IS Service Profiler

wrightia®

The Right Integration Attitude

User Guide

Project	**ISSPROF**
Document File	ISSPROF - User Guide.doc
Version	1.0.1
Version Date	12-02-2007
Status	FINAL
Author	António Abreu

TABLE OF CONTENTS

FIGURE INDEX

TABLE INDEX

1 INTRODUCTION

The Wrightia's webMethods IS Service Profiler is a tool to collect and analyze runtime information about Integration Server Services. For the remaining of this document this tool is referenced as Service Profiler.

It is intended to have a small footprint while running, be easy and fast to install and uninstall, and to require no configuration over the Services being profiled.

1.1 Document Objectives

This document is the **User Guide** for the tool.

Its main objective is to be used as the reference for the user of the tool. The guide contains information spanning from installation and setup to advanced usage.

1.2 Target Audience

This document is targeted to any user of the tool, independently of the level of intervention, responsibility or technical ability.

1.3 Pre-requisites

This document will not discuss usability subjects that are specific to the webMethods platform and/or its tools, such as Integration Server administration, Developer, Service details, Package management and usage, etc.

So, a basic knowledge of the Integration Server structure and administration, Package management and Service development is a fundamental requirement.

For the purpose of installation, some basic knowledge of the underlying operating system is also required.

1.4 Environment Requirements & Compatibility

The tool has been constructed to be restricted only by the Integration Server requisites. Therefore, the only environment it needs to run is the Integration Server, and there is no dependency on an existing Broker, Audit DB, Modeler, Developer, Workflow or any other tool.

There is no additional memory requirement above the required by the Integration Server.

The technical specifications for the tool are defined for Hardware & Operating System platform, Integration Server and JVM compatibility. The compatibility can be categorized in a combination of **Expected**, **Tested** and **Available**, as explained in **Table 1**, below.

Table 1 - Compatibility Categorization

Category	Description
Expected	According to the technical specifications and/or documented behavior, it is expected to be 100% compatible.
Tested	The specific version has been successfully tested.
Available	A version for the specific configuration is readily available.

Table 2 - Compatibility Matrix

Item	Version	Expected	Tested	Available	Comment
OS	**Windows 2000 & XP**		☒	☒	
	Windows Server 2003		☐	☒	If the Integration Server is able to run under this OS, there is no reason for the tool not to… and it has been tested with success.
	Sun Solaris	☒	☒	☒	Tested on Solaris 8 & 9.
	AIX	☒	☐	☐	A small, but fundamental, part of the tool is distributed has a native Dynamic Library. Specific OS platform availability is only limited by the existence of the generated binaries, or the existence an ANSI C compiler in the target environment for a 1st time compilation and test.
	HP-UX	☒	☒	☒	Tested on HP_UX v11.11B, both on 32-bit and 64-bit modes.
	Linux	☒	☒	☒	Tested with Red Hat and Suse on Intel processor. Tested also with Red Hat ES 4.0.
IS	**v6.0**	☒	☐	☒	
	v6.0.1 SP2	☒	☒	☒	
	v6.1 FP2	☒	☒	☒	
	v6.1 SP1	☒	☒	☒	
	v6.5	☒	☒	☒	Interface available only through the IS administration pages.
	v6.5.1	☒	☒	☒	
	v6.5.2	☒	☒	☒	
JVM	**1.3.1**	☒	☒	☒	When using a JRE other than the one bundled with the Integration Server, be sure to make the `tools.jar` library available. Default JRE distributions do not include this library. I can be retrieved from the JDK and placed in the `ext` folder of the JRE. The only tested manufacturers were IBM and SUN.
	1.4.x	☒	☐	☒	
	1.4.2	☒	☒	☒	
	1.5.0	☒	☒	☒	
	1.6.x	☐	☐	☐	Not supported.

The Integration Server and JVM version listed in **Table 2** (*above*) have been supported and tested since the first available version of the Service Profiler. However, the same cannot be alleged for the supported Operating Systems. The OSs have been tested and supported has the tool evolved, being introduced in specific tool versions, as stated in **Table 3** (*below*).

Table 3 - Service Profiler version for Operating System

OS	CPU Arch	Profiler version
Windows 2000 & XP	Intel	v1.0
Windows Server 2003	Intel	
Linux	Intel	v1.0
HP-UX	PA	v1.0
Sun Solaris	SPARC	v1.0

2 INSTALLATION AND SETUP

2.1 First-time Installation

The first-time installation assumes that the tool has never been installed or has been completely removed (*see **Uninstalling**, on page 8*).

2.1.1 Installation

With the Integration Server running, start by installing the following IS Packages in the order they are presented:

1. `WiaRoot`;
2. `WiaUtilities`;
3. `WiaServiceProfiler`.

The ZIP filenames have a version suffix appended to its name (*e.g.*, `WiaRoot_r1_0.zip`). Neither the suffixes nor the file extension are indicated on the above list. After installing the IS Packages the installation must be completed by configuring the Integration Server.

Complete the installation on an MS Windows platform

Perform the following steps:

1. Shutdown the Integration Server;
2. Copy the folder `serviceprofiler` from within the `config` folder of the installed `WiaServiceProfiler` IS Package, into the `<wm_dir>\IntegrationServer` folder;
3. Save a copy of the file `server.bat` as `server.org.bat`;
4. Edit the script file `server.bat` and insert all the lines from the `<wm_dir>\IntegrationServer\serviceprofiler\var\server.bat.include` file into the `server.bat` file at the point shown below with a ➡.

```
set CLASSPATH=%PREPEND_SYSTEM_CLASSPATH%;%CLASSPATH%;%APPEND_SYSTEM_CLASSPATH%
set PATH=%PATH%;%IS_DIR%\support\win32;%IS_DIR%\jvm\bin\classic;%IS_DIR%\lib;

rem
rem Run as an NT service ? If so, save program arguments to Registry
rem
if "1%1"=="1-service" (
   if exist LOCKFILE del LOCKFILE
   "%IS_DIR%\bin\SaveSvcParams.exe" /svcname %2 /jvm "%JAVA_DIR%" /binpath "%PATH%" /c
   goto :EOF
)

rem run integration server
title webMethods Integration Server
%JAVA_RUN% -DWM_HOME="%WM_HOME%" -classpath %CLASSPATH% %IS_PROXY_MAIN% "%IS_DIR%"\bin
```

Figure 1 - Point of editing in the `server.bat` file

5. Start the Integration Server;
   Set the License Key.
   When the tool is first installed there is no License Key set and any attempt to perform an action will result in a 'License not set' exception.
   The license is set in the licensing pages of the tool.

Complete the installation on a UNIX platform

Perform the following steps:

1. Shutdown the Integration Server;
2. Copy the folder `serviceprofiler` from within the `config` folder of the installed `N2bServiceProfiler` IS Package, into the *<wm_dir>*`/IntegrationServer` folder;
3. Save a copy of the file `server.sh` as `server.org.sh`;
4. Edit the script file `server.sh` and insert all the lines from the *<wm_dir>*`/IntegrationServer/serviceprofiler/var/server.sh.include` file into the `server.sh` file at the point shown below with a ➡.

