Fidelity Global Pricing Workstation Testing Interface:

A Web-Enabled Test Automation Tool

A Major Qualifying Project Report

submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

By

Elizabeth Carey, MIS ______________________

Daniel Dahlberg, MIS ______________________

Michael Diamant, MIS ______________________

Augustina Mills, MIS ______________________

Date: March 3, 2009

Sponsored by

Fidelity Pricing & Cash Management Services

Fidelity Investments

____

Professor Soussan Djamasbi

Project Advisor

Abstract

The goal of this project was to create a web-based tool to interface with Global Pricing Workstation (GPWS) test environments on the UNIX platform. Functional solutions to address the need for operational support tools for GPWS System Integration Testing / User Acceptance Testing (UAT) were also proposed. Perl and CGI were used to create web-based interfaces for the four high-priority deliverables for GPWS. The final dashboards that were completed for this project can be used in Fidelity's GPWS with FPCMS and can also be used as a template for other areas of the department.

Executive Summary

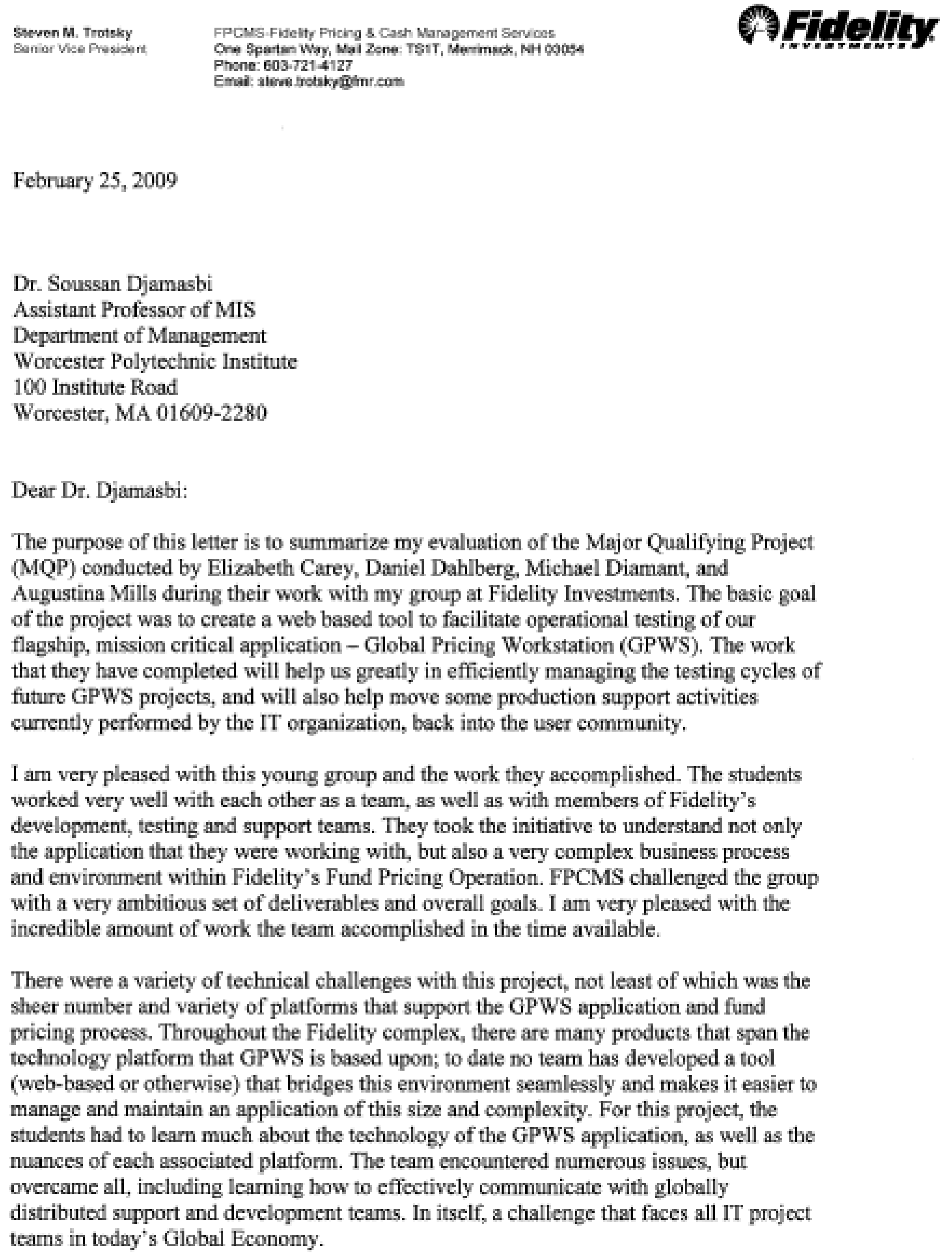
The Fidelity Global Pricing Workstation Testing Interface project was completed in collaboration with the Fidelity Pricing and Cash Management Services at Fidelity Investments Inc. in Boston, MA. The goal of this project was to create a web-based tool to interface with Global Pricing Workstation (GPWS) test environments on the UNIX platform. Along with the web-based interface, we were to propose functional solutions to address the need for operational support tools for GPWS System Integration Testing (SIT)/User Acceptance Testing (UAT), evaluate current GPWS SIT /UAT tools where appropriate, and identify areas of GPWS SIT / UAT testing that could benefit from more or better tools.

The project focused on integrating business process automation and business process improvement to deliver the selected interfaces and functionalities. There were a number of deliverables in the high-level requirements. The scope of the MQP project included the top four priority deliverables. These four deliverables were the End-of-Day Dashboard, End-of-Day Operational Support Interface, Process Control Operational Support Interface, and Test Environment Dashboard.

A key element to the project was to ensure that the deliverables would not only be useful for GPWS, but that they could also be used as a template to automate current and future applications within other groups at Fidelity. Consequently, a major part of the MQP project time was dedicated to learning about other applications that these deliverables could potentially be integrated into.

The MQP group was successful in timely completion of the project. Moreover, in addition to delivering the expected functionalities, the team provided its Fidelity sponsors with a training manual and an implementation guide. The training manual, which walks the user through the application (e.g., dashboards) and instructs them on the elements on each page, can help guide future users of this system. The implementation guide provides instruction for implementing this system in numerous environments at Fidelity.

Letter from Sponsor



Steven M. Trotsky
Senior Vice President

FPCMS-Fidelity Pricing & Cash Management Services
One Spartan Way, Mail Zone: TS1T, Merrimack, NH 03054
Phone: 603-721-4127
Email: steve.trotsky@fmr.com

February 25, 2009

Dr. Soussan Djamasbi
Assistant Professor of MIS
Department of Management
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609-2280

Dear Dr. Djamasbi:

The purpose of this letter is to summarize my evaluation of the Major Qualifying Project (MQP) conducted by Elizabeth Carey, Daniel Dahlberg, Michael Diamant, and Augustina Mills during their work with my group at Fidelity Investments. The basic goal of the project was to create a web based tool to facilitate operational testing of our flagship, mission critical application – Global Pricing Workstation (GPWS). The work that they have completed will help us greatly in efficiently managing the testing cycles of future GPWS projects, and will also help move some production support activities currently performed by the IT organization, back into the user community.

I am very pleased with this young group and the work they accomplished. The students worked very well with each other as a team, as well as with members of Fidelity's development, testing and support teams. They took the initiative to understand not only the application that they were working with, but also a very complex business process and environment within Fidelity's Fund Pricing Operation. FPCMS challenged the group with a very ambitious set of deliverables and overall goals. I am very pleased with the incredible amount of work the team accomplished in the time available.

There were a variety of technical challenges with this project, not least of which was the sheer number and variety of platforms that support the GPWS application and fund pricing process. Throughout the Fidelity complex, there are many products that span the technology platform that GPWS is based upon; to date no team has developed a tool (web-based or otherwise) that bridges this environment seamlessly and makes it easier to manage and maintain an application of this size and complexity. For this project, the students had to learn much about the technology of the GPWS application, as well as the nuances of each associated platform. The team encountered numerous issues, but overcame all, including learning how to effectively communicate with globally distributed support and development teams. In itself, a challenge that faces all IT project teams in today's Global Economy.

In summary, I am very pleased with the team and the work they accomplished. It was apparent to me that every member of the team made significant contributions to the success of the overall project. I got to know each of the students and was very impressed with their level of commitment and their professionalism. The group employed best practices for development by modularizing the code. This will allow Fidelity to adopt and scale the tool for use in many other applications in our environment. Additionally, I believe what we have learned from the MQP team's efforts will help Fidelity better manage its software development, testing, and support activities, as well as contribute to our ability to deliver products and services in a more timely and efficient manner.

They were a great group. I look forward to working with you and WPI again on future MQP projects.

Sincerely,

[illegible]

Steven M. Trotsky
Senior Vice President
Fidelity Investments – Pricing and Cash Management Services

cc: Rob Mushinski
Bill Brickley
Jennifer Koenig

Authorship Statement

Abstract – Elizabeth Carey

Executive Summary – Elizabeth Carey, Augustina Mills

Background Literature Review – Elizabeth Carey

Project Description – Michael Diamant, Augustina Mills

Planning – Elizabeth Carey, Michael Diamant

Analysis – Elizabeth Carey, Daniel Dahlberg

Design – Elizabeth Carey, Michael Diamant, Augustina Mills

Installation Guide – Daniel Dahlberg

Conclusions and Recommendations – Michael Diamant, Augustina Mills

Accumulated Weekly Meeting Agendas – Elizabeth Carey

Training Manual – Augustina Mills

Financial Analysis – Michael Diamant

Interview Reports – Elizabeth Carey

System Request – Michael Diamant

Uses Cases – Augustina Mills

Fidelity Presentation – Michael Diamant

WPI Presentation – Michael Diamant

WPI Poster – Elizabeth Carey

Acknowledgements

Sponsor: Fidelity Investments Incorporated

Special Thanks

to

Dr. Soussan Djamasbi

Washesh Mehra

Rosetin Anwar

Fidelity Pricing and Cash Management Services

Table of Contents

Table of Tables

Table of Figures

1 Background Literature Review

In this section, we will provide a background review about Fidelity Investments, the division that the MQP team worked in, and the benefits of test automation. An overview of Fidelity Investments was done for the purpose of introducing the company sponsor. The Global Pricing Workstation is explained, as it is the testing environment the MQP team worked on. Lastly, the benefits of test automation were explored because of the nature of the MQP project, ultimately helping to automate selected manual processes at Fidelity.

1.1 Fidelity Investments

Fidelity Investments is one of the world's largest mutual fund firms. It currently serves more than 23 million individual and institutional clients and manages more than 300 funds with approximately $1.5 trillion of assets under management (Hoovers, 2008). The Fidelity Pricing and Cash Management Services (FPCMS) provides accounting and investment management support to Fidelity mutual funds. Currently, Fidelity is using the Global Pricing Workstation within Fidelity Pricing and Cash Management Services to price securities held by Fidelity (Hoovers, 2008).

1.1.1 History

The Boston money management firm, Anderson & Cromwell, created the Fidelity Fund in 1930. In 1943, Edward Johnson became president of the fund when it had $3 million invested in Treasury Bills. The fund reached $10 million in 1945 after diversifying into stocks. Fidelity Management and Research was established by Johnson in 1946. The Magellan Fund started in 1962 and shortly after, the company entered the market of corporate pension plans and retirement plans (Hoovers, 2008).

Johnson gave control of Fidelity to his son, Ned, in 1972. While under the control of Ned, Fidelity Management and Research started selling directly to customers rather than through brokers as in previous years. The Magellan Fund was managed by Peter Lynch and grew from $20 million to $12 billion during the 13 years of his service. The company opened a nationwide branch network in the 1980's and soon entered the credit card business. The Magellan Fund closed to new investors in 1997 (Hoovers, 2008).

The apparent heir to Fidelity is Johnson's daughter, Abigail, who was given a 25% stake in the firm and is currently the company's largest single shareholder. During the late 1990's the firm expanded its branches in Canada and entered the Japanese market. In 1999, Fidelity developed its online brokerage services. Today, Fidelity is an international provider of financial services and investment resources. The company has a commitment to continuous improvement as well as state-of-the-art technology and customer service (Hoovers, 2008).

1.1.2 Global Pricing Workstation (GPWS)

Mutual funds are at the core of Fidelity's business and GPWS is the system of record for pricing the mutual funds. GPWS was released in August of 2005 and has been in production for the past 3 years. It is rated an AAA application within Fidelity. The AAA rating means it has the highest level of availability, 99.5%, and recoverability, 30 minutes. Currently, it is used to price approximately 58,000 securities for 4,554 funds with 275,769 holdings (Mackey, 2008). The user community for GPWS is geographically dispersed and includes employees for pricing analysts, fund accountants, and production support. There are a total of 700 users with 11 production supports. GPWS is a Java-based application with C++ client and Spring operational interface. The database used is DB2 and is runs on AIX servers.

The Test Delivery Services (TDS) work on System Integration Testing (SIT) and User Acceptance Testing (UAT). During the SIT, the TDS provides functional testing and operational support. For UAT, TDS only provides operational support. The test cycle is based on the GPWS business day. The three main areas of the business day are the start-of-day processing, pricing underway, and end-of-day processing. Functional Test Engineers and Operational Support Test Engineers are the employees who play a role in testing. The Functional Test Engineer is responsible for test planning, test cases, and business functional test execution. The Operational Support Test Engineers support functional testing, oversee environment management and have the technical subject matter expertise.

1.2 Test Automation

Automated testing refers to automating the manual testing process that is currently in use (Exforsys Inc, 2008). The purpose for automation is to increase the flexibility of time and resources by requiring less resources and making them available for other projects. Test automation is also used to avoid redundancy by having a standard way of how processes run. Finally, automation increases reliability and quality because the automated test is run exactly the same every time.

1.2.1Benefits

The benefits to automated testing are abundant. Foremost, automation is reliable. When tests are automated, they perform the exactly same tasks each time they are run, which eliminates human error. Automated tasks are repeatable, which allows greater testing on how software reacts after numerous tests. Reusability is another key benefit to automated testing. Once tests are constructed, they can be used on different versions of an application or in completely different interfaces with only minor changes. The automated tests create a good base for the development of future tests.(Exforsys Inc, 2008)

Automated testing is also a time and cost saver for testing. Tools that are automated run significantly faster than manually executed tasks. The more processes that are automated, the fewer resources are needed for a project. This lowers the costs and allows for the current manual labor done by the test support engineers to be used in other projects and work.

2 PROJECT DESCRIPTION

This section describes the MQP project and its scope. Some parts in this section are directly taken from (and/or adapted from) initial project definition documents by Fidelity such as GPWS test Automation

Project Charter and GTI Testing Interface Project Charter. Information within the objectives has been set forth by Fidelity and adjusted as needed. The full version of these documents can be found in Appendix F.

2.1 Business Opportunity

GPWS is a software application used by the Accounting and Pricing Operations group within FPCMS to collect, validate and approve the current price of securities held by Fidelity funds. Those security prices are then used to calculate and distribute the net asset values of Fidelity mutual funds. Management teams and analysts in Pricing and Fund Accounting use GPWS to perform all daily pricing functions.

Two key quality assurance activities, which ensure that GPWS releases are ready for production, are System Integration Testing (SIT) and User Acceptance Testing (UAT). GPWS SIT and UAT are the shared responsibility of the Fidelity Reporting, Accounting and Pricing Services Business Analysis team (FRAPS BA) and Test Delivery Services (TDS), part of FPCMS Center of Excellence Testing & Quality Assurance Central Services. The Test Delivery Services team focuses on validating that new functionality and procedures conform to requirements and existing functionality and procedures continue to function as before. They are also responsible for the operational support aspects of GPWS SIT / UAT and supporting the operation of the SIT / UAT test environments. The FRAPS BA team focuses on reviewing and testing GPWS business functionality and procedures during GPWS UAT. For many test scenarios, the efforts of both teams are required. Both teams are geographically dispersed across the United States and India.

2.2 Summary Project Recommendations

Test automation is a key initiative of FPCMS Testing & Quality Assurance Central Services. An integral part of the test automation effort is automation of operational tasks and support performed manually by operational support test engineer. The purpose of this project is to create a web-based tool (referred to as GPWS Testing Interface or GTI in rest of the document) to interface with GPWS test environments on UNIX platform. This will be an important step for Fidelity so it can be accessed from all Fidelity sites and be easily usable.

2.3 Preliminary Project Context

GTI is intended to automate various operational tasks that a test engineer performs during a GPWS test effort. This will enable functional test engineer to perform operational tasks without the assistance of operational engineer. A test engineer can use either Quality Center, a Fidelity application, to perform operation task as part of automated regression suite, or use the tool independently. Quality Center interfaces with Quick Test Pro, another Fidelity application, to execute operational support scripts from GTI. Quality Center also produces automated test results reports based on the responses from the GPWS server, and from the messages displayed on GTI. These can be seen below in Figure 1 and Figure 2.

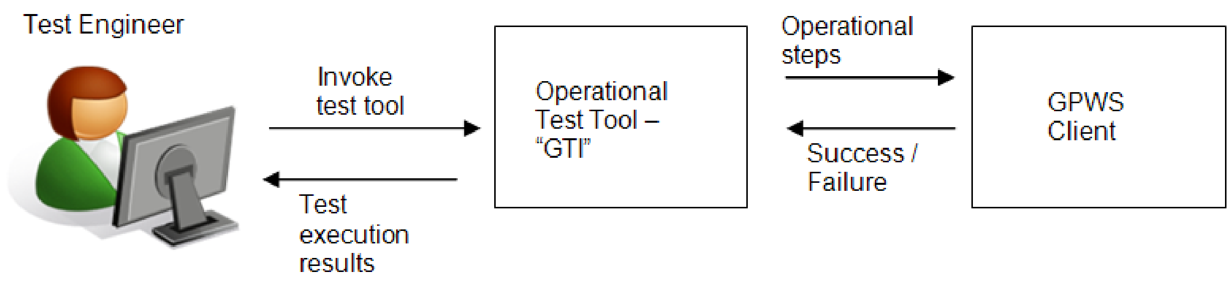


Figure 1: Independent User Context Diagram

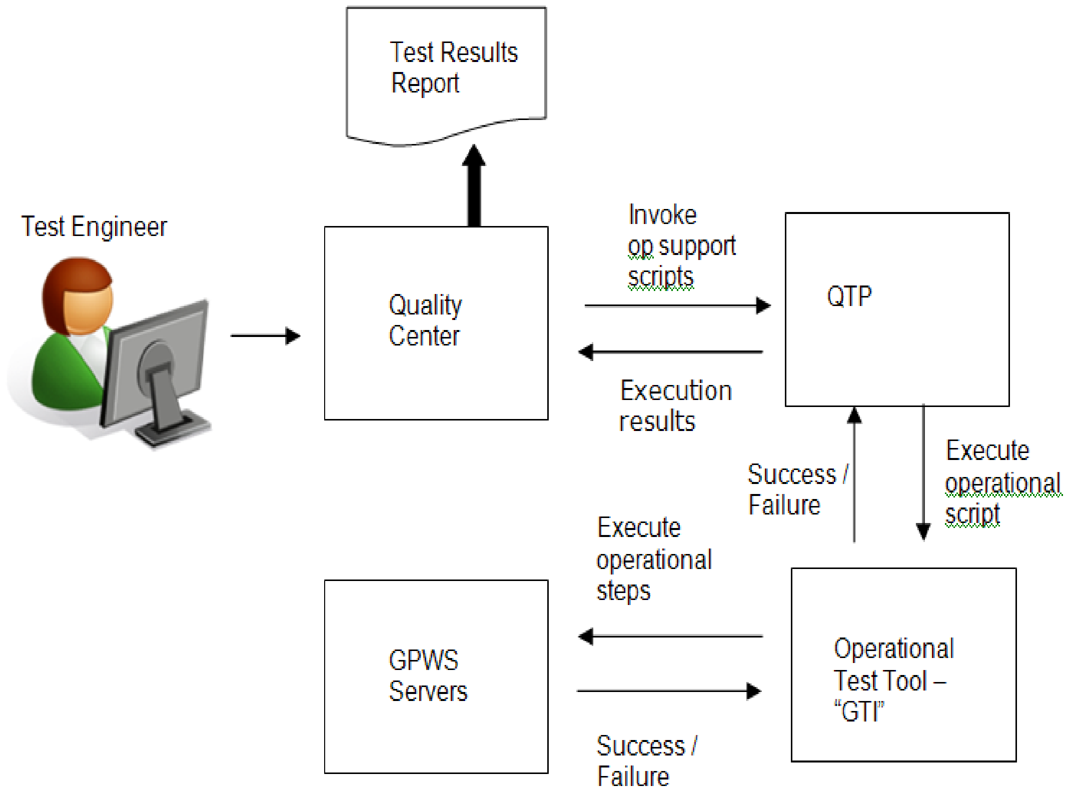


Figure 2: Operational Support Context Diagram from Fidelity GPWS Testing Interface High Level Requirements (Appendix F)

2.4 Project Scope

The project scope includes detailed and documented analysis of the high-level requirements and a documented solution of those high-level requirements. A working web application is included in the scope with a documented user guide. The GTI tool is only to be used in a testing environment, not in production, and it will be used for the GPWS application. A summarized table of what the scope includes can be found in Table 1.

