Monitoring Microsoft Project Server

eG Enterprise v6.0

Restricted Rights Legend

The information contained in this document is confidential and subject to change without notice. No part of this document may be reproduced or disclosed to others without the prior permission of eG Innovations Inc. eG Innovations Inc. makes no warranty of any kind with regard to the software and documentation, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Trademarks

Microsoft Windows, Windows NT, Windows 2003, and Windows 2000 are either registered trademarks or trademarks of Microsoft Corporation in United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Copyright

Table of Contents

Table of Figures

1

Monitoring Microsoft Project Server

Microsoft Project Server 2010 is built on Microsoft SharePoint Server 2010, bringing together powerful business collaboration platform services with structured execution capabilities to provide flexible work management solutions. Project Server 2010 unifies project and portfolio management to help organisations align resources and investments with business priorities, gain control across all types of work, and visualise performance using powerful dashboards.

Project Server 2010 innovative project and portfolio management capabilities provide:

- **Streamlined project initiation and enhanced portfolio analytics:** Simply capture all project requests in a central repository, standardise data collection and business case development and drive accountability through flexible governance workflows. Enforce a rational vs. emotional approach to investment selection and effectively identify project portfolios that best align with your business strategy under varying cost and resource constraints
- **Improved productivity with better collaboration:** Effectively connect disparate teams and easily share information to enhance collaboration and improve project success rates. Project Server 2010 is built on SharePoint Server 2010, bringing together powerful business collaboration platform services with structured execution capabilities to provide flexible work management solutions.
- **Familiar and powerful business intelligence services:** Experience the strength and depth of Microsoft Business Intelligence platform. Easily create reports and powerful audience based dashboards with Excel Services, PerformancePoint Services, Visio Services and SQL Reporting Services and more. Quickly gain visibility, control, and insights across your project portfolios
- **Easy-to-use, connected, and scalable platform:** Get started quickly and reduce training costs with the new intuitive server ribbon. Project Server 2010 connects with related Microsoft technologies to provide a familiar and unified work management platform, ensuring you can gain additional value from the Microsoft platform.

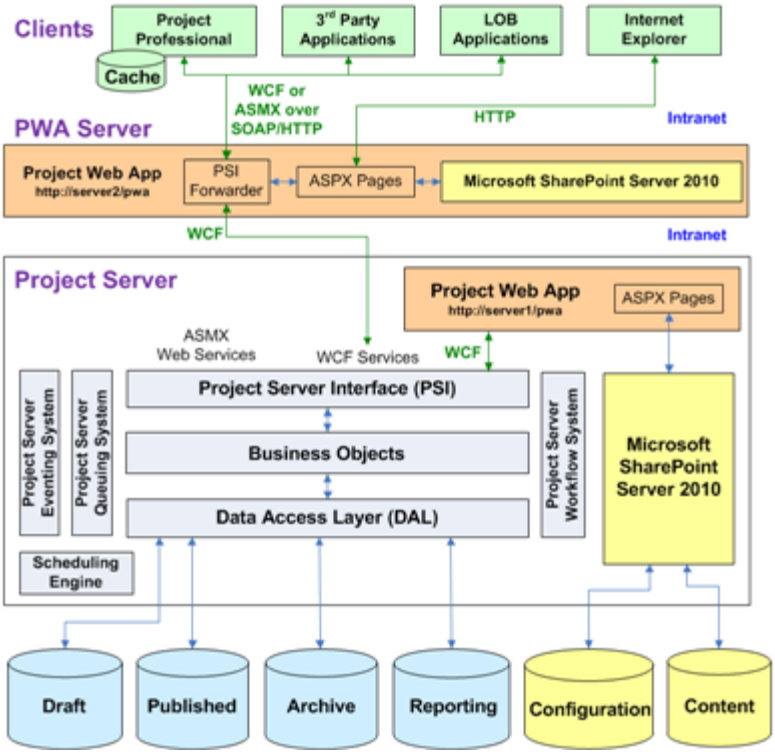

Figure 1: General architecture of the Microsoft Project Server 2010

Figure 1 shows a generalized view of the Project Server 2010 architecture, including an optional Project Web App front-end server and a Project Web App instance on Project Server. There can be multiple instances of Project Web App running on one server.