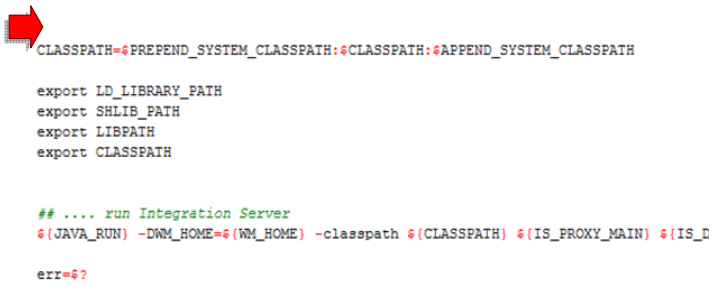

Figure 2 - Point of editing in the `server.sh` file

5. Start the Integration Server;
   Set the License Key.
   When the tool is first installed there is no License Key set and any attempt to perform an action will result in a 'License not set' exception.
   The license is set in the licensing pages of the tool.

Intentionally preventing the Profiler from loading

The Service Profiler can be intentionally prevented from loading by the Integration Server, without the need to uninstall it. This can be achieved by means of putting a file named `ISSPROF_VOID` in the `<wm_dir>/IntegrationServer` folder. The presence of this file causes the loading sequence to be skipped. A direct result is having the functionality links disabled in the tool pages menu.

Figure 3 - Preventing the Service Profiler from loading

IMPORTANT: The functionality menus will be disabled whenever the tool core libraries are prevented from loading… or and error prevented them to be successfully loaded.

IMPORTANT: When installing webMethods upgrades (e.g., Service Packs, some FIXes.) the `server.<bat|sh>` script may be overwritten thus disabling the Service Profiler start-up. When this happens, just repeat the script edit step described above.

2.1.2 Integration Server running as a Windows Service

There is no need for additional settings when using the Service Profiler on an IntegrationServer running as a Windows (NT, W2K, XP) Service.

2.2 Re-installing or Upgrading

Due to the nature of the technology involved and the way it interacts with the Integration Server, not all updates can be installed with the Service Profiler running.

There are libraries being loaded even before the Integration Server starts and that can only be replaced by exiting the JVM completely.

Even though an IS Package has its own class loader and could be replaced with the server running, the `WiaServiceProfiler` contains libraries that are directly loaded by the server class loader and thus cannot be unloaded without shutting it down. If you try to remove, or reinstall, this package using the package management pages it will not be successful… simply because the server will be locking some library files and the package files will not be deleted/moved/replaced.

As a rule-of-thumb, a reinstallation would only assured by first uninstalling the current version. However, this is not always needed and should be referenced in the release notes of the new version or upgrade.

Some patches or FIXes will have their own install instructions, which could be highly simplified and not involving a complete uninstall, not even a server shutdown.

There are a few settings (*summarized in* ***Table 4****, on page 8*) that you might be interested to safeguard before attempting an upgrade:

- Configuration and administrative settings;

  Make a copy of the `.cnf` files in the `config` folder of the `WiaServiceProfiler` package, to some place out of the package structure.

  If they happen to be removed by the (re)installation, copy those files back after the upgrade (a fresh installation does not even create them, but the start-up of the package creates empty versions).

- ACL settings.

  If you have customized the ACL settings you may have to redo those settings in the new installation.

Table 4 - Setting to safeguard before installing an upgrade

Setting	Action
Configuration and administrative	Copy the `packages/WiaServiceProfiler/config/*.cnf` files to a folder outside the package structure. These files contain specific settings customized after installation.
Security	Any ACLs settings customized after installation may have to be configured again in the server administration pages. Check those settings and write them down.

2.3 Uninstalling

Uninstalling the Service Profiler means the complete removal of its components and files. However, if the sole intent is to keep the tool from being loaded and have it performing no work at all, this can easily be accomplished without uninstalling (*please refer to* ***Intentionally preventing the Profiler from loading****, on page 6*).

To completely uninstall the tool:

- Shutdown the Integration Server;
- Revert to Integration Server launch script to the original;

  Copy the saved `server.org.<bat|sh>` onto the `server.<bat|sh>` or just remove the lines inserted upon installation.

  Please refer to the installation procedure for details on the saved file and/or the inserted lines;

- Delete the folder `<wm_dir>/IntegrationServer/serviceprofiler`;
- Delete the folder for the package `WiaServiceProfiler`;
- Start the Integration Server;
- Delete support IS Packages;

  If no longer needed (*e.g.*; not referenced by other packages), delete the packages `WiaUtilities` and `WiaRoot`.

- Remove the `WiaServiceProfilers` ACL.

3 SECURITY AND CONTROLLED ACCESS

The Service Profiler has the access controlled through its own ACL definition: `WiaServiceProfilers`.

By default, during installation, the `Developers` and the `Administrators` User Groups are associated with this ACL.

You can customize the access to the tool pages and services through the association of User Groups to this ACL. However, only users that are also associated with the Administrators ACL can use the **Licensing** pages of the tool.

4 FUNCTIONALITY GUIDE

4.1 Administration Page

This page is accessed through the **Admin** menu entry and contains the editing of settings that affect the overall tool behavior.

- Administration;

  Direct administration actions over the IS Service Profile. These actions are resumed to the **Start** and **Stop** of the tool.

  Started means that the tool is actively collecting information about the currently running IS Services.

  When the package is (re)loaded, the tool is **Stopped** and therefore any of the functionality based on snapshots will issue an (Profiler Not Started) error if used under this condition.

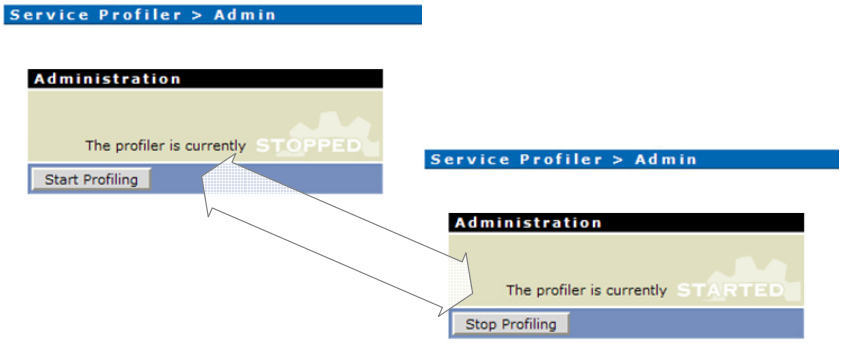

Figure 4 - **Start**/**Stop** the Profiler

The tool needs to be **Started**… and the service counters are reset.

- Settings;

  This form edits property-like settings.