Table 1: Scope of the Project

	Scope
1	Detailed and documented analysis of prioritized high level requirements
2	A documented proposed solution of the prioritized high level requirements
3	A working web application
4	Documented user guide
5	The "GTI" tool's intent is to be used in testing environments. Production environments are out of scope.
6	The tool is only for GPWS application.

The project scope does not include the GPWS restart and recovery test environment and test execution or the performance test environments and test executions. The scope does not include the GPWS development integration or the tools for production supports. The items not included in the scope are summarized in Table 2.

Table 2: Items not included in Scope

	Not included in Scope
1	GPWS restart / recovery test environments and test execution
2	GPWS performance test environments and test execution
3	GPWS development integration testing environment
4	Tools for production support purposes

2.5 Stakeholder List and Roles

There are a number of individuals in the COE TQACS department that will be included in the project and affected by this project. Fidelity has made the project sponsor, project manager, and two operational support technical leads available to the project group. A summarized table of the project team with their responsibilities can be viewed in Table 3. The stakeholders and their responsibilities can been viewed in Table 4.

Table 3: Project Team

Name(s)	Department / Organization	Project Role	Responsibilities
Roy Lockhart	COE TQACS	Project Sponsor	• Project oversight • Project reporting
Washesh Mehra	COE TQACS	Project Manager	• Project management • Project reporting • Issue follow-up • Project logistics
Barbara Mackey	COE TQACS	Operational Support Technical Lead	• Technical guidance/ oversight • Review of all deliverables • Prioritized high level requirements list
Glenn Janssen	COE TQACS	Operational Support Test Engineer	• Design/development of functional and non-functional test cases • Review of all deliverables • Testing and verification of deliverables
Dan Dahlberg Michael Diamant Elizabeth Carey Augustina Mills	WPI Students	Analysts Developers Testers Implementation	• Collect requirements • Develop solution proposal • Develop prototype • Write project and application documentation • Prepare training materials • Present migration and implementation plan

Table 4: Stakeholders

Name(s)	Department / Organization	Stakeholder Involvement	Responsibilities
Dave Erickson	COE TQACS	Environment Management Release Engineering	• Provide GPWS test environment in which to develop/test deliverables • Provide guidance on source control & RE considerations
Professor Soussan Djamasbi	WPI	Faculty Advisor	• Monitor students' and projects' progress • Review all deliverables • Be the primary contact for WPI Fidelity team
TBD	A&T	Technical Consultant	• Provide development-related guidance and standards
TBD	Software Engineering	Technical Consultant	• Provide development-related guidance and standards
Sharon Rowe	COE TQACS	Quality Management	• Provide process-related guidance and standards
Ben Gedaminski Erin Molgaard	COE TQACS	Program Management	• Provide program-related guidance and standards

2.6 Project and Business Value Objectives

The project has objectives in two areas: (1) business value objectives and (2) project objectives. In this case, project objectives refer to system implementation related goals and business value objectives refer to the value added goals for Fidelity. Objectives are categorized in this manner because the project must be thought of in the context of the business-operating environment. By explicitly highlighting system implementation and business goals, it becomes clear how the project adds value to the firm.

2.6.1 Business Value Objectives

Business value objectives highlight the benefits of implementing the proposed system for Fidelity. The system achieves business benefits through automation and by providing an easy-to-use user interface. This creates benefits recursively throughout FPCMS. The expected benefits are summarized in Table 5.

Table 5: Business Objectives and Expected Benefits

Business Objective	Expected Benefit(s)
Increase test execution consistency and efficiency by automating as many processes and procedures as possible	Operational support tasks can be repeated consistently. Time spent on operational support tasks is reduced significantly.
Ensure test execution is repeatable by using pre-defined scripts	Increase reuse of scripts/automated processes. Minimize chances of errors due to manual intervention.
Minimize data entry error of configuration items	Reduce risk of operator error for operational support tasks. Operational support tasks can be repeated reliably and consistently. Time spent on operational support tasks is reduced.
Enable functional test engineers to perform some operational support tasks, and remove dependency on Test Delivery Services for some operational support tasks.	Operational support for testing can be performed by test engineers with less or no GPWS-specific knowledge. Delays to functional testing incurred while waiting for operational support tasks to be performed are reduced. Better utilization of operational support engineers.
Improve result reporting, logging, and documentation	Reduce training time for new resources on operational support tasks. Improve the ability to debug and troubleshoot errors encountered during an operational task. Improve reliability of operational tasks.
Enable future improvements in test environments and tools	Potential reuse of the tool in other products – iTAAC, CASS etc.

2.6.2 Project Objectives

Project objectives highlight the expected benefits of system implementation. One objective is to ensure that Perl and CGI can be used to perform current manual tasks. The benefit of this objective is to ensure project feasibility. Other project objectives stem from the basis that this system must be maintainable and usable. The objectives and benefits are detailed in Table 6.

Table 6: Project Objectives and Expected Benefits

Project Objective	Expected Benefit(s)
Demonstrate that PERL and CGI can be used to execute UNIX shell scripts remotely, and perform other operational task via a Web based interface.	PERL and CGI can be added to the list of available tools/scripting options in FPCMS Test Automation program.
Demonstrate that coding standards and conventions put forth by automation team can be applied consistently regardless of tool used.	Coding standards and conventions are tool/language independent.
Design "GTI" so that it is maintainable and can support small to mid-size changes easily.	Test tool code can be changed quickly enough to support changes or enhanced functionality.
Train both functional testers and operational support test engineers how to use test tool.	Testers and engineers can transfer test tool knowledge to other testing efforts.

2.7 Project Success

The success of the project depends on many factors. This can be measured by Fidelity's continuous use and development of GTI. In addition, the GTI must function in the current and future GPWS environments.GTI will remain successful if it produces accurate results while taking less time than the current system. These project successes along with the measurements of those successes can be found in Table 7.

In Table 7, there are success criteria and measurement categories that coincide with one another. The criteria column signifies the criteria that must be met and the measurement acts as a gauge for the success of the criteria. All eleven criteria and measurement factors are listed below.

Table 7: Project Success Matrix

Criteria	Measurement
The existing technologies can be used to develop "GTI"	"GTI" will be hosted on IHS (IBM HTTP Server), and written in Perl and CGI
"GTI" must function in existing and future GPWS environments	All GPWS environments can be accessed and managed via "GTI"
"GTI" should not negatively impact the performance of the environment	Performance of GPWS application post install of GTI is same pre-install
Use of "GTI" to perform operational support task should take less time than performing the tasks manually.	Time spent in performing operational support task is less with "GTI" compared to manually.
Use of "GTI" provides accurate result reporting and unambiguous logging.	Reporting and logging functionality is easy to access and understand.
"GTI" should not interfere with the processing of operational tasks in any way that would compromise results	Operational tasks do not get impacted by "GTI"
"GTI" should be accessible and be used from all Fidelity sites – US and India	Users in all US and India sites are able to access and use "GTI"
"GTI" can be used as part of regression suite	Web based tool can be accessed using functional test automation tool – QTP.
Operational tasks can be performed reliably with minimal support or guidance from operational support engineers.	Functional engineers find "GTI" intuitive to use and perform operational tasks.
All best practices of coding standards and naming conventions must be followed as set by COE TQACS team	All coding standards and conventions are adhered to. Code passes code-review of Fidelity.
User procedures and support documents will be delivered to Fidelity at the end of the project	All documents are stored in EDMS and have been reviewed with Fidelity.

2.7.1 Critical Success Factors for MQP Team

In order for the MQP team to deliver a successful project, a few factors must be accounted for. The MQP team will need workspace and access to Fidelity's systems. Arrangements have been made so that the group is able to visit Fidelity once a week on Wednesdays while having access to necessary resources as workspace and systems in order to perform analysis, design and development toward the project. The GPWS test environment that the group is working on is available to the MQP team for regression test development and verification. In addition, the Fidelity project team is present to answer questions the

MQP team has. The MQP team is able to visit on site once a week and have all resources (space, systems, access), as well as perform analysis, design, and development, use the project team as a resource when questions arise, and lastly use the GPWS test environment for regression test development / verification. These objectives are pertinent to the sculpting of the project. The expectations and requirements will act as guide for the WPI team in the planning, design and implementation portions of the project.

3 Planning

Careful planning is a key component in making any project a success. It is in planning that we organize the project in such a way that all tasks are completed. Much thought went into the estimate of project impact. As well, it was necessary to consider other factors like competitors, customers, employees and management. All of these aspects needed to be accounted for in the planning and designing of this project. In the planning stage of this project, the group set-up initial meetings to meet with the Fidelity project team for introductions and the project layout. There was also planning that happened on a weekly basis which included weekly meetings with the project advisor, weekly meetings with the group to discuss the progress tasks, and also the assigning of relevant tasks for each stage of the project.

3.1 Project Impact

The impact of the project will be high on employees and management working within FPCMS since this tool is used and depended on by many of the engineers within FPCMS. This test interface will be utilized in a manner that will help to save time for both Fidelity and the support test engineers. The impact on the competitors and customers will be minor. Little impact will be seen to the competitors and customers as the project will remain internal within FPCMS. Competitors and customers will only observe any external improved efficiency within Fidelity as an entity. The impact on employees and upper management is explained below.

3.1.1 Employees

Employees, like the project team who is helping with the success of this project, were considered. A reason this project came about because the employees were spending too much time performing manual processes that could be automated. In this consideration, it was calculated that fewer hours would be spent on mundane processes, efficiency and consistency of the testing procedures would increase, the testing cycle should decrease, and the overall GPWS application should have less time spent on procedures. One of the main highlights of the project is the impact that is has on both employees and management.

3.1.2 Management

With the help of the test interface, management will see a positive change and effect on the way things are currently run. There will be more person-hours to dedicate to other important and/or needed tasks. The modular GTI will also bring benefits to other divisions, especially those including the ITAC environments.

3.2 Project Feasibility

To better understand the context of the project, the project is often analyzed from three perspectives: (1) technical, (2) economic, and (3) organizational. Analyzing the project from these three perspectives makes the project team aware of possible difficulties associated with project completion. The risks and difficulties identified here are later addressed through risk assessment and mitigation. Thus, analyzing project feasibility plays a critical role in identifying threats to project success.

3.2.1 Technical Feasibility

The technical feasibility gauges risk in four areas: (1) the business domain, (2) the technology, (3) project size, and (4) existing technical infrastructure compatibility. Familiarity with the business domain affects the effectiveness of the MQP team to implement a solution. Familiarity with the implementation technology also affects the project team's ability to reach project goals. Larger project sizes increase project risk because there are more logistical items to consider. Compatibility with existing technical infrastructure is critical because the solution must be capable of fitting into the operating environment. Given these areas of risk, the project is feasible, but presents moderate risk.

Risk regarding familiarity with securities pricing systems is moderate. The MQP team does have work experience in the financial services domain, which reduces risk. However, the MQP team does not have direct experience with the target system, pricing systems, which increase risk. Last, the project team does have experience creating web applications, which reduces risk.

Risk regarding familiarity with the technology is high. Not all MQP team members have web development experience, which increases risk due to the additional learning that must occur. The MQP team has limited Perl and CGI development experience. This significantly increases the risk factor because this is the implementation technology. Although the project team is unfamiliar with Perl and CGI, a large support community exists to aide in the learning process.

Risk regarding project size is low. MQP members are familiar with each other’s strengths and weaknesses, which lowers project risk. The project is highly integrated with other systems, but is isolated enough to make implementation simple. The project team is sufficiently large, four members, but not large enough to cause logistical issues.

Risk regarding existing technical infrastructure compatibility is low. A proof of concept using Perl and CGI has been completed during the analysis phase, which drastically lowers risk. The source code of the GPWS application will not be modified, which further reduces risk in this area.

3.2.2 Economic Feasibility

The economic feasibility assesses if the proposed system makes financial sense, that is, will the organization gain a monetary reward from system implementation. The economic feasibility attempts to assess project benefits and costs. Intangible benefits are often quantified to better understand the benefit of the proposed system. This particular project results in numerous intangible benefits. The benefits include improved testing process consistency, improved utilization of operational support test engineers, enhanced ability to research test exceptions, and smaller server footprint (reduction of server usage). These benefits and their estimated contributions are displayed in Figure 3.

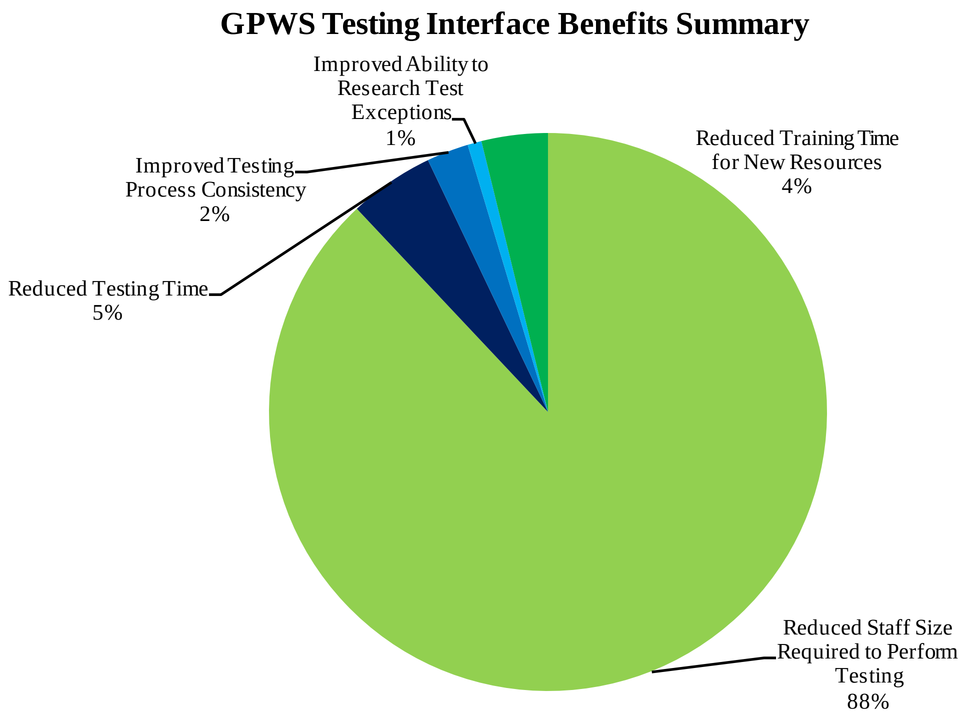


Figure 3: GPWS Testing Interface Benefits Summary

The project also results in a number of costs. The costs incurred for this project include initial user training, annual user training, annual code maintenance, development training, consultant fees, and equipment. These costs have been estimated by the team (see Table 4 and Appendix D). A summary of these estimated costs are shown in Figure 4.

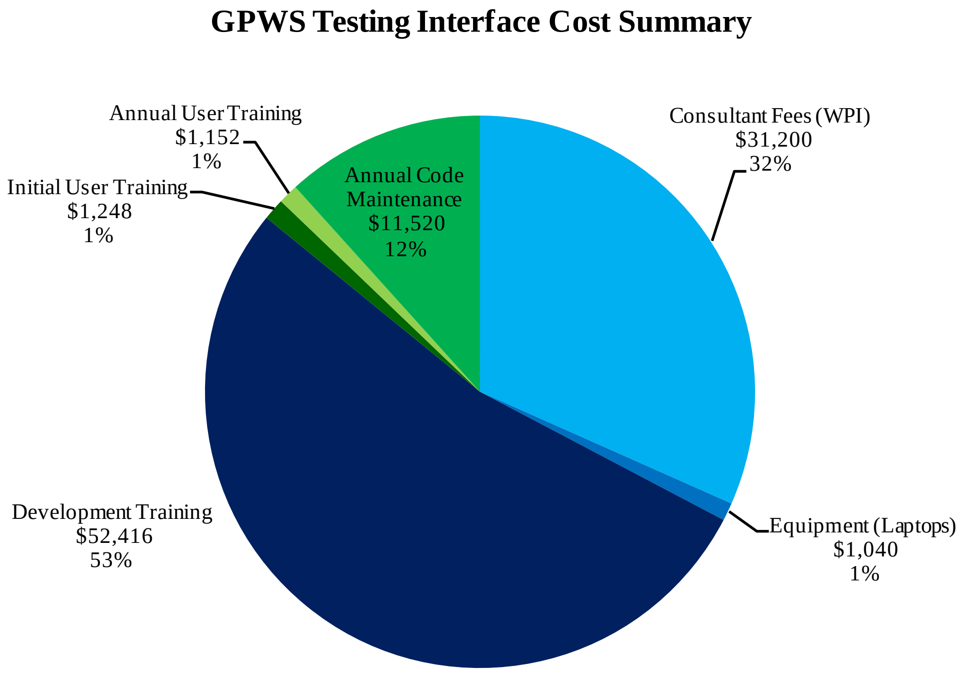


Figure 4: GPWS Testing Interface Cost Summary

The system yields tangible savings that are translated through reduced oversight by operational support test engineers during the testing process. These results in a projected cost savings of $814 per testing cycle and a total annual savings of $11,835 across all test types (see Table 8 for details). The complete set of documents for the financial analysis can be found in Appendix D.