The front-end tier includes third-party applications, Project Professional, and Project Web App. Project Web App uses Internet Explorer to display Microsoft ASP.NET 3.5 pages. The Project Web App pages use Project Server Web Parts that communicate with the PSI and also use standard SharePoint Server 2010 Web Parts.

Client applications on separate computers call the PSI through service proxies. External clients that use the WCF interface access the PSI through http://ServerName/ProjectServerName/_vti_bin/psi/ProjectServer.svc. Clients that use the ASMX Web service interface use one of the Project Web App URLs, such as http://server1/pwa/_vti_bin/psi/project.asmx. If applications do not have intranet access to Project Server, they can use a Project Web App server in a perimeter network (not shown in Figure 1).

If they can directly access the Project Server computer, client applications and back-end components of line-of-business (LOB) Web applications can use PSI proxies that use the Project Server service application URL for the PSI Web services, such as http://server1:32843/ProjectServiceApplication/project.asmx. Port 32843 is the default port for the SharePoint Web Services application in SharePoint Server 2010. Direct access to the Project Service Application virtual directory should be used only when an application needs to use impersonation or run with elevated permissions.

The middle tier includes the PSI and the business logic layer, which consists of logical business objects that represent Project Server business entities. Business objects include Project, Task, Resource, Assignment, and so forth. The PSI and the business logic tier are tightly coupled and are located on the same server. A client application calls the PSI Web services, and the PSI invokes business objects in the business logic tier.

The DAL provides communication between the middle tier and the database. All Project Server data is stored in Microsoft SQL Server databases. The Project Server databases are factored into the following stores: Draft, Published, Archive, and Reporting. Client applications can read the Reporting database for project data. Clients should use only the PSI to access the Draft, Published, or Archive databases. The Reporting Data Service (RDS, which is not shown in Figure 1) updates the Reporting database from published data in nearly real time. In Project

Server 2010, all of the Project Server databases can be located on separate servers.

The Project Web App components of Project Server also use the SharePoint Foundation 2010 configuration database for project site setup and the content database for project site content such as custom pages, workflows, management settings, documents, and lists of issues, risks, and commitments. The SharePoint configuration and content databases support additional features for project management, such as project templates and workspaces, custom lists for team collaboration, and reports.

Since the Project Server is used widely by both individuals and enterprises alike, issues in the internal operations of the Project Server or its interactions with the SQL database server, if left unnoticed or unattended, can impede the project management efforts of end-users, thus scarring their experience with Project Server 2010. For instance, the sudden non-availability of the Project Server, or an unexpected increase in the load on the server, or improper server configurations, can deny users access to the Project Server or slow down their work, thus hampering user productivity! To be able to capture such anomalies on-the-fly and resolve them swiftly, administrators should monitor the Project Server 24 x 7, identify performance deficiencies proactively, and resolve them before end-users complain. eG Enterprise offers a specialized monitoring model for the Microsoft Project Server, which does all this and more!