Table 5 - **Administration** / **Settings**

Setting	Description
Freeze Snapshot	Take a snapshot of the current profiling data and hold it in memory, using it as cached value for the next snapshot requests. This setting allows performing all the Data Analysis over the same snapshot, in opposition to always getting a new snapshot every time an operation is performed or a refresh is requested. If activated, the tool continues to gather profiling information about the running Services, but when a snapshot is requested the one cached (*aka*, frozen) is returned. When the tool is **Stopped**, this option is automatically set **ON** to allow the analysis to continue working over the last know snapshot before the **Stop** action. When this setting is **ON**, an icon (*like the one in Figure 5, below*) is shown in the affected pages, working as a visual hint to the source of the data.

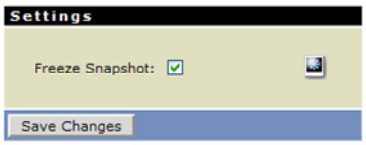

Figure 5 - **Freeze Snapshot** setting

- Package Exclusion Patterns;

  While running, the tool establishes no filters: it collects info for all running Services, because they all contribute to your system performance.

  However, these may not all be under your control or even the object of interest for your analysis. So, you can establish which packages you wish to rule out through Regular Expression patterns over the package names (the result of the applied pattern can be immediately checked, to avoid runtime expression evaluation errors).

Table 6 - **Package Exclusion Patterns** form fields

Field/Item	Description
New Pattern	New Regular Expression Pattern to add or test. When the [Save Changes] button is clicked this expression is added to the list.
Check	Click on the icon to see the list of packages it defines.
Remove	Indicate which **Name Patterns** are to be removed from the list by checking the corresponding checkbox. When the [Save Changes] button is clicked all checked expressions are removed from the list.

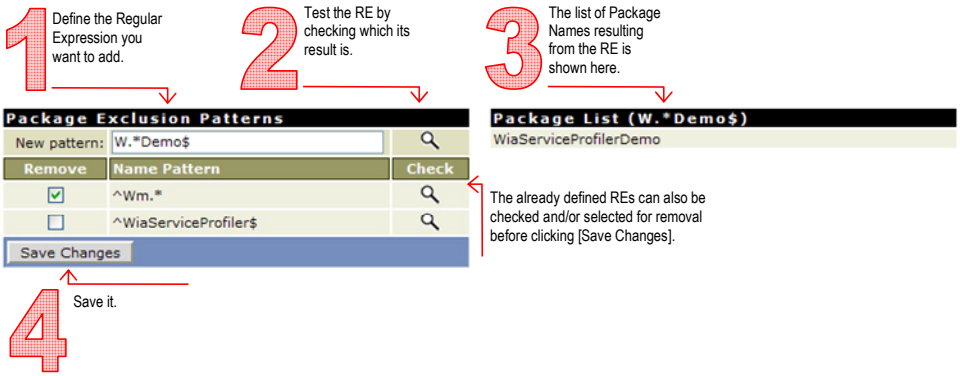

Figure 6 - Managing Package Exclusion Patterns

4.2 Licensing Pages

This functionality is only accessible to users associated with the Administrators ACL

The entry page shows the current licensing information (**Expiration Date**, **Status**, etc.) and can be used to set a new valid License Key.

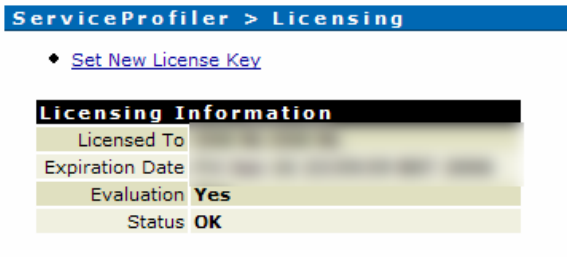

Figure 7 - Licensing main page

IMPORTANT: Take notice that the pointed functionality option is **Set New License Key** and not **Edit License Key**. There is a difference: for the first, the current License Key is never shown.

The name of the customer for whom the License Key was issued to is also part of the license information.

4.3 View Per Service

The **View Per Service** dashboard page (*see **Figure 8**, below*) is a sample Analysis Tool which builds a view/report of the generated hierarchical snapshot as a flat representation, accumulating per Service.

Dashboard > View Per Service

Snapshot Identification

Sample Key	Server Name	Begin Time	End Time
WIA-AA:5555-20061108182045	WIA-AA:5555	08-11-2006 18:20:45.758	08-11-2006 18:52:38.138

Snapshot Details per Service

Minimum Own Elapsed Time (D HH:MM:SS.m): 0.100

Reported Times are averages: ☐

Refresh

			Count		Own Code		With Child Code	
Package	Service	Type	Calls	Errors	Elapsed	Spent	Elapsed	Spent
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc2:createDocument		53	53	1:03.308	0.100	1:03.308	0.100
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc1:singleService		12	12	18.177	0.050	28.550	0.150
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc3:looping		1	0	1.811	1.452	22.132	16.374
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc3:createStringList		8412	0	2.245	1.602	5.429	3.856
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc1:looping		1	1	0.141	0.000	0.491	0.020
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc3:order		8412	0	10.444	8.012	20.341	14.921
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc2:twoLevels		53	53	53.207	0.170	1:56.545	0.270
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc3:orderStringList		8412	0	0.522	0.401	0.522	0.401
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc5.amabr_issue_nr9_svcPortType:queryEntity		15	0	0.130	0.010	2.151	0.090
WiaServiceProfilerDemo	wia.demo.issprofiler.ui.run_uc:getUseCasesMetadata		109	0	0.211	0.040	0.221	0.050
AA_Issue_9_DB_Connections	amabr.issue.nr9.DB:newInstance		1	0	0.581	0.010	0.581	0.010
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc3:orderNumericList		8412	0	1.394	0.991	1.394	0.991
AA_Issue_9_DB_Connections	amabr.issue.nr9.DB:getName		14	0	0.150	0.000	0.150	0.000
WiaServiceProfilerDemo	wia.demo.issprofiler.usecases.uc2:looping		1	1	0.251	0.000	0.541	0.010
WiaServiceProfilerDemo	wia.demo.issprofiler.ui.run_uc:triggerUseCase		108	0	0.170	0.040	0.210	0.080
WiaServiceProfilerDemo	wia.demo.issprofiler.util:randomString		168240	0	3.184	2.253	3.184	2.253
WiaServiceProfilerDemo	wia.demo.issprofiler.config:get		9697	0	0.120	0.120	0.120	0.120
WiaServiceProfilerDemo	wia.demo.issprofiler.ui.run_uc:getMenuMetadata		2	0	0.461	0.000	0.461	0.000
		Grand Totals:			2:36.507	15.252	4:26.331	39.697

Figure 8 - The **View Per Service** report

The source gathered data is an invocation tree where the same Service may be called from different parents, but this particular view disregards the service interrelations and sums all values for each individually identified Service into a single accumulator set.

All timing values are displayed in the data format explained in **Figure 9** (*below*) with the leading zeros suppressed with exception to time value smaller than one second.