Table 8: Net Present Value Estimation

	2008	2009	2010	2011	2012	Total
Benefits per Anum						
Tangible Benefits						
Reduced Staff Size Required to Perform Testing	$ -	$ 7,200	$ 7,632	$ 8,090	$ 8,575	
Reduced Testing Time	$ -	$ 128,410	$ 136,114	$ 144,281	$ 152,938	
Total Tangible Benefits	**$ -**	**$ 135,610**	**$ 143,746**	**$ 152,371**	**$ 161,513**	
Intangible Benefits						
Improved Testing Process Consistency	$ -	$ 3,551	$ 3,764	$ 3,989	$ 4,229	
Improved Ability to Research Test Exceptions	$ -	$ 1,184	$ 1,255	$ 1,330	$ 1,410	
Reduced Training Time for New Resources on Operational Support Tasks	$ -	$ 16,800	$ 17,808	$ 18,876	$ 20,009	
Total Intangible Benefits	$ -	$ 21,534	$ 22,826	$ 24,196	$ 25,647	
Total Benefits	**$ -**	**$ 157,144**	**$ 166,572**	**$ 176,567**	**$ 187,161**	
Present Value Total Benefits	**$ -**	**$ 130,953**	**$ 115,675**	**$ 102,180**	**$ 90,259**	**$ 439,067**
Costs per Anum						
Development Costs						
Consultant Fees (WPI)	$ 31,200	$ -	$ -	$ -	$ -	
Equipment (Laptops)	$ 1,040	$ -	$ -	$ -	$ -	
Development Training	$ 52,416	$ -	$ -	$ -	$ -	
Initial User Training	$ 1,248	$ -	$ -	$ -	$ -	
Total Development Costs	**$ 85,904**	**$ -**	**$ -**	**$ -**	**$ -**	
Operational Costs						
Annual User Training	$ 1,152	$ 1,221	$ 1,294	$ 1,372	$ 1,454	
Annual Code Maintenance	$ 11,520	$ 12,211	$ 12,944	$ 13,721	$ 14,544	
Total Operational Costs	**$ 12,672**	**$ 13,432**	**$ 14,238**	**$ 15,093**	**$ 15,998**	
Total Costs	**$ 98,576**	**$ 13,432**	**$ 14,238**	**$ 15,093**	**$ 15,998**	
Present Value Total Costs	**$ 98,576**	**$ 11,194**	**$ 9,888**	**$ 8,734**	**$ 7,715**	**$ 136,107**
NPV						**$ 302,960**

3.2.3 Organizational Feasibility

The project presents low organizational risk. The system aims to improve employee efficiency and to drive down costs, which is a value-added service. This is directly aligned with the goals of the automation team. The project champion is Roy Lockhart, Vice President of Quality Assurance Management. His senior position enableshim to initiate organizational change and to provide the project team withaccess to required resources and materials.

There is low risk that system users will not embrace the new system. The new system represents a product that users have been requesting since GPWS inception in June 2005. Risk is present in the user interface design. Poor user interface design can cause a dismal user experience, which may hamper user acceptance. To minimize risk, the following steps were taken. First, system users were interviewed to make interface design an interactive process. Second, extensive documentation will be provided and initial and on-going annual training will be conducted to ensure system acceptance and proper use.

4 Analysis

The analysis phase of the project was completed in A Term. The project scope was completed with the help of the Fidelity staff and includes all the deliverables for the project. The functional and non-functional requirements for each of the deliverables are explained in the following sections.

4.1 Project Scope

The following high-level requirements, as defined by FPCMS, are within the project scope. Functional requirements and non-functional requirements are identified for each deliverable. The WPI team will deliver the Test Environment Dashboard, the End-of-Day Processing Dashboard, the Process Control Interface, and the End-of-Day Processing Interface. The team with not deliver the End-of-Day Statistics Dashboard & Collection Functionality, the test Environment Burn-in, or the Common Database

Queries and Updates. The project deliverables can be seen in Table 9. The functionalities that the project team will not be delivering are summarized in Table 10.

Table 9: Project Deliverables*

Deliverable Type	**Deliverable**
Dashboards	Test Environment Dashboard
	End-of-Day Processing Dashboard
Operational Support	Process Control Interface
	End-of-Day Processing Interface

*Requirement delivery is contingent upon the fact that there must be a way to retrieve all relevant information through a UNIX machine at the time this proposal is accepted.

Table 10: Functionalities outside the scope of the project

Deliverable Type	**Deliverable**
Dashboards	End-of-Day Statistics Dashboard & Collection Functionality
Operational Support	Test Environment Burn-in
	Common Database Queries and Updates

4.2 GPWS Testing Interface

The requirements described below cover the entire Testing Interface that is delivered to FPCMS. These requirements must continue to be met within all dashboards and operational support interfaces and outside of them as well.

The following requirements have been directly transferred from the *File Load Detail Requirements* document that was provided to the WPI team by FPCMS. Since FPCMS has included the specific requirements, we are utilizing and integrating it into our documentation for ease of consistency. For further details about this section, please view the indicated document in Appendix F.

The testing interface includes functionality requirement as well as non-functional requirements. The functional requirements include requirements for the interface design, the interaction between the deliverables, the legibility of the final product, and many more. The complete list of functional requirement can be found in Figure 5.

GPWS Testing Interface Functional Requirements

- GTI implementation should include a web-based, graphical interface. The interface will allow test engineers to view test environment dashboards and perform operational support functions.
- The dashboards and operational support interfaces described in this document may be combined where appropriate.
- GTI cannot negatively affect the performance of the test environment in which it is running.
- GTI needs to function with existing GPWS testing scripts (Korn Shell, Perl).
- GTI must provide a means to monitor a function's progress and report Loaded / failure.
- All screen or log error messages must be clearly understandable by test engineers.
- All screen or log error messages must be consistently formatted and worded.
- All operational support functions must verify that the test engineer wishes to proceed before executing.
- Default values must be provided for all options.
- Deployment of GTI artifacts must ensure the GPI artifacts cannot be inadvertently deployed into the production environments.

Figure 5: GPWS Testing Interface Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

The testing interface also included non-functional requirements. These requirements include important aspects of the deliverable being used by both operational support and functional support teams. Another important requirement is that it must be able to be used in any GPWS test environment as well as having the capability to be accessed from any Fidelity Site. The non-functional requirements can be found in Figure 6.

GPWS Testing Interface Non-Functional Requirements

- GTI may be used by both operational support test engineers and functional test engineers.
- GTI may be used in any GPWS test environment.
- GTI may be accessed from any Fidelity site.
- There must be a means to block access to selected functions, such as database updates or functions that would significantly negatively affect testing if invoked at the wrong time.
- Use of GTI does not require a test engineer to have:
    - UNIX access
    - A GPWS user profile
    - Knowledge of the GPWS database schema

Figure 6: GPWS Testing Interface Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.3 Dashboards

The requirements below cover the general requirements for the dashboards that will be delivered by the WPI team to FPCMS. These requirements must continue to be met to ensure a successful deliverable. The requirements include both functional and non-functional aspects. The functional requirements can be found in Figure 7 and the non-functional requirements can be found in Figure 8.

Dashboard Functional Requirements

- The dashboards use HTML, CSS, and CGI to deliver content to the user. It uses Perl and Korn Shell scripts to retrieve the relevant information or do the relevant tasks.
- The dashboards have the Common Interface Heading (CIH) as a frame at the bottom of the screen, which contains:
    - Today date, which is current calendar date
    - Current business date
    - Test information VAH and description
    - Current GPWS Main state
- The dashboards retrieve the CIH information by using Perl to retrieve the contents of various UNIX environment variables.
- Any status indicator that is in unable to be retrieved will either display "None", or "Error", if it cannot communicate with the system due to a failure to configure the testing interface, or if it cannot communicate with the system for another reason (respectively).

Figure 7: Dashboard Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

Dashboard Non-Functional Requirements

- Each dashboard will operate as a component of the GPWS Testing Interface and users can access it from the secured entry page on the website.
- Users should already be in a secured state (logged in) at the time they access each dashboard. If they are not, it will throw an error to alert them.
- The dashboards do not require the user to have any UNIX experience to use the interface.
- The dashboards do not refresh automatically, the user will have to click the "Refresh" button at the top of the screen for updated values.
- Common status values will be displayed in a particular color depending on the value: "Up" has a green highlight, "Down" has a red highlight, "None" has a gray highlight, "Varied" and "In Progress" have a yellow highlight, "Error" has a white highlight with red text.

Figure 8: Dashboard Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.3.1 Test Environment Dashboard

Each dashboard has its own set of requirements. The following cover the requirements for the Test Environment Dashboards. Like the general dashboard requirements, the specific test environment dashboards include both functional and non-functional requirements. The functional requirements can be found in Figure 9 and the non-functional requirements can be found in Figure 10.

Test Environment Functional Requirements

- Last application start type
    - Valid successful values include: "Cold", or "Warm".
- Scenario under test
    - Valid successful values include: "Normal", "Early Market Close", "US Holiday", "Canadian Holiday", "4-to-6 Test".
- GPWS application processes (JVMs), GPWS WAS processes, GPWS IHS processes, and the GPWS database
    - Valid successful values include: "Up", or "Down".
- Enabled transmissions
    - List of all transmissions that are enabled and what have an active IP address.
- Enabled timer events
    - List of all GPWS timer events that are currently active.
- List of days in each database
    - Shows business days stored in GPWS primary / archive / history databases.

Figure 9: Test Environment Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

Test Environment Non-Functional Requirements

- Each status indicator group will be collectively located in its own box on the website (e.g. Enabled Transmissions, Enabled Timers, etc.)
- If a status value contains "Error", users will be able to click on it and view the error information (if available) in separated location.

Figure 10: Test Environment Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.3.2 File Load Dashboard

As with the GTI general requirements, the following requirements have been directly transferred from the *File Load Detail Requirements* document that was provided to the WPI team by FPCMS. Since FPCMS is building this dashboard, and they have created the requirements documentation for it, we are utilizing and integrating it into our documentation for ease of consistency. For further details about the dashboard, please view the indicated document in Appendix F.

Functional and Non-Functional Requirements

The dashboard displays heading and file load status. The summary of the functional requirements for the common interface heading can be found in Figure 11. The summary of functional requirements for the file load statute can be found in Figure 12. The nonfunctional requirements for both of these are summarize in Figure 13.

Common Interface Heading Functional Requirements

- Today date, which is current calendar date
- Current business date
- Test information VAH and description
- Current GPWS Main state

Figure 11: Common Interface Heading Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

File Load Status Functional Requirements

- GPWS file loads are grouped into Start Of Day, Intraday, APT, Fulfillment, Bond Vendor and ECN
- User has an option to display ALL file load status in ALL groups
- User clicks the group to get status on files in that group. If there is any file load failure in the group, the dashboard will automatically expand that particular group to show each file status
- Source of each files will:
    - Display ProdSaved, Test or AdHoc if the file has been loaded.
    - Display blank if the file has not been loaded
- Display No Load indicator (ON/blank) on each file:
    - Display "ON" when the file does not planned to load for given business day
    - Otherwise display blank
- For each file, display one of the file load status: Loaded, Not Loaded, Fail or Blocked
    - Status Not Loaded when the file has not been loaded
    - Status Loaded when the file has been successfully loaded
    - Status Blocked when the No Load indicator is ON
    - Status Fail when the file load failed. Fail status should be displayed in Red
- When the status display Fail, user can click the word Fail to display details for that specific file
- For files that come multiple times in a business day e.g. intraday, the dashboard only displays one entry
- Display status of fulfillment file load on each hour (0000-1810) from Primary Server only (MBP) for each source (Reuters and S&P)
- The dashboard will not refresh automatically. User clicks the 'Refresh' button to refresh dashboard
- Display these buttons:
    - Button "GPWS Test Interface" – to go back to the GTI main screen
    - Button "File Load Interface" – to go to file load interface
    - Button "Refresh" – to refresh file load dashboard

Figure 12: File Load Status Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

File Load Dashboard Non-Functional Requirements

- User (test engineer or functional tester) access File Load Dashboard via GPWS Testing Interface main screen
- It has open access, no password is required to login [into the dashboard]
- Each GPWS test environment has its own file load dashboard.

Figure 13: File Load Dashboard Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.3.3 End-of-Day Processing Dashboard

The following requirements have also been directly transferred from the *File Load Detail Requirements* document that was provided to the WPI team by FPCMS. Since FPCMS is building this dashboard, and they have created the requirements documentation for it, we are utilizing and integrating it into our documentation for ease of consistency. For further details about the dashboard, please view the indicated document in Appendix F. The functional requirements will show the status, and respective values, of the previous and current End-of-Day heading. The requirements for the previous End-of-Day heading are listed in Figure 14 and the requirements for the current End-of-Day heading are listed in Figure 15. The non-functional requirements for both of these headings can be found in Figure 16.

Previous End-of-Day Heading Functional Requirements

- Calendar date of last EOD processing
    - Displayed as a numerical date.
- Business date of the last EOD processing
    - Displayed as a numerical date.
- Calendar date of last database archiving
    - Displayed as a numerical date.
- Calendar date of last database optimization
    - Displayed as a numerical date.

Figure 14: Previous End-of-Day Heading Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

Current End-of-Day Heading Functional Requirements

- Status of valuation point closure
    - Verify whether the functional test engineers have closed the GPWS valuation points.
- Failed state machine indicator
    - Verifies whether there are any failed state machines.
    - Valid successful values include: “Yes”, and “No”.
        - Users can click on “No” and view which state machine has failed
- Status of “end of day complete”
    - Verifies whether the GPWSMainState state machine has transitioned into EndofDayComplete.
    - Valid successful values include: “Yes”, and “No”
- Status of NASDAQ connections
    - Verifies whether the NASDAQ connections have been deleted
    - Valid successful values include: “Up”, and “Down”
- Status of GPWS application processes (JVMs)
    - Verifies the status of **all** GPWS application processes
    - Valid successful values include: “Up”, “Down”, “Varied”
        - If a user clicks “Varied”, they will get list of which machines are up and down
- Status of outbound files
    - Verifies if all the outbound files have been saved.
    - Valid successful values include: “Yes”, and “No”
- Status of file cleanup
    - Verifies if all inbound, testing, error log, and outbound files have been deleted.
    - Valid successful values include: “Yes”, “No”, “Varied”
        - If a user clicks “Varied”, they will get list of which set of files have been deleted, and which have not
- Status of database connections
    - Verifies if there are any active connections to any of the GPWS databases
    - Valid successful values include: “Yes”, “No”, “Varied”
        - If a user clicks “Varied”, they will get list of which set of files have been deleted, and which have not
- Status of nightly cycle cleanup
    - Valid successful values include: “Run”, “No Run”, “In Progress”
- Status of database archiving
    - Primary to archive database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
    - Archive to history database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
- Status of database optimization
    - Primary database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
    - Archive database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
    - History database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
- Status of database exports
    - Primary database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
    - Archive database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
    - History database
        - Valid successful values include: “Run”, “No Run”, “In Progress”
- Status of EOD statistics
    - Valid successful values include: “Collected”, “Not collected”, “In Progress”

Figure 15: Current End-of-Day Heading Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

End-of-Day Processing Non-Functional Requirements

- Users should already be in a secured state (logged in) at the time they access this dashboard. If they are not, it will throw an error to alert them.
- It does not require the user to have any UNIX experience to use the interface.

Figure 16: End-of-Day Processing Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.4 Operational Support Interfaces

All Operation Support interfaces will have functional and non-functional requirements. The following tables show the general requirements for the operational support interfaces. The functional requirements can be found in Figure 17 and the non-functional requirements can be found in Figure 18.

Operational Support Interface Functional Requirements

- The interfaces use HTML, CSS, and CGI to deliver content to the user. It uses Perl and Korn Shell scripts to retrieve the relevant information or do the relevant tasks.
- The interfaces have the Common Interface Heading (CIH) as a frame at the bottom of the screen, which contains:
    - Today date, which is current calendar date
    - Current business date
    - Test information VAH and description
    - Current GPWS Main state
- The interfaces retrieve the CIH information by using Perl to retrieve the contents of various UNIX environment variables.

Figure 17: Operational Support Interface Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

Operational Support Interface Non-Functional Requirements

- Each task can be clicked by the user to see what command would normally run on the system.
- Each task will have a status indicator specifying whether a task has been completed.
    - Valid successful values include: “Complete”, “Not started” and “In Progress”
    - Valid unsuccessful values include: “None”, and “Failed”
- Common status values will be displayed in a particular color depending on the value: “Completed” has a green highlight, “Not started” has a red highlight, “None” has a gray highlight, “In Progress” has a yellow highlight, and “Failed” has a white highlight with red text.

Figure 18: Operational Support Interface Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.4.1 Process Control

The specific requirements for Process Control are included in the functional and non-functional requirements tables below. The functional requirements will depict what the users will be able to do with the interface and can be seen in Figure 19. The non-function requirements can be found in Figure 20.

Process Control Functional Requirements

- GPWS application processes (JVMs)
    - Tasks to start / stop all or individual JVMs
- GPWS WAS processes
    - Tasks to start / stop all or individual WAS processes
- GPWS IHS processes
    - Tasks to start / stop all of individual IHS processes
- GPWS "Cold" start
    - Task to enable the user to do a "cold" start of GPWS.
- Error log rotation
    - Task to create a new set of application logs, renames a predefined number of old logs, and deletes undesired logs.

Figure 19: Process Control Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

Process Control Non-Functional Requirements

- Verify that a test engineer really wants to stop / start process
- Prevent a test engineer from starting a process that is already running (without stopping the process first).
- Prevent a test engineer from attempting to stop a process that is already stopped.
- For "cold" start:
    - Allow the user to specify a business date
    - Verify that GPWS is in the "end of day complete" state before allowing the "cold" start to run
    - Issue a warning before performing the "cold" start and request additional verification

Figure 20: Process Control Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.4.2 File Load Operational Support Interface

As with the GTI general requirements, the following requirements have been directly transferred from the *File Load Detail Requirements* document that was provided to the WPI team by FPCMS. Since FPCMS is building this interface, and they have created the requirements documentation for it, we are utilizing and integrating it into our documentation for ease of consistency. For further details about the interface, please view the indicated document in Appendix F. The functional requirements for the file load operational support interface can be found in Figure 21 for the file load selection, Figure 22 for file load status, and Figure 23 for log webpage. The non-functional requirements for the file load support interface are that the Operation Engineer is responsible to provide Test / AdHoc files and place them in appropriate directory.

File Load Operational Support Interface Functional Requirements for File Load Selection

- File Load Selection:
  - File load Interface displays the file load groups and their corresponding files:
    - Display groups of file load (Start of Day, Intraday, APT, Fulfillment, Bond Vendor and ECN). Display source (ProdSaved, Test or AdHoc), No Load indicator and Status for each group
    - Display files within each group. Display source (ProdSaved, Test or AdHoc) and No Load indicator and Status for each file
    - If user makes selection on a group level, the individual files within the group are disabled to prevent them to be selected.
    - If user makes selection on the file level, the associated group is disabled
  - User can make multiple selections on groups and on files in different group
  - The source field is display as dropdown. User make selection on one source only, multiple source selection is not allowed.
  - The No Load indicator display as check box. It is checked if the file does not plan to load for the business day. User has ability to reverse the No Load indicator if needed
  - If user set No Load indicator to ON, then the source will be blank and disabled
  - User has an option to load Fulfillment files for :
    - Each hour: 0000, 0730, 1000,…1600, …1810
    - ALL hours (0000 – 1600) with an exclusion of the 1630, 1700, 1810 file load
  - The file load interface enforce the order of file load
  - If prior file load has not been completed when user execute file load, then the interface display error
  - For the files that come multiple times in a business day e.g. intraday files e.g. pg_sw_up, the interface only displays one entry for the file. But it will load all the files
  - If user wants to repeat a file load that is already completed, the interface will ask for confirmation
  - The file load process will abort when having these conditions:
    - Failure on creating file load log
    - Failure in input parameters validation
    - The file to be loaded is missing in source directory
    - If find multiple files when expected to see only one file in the source directory
    - Failure when loading any Start of Day file
  - The file load process will not abort when one or more of multiple file load fails e.g. APT. But the errors are displayed in the return webpage
  - Display these buttons:
    - Button "GPWS Test Interface" – to go back to the GTI main screen
    - Button "File Load Dashboard" – to access file load dashboard
    - Button "Display Log" – bring user to Log Webpage to view file load activities from GPWS log file
    - Button "Refresh" – to refresh the file load interface

Figure 21: File Load Operational Support Interface Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

File Load Operational Support Interface Functional Requirements for File Load Status

- File Load Status:
  - Display file load status as Blank, Loaded, Blocked or Fail
    - Status Blank if the file has not been loaded
    - Status Loaded if the file has been successfully loaded
    - Status Blocked when the No Load indicator is ON
    - Status Fail when the file load failed.
  - Status is displayed on file level only and not on group level
  - Fail status is displayed in Red. User can click the word Fail to display details for that specific file

Figure 22: File Load Operational Support Interface Functional Requirements File Load Status (Anwar, Helbick, Mehra, & Molgaard, 2008)

File Load Operational Support Interface Functional Requirements for Log Webpage

- Log Webpage:
  - Display file load activities based on user selection:
    - Current business day
    - File load date timestamp
    - INFO messages from gpws.log
    - ERROR messages from gpws.log
    - Status of file load

Figure 23: File Load Operational Support Interface Functional Requirements Log Webpage (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.4.3 End-of-day Processing Interface

The end-of-day processing interface requirements are included in the two following tables. The functional requirements describe the interactions the user will have with the interface and the tasks the user will be able to accomplish. The requirements can be found in Figure 24. The non-functional requirements can be found in Figure 25.

