtime components will be written to either the primary Portrait Foundation database or the transient database, depending on the settings chosen at installation time. The database management procedures discussed below may therefore need to be applied to either of these databases.
2. The counter tables require some housekeeping operations to prevent them growing excessively large. Procedures are provided.
3. In a production environment the counters should be extracted from the live database into an extract database before users access it with the Profiler or ad hoc queries. In test environments it may be acceptable to leave the counters in the main database for analysis.

6.1 Maintenance of the counter tables

The amount of data generated by the runtime components can be quite large in a production environment, particularly if there are multiple Process Servers involved. For example, a typical Portrait Foundation implementation may have around 500 models containing an average of 10 nodes each. If all of these models are executed regularly (i.e. during the recording period of a "group"), approximately 500 model counter records and 5000 node counter records will be written to the database for each group. If there are five Process Servers in the system and they are configured to measure in one-hour periods, there will be:

24 (hours) x 5 (servers) x 5500 (models+ node counters)
= 660,000 records written per day.

A stored procedure, `p_amc_dep_purge_counters`, is provided to purge the accumulated model and node counters on a regular basis. This procedure takes a single parameter – the minimum age of the data to be purged. It deletes all groups, node counters and model counters that are older than the specified age. The age is specified in hours. If no age is specified, all data in these tables will be purged.

It is recommended that a SQL Server job is created to run this procedure regularly. A typical scenario would be to run the job daily and purge all data more than a week old (168 hours). However, the amount of data accumulated will depend on the rate at which the servers are generating it (i.e. the flush interval) so it may be appropriate to keep, say, a month of data if the system is flushing only once a day.

6.2 Extraction of counters

It is preferable to execute the Profiler tool on a copy of the data instead of on the live database server. It is therefore necessary to copy the counter data and associated configuration (model definitions, etc) into an extract database. The configuration data must always be copied from the primary database. The counters need to be copied from either the primary database or the transient database, depending on the database chosen at installation time.

A SQL script, `amc_utl_extract_node_audit.sql`, is provided on the Portrait Foundation installation CD for the purpose of copying the counters and the configuration into an extract database.

The extraction process consists of the following automated steps:

1. Create a new database shell for the Audit database. This will have the prefix "*Audit_*" and then the name of the operational database.
2. Copy the relevant performance audit files into it from the Live/Transient databases.
3. Create the necessary stored procedures and views in the new audit database.
4. Optionally;
    a. Create a compressed file backup of the new Audit database,
    b. make sure that we know where the backup is on the database server's file system, and
    c. delete this new Audit database from the database server keeping only the compressed backup which can be distributed for investigation on another server.

DBAs should read the comments in the script before executing it. There are a number of parameters at the top of the script that need to be filled before the script can be run. They include:

- The operational database name. This can be left as db_name() if the script is being run against the operational database.
- The transient database name. If the performance auditing data is being collected on a transient database, then use its name here. Note that it MUST be located on the same database server as the operational database.
- Either the number of days of data to collect *or* a start date and an end date of the periods where data is being collected.
- The optional flag that determines whether to create a backup of the output data and delete the generated audit database. This is useful to ensure that the audit database does not get left on the server. It is also necessary if you wish to analyse the data on a database server other than the current one. E.g. if requested by Portrait Support.

⇒ Note

The script requires that the user have a number of administrative permissions on the database server granted. These permissions are checked at the beginning of the script to ensure that the user will be able to complete all of the tasks within the script. If the relevant permissions cannot be granted to the user running the script, then a DBA should manually follow the same steps using different accounts that do have the correct permissions.