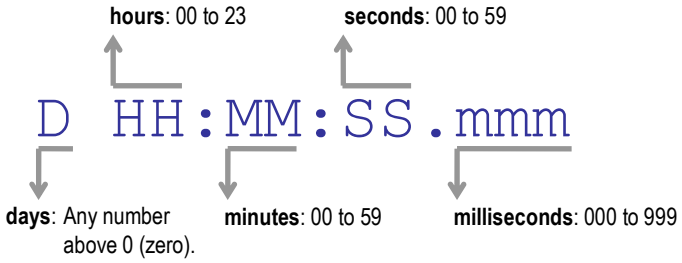

Figure 9 - Timing display format

The table can be sorted by any of the detail columns by clicking on the corresponding column header. Clicking multiple time toggles between ascending and descending sort order. Default (first click) sorting order depends on the data type: numeric and time columns start by being sorted descending while the others are ascending.

The information is laid in two separate tables:

- One for the **Snapshot Header**;
  This table contains data that identifies the snapshot,
- One for the **Snapshot Details**.
  This table contains the information about each Service found in the snapshot, on an *one-Service-per-line* layout.

Table 7 - **View Per Service** table for **Snapshot Header**

Snapshot Header	Description
Sample Key	A unique key that identifies the current snapshot.
Server Name	The net name of the Integration Server host.
Begin Time	The timestamp of when the profiling was started.
End Time	The timestamp of when the snapshot was requested.

Table 8 - **View Per Service** table for **Snapshot Details**

Snapshot Details	Description
Package	The name of the Package where the profiled Service is defined.
Service	The fully-qualified runtime name of the profiled Service.
Type	The Service Type: • Flow Service • Java Service • Adapter Service • C/C++ Service • Webservice • XSLT Service[1] • Warning! Service name not found. This may happen if the package is disabled or the Service is renamed during the profiling period. The service type icon is also a link to the **Browse Snapshot** page (*see page 17*), where the occurrences of the corresponding service will be automatically highlighted. The link is made on the icon, and not on the Service column, to save horizontal space on the page: due to webMethods styling for links, their text appears as bold and this would widen considerably the Service table column.
Call Count	The accumulated number of times the Service has been invoked. The Service can be called from a number of other Services and has a Call Count for each of those calling nodes. However, this value sums all those counters.
Error Count	The accumulated number of exceptions the Service has raised. The Service can be called from a number of other Services and has an Exception Count for each of those calling nodes. However, this value sums all those counters.
Own Code	Timings for the profiled Service Own Code. This means that is does not include the time spent in Services it calls, only the time it took executing its own functionality code.
With Child Code	Timings for the profiled Service Own Code plus the time spent in Services it calls. The **With Child...** columns have defined a tip label showing the minimum and maximum timing values the corresponding Service took on a single call. Just place the mouse cursor over the table cell and the tip will appear for a short while.
Elapsed	Clock time passed from the beginning of the call until it ends.
Spent	Actual time spend in the CPU doing work.

Additionally, there are filtering options that further limits the amount of data when the page is refreshed, by means of clicking on the [Refresh] button. These options are:

- **Minimum Own Elapsed Time**;

  Do not show Services that have an **Own Elapsed Time** less than the established value.

  This allows you to concentrate the analysis only on the Services that take the longest... or significant timings.

- **Reported Times are averages**.

  When checked, the values of **Elapsed** and **Spent** are the calculated averages for the **Call Count**.

4.4 Code Coverage

The **Code Coverage** report (***Figure 10**, below*) is another example of an Analysis Tool.

It presents a view over the Snapshot data, reporting the percentage of Services in a Package that have effectively been run… while the profile was active.

[1] This kind of service was introduced with v6.5 of the webMethods Integration Platform.

The results are based solely on the number of Services for each package present in the snapshot against the universe of services defined by their respective packages. From this, the percentage of services used per package is calculated.

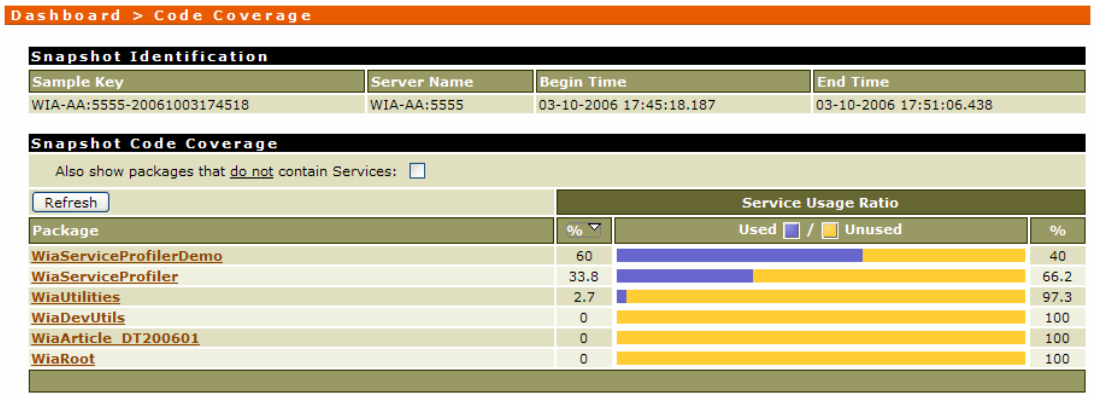

Figure 10 - The **Code Coverage** report

The table can be sorted by clicking on the corresponding column header on any of the detail columns, with exception to the percentage bar. Clicking multiple times toggles between ascending and descending sort order.

Each package name (that contains Services defined) is presented as a link to a new page where the complete list of Services from that package is presented, together with an indication on whether is has been used/invoked or not while the profiler was **Active** (*see **Figure 11**, below*).

Additionally, there is a filtering option that changes the amount of data when the page is refreshed, by means of clicking on the [Refresh] button. The option is:

- **Also show packages that do not contain Services**.

  When checked, the entire list of packages, even if they do not have any Services defined, is shown.

  This is just a simple option to allow viewing the entire universe of Package names. The Packages that do not contain Services are not available as links for drilling-down.