End-of-Day Processing Interface Functional Requirements

- Verification that business processing is done for the business day
- Deletion of the NASDAQ connections
- Process termination
- Collection of outbound files
- Directory cleanup
- Termination of active database connections
- Running nightly cycle cleanup
- Performing database archiving and optimization
- Performing database exports
- Collection of EOD statistics

Figure 24: End-of-Day Processing Interface Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

End-of-Day Processing Interface Non-Functional Requirements

- Automatically execute the EOD checklist tasks
- Override selected tasks where applicable
- Prevent the override of selected tasks where applicable
- Determine when a task is completed successfully or fails
- Enforce the order of EOD tasks where appropriate
- Prevent a test engineer from executing an EOD task before previous tasks are complete
- Verify that the test engineer really wants to repeat an EOD task that is already completed

Figure 25: End-of-Day Processing Interface Non-Functional Requirements (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.5 PIECES Framework

The PIECES framework is an analytical framework that ensures problems and opportunities are classified appropriately, and it provides a structured way to approach the identification of business opportunities. The PIECES framework analyzes a situation from the following perspectives: performance, information, economics, control, efficiency, and service. Brief explanations of these perspectives are provided in the following sections.

4.5.1 Performance

This project aims to reduce test time duration through process automation. By automating process execution, GPWS staff can focus on other tasks (e.g. development) to add value to the organization. This performance increase also coincides with a reduction in testing errors, which further streamlines the testing process.

Additionally, this project increases performance through the implementation of an intuitive, web-based interface that allows functional test engineers to execute processes previously done by operational support test engineers. This is a performance increase because functional test engineers are no longer dependent upon operational support test engineers to perform critical test functions. This reduces testing time duration and frees operational support test engineers to perform value-added activities.

4.5.2 Information

One of the primary goals of this project is to make test environment and test process data more visible to end users. The proposed solution enables functional test engineers and operation support test engineers to more easily access critical testing data and statistics through graphical interfaces.

4.5.3 Economics

This project strives to automate numerous testing procedures, which enables Fidelity Investments to better allocate its resources. This simultaneously cuts costs and adds value to the organization. Costs are cut because fewer staff is required to execute GPWS testing. Value is added because unneeded staff can reallocated to projects (e.g. development) where value is added.

4.5.4 Control

The current system is prone to significant testing errors that may harm testing progress or may affect development environments. By automating testing processes and including a system of checks and balances, testing control is increased and the probability of testing error is reduced.

4.5.5 Efficiency

The underlying issue addressed by the proposed solution is efficiency. The current testing process is inefficient because it requires excessive manual labor in situations where automation can be applied.

Through a combination of business process automation and business process improvement, GPWS testing duration will be reduced. Time and resources gained from the proposed solution's efficiency increases can directed towards value adding functions in the organization.

4.5.6 Service

Since GPWS testing is not a forward facing operation, service impact is minimal. One possible service benefit is increased development team satisfaction because the testing team is able to process GPWS updates faster.

4.6 Scope Management

Project scope management is necessary to prevent scope creep, which can lead to schedule and cost overruns. To minimize scope creep, project scope changes will only be allowed at two decision points: 1 and 2.Decision Point 1 refers to activities conducted before any business analysis is conducted. This includes onsite conversations with GPWS staff, onsite observation, and consulting group discussion. Decision Point 2 is the final decision point in this project during which the project scope can be changed. This decision point immediately precedes the design phase. Any recommended to changes to project scope after this decision point can only be reflected in future releases.

4.7 Project Approach

The project approach includes the project strategies and the staffing that we have for the project. The project strategies were established by the team at the opening of the project. Many of these strategies were implemented to ensure a successful project. The staffing of this project is mainly the WPI project team.

4.7.1 Project Strategies

The following strategies will be employed by the WPI MQP team to maximize the chance of project success. These principles will help ensure that the project deadline is reached, the cost benefit analysis of this project is achieved, and the users at Fidelity find the final product fitting to their specified needs. The project strategies can be found in Figure 26.

Project Strategies

- Use methodical approach to frame thoughts, such as the PIECES framework.
- Seek to understand the business value and the business implications of an issue first.
- Apply a technological solution to an issue only after understanding the business implications.
- Create a feedback loop with end users to ensure that the progress on the GPWS interface is meeting the indicated needs.
- Establish clear control mechanisms for handling project scope creep.
- Develop prototypes for end users to provide feedback before completing entire system.
- Employ proactive risk management practices to mitigate risks. Risks will be identified, assessed, and mitigated by identifying ways to minimize the effect of the risk.
- Employ a detailed change management plan to maximize success of system rollout. The rollout strategy entails a two week training period in which departmental employees will be trained.
- Develop maintenance and support strategies to ensure project success after implementation.
- Communicate openly among group members to ensure project work remains coordinated. The group will meet at least four times a week on Sunday, Monday, Tuesday (with WPI advisor), and Wednesday (at Fidelity). The group will meet as often as necessary. Tasks are to be completed together as a team. Communication is always ongoing over e-mail, phone, and instant messenger.
- Manage the project by utilizing e-mail. E-mail is used in-between the completion of project deliverables.

Figure 26: Project Strategies (Anwar, Helbick, Mehra, & Molgaard, 2008)

4.7.2 Major Project Milestones

The project team has established three major project milestones. In A term, the analysis portion of the project will be completed. Design will be completed in B term and then pass on to the Fidelity staff to test over the winter recess. In C term, the implementation phase will be completed along with finalizing documentation. The milestones can be found in Table 11.

Table 11: Major Project Milestones

Milestone/Deliverable	Target Date
Analysis – A term	October 15, 2008
Design – B term	December 17, 2008
Implementation – C term	March 6, 2008

4.7.3 Project Staffing

Four senior students staff the project: Elizabeth Carey, Dan Dahlberg, Michael Diamant, and Augustina Mills.

Elizabeth is a Management Information Systems major. She is currently enrolled in Psychology classes. The information she gains from the psychological standpoint will help the group better understand the underlying personal issues associated with this project. Her analytical skills will help in implementing the system requirements.

Dan has a strong background in both Management Information Systems and Computer Science. His skills will be used for the implementation portion. His previous general knowledge of Perl/CGI will help in the coding and documentation of the project. His background in IT Security and Support will also add great overall knowledge to the team.

Michael has a background in Management Information Systems and Computer Science, which will be needed for the implementation phase of the project. Michael's analytical skills will help translate business requirements into technical requirements.

Augustina is a Management Information Systems major. She is enrolled in several Organization Behavior and Change courses this year. The information gathered from these courses will help in maintaining a healthy work environment and positive group dynamics. Her knowledge of group relations and information gathering can help the team maintain good relations with Fidelity as well as continuing to ensure the company's needs are met.

Collectively, the WPI team has a well-balanced background to understand the business, human, and technical needs of this project. The group is confident in their abilities to present a proper testing interface for GPWS.

4.8 Risk Assessment and Management

In addition to project risks stated in feasibility analysis, additional risks are assessed below. Mitigation steps are described to minimize the impact of each potential project risk. The probability of risk occurrence and risk impact are judged on a scale from 1 to 5, where 1 indicates minimal likelihood or impact and 5 indicates maximum likelihood or impact. The risk assessment below in Table 12 includes the probability of the risk on a 1-5 scale, the impact of the risk on a 1-5 scale, and mitigation steps that can be taken in regards to the risk.

Table 12: Risk Assessment

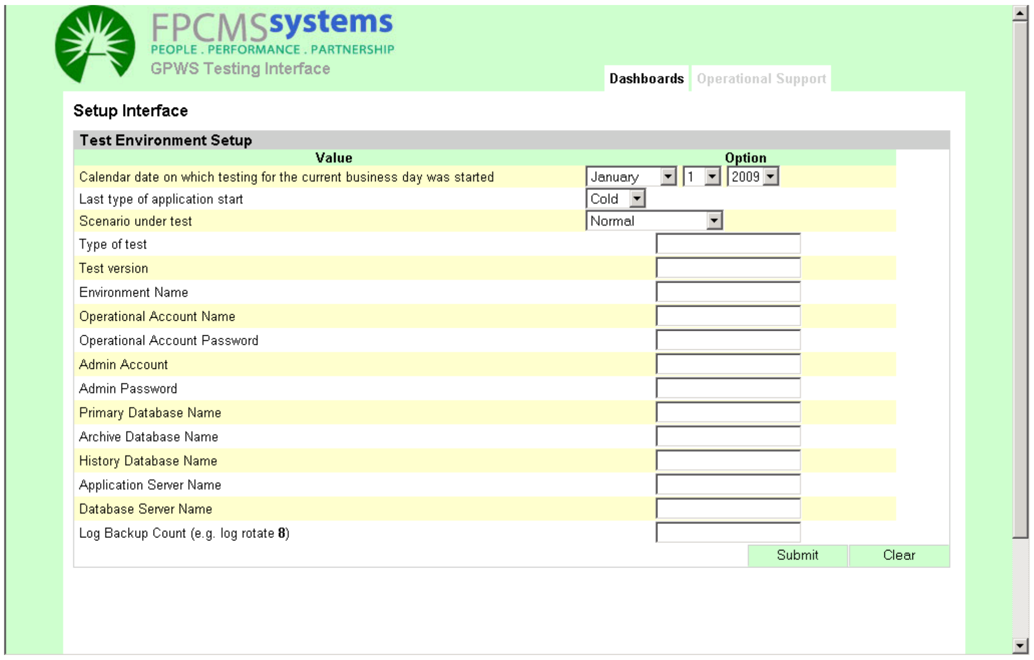
Risk Description	Probability	Impact	Mitigation Steps
WPI project team is not familiar with all business-related processes.	3	2	Communicate unknowns to appropriate resource(s) within FPCMS.
WPI project lacks required access rights to GPWS servers, documentation, etc.	1	3	Communicate problem(s) to Fidelity team resources.
WPI project team is unable to solve a technical problem due to Perl unfamiliarity.	4	5	Reuse Perl scripts (e.g. Fidelity proof of concept). Utilize online resources (e.g. Safari Books Online) and leverage FPCMS knowledge base. Pad project schedule to allow time for uncertainties.
Fidelity team resources are unable to respond quickly to WPI project team concerns due to other project involvement.	2	4	Ask Fidelity resources to provide contact information of other candidate problem resolvers. Pursue additional resources through HR representative, Lori Keenan.
Scripts selected for automation require modification to be implemented.	2	2	Determine early in project development, which scripts are needed and assess necessary modifications. Pad project schedule to account for unexpected script modifications.
GPWS testing interface does not match expectations of end users (functional test engineers and operational support test engineers).	2	5	Generate interface mock ups before interface development and request feedback from end users to ensure good fit between development and expectations.
Code fails to meet Fidelity coding standards.	1	5	Review Fidelity Perl community's best practices for Perl and CGI development. Leverage Fidelity team resources as soon as uncertainty arises. Schedule code review sessions to verify consistent development practices.
WPI project team fails to develop promised deliverables.	2	5	Learn Perl and develop proofs of concept before requirement specification. Review proposed deliverables with Fidelity team resources. Pad project schedule for unexpected development issues.

5 Design

The interface was carefully designed by the MQP team with careful consideration of the reviews given by current Fidelity employees. The following sections include screenshots of the developed interfaces along with a description of the function of the page. The training manual can be found in Appendix C.

5.1 Setup Interface

The Setup Interface is accessed the first time that the web application is put in a new environment. The interface allows users to enter in values that are needed for the application to run. If there is no setup file detected, the user is automatically directed to the setup page. The setup interface can be seen below in Figure 27.



Setup Interface

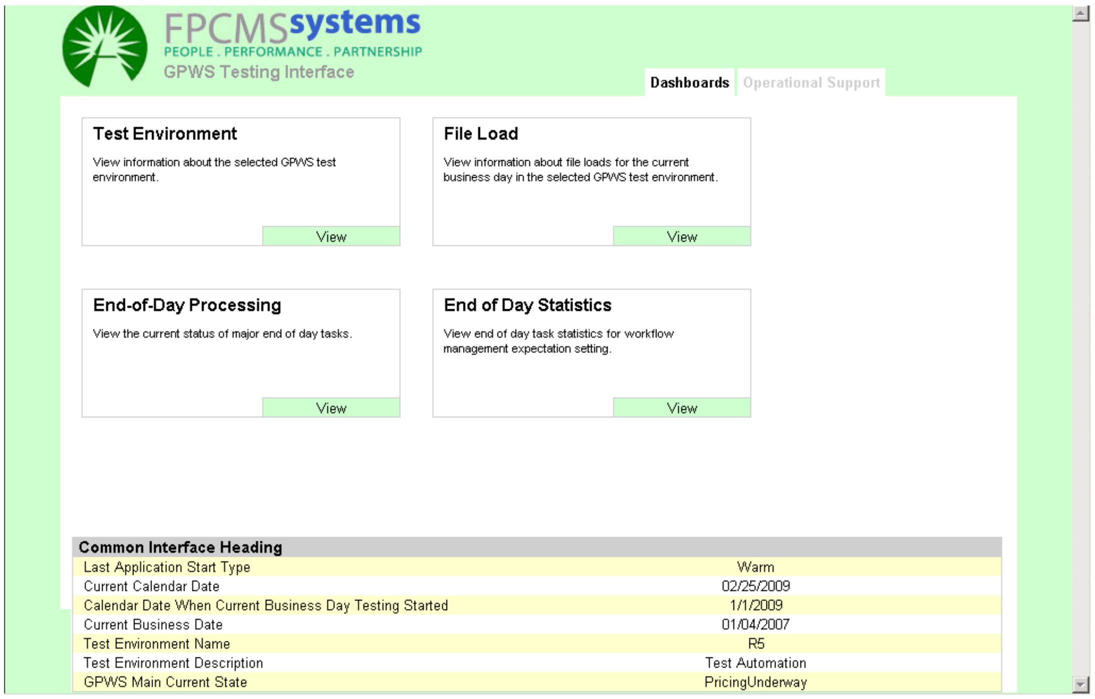
Test Environment Setup

Value	Option
Calendar date on which testing for the current business day was started	January 1 2009
Last type of application start	Cold
Scenario under test	Normal
Type of test	
Test version	
Environment Name	
Operational Account Name	
Operational Account Password	
Admin Account	
Admin Password	
Primary Database Name	
Archive Database Name	
History Database Name	
Application Server Name	
Database Server Name	
Log Backup Count (e.g. log rotate **8**)	

Figure 27: Setup Interface

5.2 Dashboard Index

The Dashboard Index is the main page to navigate through the dashboards. The dashboard tab is bolded at the top to confirm what tab you have navigated to. From this tab you can view the Test Environment or the End-of-Day Processing. The index includes the Common Interface Heading, which is standard throughout every page. The dashboard index can be seen in Figure 28.



Test Environment

View information about the selected GPWS test environment.

File Load

View information about file loads for the current business day in the selected GPWS test environment.

End-of-Day Processing

View the current status of major end of day tasks.

End of Day Statistics

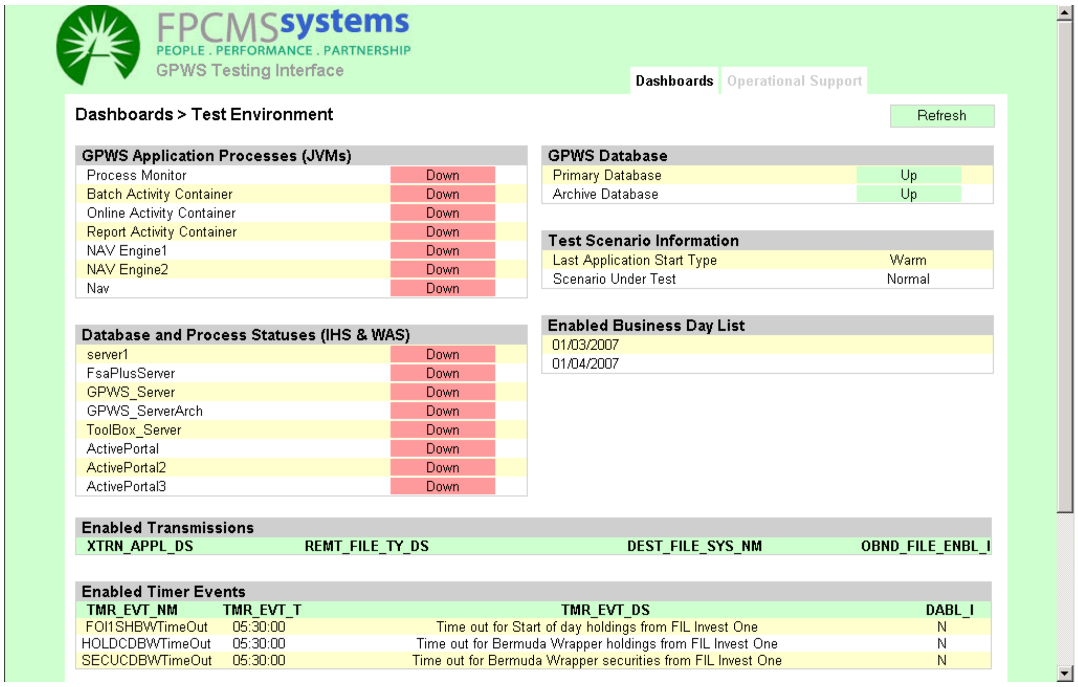
View end of day task statistics for workflow management expectation setting.

Common Interface Heading	
Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

Figure 28: Dashboard Index

5.3 Test Environment Dashboard

The Test Environment Dashboard can be seen once the view button is clicked on the previously mentioned Dashboard Index. All information relevant to the test environment is displayed on this page. The Test Environment Dashboard also includes the status of different elements in the environment. Below in Figure 29 is a screen shot of the Test Environment Dashboard, which you can see the types of information included on the page.