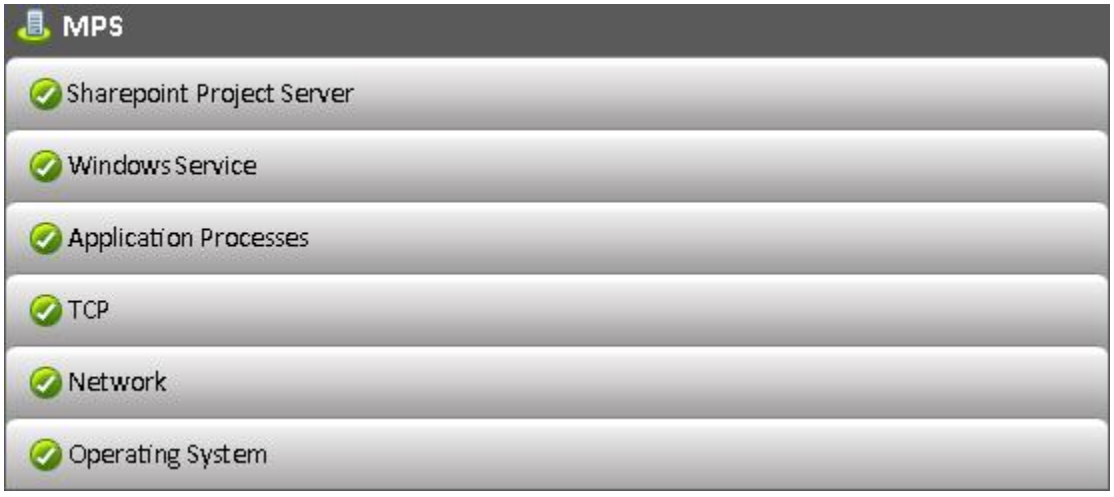

Figure 2: The layer model of the Microsoft Project Server

Each layer of Figure 2 is mapped to a wide variety of tests that report a number of metrics related to the performance of the Microsoft Project server. Using these metrics, the administrators can find quick and accurate answers for the following performance queries:

- Does any queue consist of too many unprocessed jobs? If so, which queue is it and why is that queue unable to process requests quickly? Has the queue used up all its processor threads? Should the queue be configured with more threads?
- Is any queue failing often to poll the database for jobs? If so, which queue is it?
- Is the Project Server taking too long to process jobs in queue? Which type of jobs in particular are being processed slowly? Where are these jobs spending maximum time - while waiting for processing? or when being processed?
- Which job types are failing often and why?
- What is increasing the stress on the SQL database server used by the Project Server - too many PSI calls made by external applications to the Project Server database? or too many SSP schedule job protocol queries to the database?

- Are full project saves to the database kept at a minimum?
- Do projects open quickly on the Project Server?

The **Operating System**, **Network**, **TCP**, **Application Processes** and **Windows Services** layers of a *Micrososft Project Server* model are similar to that of a *Windows Generic* server model. Since these tests have been dealt with in the *Monitoring Unix and Windows Servers* document, Section 1.1 focuses on the **Sharepoint Project Server** layer.

1.1 The Sharepoint Project Server Layer

This layer monitors the Microsoft Project Server and tracks its critical performance statistics pertaining to the jobs in the queue, the user activity on the server and the statistics of the project such as the number of incremental saves, full saves, full open etc.

Figure 3: The tests mapped to the Sharepoint Project Server layer

1.1.1 Project Server Queues Test

A queue is a waiting line that becomes a necessity when the number of service requests becomes greater than the optimum serving capacity

The purpose of the Office Project Server Queuing System is to handle such abrupt changes in demand gracefully and reliably. The Office Project Server Queuing System takes all the users' input, records entries for the requests in Microsoft SQL Server, and then processes the data asynchronously on a first-come, first-served basis. Having a queue ensures that the Office Project Server does not stop working when a spike in demand occurs.

The Office Project Server 2007 Queuing System is composed of two separate queues:

- **Project Queue** Primarily used for project-related messages such as Saving, Publishing, Reporting, and Cube Building, although other types of messages may be sent to this queue as well. Its tables and stored procedures are stored in the Office Project Server 2007 Draft database.
- **Timesheet Queue** Primarily used for timesheet-related messages such as Timesheet Save and Timesheet Submit, although other types of messages may be sent to this queue as well. Its tables and stored procedures are stored in the Office Project Server 2007 Published database.

The two queues are designed the same way, except that their jobs reside in different databases.

The true test of the efficiency of these queues lies in how quickly they process the requests lined up within. If a queue consists of too many unprocessed jobs, it is a clear indicator of a processing bottleneck in the queue, which if not cleared promptly, can end up overloading the Microsoft Project Server and significantly degrading its overall performance. To pre-empt such slowdowns, administrators will have to continuously track the jobs in each queue, look out for a steady growth in queue length, and make sure that the queue is configured with enough processing power to handle its current and anticipated load. This is where the **Project Server Queues** test helps!

This test monitors each queue of the Project server, counts the unprocessed jobs in every queue, and thus pinpoints the queue that is unable to process requests quickly. By reporting the number of threads the queue is currently using for processing requests, the test also indicates whether/not the lack of enough processor threads is the reason for the poor processing ability of the queue. This way, the test not only alerts administrators to a processing bottleneck in a queue, but also leads to the probable cause of the same.