Dashboard > Code Coverage > Package Services

- Back to Code Coverage

Services From Package 'WiaServiceProfilerDemo'

Service Name	Used?
wia.demo.issprofiler.usecases.uc1:looping	Yes
wia.demo.issprofiler.config:get	Yes
wia.demo.issprofiler.usecases.uc1:singleService	Yes
wia.demo.issprofiler.usecases.uc3:order	Yes
wia.demo.issprofiler.usecases.uc3:createStringList	Yes
wia.demo.issprofiler.util:randomString	Yes
wia.demo.issprofiler.usecases.uc3:orderStringList	Yes
wia.demo.issprofiler.usecases.uc3:orderNumericList	Yes
wia.demo.issprofiler.usecases.uc3:looping	Yes
wia.demo.issprofiler.ui.run_uc:getMenuMetadata	Yes
wia.demo.issprofiler.usecases.uc2:looping	Yes
wia.demo.issprofiler.usecases.uc2:twoLevels	Yes
wia.demo.issprofiler.usecases.uc2:createDocument	Yes
wia.demo.issprofiler.ui.run_uc:getUseCasesMetadata	Yes
wia.demo.issprofiler.config:list	Yes
wia.demo.issprofiler.ui.run_uc:triggerUseCase	Yes
wia.demo.issprofiler.config:set	Yes
wia.demo.issprofiler.util:doThreadInvoke	Yes
wia.demo.issprofiler.usecases.uc2:zip	No
wia.demo.issprofiler.admin:initialize	No
wia.demo.issprofiler.usecases.uc2:unzip	No
wia.demo.issprofiler.util:getThisPackageHomeDir	No
wia.demo.issprofiler.config:loadConfig	No
wia.demo.issprofiler.util:sleep	No
wia.demo.issprofiler.usecases.uc2.wsc:processEntry	No
wia.demo.issprofiler.usecases.uc2.wsc.uc2:processEntry	No
wia.demo.issprofiler.usecases.uc2:sleep	No
wia.demo.issprofiler.admin:startup	No
wia.demo.issprofiler.usecases.uc2:random	No
wia.demo.issprofiler.util:randomNumber	No

Figure 11 - The **Code Coverage** service usage detail

4.5 Browse Snapshot

This Analysis Tool simply shows the Snapshot tree of called Services.

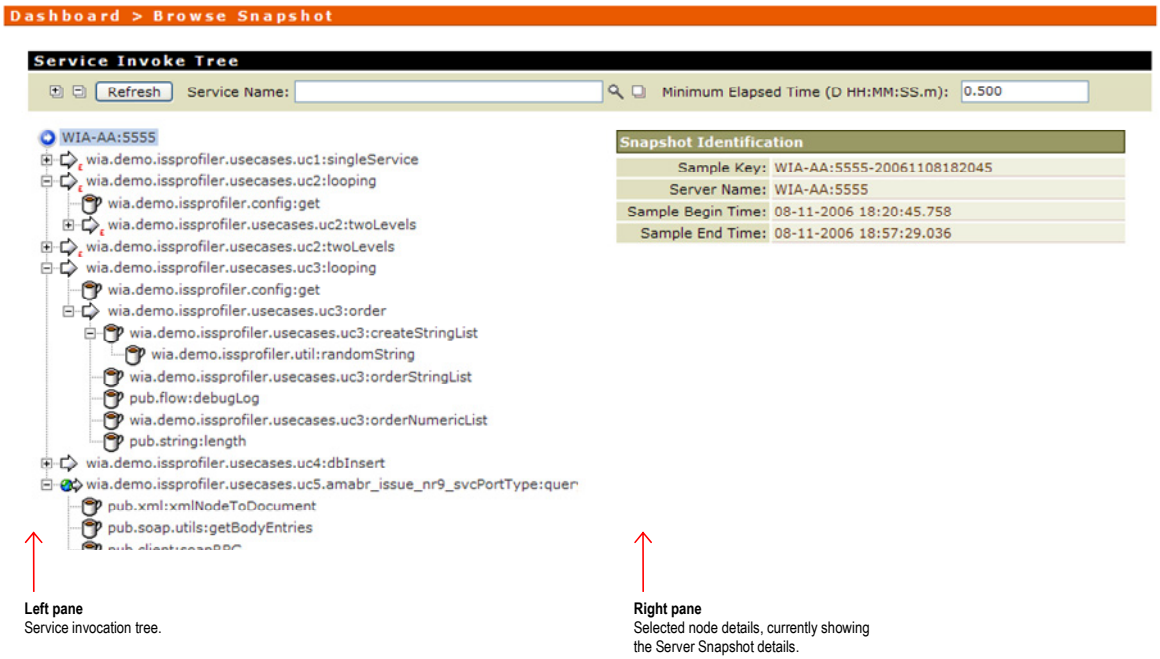

Figure 12 - The **Browse Snapshot** page

Placing the mouse cursor over the node name will show a tip with context summarized information. By clicking on the node name, contextual detail information is shown in the panel on the right:

- The Integration Server node stems snapshot identification data (*see Figure 12, above*);
  The nature of this information is the same as the one presented on the other Analysis Tools.
- A service node stems details about that Service in that calling context (*see Figure 13, below*);
  The **Accumulated With Children** table presents timings that include the service own timing plus the timings of all the services in its child nodes.

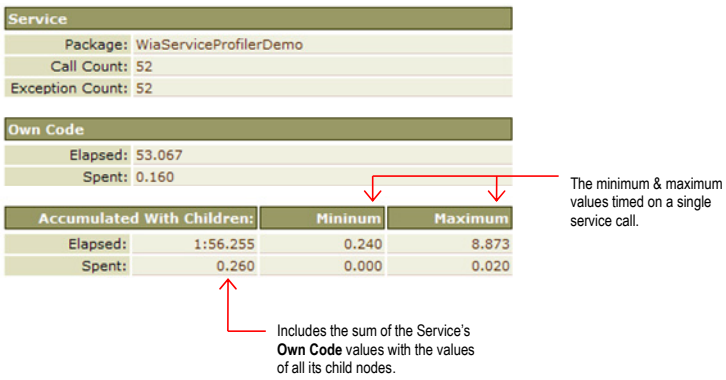

Figure 13 - The **Browse Snapshot** example of detail for a selected Service tree node

There are no sorting options: the root nodes are always ordered ascending.

The **Package Exclusion Patterns** (*see **Administration Page**, on page 10*) and the **Minimum Elapsed Time** filtering options are only applied to the root nodes.

Table 9 - **Snapshot Browser** options

Option	Description
	Expand all tree nodes.
	Collapse all tree nodes.
Service Name	Locates all occurrences of a service name in the call tree. Enter a fully qualified service name and press  . The tree is expanded to where the service is found and the tree node(s) is(are) highlighted (but not selected).
	Clear all highlighted tree nodes (does not deselect any currently selected tree node).
Minimum Elapsed Time	Applied only to root services. Only shows root nodes for which the Elapsed Time is above the stated value. The filter is applied by pressing the [Refresh] button.

On the tree nodes, which represent **Service Calls**, there may appear some visual hints overlaid to the node icon that give it some extra meaning. These tree node visual hints are explained in **Table 10** (*below*).

Table 10 - **Snapshot Browser** node icons visual hints

Tree Node Hint	Description
	False root. This can only occur on the node that is representing a Top Level Service or Root Service, *i.e.*, a Service at the top of the tree. A **False root** occurs when the Service Profiler is started, a service running path is already on the run and the first collected data happens in the middle of the call stack: for the Service Profiler this is a root because its has no recollection of the services up in the call stack. However, it is recognizably not a true Top Level Service because its parent is not `null`.
	This hint indicates that that Service has raised exceptions. Select it to view its details, where an exception count is also presented.