Dashboards > Test Environment

GPWS Application Processes (JVMs)	
Process Monitor	Down
Batch Activity Container	Down
Online Activity Container	Down
Report Activity Container	Down
NAV Engine1	Down
NAV Engine2	Down
Nav	Down

GPWS Database	
Primary Database	Up
Archive Database	Up

Test Scenario Information	
Last Application Start Type	Warm
Scenario Under Test	Normal

Enabled Business Day List
01/03/2007
01/04/2007

Database and Process Statuses (IHS & WAS)	
server1	Down
FsaPlusServer	Down
GPWS_Server	Down
GPWS_ServerArch	Down
ToolBox_Server	Down
ActivePortal	Down
ActivePortal2	Down
ActivePortal3	Down

Enabled Transmissions

XTRN_APPL_DS	REMT_FILE_TY_DS	DEST_FILE_SYS_NM	OBND_FILE_ENBL_I

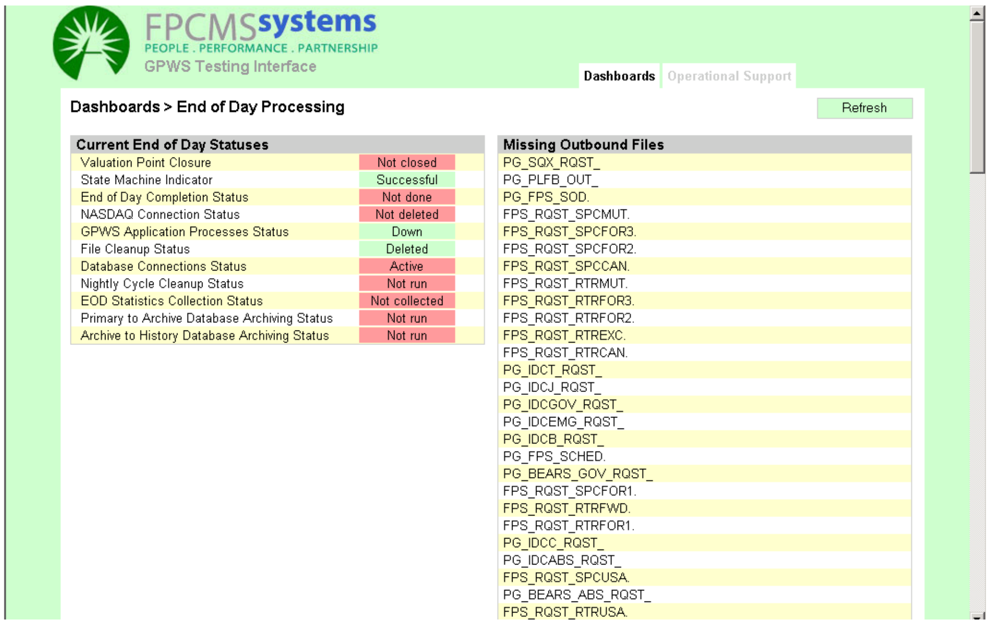
Enabled Timer Events

TMR_EVT_NM	TMR_EVT_T	TMR_EVT_DS	DABL_I
FOI1SHBWTimeOut	05:30:00	Time out for Start of day holdings from FIL Invest One	N
HOLDCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper holdings from FIL Invest One	N
SECUCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper securities from FIL Invest One	N

Figure 29: Test Environment Dashboard

5.4 End-of-Day Processing Dashboard

The End-of-Day Processing Dashboard can be seen once the view button is clicked on the previously mentioned Dashboard Index. All information relevant to the end-of-day process is displayed on this page. The End-of-Day Processing Dashboard also includes the status of different elements in the environment. Below in Figure 30 is a screen shot of the dashboard showing the Current End of Day Status as well as a portion of the Missing Outbound Files.



Dashboards > End of Day Processing

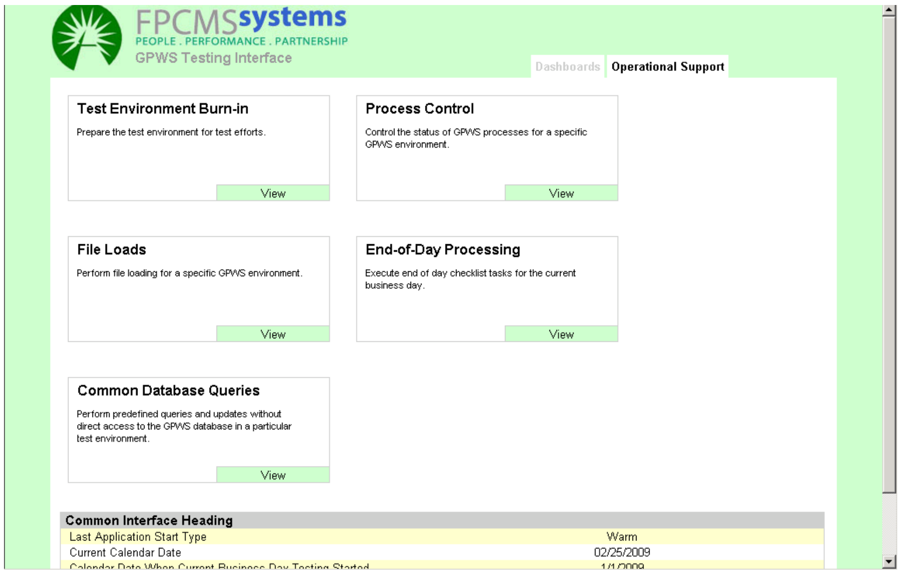
Current End of Day Statuses	
Valuation Point Closure	Not closed
State Machine Indicator	Successful
End of Day Completion Status	Not done
NASDAQ Connection Status	Not deleted
GPWS Application Processes Status	Down
File Cleanup Status	Deleted
Database Connections Status	Active
Nightly Cycle Cleanup Status	Not run
EOD Statistics Collection Status	Not collected
Primary to Archive Database Archiving Status	Not run
Archive to History Database Archiving Status	Not run

Missing Outbound Files
PG_SQX_RQST_
PG_PLFB_OUT_
PG_FPS_SOD.
FPS_RQST_SPCMUT.
FPS_RQST_SPCFOR3.
FPS_RQST_SPCFOR2.
FPS_RQST_SPCCAN.
FPS_RQST_RTRMUT.
FPS_RQST_RTRFOR3.
FPS_RQST_RTRFOR2.
FPS_RQST_RTREXC.
FPS_RQST_RTRCAN.
PG_IDCT_RQST_
PG_IDCJ_RQST_
PG_IDCGOV_RQST_
PG_IDCEMG_RQST_
PG_IDCB_RQST_
PG_FPS_SCHED.
PG_BEARS_GOV_RQST_
FPS_RQST_SPCFOR1.
FPS_RQST_RTRFWD.
FPS_RQST_RTRFOR1.
PG_IDCC_RQST_
PG_IDCABS_RQST_
FPS_RQST_SPCUSA.
PG_BEARS_ABS_RQST_
FPS_RQST_RTRUSA.

Figure 30: End-of-Day Processing

5.5 Operational Support Index

The Operational Support Index is the main page to navigate through the Operational Support pages. Like with the Dashboard Index, Operational Support is bolded at the top to confirm what tab you have navigated to. From this tab, you can view the status of both Process Control and End-of-Day Processing. The index also includes the Common Interface Heading, which is standard throughout every page. Below you can view the Operational Support Index in Figure 31.



Test Environment Burn-in
Prepare the test environment for test efforts.
View

Process Control
Control the status of GPWS processes for a specific GPWS environment.
View

File Loads
Perform file loading for a specific GPWS environment.
View

End-of-Day Processing
Execute end of day checklist tasks for the current business day.
View

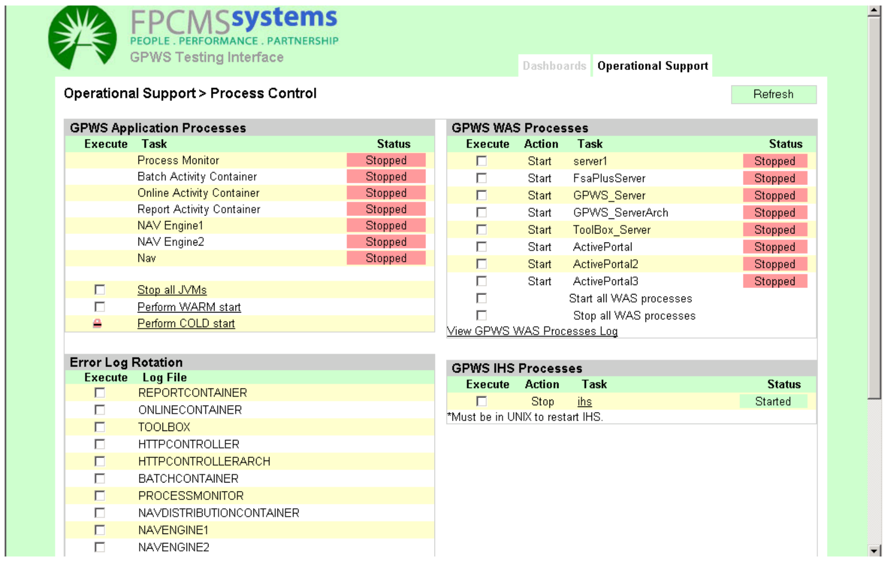
Common Database Queries
Perform predefined queries and updates without direct access to the GPWS database in a particular test environment.
View

Common Interface Heading	
Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009

Figure 31: Operational Support Index

5.6 Process Control

The Process Control page includes the background processes for GPWS to work with status. If a particular function is stopped, you can start it on this page, and likewise if it is started you can stop it. An Error Log Rotation is also included on this page. Below you can view what processes are include on the page in Figure 32.



Operational Support > Process Control

GPWS Application Processes

Execute	Task	Status
	Process Monitor	Stopped
	Batch Activity Container	Stopped
	Online Activity Container	Stopped
	Report Activity Container	Stopped
	NAV Engine1	Stopped
	NAV Engine2	Stopped
	Nav	Stopped
☐	Stop all JVMs	
☐	Perform WARM start	
	Perform COLD start	

GPWS WAS Processes

Execute	Action	Task	Status
☐	Start	server1	Stopped
☐	Start	FsaPlusServer	Stopped
☐	Start	GPWS_Server	Stopped
☐	Start	GPWS_ServerArch	Stopped
☐	Start	ToolBox_Server	Stopped
☐	Start	ActivePortal	Stopped
☐	Start	ActivePortal2	Stopped
☐	Start	ActivePortal3	Stopped
☐		Start all WAS processes	
☐		Stop all WAS processes	

View GPWS WAS Processes Log

Error Log Rotation

Execute	Log File
☐	REPORTCONTAINER
☐	ONLINECONTAINER
☐	TOOLBOX
☐	HTTPCONTROLLER
☐	HTTPCONTROLLERARCH
☐	BATCHCONTAINER
☐	PROCESSMONITOR
☐	NAVDISTRIBUTIONCONTAINER
☐	NAVENGINE1
☐	NAVENGINE2

GPWS IHS Processes

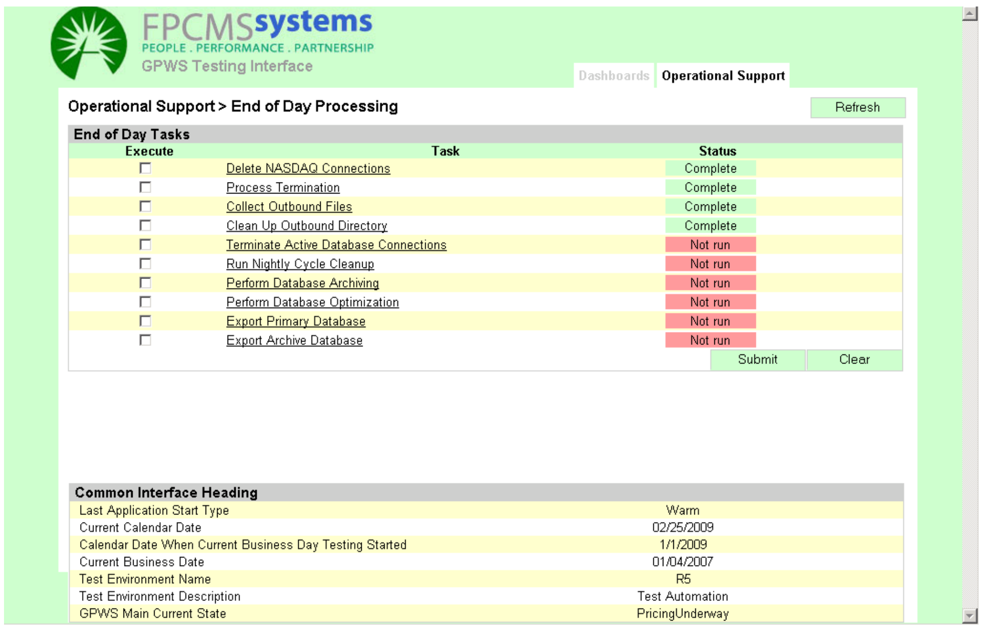
Execute	Action	Task	Status
☐	Stop	ihs	Started

*Must be in UNIX to restart IHS.

Figure 32: Process Control

5.7 End-of-Day Processing

The End-of-Day Processing Interface allows you to execute processes by checking of the execute box. To view a log for a particular task, the user must click on the task title. The tasks are run in order of the list from top to bottom. If the user clicks on a task that is not next to be executed, an error message will appear. Below in Figure 33 is a screen shot of the End-of-Day Processing interface.



Operational Support > End of Day Processing

End of Day Tasks

Execute	Task	Status
☐	Delete NASDAQ Connections	Complete
☐	Process Termination	Complete
☐	Collect Outbound Files	Complete
☐	Clean Up Outbound Directory	Complete
☐	Terminate Active Database Connections	Not run
☐	Run Nightly Cycle Cleanup	Not run
☐	Perform Database Archiving	Not run
☐	Perform Database Optimization	Not run
☐	Export Primary Database	Not run
☐	Export Archive Database	Not run

Common Interface Heading

Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

Figure 33: End-of-Day Processing

6 Installation Guide

The installation guide provides information on how to implement GTI. It reviews the structure of GTI and provides the location for the final version of the interface that was created. The IBM HTTP Server (IHS) configuration is also included in this section which includes code that will be needed for implementation. The final section refers to the error codes. Eight error codes are highlighted followed by the reason for the error and troubleshooting.

6.1 GTI structure

The final version of the interface should be contained in a zip file labeled 'gti-20090302.zip'. However, if any subsequent in-house updates or modifications were made since this document was originally written, use the latest version available. The directory structure of the new versions should be consistent with the structure of the original version, which is as follows in Figure 34.

Directory Structures

- ./GTI/cgi
    - Contains all CGI files that do not need to be password protected to access. These include index.cgi, setup.cgi, and the available dashboards.
- ./GTI/cgi-protected
    - This folder contains all the CGI files that should not be accessible by everyone. These include the operational support interfaces.
- ./GTI/config
    - Any configuration document used in the operation of the interface is stored here. This includes the setup.cfg, which contains the values that were entered in the setup.cgi.
- ./GTI/files
    - Contains the status values that are used to track the progress of EOD operational support interface and the general condition and state of GPWS in the EOD dashboard
- ./GTI/log
    - Every executed command and task has a log file written to this directory. The openlog.cgi is contained here as well, opening the corresponding log when requested.
- ./GTI/scripts
    - This directory has all the various Perl scripts that are executed by and included in the CGI pages. This may be scripts that execute operational support tasks (e.g. handle_process_control.pl) to subroutines that are used throughout the interface (common.pl)

Figure 34: Directory structures based on original version

As long as the directory structure remains the same when uploading the interface, the location where the files are placed on the server are irrelevant for reasons discussed in the next section.

6.2 IHS configuration

There are multiple changes that need to be made to the httpd.conf of IHS for the GPWS Testing Interface to operate appropriately. These two sections focus on the addition of aliases and the configuration of the protected CGI directory. If you are already familiar with how to configure a CGI website, then you can feel free to skip the major portions of this document and focus on the actual configuration, highlighted in bold.

There are two types of aliases in httpd.conf , 'Alias' and 'ScriptAlias'. Our interface requires two additional ScriptAlias lines and one additional Alias added to the default httpd.conf. Our interface does not require itself to be in any specific directory, in fact, it can be put into any directory on the AIX server. This is possible because the configuration that we add to the httpd.conf specifically addresses where on the server the interface is located.

The 'Alias' tag maps out a directory on the local server to a virtual directory on the web server. For example, the Alias line:

Alias /gti/ /var/www/html/applications/gti/

Tells the web server that the contents within the /var/www/html/applications/gti/ directory should appear to the user as if they were in http://webserver/gti/. Without this alias, the web server would not even know that directory on the local server even exists.

The 'ScriptAlias' tag incorporates the same features from the 'Alias' tag but is also used to inform the IHS server that there are only scripts located in this directory and that they need to be executed by the 'apache' user. The executed scripts are then displayed to the user trying to browse that script. In this context, the word 'script' is synonymous with 'CGI script' but it does not imply that a user can execute a standard Perl script from this directory either. While it may be feasibly possible to do this, it is not standard practice since the user viewing the page will not know what is happening as the output from traditional Perl scripts is not formatted in HTML.

For the GTI interface, the followings lines in Figure 35 will have to be added to httpd.conf.

```
Lines for httpd.conf

ScriptAlias /gti/cgi-protected/ /path/to/gti/cgi-protected/

ScriptAlias /gti/cgi/ /path/to/gti/cgi/

Alias /gti/ /path/to/gti/
```

Figure 35: Lines to add to httpd.conf for GTI Interface

Ensure that “/path/to/gti” has been substituted by the correct path. Once these lines have been added, IHS will have to be restarted. The interface should be accessible by now going to http://webserver/gti/cgi/index.cgi

One feature of the GPWS Testing Interface is to not allow individuals without a username and password to access the CGI pages that are deemed as 'protected.' These pages are located in a special 'cgi-protected' directory in the main tree of GTI. Within this directory is an .htaccess and .htpasswd file. These files determine the usernames and passwords that will allow users to access the page. The files can be configured in such a way that each user can have his or her own password. However, the current installation is setup with only one username and password, which are both “gpws”.

By default, the IHS server does not support such a configuration that we have made for GTI. To allow the password protection to be used, the specific “protected” directory will need to be configured in the httpd.conf. The following lines in Figure 36 need to be added to the file.

```
Password Protection

<Directory /path/to/gti/cgi-protected>

AuthUserFile /path/to/gti/cgi-protected/.htpasswd

AuthName "GPWS Testing Interface Operational Support Interfaces"

AuthType Basic

require valid-user

</Directory>
```

Figure 36: Lines to be added to allow password protection

Ensure that "/path/to/gti" has been substituted by the correct path. Once these lines have been added, IHS will have to be restarted.

6.3 Error Codes

The following section includes eight error codes. The information that follows each error is an explanation of why this code may appear and how to troubleshoot a particular error.

Error Code 1: "A process is either starting or stopping, or information about the WAS processes is being gathered by another user. Please try again later."

This message is triggered whenever the interface notices that there is an existing "startwas", "stopwas", or "statwas" process running. This is done since starting a "statwas" process while another process is starting will generally freeze the "statwas" script, causing the page to never load. Simply refresh the page a minute or two later, and as long as those processes are not running anymore, the page will load as normal.

Error Code 2: "Untaint error: Bad data in ..."

An untaint error occurs whenever the interface notices non-traditional characters being passed in a variable that only expects alphanumerical characters for traditional variables and anything besides alphanumerical characters, slashes, hyphens, and underscores for variables that contain paths. This error should not occur often unless something has changed with GPWS itself that GTI did not take into account for (e.g., file structure and changed locations for binary files).

Error Code 3: "Bad data in .."

Same error as the one located above. This error should not occur often unless something has changed with GPWS itself that GTI did not take into account for (e.g., file structure and changed locations for binary files).

Error Code 4: "Failed to open statuses file: ..."

If the permissions are incorrectly set or the file is not in the correct directory, this error will trigger.

Error Code 5: "Could not create ksh script for error log rotation."

This error will display if the user writing the ksh script (the web server user) does not have permissions to generate a file in the desired directory.

Error Code 6: "Failed to open config file: ..."

If the config file cannot be located, or if the permissions on the file or directory are incorrectly set, this error will appear.

Error Code 7: "Failed to open log file"

If the config file cannot be located, or if the permissions on the file or directory are incorrectly set, this error will appear.

Error Code 8: "Could not open error log rotation file list."

If the config file cannot be located, or if the permissions on the file or directory are incorrectly set, this error will appear.

7 Conclusion & Recommendations

The main objective of this project was to improve the efficiency of Fidelity Investment's fund price verification process through business process automation and user-friendly interface development. The MQP team achieved this goal by creating a web-based tool to interface with Global Pricing Workstation (GPWS) test environments on the UNIX platform. In particular, the project was completed in a way to ensure that it could also be used as a template for automating current and/or future applications within other groups at Fidelity.