Purpose	Monitors each queue of the Project server, counts the unprocessed jobs in every queue, and thus pinpoints the queue that is unable to process requests quickly. By reporting the number of threads the queue is currently using for processing requests, the test also indicates whether/not the lack of enough processor threads is the reason for the poor processing ability of the queue. This way, the test not only alerts administrators to a processing bottleneck in a queue, but also leads to the probable cause of the same.		
Target of the test	A Microsoft Project Server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured. 3. **PORT** – The port number at which the specified **HOST** listens to. By default, this is *NULL*.		
Outputs of the test	One set of results for each queue that is to be monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Active job processing threads: Indicates the number of threads that are currently active - i.e., the threads that are currently processing jobs in this queue.	Number	Comparing the value of this measure across queues will reveal the queue which has the highest number of threads that are currently processing the jobs.

	Average unprocessed jobs: Indicates the average number of jobs that remain unprocessed per day in this queue.	Number	A consistent increase in the value of this measure is a cause for concern, as it could indicate that the workload of the queue is steadily increasing, but the queue is unable to handle this load. In such situations, you may want to time-correlate the values of this measure with that of the *Active job processing threads* measure. If this analysis reveals that the queue is not employing more threads to handle the additional load, it could mean that the queue does not have more processor threads. You may then want to consider fine-tuning the maximum thread configuration of the queue. To do this, you will have to use the **Queue** section in the **Queue Settings** page of the **Microsoft Project Server 2010 Server Settings page**. In this section, you can configure the **Maximum Number of Job Processor Threads** . This setting determines how many job processor threads are available for use for the selected queue type (**Project** or **Timesheet**). Before modifying this setting, consider the following recommendations: • As a starting point, Microsoft recommends that you set the maximum number of processor threads settings based on the number of available processors (or cores). For example, if the Project Server application server uses a single dual-core processor, configuring the settings for two threads per queue is a good starting point. If your application server uses a quad dual-core processor, you might be able to use eight threads per queue. You can adjust these settings accordingly based not only on the volume of transactions, but also the average size of the transactions (for example, publishing 10-line projects versus 1000-line projects). • You should also think about the farm topology and other applications that are running on the farm. If you have four application servers on the farm and if you set the **Maximum Number of Job Processor Threads** setting to **4**, it gives you the potential for 16 threads to be operating. You should adjust the setting accordingly if your application server is also serving as a front-end Web server or running search or other processor-intensive activities.

			• Along with the consideration of the processing power of application servers, it is important to consider the throughput of the computer that is running SQL Server that is hosting the Project Server databases. For instance, suppose that you have eight application servers that can process threads and the **Maximum Number of Job Processor Threads** setting is at **4** (potential for 32 threads all processing jobs). SQL Server may start having contention issues because all the threads are operating on the same table.
	Current unprocessed jobs: Indicates the number of unprocessed jobs that are currently in this queue.	Number	If the value of the *Current unprocessed jobs* measure is abnormally high, you may want to compare the value of the *New jobs* measure with that of the *Current unprocessed jobs* measure to know exactly when the spike occurred – in the last minute? Or after that?
	New jobs: Indicates the number of new jobs that entered this queue in the last minute.	Number	
	SQL calls: Indicates the number of SQL calls made per minute by this queue.	Number	
	SQL retries per minute: Indicates the number of SQL calls that were retried per minute by this queue.	Number	The queue polls the database at set intervals for jobs that need processing. If the query fails, then you can configure the queue to retry the query a specific number of times. Likewise, you can also configure the interval between retries. A high value for these measures could indicate one/all of the following: • Too many queries are failing; • The **SQL retry limit** is set to a high value, allowing too many retries; • The **SQL Retry Interval** is set to a low value, allowing queries to be retried frequently;
	SQL retries per hour: Indicates the percentage of SQL calls that were retried per hour by this queue.	Percent	

1.1.2 Project Server Queue Jobs Test

Nearly all critical operations/jobs in the Project Server go through the Queuing System. These include:

- Project Save
- Project Publish
- Timesheet Save
- Timesheet Submit
- Project Backup/Recovery
- Report Data Service operations
- Cube Building Service operations
- Server Side Scheduling (and node-consistency processing)

If the Project Server is unable to process these jobs quickly, it would result in an overload condition characterized by long-winding job queues and a general server slowdown. In the event of such abnormalities, administrators will have to instantly figure out which type of jobs are contributing to the overload and why – is it because jobs of this type are failing frequently owing to errors? Or is it because the server is not adequately configured to handle these jobs? The Project Server Queue Jobs test helps administrators answer these questions!