IMPORTANT: The **View Per Service** reports accumulated timings per service independently from its position in the calling tree. This may induce timing differences for a specific node by comparison because that service may be called by different parents which may even be currently filtered out.

IMPORTANT: Some service nodes that are presented in the root may in fact not be top-level services. They may happen to be on the root node just because they were first scanned after the time when the profiler was started.

4.6 Export To File

This functionality allows exporting the current Snapshot to a file.

When you choose the file format/type, adequate options appear corresponding to that file format/type.

When the [Generate File] button is clicked, the file generation process is started. Upon completion, and on success, you are presented with a new page with a file name link.

The link is intended to be used as a Save As... option to save it to your local disk.

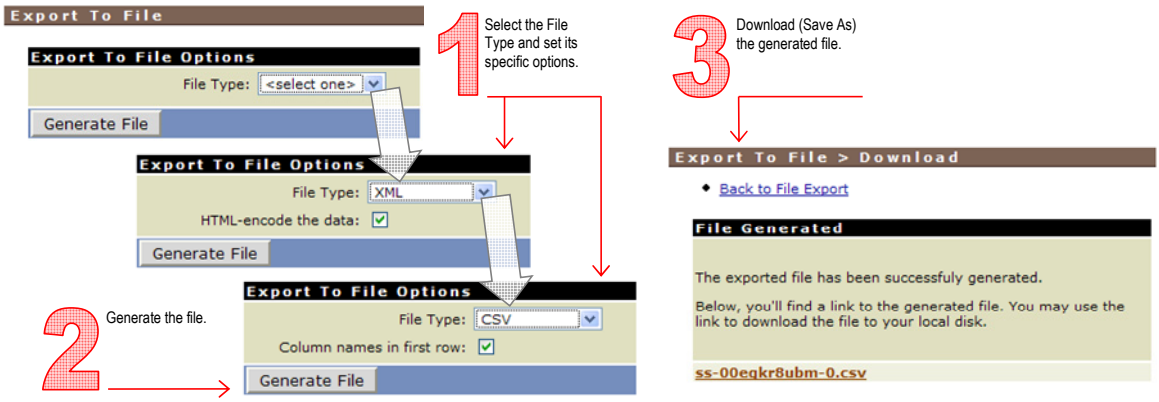

Figure 14 - Exporting Snapshot as a file

4.6.1 CSV Format

The standard CSV (Comma Separated Values) format is widely know but still leaves some minor details for the implementation. Therefore, the Service Profiler implementation fixes these rules:

- The value separator is always the comma (",");
- The **Service Name** includes the package name as a prefix separated by a forward-slash ("/");
  For example: `WiaServiceProfiler/wia.pub.issprofiler.analyze:viewPerService`
- Only values that include the value separator are quoted;
  When this happens the value is enclosed in double-quotes.
- Header names are not quoted;
  Header names are optional, and are not quoted in accordance with the rule to quote values.
- The record separator used is dependent of the Operating System the Integration Server is running under;
  The record separator is the new-line sequence used for normal text files, following the rules for the hosting Operating System.
- Column header names are:

```
Node Id
Parent Node Id
Sample Key
Server Name
Sample Begin Time
Sample End Time
Service Name
Call Count
Exception Count
Elapsed milliseconds
```

```
CPU nanoseconds
Max Elapsed milliseconds
Min Elapsed milliseconds
Max CPU nanoseconds
Min CPU nanoseconds
```

4.6.2 XML Format

The produced XML is syntactically formatted according to standard rules, which includes starting the file with the line:

```
<?xml version="1.0" encoding="UTF-8"?>
```

The one specificity that the produced XML contains is the used dictionary. Because the API to the core functions already uses XML to provide the Snapshot, the same structure is used here, both for simplicity and enhanced performance.

As an option, the data values can be HTML-encoded entities, *i.e.*, where values contain characters that are used by the tagging rules or may create incompatibilities these characters are encoded in the form `&<code>;`. Depending on the application that will be the target of the produced XML, decoding these entities may be automatic or not.

Table 11 - Snapshot XML dictionary

Tag	Level	Attribute	Comment
service-counters	0		Main enclosing tag.
		sample-key	Snapshot ID.
		server-name	Integration Server net-name (or IP address) and port number.
		sample-begin-time	Timestamp of when the profiler started collecting data, formatted as "dd-MM-yyyy kk:mm:ss.SSS".
		sample-end-time	Timestamp of when Snapshot was taken, formatted as "dd-MM-yyyy kk:mm:ss.SSS".
invocation	1 .. n		
		service	Name of the Service. It contains the name of the Package has a prefix separated by a forward-slash, for example: MyPackage/MyAPP.util:getMaxValue
		calling-service	The name of the parent Service, *i.e.*, the service that called this one. "null" if there is no registry of what Service called this one[2].
		call-count	Number of times this Service was called… in the context of this call path node.
		exception-count	Number of time this Service raised an exception.
		elapsed-ms	Sum of the elapsed milliseconds for the calls in the context of this call path node.
		cpu-ns	Sum of the CPU spent nanoseconds for the calls in the context of this call path node.
		min-elapsed-ms	Reference value. The minimum timed value on a single service call.
		min-cpu-ns	
		max-elapsed-ms	Reference value. The maximum timed value on a single service call.
		max-cpu-ns	
child-invocations	n+1		If defined, contains the invocations to Services from the current Service.
invocation	…	…	Recursive…

[2] A Service that is not called by another Service is generally called a Top-Level-Service or a Root-Service, and these will appear at the top of the tree. However, depending on when the profiler is started, the Service path may be intercepted already on-the-go producing the registration of Service invocations without the register for their calling Services, thus generating false top level services, *aka* **False-Root** or **Faux-Root**.
It is not difficult to make a distinction between a true top level service and a false one: if a Service is on the top-level and its parent Service is not "null"… it is a False-Top-Level-Service.

5 EXTENSIBILITY

IMPORTANT: The information presented in this section may change in the future to accommodate the evolution of the tool.
These changes will eventually be additions to the presented structures and extensibility mechanisms which, to the extend of the possible and reasonable, will be compatible with previous versions of the same mechanism. However, this compatibility compromise will be superseded when it impairs the progression of the tool evolution/betterment.

5.1 Overview

The Service Profiler core functionality is to gather raw information about the running Services. From that information, Analysis Tools may be implemented. The tool already includes some tools of that nature. However, to avoid limiting the analysis of the data to the tool's included functionality, external accessibility to the snapshot data is provided.

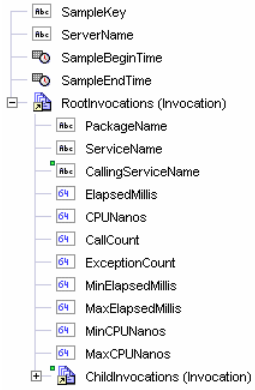

Figure 15 - The Snapshot as a Document structure

A Snapshot is a tree structure (*see **Figure 15**, above*) with common information on the root note plus the entire top-level Services, each of them being a node and with (possible) child nodes in them representing the Services they have called. These child codes can also have their own child nodes in them representing the Services they have called… and so on.