As FPCMS scales the GPWS testing interface to other applications, FPCMS should consider implementing AJAX (asynchronous JavaScript and XML). AJAX would be most helpful in updating status values on dashboards. Using AJAX to refresh status values will save bandwidth and more importantly, it will save users time. The GPWS testing interface is not performance tuned. As such, users may encounter long load times because of the numerous scripts required to execute in the background. When users refresh an interface page, such as a dashboard, all scripts must be executed again. This leads to a long load time that may be an inconvenience to the user. An AJAX implementation for the status values on a page will curtail the amount of scripts running unnecessarily, which will translate into a shorter load time and a better user experience.

The lessons that the MQP team has gained from this experience can be categorized into two sections, technical and intrapersonal lessons. The MQP team grasped the concept and importance of creating reusable code. Creating code was necessary so that it can be used toward the building out of the remaining pieces of GPWS. Reusable code can also be used to construct other testing interfaces like ITAC. Another technical lesson we learned was the necessity of detailed requirements. Since the MQP team was working with a large system, the room for error and miscommunication may have been great if the team was not provided with detailed instructions. The value learned was that with good requirements, a project has a stronger chance of success.

In addition, many intrapersonal lessons were learned as well. It was important for there to be an integration between the business and the technology. The MQP team saw the importance of where our major fits in with both subjects. To complete this project it was essential to integrate our financial domain expertise and our technical skills to implement testing functionalities on the interface. The other main intrapersonal lesson we learned was the perspective of a consultant role. The MQP team was able to gain a sense of what a consultant's work entails. The MQP team was faced with a very large application that were not familiar with as well as being introduced to a new corporation and culture. In a short amount of time, we had to understand how the system being worked on functioned and how to implement new functionality. This all required communication with the members of FPCMS and in turn take their feedback and implement this into the project solution.

In all, the team strongly believes that the MQP experience was quite valuable. We learned a great deal about the business and technology of Fidelity as well as had the opportunity to scholastically challenge ourselves. These experiences will be of use to each member of this team in our future career paths.

8 Bibliography

Anwar, R., Helbick, K., Mehra, W., & Molgaard, E. (2008). *GPWS Testing Interface: Fileload Detail Requirements*. Fidelity Investments.

Exforsys Inc. (2008). *Automated Testing Advantages, Disadvantages and Guidelines*. Retrieved October 2008, from Exforsys Inc: http://www.exforsys.com/tutorials/testing/automated-testing-advantages-disadvantages-and-guidelines.html

FMR LLC. (2008). *Company Overview*. Retrieved September 2008, from Inside Fidelity: http://personal.fidelity.com/myfidelity/InsideFidelity/index.html

Hoovers. (2008). *FMR LLC*. Retrieved September 2008, from http://premium.hoovers.com/subscribe/co/history.xhtml?ID=ffffcfrhffshxhjyyt

9 Glossary

Acronym	Term	Definition
BA	Business Analyst	Individual responsible for analyzing the business needs of their clients to help identify business problems and propose solutions.
FPCMS	Fidelity Pricing and Cash Management Services	Overarching department that contains FRAPS BA and TDS.
FRAPS BA	Fidelity Reporting, Accounting and Pricing Services Business Analysis Team	Responsible for reviewing and testing GPWS business functionality during GPWS UAT.
GPWS	Global Pricing Workstation	Software application used by Accounting and Pricing Operations group within FPCMS to collect, validate, and approve the current price of securities held by .Fidelity funds.
GTI	GPWS Testing Interface	Result of completed project.
SIT	System Integration Testing	Testing process that exercises a software system's coexistence with others.
TDS	Test Delivery Services	Part of the FPCMS Center of Excellence Testing & Quality Assurance Central Services that is responsible for supporting SIT/UAT operational tasks (including new and existing functionality review) and test environments.
UAT	User Acceptance Testing	System users perform tests to verify business requirements implementation.

10 Appendices

A. Weekly Meeting Agendas

Fidelity MQP Meeting

Tuesday, September2nd, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Dan Dahlberg (Moderator), Michael Diamant (Secretary), Elizabeth Carey, Augustina Mills

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Project Goals and Objectives
4. Review of Introduction Meeting
5. Discussion of Project Charter
6. Outline of MQP Paper Structure
7. Questions / Comments

Minutes

Professor Djamasbi discussed how the project is going to be structured what we expect to do the first day we visit Fidelity. Various meetings are scheduled to give the students an introduction to the project. Professor Djamasbi discussed that this project will be very similar to the Systems Analysis Class (MIS4720). She continued by discussing what we expect to do each term: A term: Requirement analysis to generate proposal, B term: Implement the proposal, have them test it over winter break, C term: Generate documentation and conduct training. She reminded the students to dress and act professionally as we will be representing WPI throughout the entire project.

Attachments

I. WPI Project: GPWS Testing Interface (already received by all parties)
II. GPWS Testing Interface Requirements (already received by all parties)

Fidelity MQP Meeting

Tuesday, September9th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Michael Diamant (Moderator), Augustina Mills (Secretary), Elizabeth Carey, Dan Dahlberg

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review of Project Plan
4. Review of Day Planned at Fidelity
5. Questions / Comments

Minutes

Dan Dahlberg began the meeting by reviewing the previous week's minutes. This led into a discussion about project goals and objectives. Each project member gave his/her thoughts about the product of the project. Following the project goals discussion, Dan reviewed the project charter that was given to the group during the previous Fidelity visit. Finally, Professor Djamasbi provided an overview of MQP documentation structure and electronically distributed two documents: (1) MQP documentation checklist and (2) Guide to Successful MIS MQPs. The deliverable for the next meeting is a project plan.

Attachment

I. Current version of Project Plan

Fidelity MQP Meeting

Tuesday, September16th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Augustina Mills (Moderator), Elizabeth Carey (Secretary), Dan Dahlberg, Michael Diamant

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review of Project Plan
4. Review Attachments
   a. Revisions to Documentation
   b. Review project questions
   c. Acronyms
   d. Feasibility Analysis
   e. In Progress: Literature Review and Organizational Chart
5. Review of Day Planned at Fidelity (leaving early)
6. Questions / Comments

Minutes

Michael began the meeting by reviewing/approving last week's meeting minutes. This led into discussion about the project plan. We then reviewed the group's day plans at Fidelity. Professor Djamasbi then discussed the need to delegate tasks for what each team member will be responsible for. The group also discussed UPOD and only promising Fidelity delivery on items we are confident will be completed. We were encouraged to begin looking up Perl tutorials. Professor Djamasbi asked that the group take time to write down specific questions to ask while at Fidelity (to get the most of our time on Wednesdays) as well as create an outline of the group's understanding of the requirements. We discussed our plan of action in speaking with Barbara regarding the career fair time conflicts and possibly leaving early.

Attachments

I. Updated version of Project Plan
II. Revised Work Plan
III. Documentation Revisions
IV. Project Questions
V. Acronyms
VI. Feasibility Analysis

Fidelity MQP Meeting

Tuesday, September 23rd, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey (Moderator), Dan Dahlberg (Secretary), Michael Diamant, Augustina Mills

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review of Project Plan
4. Review Attachments
    a. Revisions to documentation
    b. Review project questions
    c. Acronyms
    d. Feasibility Analysis
    e. In Progress: Literature Review and Organizational Chart
5. Review last Wednesday at Fidelity
    a. Issue with payment
6. Plan for this week at Fidelity
7. Questions / Comments

Minutes

Augustina began the meeting by reviewing/approving last week's meeting minutes. This led into discussion about the project plan. After reviewing the project plan, we then proceeded to review the revisions to the documentation. Michael noted that we have started an acronyms list that will aid us in discussions at Fidelity. The feasibility analysis was then reviewed. Professor Djamasbi suggested that we cite our information in the analysis. It was also suggested that we share our agenda with Fidelity on our Wednesday visits. We spoke about our plans for the upcoming Wednesday. We also confirmed that we will keep our meeting next Tuesday at 3pm and have Professor Djamasbi talk with us by phone.

Attachments

I. Updated version of Project Plan
II. Revised Work Plan
III. Feasibility Analysis

Fidelity MQP Meeting

Tuesday, September 30th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Dan Dahlberg (Moderator), Michael Diamant (Secretary), Augustina Mills, Elizabeth Carey

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Presentation Date Change
4. Refined Feasibility Analysis
5. Modified MQP Documentation
6. Initial Deliverable Assessment
7. Review last Wednesday at Fidelity
8. Plan for this week at Fidelity
9. Questions / Comments

Minutes

Liz began the meeting by discussing the agenda and providing a brief overview of the minutes. Everyone subsequently accepted them and Liz indicated that the project plan remains intact and on schedule and that nothing had been modified. Mike explained the changes he had made to the feasibility analysis and demonstrated how he generated some of the numbers. Liz continued by discussing the upcoming Wednesday's plans at Fidelity, which included a 9AM meeting with Veritude and a meeting with Barbara to show what the team has come up with so far. Dan spoke about how has been learning Perl / CGI in order to properly assess what the team could provide Fidelity as deliverables, relative to Fidelity's requirements definition, and that an initial assessment will be made for the next time the team met.

Attachments

I. Refined Feasibility Analysis
II. Modified MQP Documentation
III. Initial Deliverable Assessment
IV. Economic Feasibility Questions
V. GPWS Process Questions

Fidelity MQP Meeting

Tuesday, October 7th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Dan Dahlberg (Moderator), Michael Diamant (Moderator, Augustina Mills (Secretary), Elizabeth Carey

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review last Wednesday at Fidelity
4. Project Update
5. Website interface design/Login POC
6. Plan for this week at Fidelity
7. Questions / Comments

Minutes

Professor Djamasbi clarified the Fidelity payment issue; the project team is cleared to accept payment from Fidelity. Next, Professor Djamasbi approved the previous week's minutes. Then, the project team overviewed the retooled feasibility analysis and risk management assessment. Dan informed the project team and Professor Djamasbi that programming will not be the main project concern. Instead, clarification of project requirements obfuscates project progress. The project team agreed to reduce project uncertainty by October 8th, to remove uncertainty by October 15th, and to submit the MQP proposal by October 21st.

Attachments

I. Website interface design (external link)
II. Login proof-of-concept (external link)

Fidelity MQP Meeting

Tuesday, October 14th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Daniel Dahlberg, Michael Diamant, Augustina Mills (Moderator), Elizabeth Carey (Secretary)

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review last Wednesday at Fidelity
4. Project Update
5. Review revised mock-ups
6. Presentation
7. Plan for this tomorrow at Fidelity
8. Questions / Comments

Minutes

We started the meeting by review the previous week's notes. We talked about working at Fidelity and the work environment. We students should be leveraging the real work experience we have at Fidelity. In a sense, Fidelity can be seen as a new world due to the vast difference in culture, acronyms, etc. It is our job to adapt to this environment. After our brief discussion about our Fidelity experience thus far, we proceeded to login on GTI. Questions/Suggestions for Fidelity: Ask if there is a Camtasia-like tutorial we could view to see the actual process of performing a test. Mention a possible video conference session with Doug. Regarding the writing portion of the project, the professor reminded us to be mindful of our transitions between sections in the paper. Before submitting a copy to the advisor we must proof read our sections and then turn in for content editing. This will save the professor less time correcting trivial errors and such.

Attachments

Fidelity MQP Meeting

Tuesday, October 28th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey (Moderator), Daniel Dahlberg (Secretary), Michael Diamant, Augustina Mills

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review last Wednesday at Fidelity
4. Project Update/ Progress of Documentation
5. Presentation at Fidelity Wednesday
6. Questions / Comments

Minutes

The meeting was opened with reviewing the agenda. The previous minutes were adjusted and approved as adjusted by Professor Djamasbi. Then, the project team overviewed their previous week at Fidelity, which included a meeting with Dan at Fidelity. The economic feasibility analysis was discussed and decisions were made to have a feasibility analysis concerning what Fidelity needs and an analysis of what the group needs for WPI. The proposal presentation was then reviewed and adjusted with input from Professor Djamasbi and the project team. The documentation was to be turned in by Friday October 17th.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, November 4th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Daniel Dahlberg (Moderator), Michael Diamant (Secretary), Augustina Mills, Elizabeth Carey

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Discussion and Feedback on Proposal Presentation
4. Review last Wednesday at Fidelity
5. Progress of Documentation
6. Questions / Comments

Minutes

The meeting began with a review of the agenda and approval of the minutes by all attendees. Liz detailed the group's interaction with Fidelity last Wednesday, including steps the group has done to setup wireless and VPN access on the laptops. The group also did a 'dry-run' of the presentation with Washesh and Barbara, who generally approved of the presentation. Professor Djamasbi then discussed her method of grading the MQP proposal and common mistakes to avoid. She also emphasized that we are providing a service based on the project scope, not based on our own limitations. Finally, Dan discussed the lack of information the group currently has with the back-end Fidelity scripts and how the group plans on improving that situation.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, November 11th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Daniel Dahlberg, Michael Diamant (Moderator), Augustina Mills (Secretary), Elizabeth Carey

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review last Wednesday at Fidelity
4. Progress of Documentation
5. Update on Programming State
6. Questions / Comments

Minutes

The meeting opened with an agenda review and minutes approval. Professor Djamasbi provided positive remarks regarding last week's MQP proposal presentation at Fidelity. Professor Djamasbi emphasized the importance of a strong presentation in the context of Fidelity's interests and the MQP group's interests. Additionally, it was emphasized to the group to follow up with Fidelity sponsors for a letter of reference at the project conclusion. The group then discussed the possibility of meeting Fidelity's CIO and its implications. Next, the group provided an overview of documentation updates. Professor Djamasbi requested completed documentation by the end of the term. The group is to arrange an appointment prior to Thanksgiving to review the documentation with Professor Djamasbi. Additionally, the group is to provide code samples for review. The meeting concluded with long-term planning for C-term. C-term activities will include training manual creation and presentation creation and review.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, November18th, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Augustina Mills (Moderator), Elizabeth Carey (Secretary)

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review last Wednesday at Fidelity
4. Progress of Documentation
5. Update on Programming State
6. Questions / Comments

Minutes

The meeting opened with an agenda review and minutes approval. Last meeting a few opportunities arose in creating a fifth dashboard along with a lack of access to the VPN and testing environment. However, Professor Djamasbi reminded us that we needed to get this project done. Any issue that arises should be tended to, but ultimately the success of this project lies within our hands, the champions of this project. The Professor suggested we create a set agenda when going to Fidelity on Wednesday's. The agenda would map out all the tasks that our group must get done throughout the day in Boston. This would help in managing our time and being most productive. We then talked about the documentation plan, which is to be submitted before Thanksgiving break. Mike and Dan presented the programming status afterwards. Meeting adjourned at 3:50pm.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, December 2nd, 2008

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey (Moderator), Michael Diamant (Secretary), Daniel Dahlberg, Augustina Mills

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review previous Wednesday at Fidelity
4. Progress of Documentation
5. Update on Programming State
6. Questions / Comments

Minutes

The meeting opened with an agenda review and minutes approval. Michael and Daniel were not in attendance for the meeting due to other commitments. The group reviewed the previous Wednesday at Fidelity and spoke about how to deal with the current situation. We were instructed to send Barbara an email from the group. Professor Djamasbi then spoke at length about the documentation and instructed the group to hand in a draft on Friday. A meeting was set for before Thanksgiving to review the documentation. The state of programming was then discussed and will be further discussed at the next meeting when Michael and Daniel are present. Professor Djamasbi informed the group of the timeline for the rest of the term and requested to have material turned into her at the beginning of the month. The meeting on December 9th may need to be a phone conference with Professor Djamasbi.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, January 20th, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Daniel Dahlberg (Moderator), Elizabeth Carey (Secretary), Augustina Mills, Michael Diamant

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Review of upcoming visit to Fidelity
4. Discussion of Professor Djamasbi's Feedback on Documentation
5. Update on Code Changes over Break and Current Progress
6. Questions / Comments

Minutes

- Last Wednesday the group did not go into Boston because neither Rosetin nor Washesh would be in the office.
- Mike and Dan have been working hard on completing the scripts
- Augustina and Liz completed the changes in the documentation (attached the this email)
- We will be doing a demonstration tomorrow to walk Rosetin and Washesh through the interface and discuss how it functions

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, January 27th, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey (Moderator), Augustina Mills (Secretary), Michael Diamant, Daniel Dahlberg

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Last week at Fidelity
4. No Fidelity visit this week- plans for on campus
5. Update on Code
6. Documentation Timeline
7. Questions / Comments

Minutes

Meeting opened with reviewing the minutes from the last meeting and minutes were approved. The group then discussed the plans for Wednesday at Fidelity. We will be speaking with Rosetin to address questions that have come up over the break. In addition, we will be meeting with the CIO of FPCMS. Michael and Dan reviewed the progress of the code and informed Professor Djamasbi that they have created a task list that in continuously updated. Augustina will be checking on use cases with Washesh and will get feedback. The group then spoke about the documentation progress and will be restructuring the paper. We need to set a date for the final presentation that works for the WPI team and for Fidelity. In addition, we need to address the letter of acknowledgement with Washesh on Wednesday. For next meeting we are to have a timeline for the documentation and the user manual as well as a date for the project presentation.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, February 3rd, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey, Augustina Mills (Moderator), Michael Diamant (Secretary), Daniel Dahlberg

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Fidelity this coming week
4. Update on Code
5. Documentation Timeline
6. Questions / Comments

Minutes

Meeting opened with reviewing the minutes from the last meeting and minutes were approved. The group then discussed the plans for Wednesday at Fidelity. Last meeting, Michael and Dan reviewed the progress of the code. Liz discussed the documentation timeline and the group decided on an official presentation date. The date and time is: March 5th, 2009, at 10am at the WTC in Boston. Last week, a phone call to Fidelity was supposed to be held on Wednesday, but that never occurred. We will be using this week's time to its fullest, since we missed last week. Augustina will be checking on use cases with Washesh and will get feedback. Liz and Augustina will be discussing training material options with Washesh.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, February 10th, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Elizabeth Carey, Augustina Mills, Michael Diamant (Moderator), Daniel Dahlberg (Secretary)

Agenda

1. Overview of Agenda
2. Approval of Previous Weeks Minutes
3. Fidelity this coming week
4. Update on Code
5. Questions / Comments

Minutes

Meeting opened with reviewing the minutes from the last meeting and minutes were approved. The group then discussed the plans for Wednesday at Fidelity. We discussed potential dates for the MQP, including the final week of the term and the first week of D-term. Michael and Dan are focused on reviewing testing requirements for GPWS and they intend to speak with Rosetin about this issue on Wednesday. Augustina and Liz are planning to meet with Michael and Dan to better understand the user interface for the purpose of documentation.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, February 17^{h}, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Augustina Mills (Secretary), Michael Diamant, Dan Dahlberg (Moderator)

Agenda

1. Overview of agenda
2. Approval of previous weeks minutes
3. Fidelity visit the previous week
4. Update on programming status
    a. Programming mistake
    b. Final coding action items
5. Documentation update
6. Questions / comments

Minutes

Meeting started with Mike detailing the (missing) agenda and detailing a summation of the previous weeks minutes. Mike then moved on to explain what the group did the last week at Fidelity, which included a meeting with Rosetin about major task items and discussions with Washesh. Dan gave an update on the code indicating that we have a list of task items that we have to address with Rosetin the next visit. Augustina and Liz explained the changes and outline they have for the documentation in the coming weeks. The meeting concluded with a discussion regarding preparing for the MQP award.

Attachments

No attachments available.