7 Using the Model and Node Profiler

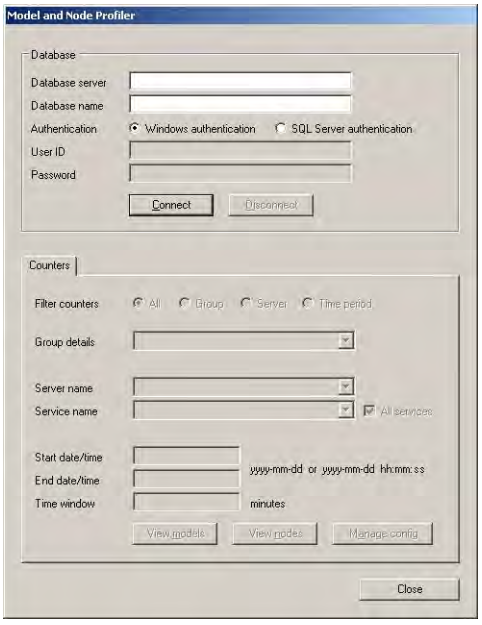
The Model and Node Profiler enables counters to be browsed offline from the system where they were captured. The Profiler offers the following features:

- Connect to a database using specified logon credentials (either Windows authentication or SQL Server Authentication).
- Set up a filter to restrict the data to be viewed. The server(s) in the runtime environment write the counters to the database in *groups*. A group consists of sets of counters that were recorded on a single machine during a certain time period. (The time periods relate to the settings in the Management Console.) The filter enables the selection of one or more groups meeting server or time criteria.
- View a list of all models executed within the selected group(s). The models can be sorted by name, type, minimum, maximum or average execution times.
- View a list of all nodes executed within the selected group(s). The nodes can be sorted by name, type, minimum, maximum or average execution times or by the model in which they were executed.
- View a graphical representation of a model with various counter information overlaid.
- Prepare the node configuration in the database. This is a one-off activity that must be performed before attempting to view the node and model counters.

7.1 Database Selection

Enter the database information and logon credentials into the Profiler's main screen, then press Connect to validate the information.

Figure 6 – Profiler main screen

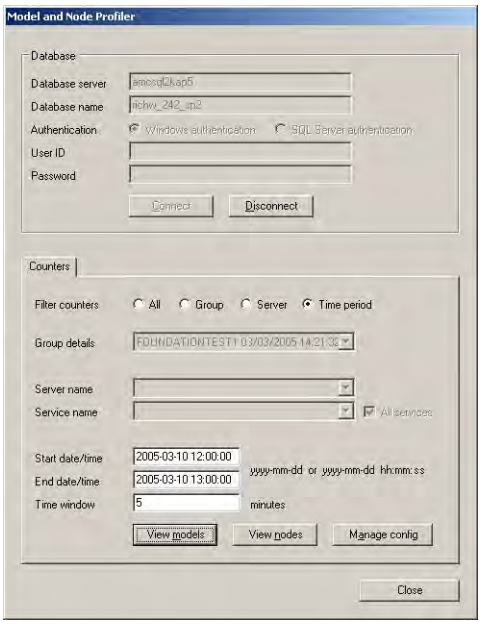


If the connection is successful, various controls on the screen will become enabled.

7.2 Setting up a filter

The lower part of the main screen enables various filter criteria to be specified. The default is to process all counter information in the database, but by specifying a filter it is possible to drill into counters specific to a certain time period or a certain server.

Figure 7 – Filtering by time



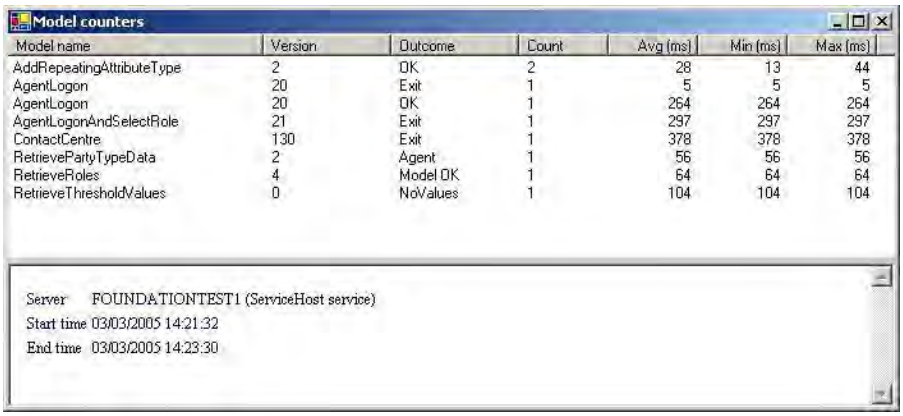
The filter options are:

- All – Show all of the counter information in the selected database.
- Group – Select a specific group to analyse. A Group consists of all of the counters that were recorded during a certain time period on a specified server. The *Group details* combo box shows the name of the server and the time at which the group started recording counters.
- Server – Select a specific server from the Server name combo box. In some circumstances, there may be multiple instances of the Portrait Process Server executing on a single, physical machine. In this case each instance will have the same server name by will have its own unique service name. If required, a specific service can be specified in the *Service name* combo box. However, the default of *All services* should be appropriate in most circumstances.
- Time period – Enter a time period for which the counters should be profiled. Any groups that started and ended within this period will be included in the filers. Both the start time and the end time are optional, but if specified they must be in the format YYYY-MM-DD HH:MM:SS (year, month, day, hours, minutes, seconds). Optionally a *Time window* can be specified in minutes. This gives a "tolerance" in minutes for the start and end times, to allow for the fact that groups written by different machines may not be perfectly aligned on time boundaries.

7.3 Viewing model counters

Select the *View models* button on the main screen to display the model counters screen. This screen shows all of the models that were executed within any of the groups that meet the filter criteria. The lower part of the screen displays the filter criteria being applied.

Figure 8 – Model counters screen



Counters are collected separately for each version of each model, and for each outcome of each model. Hence there may be several entries in the list for the same model name. This enables the relative performance of different versions of a model to be compared, and also shows how performance varies depending on the outcome. For example, in Figure 8 it can be seen that the AgentLogon model took an average of 264ms when the user completed the logon process (outcome = "OK"), and only 5ms when the user quit (outcome = "Exit").

The *Count* column shows the total number of times that the model was executed. The *Avg*, *Min* and *Max* columns show the average, minimum and maximum time in milliseconds for each model.

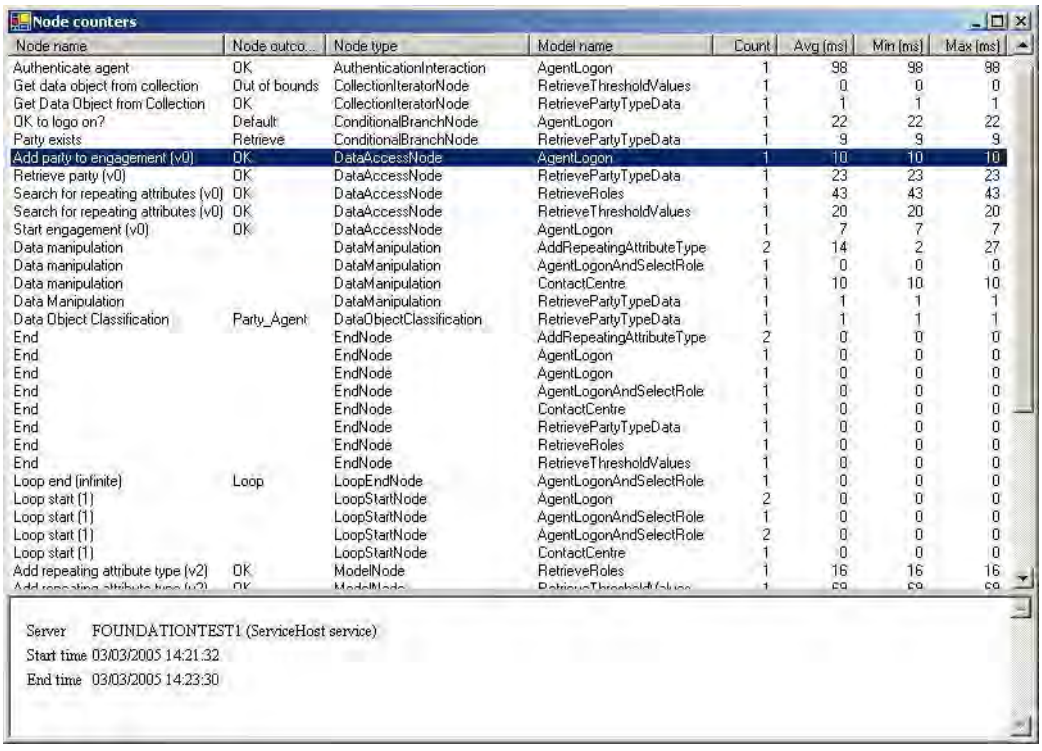
The data on this screen can be sorted by any of the columns by clicking a column header. Clicking the same header a second time will reverse the sort order.