The Snapshot can currently be exported as an XML file or a CSV (*see **Export To File**, on page 18*).

Beyond the export as file feature, and closer to the Integration Server environment, the Service Profiler provides a public interface (API) of Services and Document Types that can be used by any external IS Service to request Snapshots and process them.

The term public determines that the interface exposes controlled functionality and that the retro-compatibility of that interface will be taken into consideration when upgrades are issued, but is not guaranteed. The later may be justified by the integration of new and incompatible functionality, but whenever it may occur, it will be clearly described in the Release Notes.

5.2 Public API

The Service Profiler specific functionality is enclosed in the `WiaServiceProfiler` package.

The `WiaRoot` and `WiaUtilities` are also distributed as part of the installation, but these are support packages that are shared by other Wrightia products. The first contains no usable components but the latter contains some generic use services, which are not object of this documentation. However, the included services are fairly self-commented, through the use of Service and Field comment entries.

The `WiaServiceProfiler` package public interface is rooted on the `wia.pub.issprofiler` folder. To execute the enclosed services, the user must be part of the WiaServiceProfilers ACL. These components are fairly self-commented, through the use of Service and Field comment entries.

The `wia.pub.issprofiler.ui` folder contains the functionality that serves exclusively the product DSPs. These are made public for the sole reason that they provide the exact same information that is viewed on the corresponding pages and are therefore not object of the current documentation.

5.2.1 Documents

The several existing views of a Snapshot are defined as Documents, which are used as input and/or output parameters of the Services on the public API.

On **Table 12** (below) are listed the public interface Documents.

Table 12 - Documents defined as public

wia.pub.issprofiler.docs		
Interface	**Document Name**	**Description**
Take Snapshot	`ServiceCounters`	This document represents a complete, raw, Snapshot as provided by the profiler core. This document is a translation to IData from the native XML dictionary… not a direct conversion of tags and attributes.
	`Invocation`	This in the representation of a single Service Invocation node of the `ServiceCounters` document.
View Per Service	`ViewPerService`	The snapshot represented as a **View Per Service** result.
	`InvocationAccumulator`	`ViewPerService` child document representing an accumulator for a particular Service.
Code Coverage	`CodeCoverage`	The snapshot represented as a **Code Coverage** result.
	`PackageCoverage`	`CodeCoverage` child document representing the coverage statistics for a particular Package.
Browse Snapshot	`BrowseSnapshot`	The snapshot represented as a **Browse Snapshot** result.
	`BrowseNodeAccumulator`	`BrowseSnapshot` child document, representing a Service node in the invocation path where its invocation statistics are kept.
Export To File	`ExportedFile`	The link representation of the result of an **Export To File**.

For details on the structure of each of these documents, please use Developer (or any other capable tool) to view the Document's field comments.

5.2.2 Services

This section describes the public API Services.

IMPORTANT: The public API *per se* does not include Services for administrative purposes. However, that functionality can be accessed thrown the UI Services for the administration pages.

Table 13 - Snapshot request Services

wia.pub.issprofiler.snapshot:	
Service Name	**Comments**
take	Take a Snapshot of the current (possibly frozen) counters.
freeze	Takes a snapshot and freezes it in memory, caching for the next requests. Doesn't need to check if already frozen...
unfreeze	Unlock freeze settings, allowing each new request to get the most current snapshot. Doesn't need to check if already unfrozen: if not frozen, it does nothing.

Table 14 - Out-of-the-box Snapshot Analysis Services

wia.pub.issprofiler.analyze:	
Service Name	**Comments**
viewPerService	Analyzes the statistical Service Counter tree structure and produces a flat (array) of accumulators: one element per service. Optionally, the accumulator list can be filtered to only include those that are equal or above a given **Minimum Elapsed Time** value, and/or timings be generated as averages related to the **Call Count**.
codeCoverage	Analyzes the statistical Service Counter tree structure and produces a **Service Code Coverage** assessment for the snapshot. The analysis result is produced only for a given list of package names. Optionally, the resulting list may contain Packages that do not hold Services.
browseSnapshot	Analyzes the statistical Service Counter tree structure and produces accumulators per tree node. Optionally, the accumulator list can be filtered to only include those that are equal or above a given **Minimum Elapsed Time** value.

Table 15 - Export Snapshot to file Services

wia.pub.issprofiler.export:	
Service Name	**Comments**
to_csv	Takes the current Snapshot and exports it to a Comma Separated Values (CSV) formatted file. The Snapshot is automatically taken by this service as part of the service workflow, and it is thus affected by global filtering settings and the snapshot freeze setting. See **CSV Format**, in page 19, for a reference of the CSV data structure.
to_xml	Takes the current Snapshot to a XML formatted file. The XML dictionary is the one used internally by the tool to provide the current Snapshot. The Snapshot is automatically taken by this service as part of the service workflow, and it is thus affected by global filtering settings and the snapshot freeze setting. See **XML Format**, in page 20, for a reference of the XML data structure.

Table 16 - Utility Services

wia.pub.issprofiler.util:	
Service Name	**Comments**
`getPackageNames`	Get the current list of package names, based on the complete list of Packages and the current **Package Exclusion Patterns** settings.

For details on the input and output parameters for the listed Services, please use Developer to view their signature definitions and the comments filled for the fields. It's bound to stand true that those comments are more complete and up-to-date than any detached documentation.

6 TROUBLESHOOTING

Consult this section prior to asking for support.

This section describes known solutions to known situations caused by common pitfalls.

A listed effect may be triggered by a non-listed cause and thus not having a description of a known sequence of actions to overcome it. When this happens, please report it to support.

The Licensing menu option does not appear

This is not the case where the menu option exists but is disabled… in this case the menu option is not listed at all.

On some installations, this may happen on the first time the Service Profiler pages are accessed: refreshing the page usually makes the link appear.

However, only users with `Administrators` ACL can see this link: make sure the user you are using obeys to this definition.

(Access Denied) Cannot login onto the Service Profiler pages

The reason for this kind of problem is manifold. It can be a problem with the browser itself not being able to let go of the session status. Close all browser windows and try again.

Occasionally, it may happen after the installation of an upgrade of the tool. Due to the incomplete or mixed settings of the ACLs caused by things being removed while others are added, the loaded settings do not match the runtime environment. Delete the `WiaServiceProfilers` ACL and reload the `WiaServiceProfiler` package. The security settings are reset to the default (*refer to* ***Security and Controlled Access****, on page 9, for details about which are these default settings*).

Functionality menu options appear disabled

Basically, the menu options are disabled if the core libraries are not loaded.

This can happen in the following known situations:

- The installation is not complete;

  When the Service Profiler package is installed just via the IS Admin pages, the menu options are found disabled because the installation is not complete (*see* ***Installation and Setup****, on page 5 for further details*).

- A file named `ISSPROF_VOID` exists in the `<wm_is>/IntegrationServer` folder;

  Delete the file.