Fidelity MQP Meeting

Tuesday, February 24th, 2009

Attendees: Soussan Djamasbi (Faculty Advisor), Augustina Mills (Moderator), Michael Diamant, Dan Dahlberg, Liz Carey (Secretary)

Agenda

1. Overview of agenda
2. Approval of previous weeks minutes
3. Fidelity visit the previous week
4. Update on programming status
    a. Programming mistake (update)
    b. Final coding action items
5. Documentation update
    a. Training manual screenshots (done this morning)
    b. Find a meeting time to review Documentation for final touch-ups
6. Questions / comments

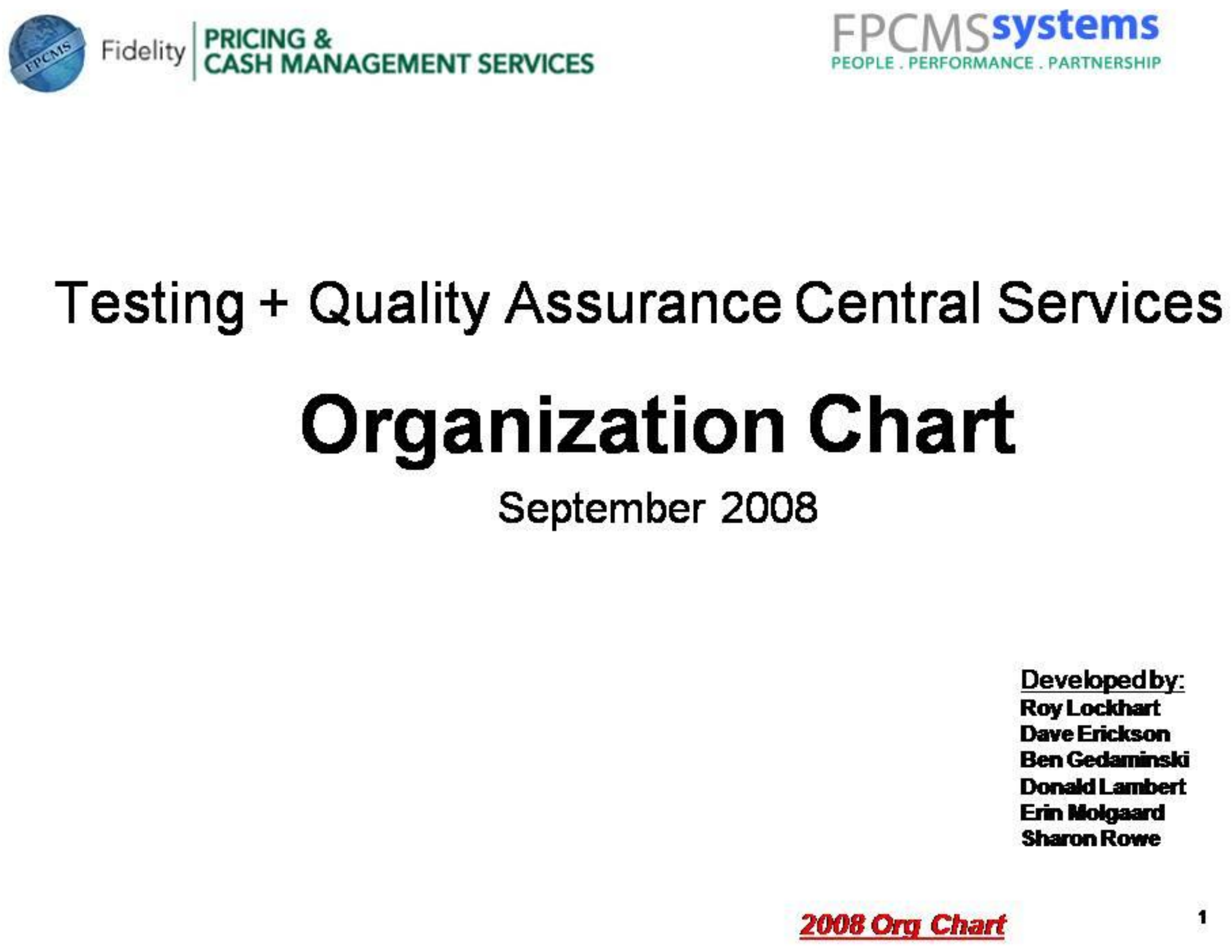
Minutes

Meeting started with Mike discussing the deletion error that occurred last week. Dan then discussed the agenda and detailing a summation of the previous weeks minutes. Mike then moved on to explain what the group did the last week at Fidelity, which included a meeting with Rosetin about major task items and discussions with Washesh. Dan gave an update on the code. Augustina explained the issue with the training manual and how screenshots will be done once Fidelity can restore the files that were deleted. The meeting concluded with a discussion regarding preparing for the MQP award.

Attachments

No attachments available.

B. Fidelity Organizational Chart



Fidelity | PRICING & CASH MANAGEMENT SERVICES

FPCMSsystems
PEOPLE . PERFORMANCE . PARTNERSHIP

Testing + Quality Assurance Central Services

Organization Chart

September 2008

Developed by:
Roy Lockhart
Dave Erickson
Ben Gedaminski
Donald Lambert
Erin Molgaard
Sharon Rowe

2008 Org Chart 1

FPCMS Systems - C.O.E. - Testing + Quality Assurance Central Services

September 2008

Roy Lockhart
Vice President

Ben Gedaminski
Director
Test Delivery Services

- Katie Brunquell
- Doug Carriger
- Meredith Frost
- Glenn Janssen
- Brian Schultz

Offshore:

- TBH-Replace-Reetika Ghai
- TBH-Replace-Gaurav Chandna
- TBH-Replace-Ayush Gupta
- TBH-Replace-Piyush Mehta
- TBH-Replace-Lakshmi Narain
- TBH-Replace-Geeta Sahi

Donald Lambert
Principal Quality Assurance Engineer

Sharon Rowe
Director
Test Management Office /Quality Assurance Management Office

- Irina Doudova
- Ray Fournier
- Evelyn Fernandes (09-22)
- Barbara Mackey
- Marie Mikalauskis

Offshore:

- B.S Kempe Gowda
- Avinash Mehrotra
- Venugopal Nallani

Erin Molgaard
Senior Manager
Tools, Automation, + Performance Test Services

- Roselin Anwar
- Washesh Mehra
- Kevin Ward

Offshore:

- Naveen Kumar
- Latha Guthi
- Mallika Karkada
- Sivakumar Mungamda

Professional Services:

- Surendra Bysani (Infosys)
- Merin Carnal (Infosys)
- Ramya Gogu (Infosys)
- Umesha Kannakatti (Infosys)
- Vaibhav Khandelwal (Infosys)
- Junaid Kochak (Infosys)
- Rekha Manoharan (Infosys)
- Kim Helbick (FTG)
- Elizabeth Carey (WPI Intern)
- Daniel Dahlberg (WPI Intern)
- Michael Diamant (WPI Intern)
- Augustina Mills (WPI Intern)

Dave Erickson
Director
Release Engineering/ Environment Management

- Bobby Jones
- Shane Grady
- Diane Laverdu
- Pedro Morales
- Walter Moreira
- Chris Rossino
- Jim Stewart

Offshore:

- TBH-Replace-Alok Kumar
- Chandan Menghani
- Kelki Anand

As of: September, 2008
FPCMS

2

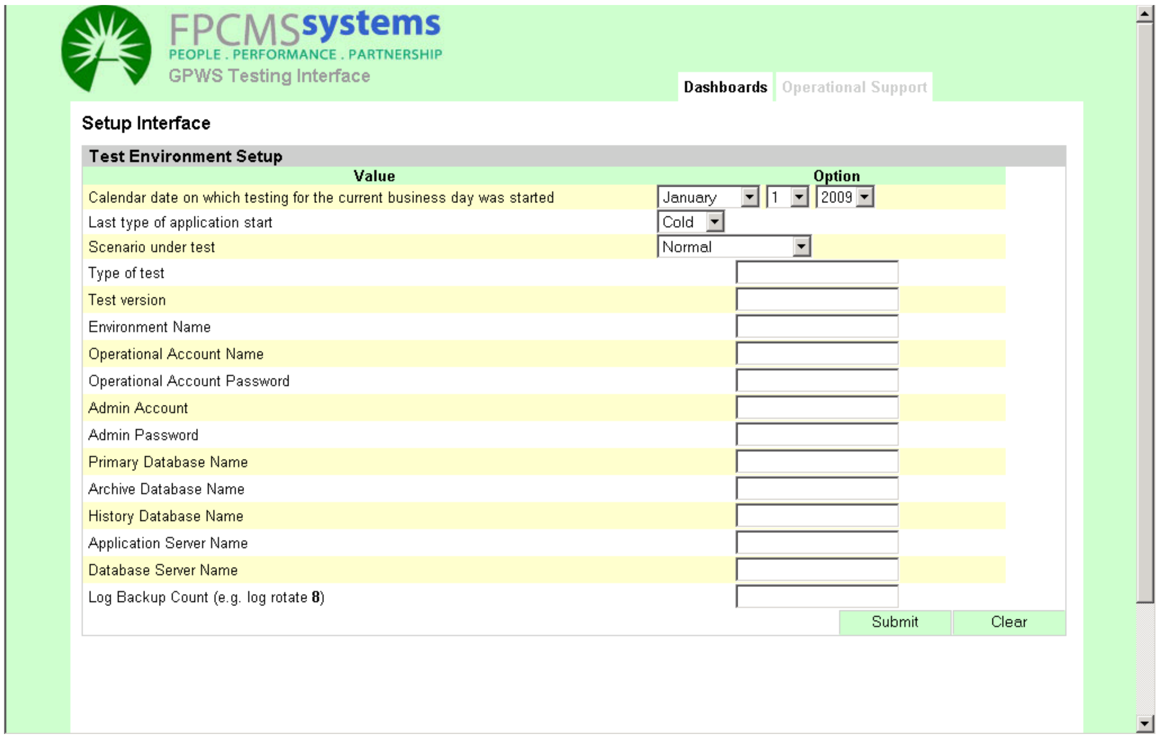
2008 Org Chart

C. Training Manual

GPWS Testing Interface

Training Manual

Setup Interface



Setup Interface

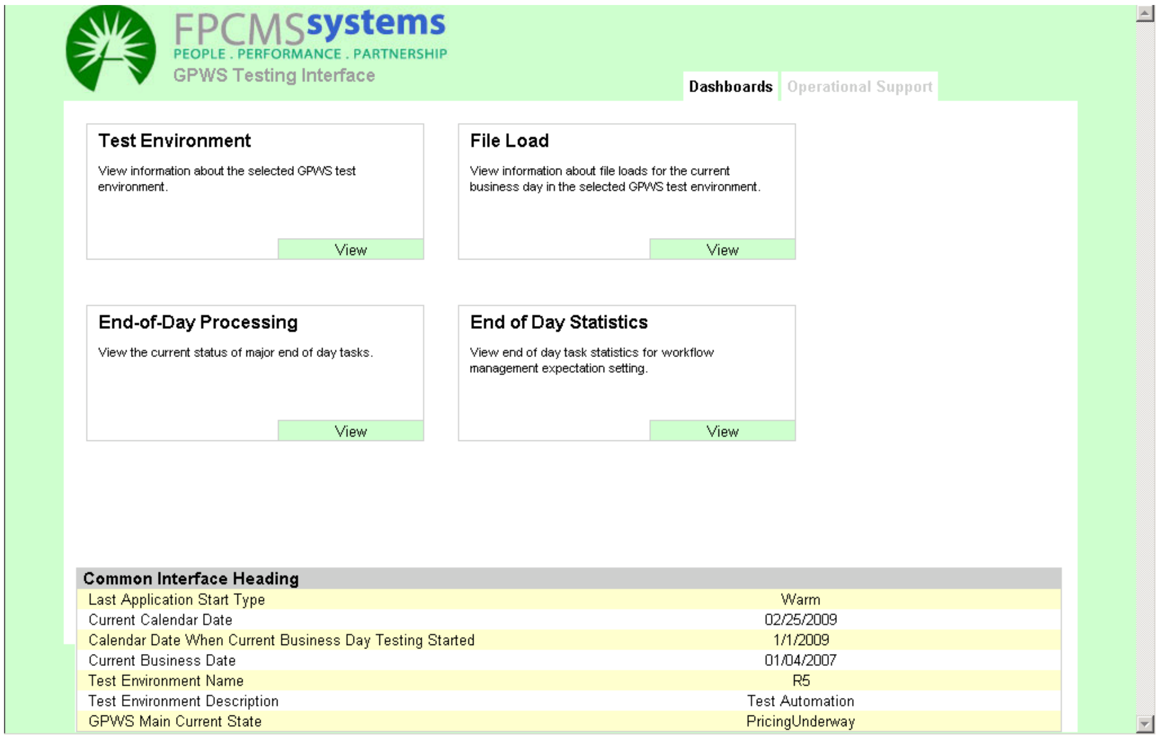
Test Environment Setup

Value	Option
Calendar date on which testing for the current business day was started	January 1 2009
Last type of application start	Cold
Scenario under test	Normal
Type of test	
Test version	
Environment Name	
Operational Account Name	
Operational Account Password	
Admin Account	
Admin Password	
Primary Database Name	
Archive Database Name	
History Database Name	
Application Server Name	
Database Server Name	
Log Backup Count (e.g. log rotate **8**)	

The setup interface will run only once for each individual testing date. Fill in all the appropriate information and then click "submit" when done.

Index –

Dashboards



Test Environment

View information about the selected GPWS test environment.

View

File Load

View information about file loads for the current business day in the selected GPWS test environment.

View

End-of-Day Processing

View the current status of major end of day tasks.

View

End of Day Statistics

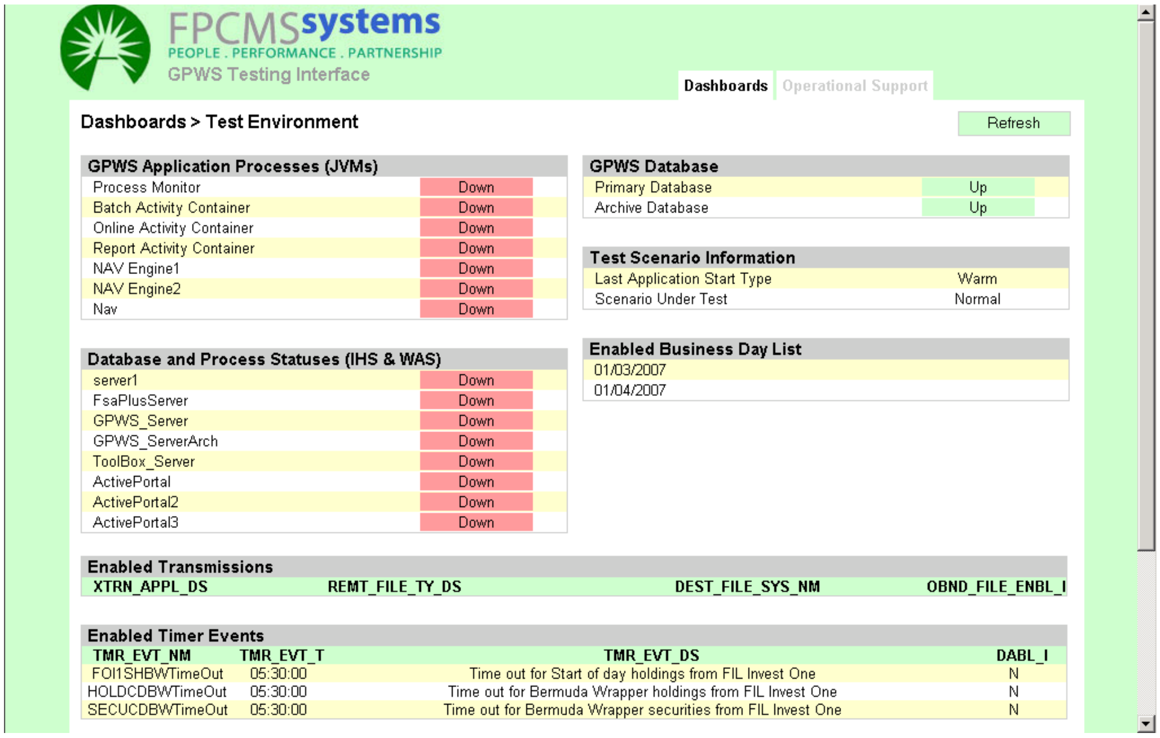
View end of day task statistics for workflow management expectation setting.

View

Common Interface Heading	
Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

This is next screen that will appear after you hit “submit”. This is the Dashboards tab, which allows you to view the Test Environment and End-of-Day Processing. Click “View” for either option to view the status.

Test Environment



Dashboards > Test Environment

GPWS Application Processes (JVMs)	
Process Monitor	Down
Batch Activity Container	Down
Online Activity Container	Down
Report Activity Container	Down
NAV Engine1	Down
NAV Engine2	Down
Nav	Down

GPWS Database	
Primary Database	Up
Archive Database	Up

Test Scenario Information	
Last Application Start Type	Warm
Scenario Under Test	Normal

Database and Process Statuses (IHS & WAS)	
server1	Down
FsaPlusServer	Down
GPWS_Server	Down
GPWS_ServerArch	Down
ToolBox_Server	Down
ActivePortal	Down
ActivePortal2	Down
ActivePortal3	Down

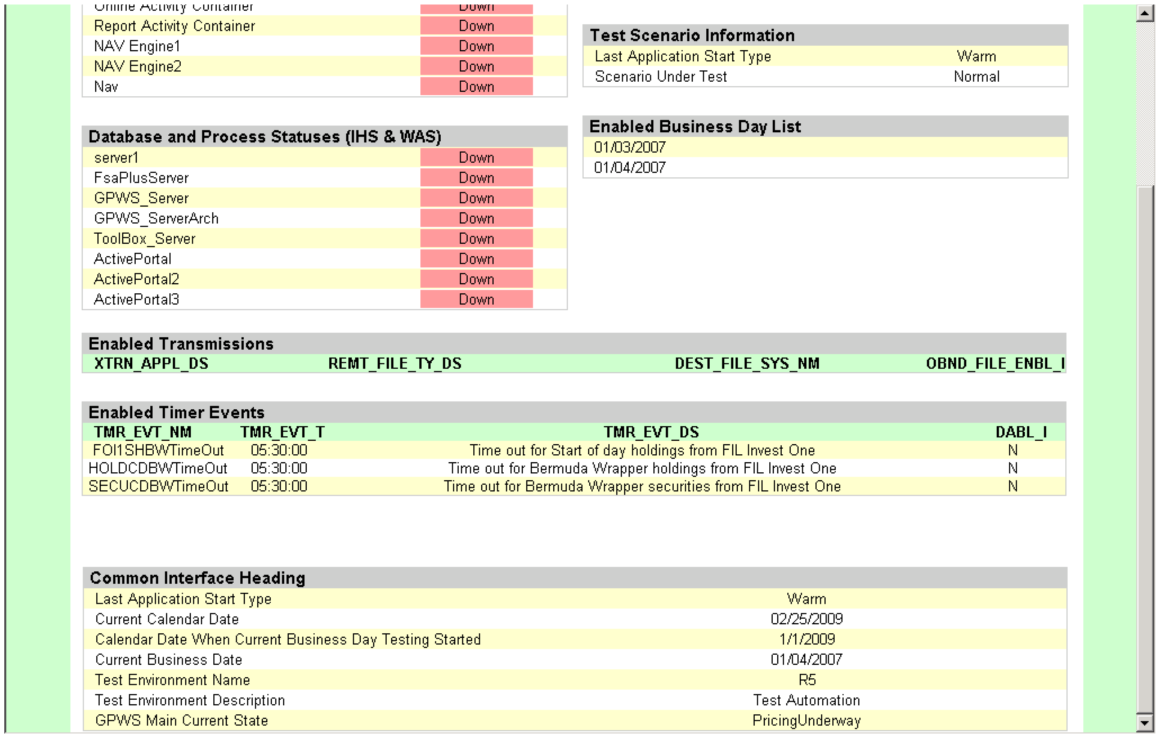
Enabled Business Day List
01/03/2007
01/04/2007