Double-clicking any model in the list will display the *Model viewer* screen.

7.4 Viewing node counters

Select the *View nodes* button on the main screen to display the node counters screen. This screen shows all of the nodes that were executed in any model within any of the groups that meet the filter criteria.

Figure 9 – Node counters screen



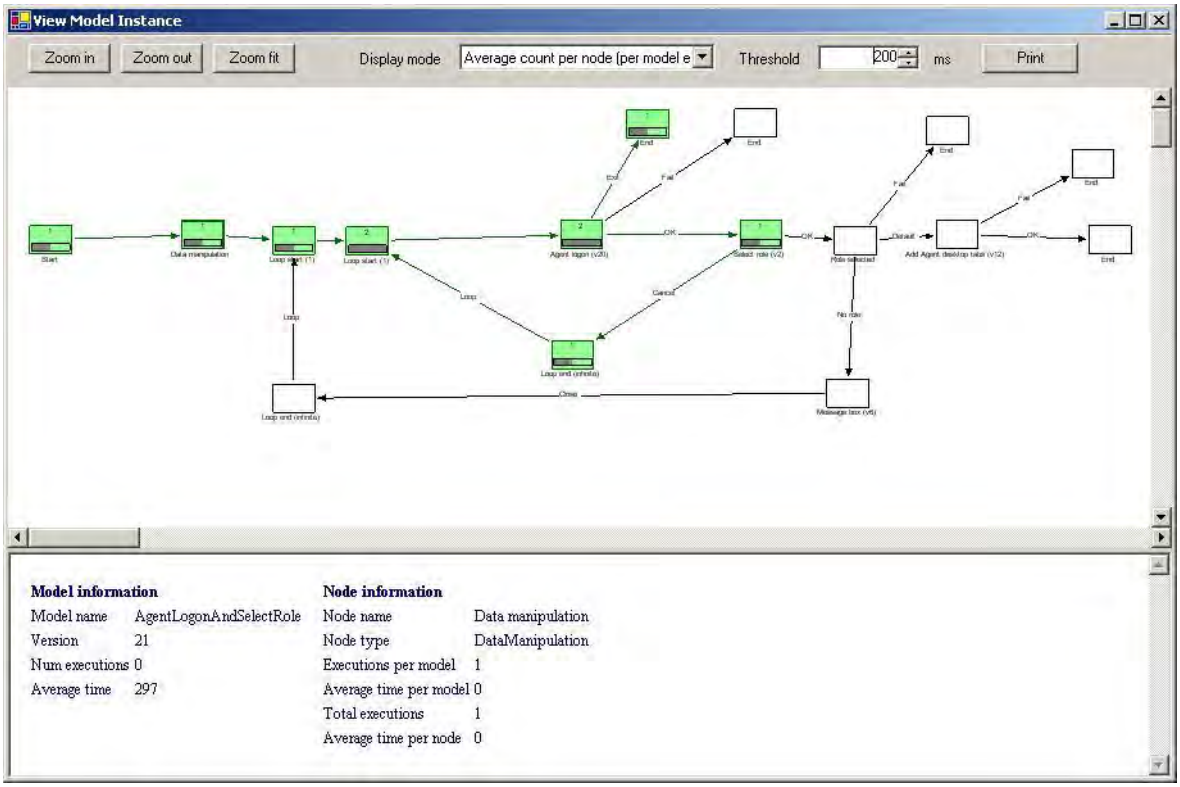
Counters are collected separately for each node in each model, and for each outcome of each node. The type of each node is displayed, along with the *Count*, *Avg*, *Min* and *Max*, as per the model counters screen.