This test auto-discovers the type of jobs in queue, and for each job type, reports the count of jobs processed, the processing time of jobs, and the percentage of job failures. This way, the test sheds light on job types that fail often, those that are taking too long to complete, and the probable reasons for the same.

Purpose	Auto-discovers the type of jobs in queue, and for each job type, reports the count of jobs processed, the processing time of jobs, and the percentage of job failures. This way, the test sheds light on job types that fail often, those that are taking too long to complete, and the probable reasons for the same.		
Target of the test	A Microsoft Project Server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured. 3. **PORT** – The port number at which the specified **HOST** listens to. By default, this is *NULL*.		
Outputs of the test	One set of results for each job type that is to be monitored		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Average processing time:** Indicates the average time taken by the Project server Queue system to process the jobs of this type.	Minutes	The value of this measure would vary based on the type and size of the job that is being processed. Ideally, the value of this measure should be low. A consistent increase in the value of this measure for any job type could indicate a bottleneck when processing jobs of this type. Administrators may then want to investigate further to figure out where the bottleneck lies.
	Average wait time: Indicates the average time the jobs of this type had to wait in queue to be processed by the Project server Queue system.	Minutes	Ideally, the value of this measure should be low. A high value indicates that the jobs are spending too much time in queue. Possible causes for this abnormality are as follows: • Jobs are blocked in the queue of the Project Server application server. • The Project Server is overloaded • The Project Server application server may have fewer job processor threads than required to meet the inflow of jobs To address these issues, administrators can do the following: • Verify whether jobs are blocked in the queue by reviewing the Windows Event log and trace logs, and troubleshoot accordingly. • If the problem persists, determine whether the problem is due to an overloaded Project Server by monitoring the following performance counters on the affected server • Try increasing the number of job processor threads
	Jobs retried per hour: Indicates the percentage of jobs of this type that were retried over the past hour by the Project server Queue system.	Percent	Ideally, the value of these measure should be zero. A high value for these measures could indicate one/all of the following: • Job failures are high; • The retry interval setting for jobs that failed due to SQL-related issues (such as SQL deadlocks), is small;
	Jobs retried per minute: Indicates the number of jobs of this type that were retried during the last minute by the Project server Queue system.	Number	

	Jobs failed per hour:
	Indicates the percentage of jobs of this type that failed over the last hour in the Project server Queue system.
		Percent	Ideally, the value of these measures should be zero. A non-zero value indicates that one/more jobs failed; this is a cause for concern and requires investigation.
			Perform the following steps to determine the reason for these failures and troubleshoot accordingly.
			• View errors in Microsoft Project Web App:
			  • Log on to Project Web App as an administrator.
			  • In the Quick Launch, click **Server Settings**.
			  • On the Server Settings page, in the **Queue** section, click **Manage Queue**.
			  • On the Manage Queue Jobs page, find the jobs that have the job state of **Failed and blocking** or **Failed and not blocking correlation**.
			  • View more information about the failed jobs by clicking the error IDs that appear in the **Error** column of the failed jobs.
			• View errors in the Windows Event log of the Project Server to determine whether any errors occurred at around the same time as the job failures.
			• View the trace logs on the Project Server to determine whether any errors occurred at approximately the same time as this alert that may have caused the problem.

	Jobs failed per minute: Indicates the number of jobs of this type that failed over the last minute in the Project server Queue system.	Number	
	Jobs processed: Indicates the number of jobs of this type that were processed per minute by the Project Server Queue system.	Number	A high value is desired for this measure. A consistent decrease in the value of this measure clearly indicates a processing bottleneck.

1.1.3 Project Server User Activity Test

A high level of user activity on the Project Server often increases the workload of the SQL server on which the Project Server databases reside! For instance, if many users make PSI calls to the Project server database via third-party applications or if users execute multiple queries on the database to create/modify/retrieve SSP job schedules, the

load on the SQL server hosting these databases is bound to increase; in the long run, this can significantly impact the overall performance of not just the SQL server but also the Project Server associated with it. To avoid this, administrators can use the **Project Server User Activity** test to monitor the PSI calls made and SSP scheduled job stored procedures executed on the SQL database server used by the Project server. Doing so enables administrators to rapidly detect abnormal activity on the server and promptly initiate measures to curb the activity, so that user experience with the Project Server does not suffer.