  It's not automatically created. It's a manual mechanism to prevent the Service Profiler from loading without the need to uninstall.

- **LIB**rary paths and **CLASSPATH**s are not correctly mounted or do not include the Service Profiler directories;

  The directories have been removed or the `server.<sh|bat>` script has been replaced or edited.

Check is there was any webMethods upgrade or (re)installation performed recently. These are known to sometimes replace the server launch script.

- The Operating System platform is not supported;

  The Service Profiler libraries depend on native shared libraries which are available only for a limited list of operating systems & CPU architectures. If the expected shared library is not found the libraries will not load correctly and the menus are kept disabled.

  Depending on the Service Profiler version, native code loading errors may not appear on logs and the list of supported platforms may differ.

  (See ***Environment Requirements & Compatibility***, on page 3, for further details)

- File permissions;

  This may happen in UNIX environments. However, it will rarely occur because Service Profiler shell scripts set the needed execution permissions.

  Nevertheless, depending on how the files have been copied, who is the owner and what user is being used to start the Integration Server… some glitch may happen and the files do not get the needed execute permissions.

- Errors during launch.

  Check the console, server logs and the `<wm_is>/IntegrationServer/serviceprofiler/logs/issprof.log` file for errors during the launch.

  To specifically know if an error occurred while loading the native libraries, you can use the Developer tool and run the service `wia.issprofiler.admin:getStatus`. If an exception was raised, a variable named `stackTrace` will be placed on the pipeline containing the stack trace for it. Please pick its contents and report to support.

Timeout or long-running script alert on browser

With very large snapshots (1000, 2000 or more service nodes), the browser may stop the page loading with an alert/warning that the running JavaScript it taking too long. The message and the conditions for raising this warning vary from browser-to-browser and from environment-to-environment.

On the environment side of the issue, and again depending on the browser, little available CPU and RAM may increase the chance of raising the alarm.

The alert message is a security protection that works differently depending on the browser: for instance, while IE evaluates the number of executed script statements; Firefox evaluates the number of elapsed seconds. If the number of service nodes in the snapshot is border-line with the limits the browser evaluates, tweaking its setting may successfully eliminate the alert. But if the number of service nodes is too high (above 3000) it will have to be a job for the Service Profiler tool itself to overcome the limitation (read further).

Assuming that these are the most used browsers, some tests with around 3000 service nodes have been made with them, and so providing some performance and tweaking info:

Table 17 - Script timeout browser related info

Browser	Result
Internet Explorer 6	The slowest in the test, both 0n page loading and script execution. With 3000 nodes, the problem happens on every page load. **Open All** and **Close All** are very slow but do not raise the alarm. IE has a way of increasing the threshold of when the alarm is raised (http://support.microsoft.com/kb/175500). Only with a setting of **50,000,000** was possible to eliminate the alert. Browser restart is needed. No tests have been done with IE7.
Firefox 2.0	Highest size in memory. With 3000 nodes, the problem happens on every page load and while opening all nodes (but only on the first time). **Close All** is faster than **Open All**, but the latter is only slow on the first time it is called. Firefox has a setting to change the script timeout, in seconds: open the page `about:config` page and change the `dom.max_script_run_time` value. The alert was suppressed by setting its value to **20**. No browser restart is needed.
Opera 9	The fastest in the test, both on page loading and script execution. The smallest size in memory. With 3000 nodes, no problem ever arises. With 4000 nodes, no problem ever arises, apart from taking a little longer to load. However, in spite of the increased load, the **Open All** and **Close All** operations are quite fast. Opera also has a configuration setting page (`opera:config`), but no setting related to a script timeout was identified.

However, if the number of service nodes on the snapshot is above 1000, the **Browse Snapshot** and **Browse Running Services** dashboards automatically set a paging control for the tree of nodes. The setting of **1000** nodes is the default, but can be set to a different value by entering the `issprof.browsesnapshot.nodesperpage` key on the `issprof.cnf` file:

```
…
issprof.browsesnapshot.nodesperpage = 2000
…
```

The setting change is done by editing the properties file with a text editor and entering a new line as demonstrated above. The properties file is located in the `config` folder of the tool's package.

The first time the value is entered in the properties file, no further action is needed. However, if the setting is afterwards changed, a package reload is required.

A **limitation** exists, however:

- If the number of nodes in a Snapshot triggers the tree pagination in the dashboard, the **find** a node in the tree is limited to the current page.

  On the special case of the navigation link from the **View Per Service** to the **Browse Snapshot**, the navigation & find is always (and only) made onto the first page. However, jumping to the next/previous page and repeat the find is always possible, but hits on the entire snapshot are not seen all at once.

Profiler not correctly initialized: E01

This error will only happen if the Java instances of the tool classes are remove from memory in runtime… for whatever reason… and should never happen.

To correct the situation you can either reload the tool package of just run the `wia.issprofiler.admin:startup` service from the Developer tool or using a browser to do an HTTP service invoke.

7 FURTHER READING

You may find useful the reading of the Profiling Whitepaper. This document is provided separately and the latest version can be requested directly to Wrightia, free of charge. Rather than providing additional information about the tool, the whitepaper focuses on profiling in general, the usefulness of a Profiler and the reasoning for using a tool such as the Service Profiler.

INDEX

APPENDIX A GLOSSARY

Item	Definition
Analysis Tool	A dashboard view of the Snapshot data organized in a way that reveals relations on the collected data, either interactively or in the form of a report.
False Root	A False Root Service is one that is presented at the top of the call tree but because that's when the Service Profiler has first collected it, but in fact, it is being called by another Service… not collected by the Service Profiler. See also Top Level Service.
Faux Root	See False Root.
Own Code	In opposition to **Whit Child Code**, this set of timings relates solely to the code that, in itself, is the body of the subject Service. This excludes times measured while in any Services the current one calls as part of its designated functionality.
Snapshot	A Shapshot is an instant sample of the data structures internal to the Service Profiler.
Time Elapsed	This is the absolute time that has elapsed as measured by an external chronograph, or even on your wrist watch.
Time Spent	In opposition to **Time Elapsed**, this value measures only the time the Service was in the CPU doing real work. The CPU is being used by more than one thread. The scheduler slices the availability of the CPU by giving each thread an opportunity to run. This value counts the time the Service spend effectively using the CPU. A Service may have a Time Elapsed much larger that the Time Spent if it is given little opportunity to run. This can be caused by: • A thread with much higher priority hogging the CPU availability; • A badly dimensioned system with not enough threads allocated, resulting in an activity of context switching; • Not enough resources (RAM, network bandwidth, etc.) causing the Service to spend most of the time waiting; • Etc.
Top Level Service	A Service that is not called by another service, thus being at the top of the call tree. This kind of service may also appear referenced as Root Service.
Root Service	See Top Level Service.
With Child Code	This set of timings corresponds to the current **Service Own Code** plus any times measured while in any Services the current one calls as part of its designated functionality.
…	…