Double-clicking any node in the list will display the containing model in the *Model viewer* screen. The relevant node will be automatically selected within the model viewer.

7.5 Viewing models graphically

The *Model viewer* screen can be launched by double-clicking on a model in the *Model counters* screen or a node in the *Node counters* screen. The *Model viewer* displays a graphical representation of the model, with various counter information overlaid, as shown in Figure 10.

Figure 10 – The Model Viewer



The layout of the model is based on the way that it was designed in the Configuration Suite. However, the representation of the nodes is different in the *Model viewer* because its purpose is to show performance and execution information.

Nodes that have been executed at least once are shaded in green, whilst those not executed are empty. Clicking on any node will display additional information at the bottom of the screen, consisting of:

- Name of the node (as defined at configuration time)
- Node type (for example, data access, generated interaction, etc)
- Average number of executions per model instance
- Average amount of time spent by this node in each model instance (e.g. a node that takes 100ms and gets executed three times in a loop will have a total time of 300ms)
- Total number of executions of the node (across all model instances)
- Average time for the node across all of its executions

The toolbar at the top of the window enables zooming of the model within the window and printing the model at its current zoom level. The default printer and settings (such as page orientation) will be used.

The toolbar also enables a display mode to be selected, which determines what information gets overlaid on top of each node. The display modes are as follows:

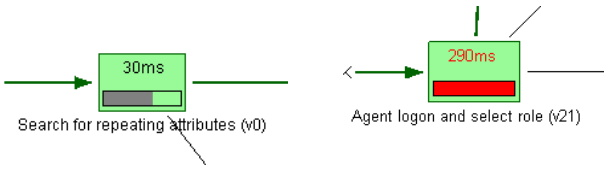
- Total time per node (per model execution)
- Average time per node (across all model executions)
- Max time per node (across all model executions)
- Average count per node (per model execution)
- Total count per node (across all model executions)

The numbers on the node represent the absolute value of the chosen counter (average time, total count, etc) and the bar below the number shows a relative value, compared with the highest value of the counter across all nodes in the model.

The other control on the toolbar is the *Threshold* indicator. By default the threshold is set to 200ms, but this can be changed by the user. When displaying any of the time-based counters, any node exceeding the threshold will have its text and bar shown in red instead of black. The threshold has no effect when displaying iteration-based information, such as average count.

Figure 11 shows some example nodes. The node on the left took 30ms, which was approximate 65% of the time taken by the longest-running node in that model. The node on the right, taken from a different model, took 290ms and was the longest-running node in this model (because the bar is full). We can also see that 290ms is above the current threshold because the text is displayed in red.