Purpose	Monitors the PSI calls made and SSP scheduled job stored procedures executed on the SQL database server used by the Project server		
Target of the test	A Microsoft Project Server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured. 3. **PORT** – The port number at which the specified **HOST** listens to. By default, this is *NULL*.		
Outputs of the test	One set of results for the Microsoft Project server that is being monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	PSI calls: Indicates the number of Web Service calls from the clients to the PSI API per second.	Number	The PSI is the API of Project Server. The PSI object model exposes Project Server functionality to all external applications. Project Professional 2010, Project Web App, LOB, and other third-party applications use the PSI to access Project Server data in the Draft, Published, and Archive databases. The PSI is available through WCF services and through ASMX Web service calls by back-end LOB applications, or through a PSI proxy. The value of this measure is an indicator of the load imposed by external applications on the Project server.

	SSP jobs scheduled: Indicates the number of SSP (Shared Services Providers) jobs that were scheduled per second.	Number	A Shared Services Provider (SSP) provides a common set of services and service data to a logical grouping of Web applications and their associated sites. The Shared Services Provider Scheduled Jobs protocol allows clients to add, modify, refresh, and delete scheduled jobs from a store on the protocol server, as well as retrieve those scheduled jobs by using predefined criteria such as a unique identifier or jobs scheduled to be run within a specific interval. In addition, the protocol specifies the actions that the client takes when a particular scheduled job is retrieved and is to be executed in the next specified interval. This measure reports the number of SSP scheduled jobs that were added every second to the database. A high value indicates that many jobs have been scheduled, which in turn signals an increase in the load on the server in the immediate future.

1.1.4 Project Server WinProjects Test

How a project is opened on the Project Server and how it is saved on the SQL database server can impact the health and overall performance of the Project Server and the SQL server it uses. By keeping an eye on project opens and saves, administrators can proactively detect potential overload conditions / slowdowns in the Project Server / SQL server, and promptly initiate measures to avert such anomalies. The **Project Server WinProjects** test helps administrators with this! This test reports the count of incremental saves, full saves, incremental opens and full opens that were attempted on the Project Server and its database server, measures the time taken for a project to open, and thus leads administrators to probable performance slowdowns with the Project Server.

Purpose	Reports the count of incremental saves, full saves, incremental opens and full opens that were attempted on the Project Server and its database server, measures the time taken for a project to open, and thus leads administrators to probable performance slowdowns with the Project Server
Target of the test	A Microsoft Project Server
Agent deploying the test	An internal agent
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured. 3. **PORT** – The port number at which the specified **HOST** listens to. By default, this is *NULL*.
Outputs of the test	One set of results for the Microsoft Project server that is to be monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Winproj incremental saves in the last hour: Indicates the number of incremental saves that occurred for a project created on the Project server during the last hour.	Number	The first time that a project is created on the Microsoft Project Server application server from Microsoft Project Professional, it is saved using a full save, which means that all of the project data is transmitted and saved in the database used by the Project Server application server. Subsequent saves of the project are saved incrementally, by default. This means that only the project data that has **CHANGED** is transmitted to the server and saved in the database. If this default setting is overridden, it may increase the number of full saves, which is detrimental to the optimal performance of the Project Server and the SQL database.
	Winproj full saves in the last hour: Indicates the number of full saves of the project that was created on the Project server 2010 during the last hour.	Number	
	Winproj incremental opens in the last hour: Indicates the number of times the project created on the Microsoft Project Server application server was opened during the last hour.	Number	
	Winproj full opens in the last hour: Indicates the number of times a '*full open*' was attempted on the projects that were created on the Project server 2010 during the last hour.	Number	
	Avg time taken to open project: Indicates the average time taken for a project to open on the Project server application server that uses Microsoft Project Professional.	Secs	Ideally, the value of this measure should be less than 120 seconds. The value of this measure tracks the time taken for a project to open on the server only — it does not include the communication overhead between the client application (in this case Microsoft Project Professional) and the Project Server application server. A value higher than 120 seconds is a cause of concern and is an indication of an overloaded Project Server application server or an overloaded SQL Server computer that is hosting the databases of the Project Server application server.

	Incremental save to full save ratio: Indicates the percentage of incremental saves to full saves.	Percent	For optimal performance, the value of this measure should be low.

2

Conclusion

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

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