Figure 11 – Node examples



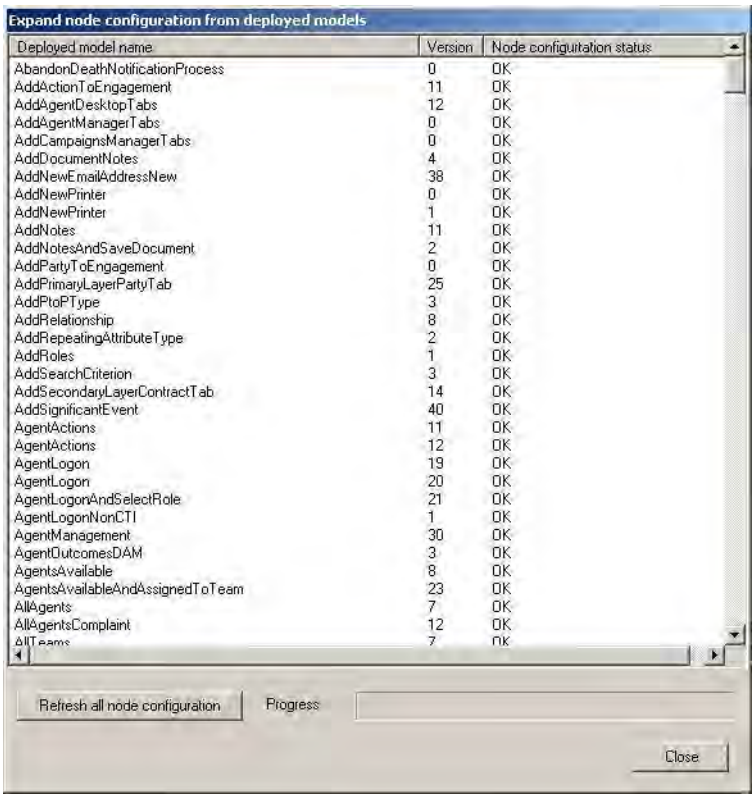
7.6 Preparing node configuration

When a Portrait Foundation system is deployed from the Configuration Suite, model definitions are saved to the database in a binary, streamed format that is required by the Process Engine at runtime. Amongst other things, this streamed information contains details about the individual nodes that make up the model. This node information is not deployed in any other format.

In order to make sense of the node counters saved to the database by the Process Engine, it is necessary to have the node configuration information available in a standard, relational format instead of the binary stream. The Model and Node Profiler requires this information when displaying its *Node counter* screen and it may also be required by other ad hoc query or reporting tools. Node configuration information is obtained by "expanding" the configuration of the model that contains the node. This must be done once for each database used with the Node Profiler. If configuration data is re-deployed to that database, the expansion process must be re-run.

The Profiler's main screen contains a *Manage config* button which brings up the *Expand node configuration from deployed models* screen.

Figure 12 – Expand node configuration screen



This screen lists all of the models that are deployed in the database, along with their version number. (Note that there may by multiple versions of the same model deployed, in which case the model will appear multiple times in the list). For each model/version combination, a node configuration status is displayed. The status is one these values:

- OK – The model configuration has already been expanded into its constituent node configurations and is up to date.
- Node config not present – The model has not been expanded into its constituent nodes, so it will not currently be possible to view the nodes in that model in the *Node counters* screen.
- Node config out of date – The model has previously been expanded, but the node configuration is older than the model configuration, suggesting that the model has been re-deployed since it was last expanded. Some node configuration information may be missing.

If all of the models are in an OK state, model expansion has already been carried out and no further action is necessary. If this is not the case, press the *Refresh all node configuration* button to start the process of expanding all model definitions. Note that this process may take several minutes to complete. The process can be cancelled once started, but this will leave the node configuration information in an intermediate state with some models expanded and others not.

The expansion process needs to be carried out once when a database is first used for the counters, and again whenever new configuration is deployed to that database.

8 Ad hoc queries and reporting

The Profiler tool enables interactive analysis of a snapshot of Model and Node Counters during a certain time period. It is intended to assist Configurers and Developers in diagnosing "hot spots" within the configuration.

However, the raw counter information can also be used for other purposes. Examples include:

- Monitoring how model usage changes over time. For example, reporting on the total executions of each type of model in a 24 hour period, and the average response times of those models, and plotting the results over a period of weeks or months.
- Identifying "hot spots" at different times of the day. For example, reporting on the average response times of certain key nodes (such as Data Access Nodes) over 1-hour periods throughout the day.
- Filtering and amalgamating data from different models or nodes. For example, reporting on the average response times of all Data Access Nodes by target system (Portrait Foundation database, external host system, etc). This may require some detailed knowledge of the way the Portrait Foundation system has been configured.
- Comparing the relative throughput and response times of different servers in a Process Server farm, or different instances of Portrait Foundation running on the same server in a multi-instance configuration.

When using the counters for these purposes, it is necessary to query the data directly from the database. A query tool (such as SQL Server Query Analyser) or a reporting tool (such as SQL Server Reporting Services) can be used. Sample queries and reports may be offered in a future release, but are not currently available. Users will therefore need to develop their own queries and reports when using the current Portrait Foundation releases.

Mode and Node Counters can easily be retrieved through a pair of database views – `v_amc_dep_model_counters` and `v_amc_dep_node_counters`. As the names suggest, the first view returns information about all of the model counters in a database and the second view returns all of the node counters.

The columns returned by `v_amc_dep_model_counters` are:

perf_group_id	Unique identifier for the performance counter group (set of related counters written by a single service on a single server during a certain time period)
server_name	Name of the server that collected and saved the group of counters.
Service_name	Name of the service on the server. If there is only once instance of Portrait Foundation on the server (the default case), the service name will normally be ServiceHost. If multiple instances are present, the names will normally be ServiceHostA, ServiceHostB, etc.
group_start_time	The date and time at which this group started recording counters.
Group_end_time	The date and time at which this group stopped recording counters.
Model_name	System name of the model.
Model_version	Version of the model. Note that when multiple versions of a model exist in a system, one counter will be generated for each version that is executed within a group.

Model_outcome	Outcome of the model. If different instances of a model complete with different outcomes, there will be one counter for each outcome.
Model_instance_count	Number of instances of the model/version/outcome combination in this group.
Model_total_time	Total time taken by all instances of this model/version/outcome combination in this group (in ms).
Model_min_time	Minimum time taken by any instance of this model/version/outcome combination in this group (in ms).
Model_max_time	Maximum time taken by any instance of this model/version/outcome combination in this group (in ms).
Model_avg_time	Average time taken by all instances of this model/version/outcome combination in this group (in ms).

The columns returned by `v_amc_dep_node_counters` are:

perf_group_id	Same as for v_amc_dep_model_counters.
Server_name	Same as for v_amc_dep_model_counters.
Service_name	Same as for v_amc_dep_model_counters.
Group_start_time	Same as for v_amc_dep_model_counters.
Group_end_time	Same as for v_amc_dep_model_counters.
Model_config_data_guid	Unique identifier for the model. Relates to a column in the deployed model configuration table.
Model_name	System name of the model within which the node was executed.
Model_version	Version of the model within which the node was executed.
Node_config_guid	Identifier of the node usage within a model (unique within the scope of a model).
Node_type	Name of the node type (for example "`DataAccessNode`" or "`ConditionalBranchNode`")
node_name	Name of the node, as it appears within its model.
Node_outcome	Outcome of the node. This column contains an empty string (not a null value) if the node had no outcome. If different instances of a node complete with different outcomes, there will be one counter for each outcome.
Instance_count	Number of instances of this node (within this model/version combination) that completed with this outcome.
Node_total_time	Total time taken by all instances of this node/outcome combination in this group (in ms).
Node_min_time	Minimum time taken by any instance of this node/outcome combination in this group (in ms).
Node_max_time	Maximum time taken by any instance of this node/outcome combination in this group (in ms).
Node_avg_time	Average time taken by all instances of this node/outcome combination in this group (in ms).

It is likely that most reports will make use of one or other of these views, not both. However, if required, it is possible to relate the data in the two views through the model name and version columns.