Enabled Transmissions

XTRN_APPL_DS	REMT_FILE_TY_DS	DEST_FILE_SYS_NM	OBND_FILE_ENBL_I

Enabled Timer Events

TMR_EVT_NM	TMR_EVT_T	TMR_EVT_DS	DABL_I
FOI1SHBWTimeOut	05:30:00	Time out for Start of day holdings from FIL Invest One	N
HOLDCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper holdings from FIL Invest One	N
SECUCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper securities from FIL Invest One	N



Online Activity Container	Down
Report Activity Container	Down
NAV Engine1	Down
NAV Engine2	Down
Nav	Down

Test Scenario Information	
Last Application Start Type	Warm
Scenario Under Test	Normal

Database and Process Statuses (IHS & WAS)	
server1	Down
FsaPlusServer	Down
GPWS_Server	Down
GPWS_ServerArch	Down
ToolBox_Server	Down
ActivePortal	Down
ActivePortal2	Down
ActivePortal3	Down

Enabled Business Day List
01/03/2007
01/04/2007

Enabled Transmissions

XTRN_APPL_DS	REMT_FILE_TY_DS	DEST_FILE_SYS_NM	OBND_FILE_ENBL_I

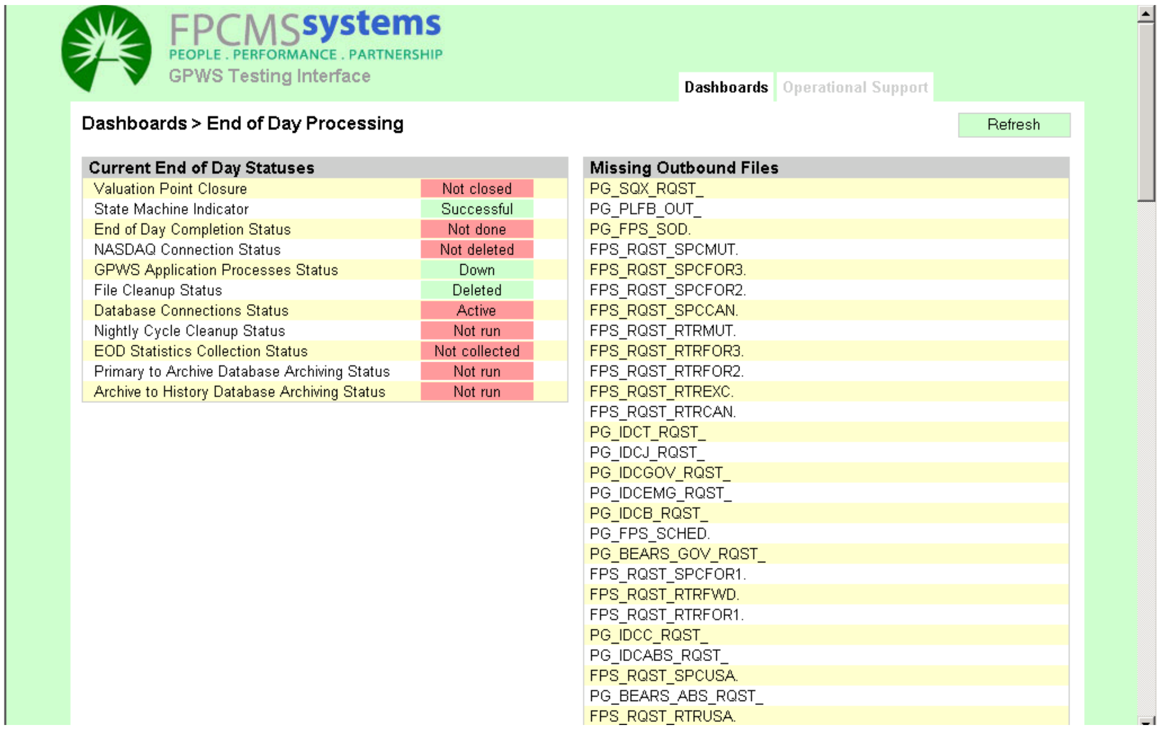
Enabled Timer Events

TMR_EVT_NM	TMR_EVT_T	TMR_EVT_DS	DABL_I
FOI1SHBWTimeOut	05:30:00	Time out for Start of day holdings from FIL Invest One	N
HOLDCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper holdings from FIL Invest One	N
SECUCDBWTimeOut	05:30:00	Time out for Bermuda Wrapper securities from FIL Invest One	N

Common Interface Heading	
Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

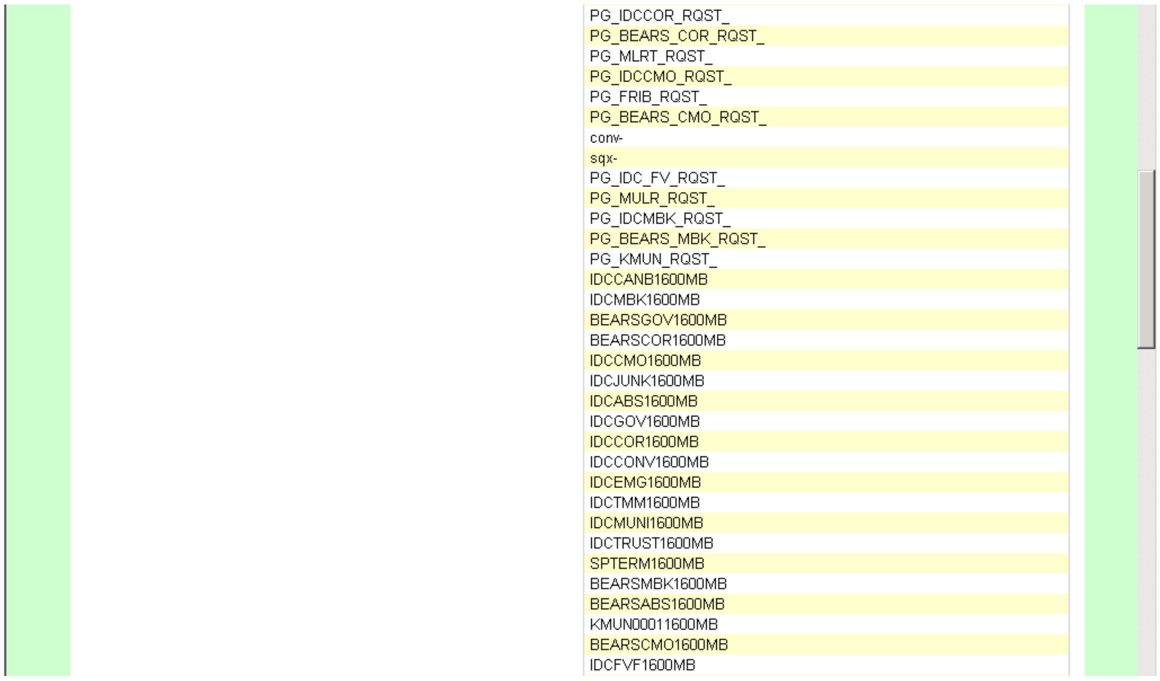
When you click "View" on test environment, this screen will appear and show the several statuses listed above.

End-of-Day (EOD) Processing

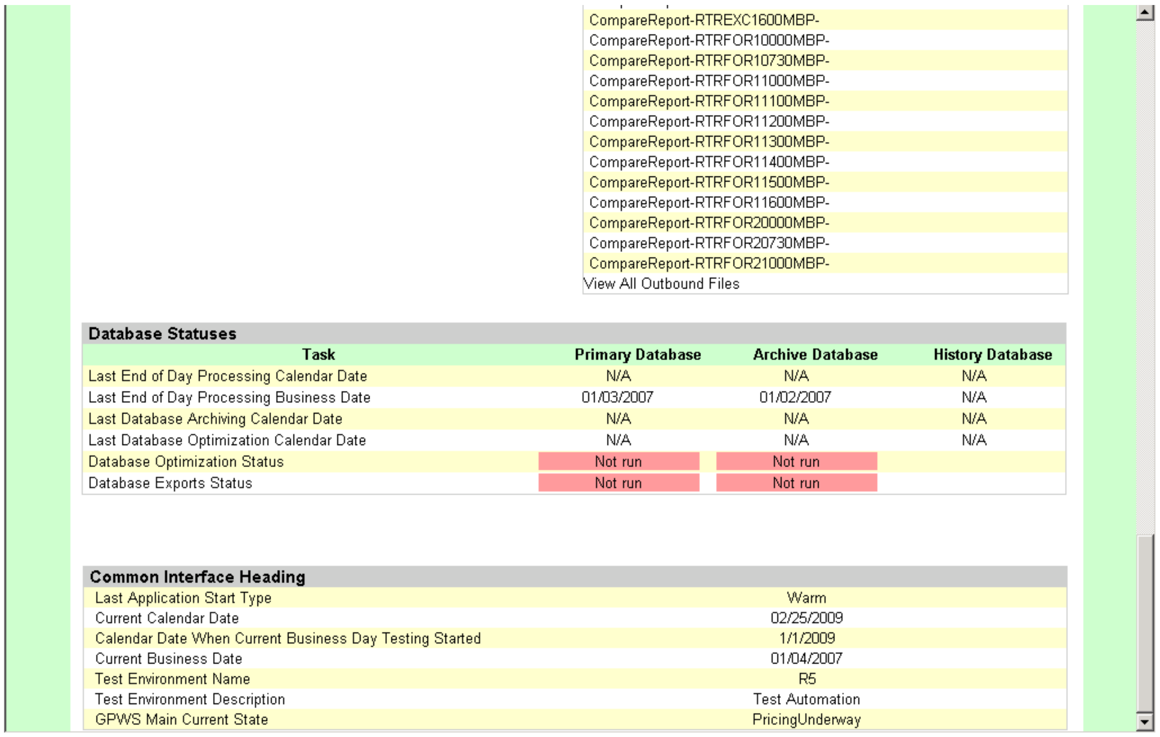


Dashboards > End of Day Processing

Current End of Day Statuses	
Valuation Point Closure	Not closed
State Machine Indicator	Successful
End of Day Completion Status	Not done
NASDAQ Connection Status	Not deleted
GPWS Application Processes Status	Down
File Cleanup Status	Deleted
Database Connections Status	Active
Nightly Cycle Cleanup Status	Not run
EOD Statistics Collection Status	Not collected
Primary to Archive Database Archiving Status	Not run
Archive to History Database Archiving Status	Not run



Missing Outbound Files
PG_SQX_RQST_
PG_PLFB_OUT_
PG_FPS_SOD.
FPS_RQST_SPCMUT.
FPS_RQST_SPCFOR3.
FPS_RQST_SPCFOR2.
FPS_RQST_SPCCAN.
FPS_RQST_RTRMUT.
FPS_RQST_RTRFOR3.
FPS_RQST_RTRFOR2.
FPS_RQST_RTREXC.
FPS_RQST_RTRCAN.
PG_IDCT_RQST_
PG_IDCJ_RQST_
PG_IDCGOV_RQST_
PG_IDCEMG_RQST_
PG_IDCB_RQST_
PG_FPS_SCHED.
PG_BEARS_GOV_RQST_
FPS_RQST_SPCFOR1.
FPS_RQST_RTRFWD.
FPS_RQST_RTRFOR1.
PG_IDCC_RQST_
PG_IDCABS_RQST_
FPS_RQST_SPCUSA.
PG_BEARS_ABS_RQST_
FPS_RQST_RTRUSA.
PG_IDCCOR_RQST_
PG_BEARS_COR_RQST_
PG_MLRT_RQST_
PG_IDCCMO_RQST_
PG_FRIB_RQST_
PG_BEARS_CMO_RQST_
conv-
sqx-
PG_IDC_FV_RQST_
PG_MULR_RQST_
PG_IDCMBK_RQST_
PG_BEARS_MBK_RQST_
PG_KMUN_RQST_
IDCCANB1600MB
IDCMBK1600MB
BEARSGOV1600MB
BEARSCOR1600MB
IDCCMO1600MB
IDCJUNK1600MB
IDCABS1600MB
IDCGOV1600MB
IDCCOR1600MB
IDCCONV1600MB
IDCEMG1600MB
IDCTMM1600MB
IDCMUNI1600MB
IDCTRUST1600MB
SPTERM1600MB
BEARSMBK1600MB
BEARSABS1600MB
KMUN00011600MB
BEARSCMO1600MB
IDCFVF1600MB



CompareReport-RTREXC1600MBP-
CompareReport-RTRFOR10000MBP-
CompareReport-RTRFOR10730MBP-
CompareReport-RTRFOR11000MBP-
CompareReport-RTRFOR11100MBP-
CompareReport-RTRFOR11200MBP-
CompareReport-RTRFOR11300MBP-
CompareReport-RTRFOR11400MBP-
CompareReport-RTRFOR11500MBP-
CompareReport-RTRFOR11600MBP-
CompareReport-RTRFOR20000MBP-
CompareReport-RTRFOR20730MBP-
CompareReport-RTRFOR21000MBP-
View All Outbound Files

Database Statuses

Task	Primary Database	Archive Database	History Database
Last End of Day Processing Calendar Date	N/A	N/A	N/A
Last End of Day Processing Business Date	01/03/2007	01/02/2007	N/A
Last Database Archiving Calendar Date	N/A	N/A	N/A
Last Database Optimization Calendar Date	N/A	N/A	N/A
Database Optimization Status	Not run	Not run	
Database Exports Status	Not run	Not run	

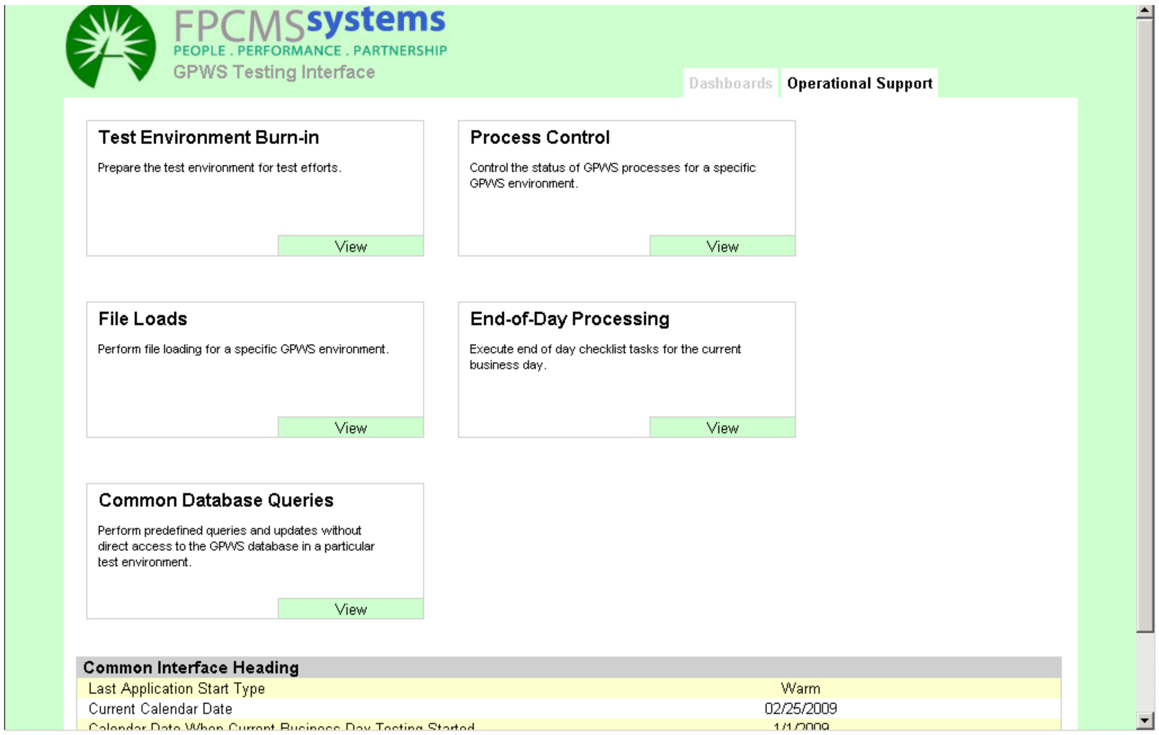
Common Interface Heading

Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

When you click “View” on End-of-Day processing, this screen will appear and will show the EOD statuses.

Index –

Operational Support



Test Environment Burn-in

Prepare the test environment for test efforts.

View

Process Control

Control the status of GPWS processes for a specific GPWS environment.

View

File Loads

Perform file loading for a specific GPWS environment.

View

End-of-Day Processing

Execute end of day checklist tasks for the current business day.

View

Common Database Queries

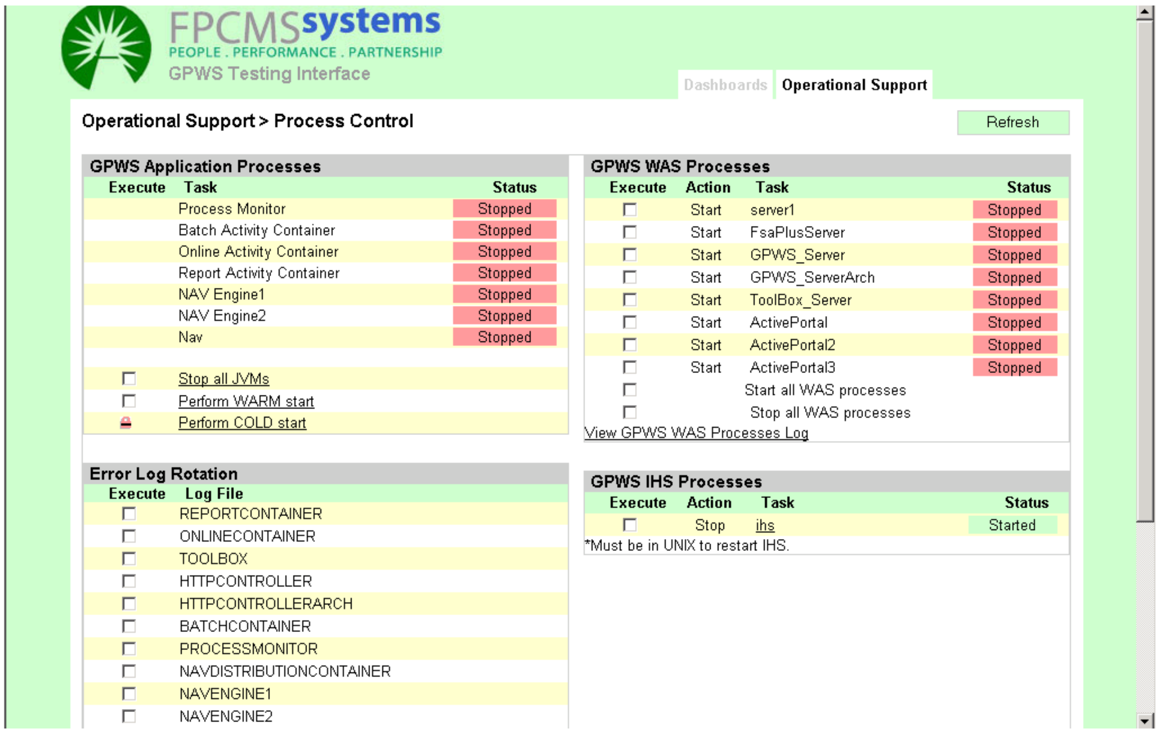
Perform predefined queries and updates without direct access to the GPWS database in a particular test environment.

View

Common Interface Heading	
Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009

This is the other tab, "Operational Support". You can see the status of both Process Control and End-of-Day Processing by clicking "View".

Process Control



FPCMSsystems
PEOPLE . PERFORMANCE . PARTNERSHIP
GPWS Testing Interface

Dashboards | Operational Support

Operational Support > Process Control

Refresh

GPWS Application Processes

Execute	Task	Status
	Process Monitor	Stopped
	Batch Activity Container	Stopped
	Online Activity Container	Stopped
	Report Activity Container	Stopped
	NAV Engine1	Stopped
	NAV Engine2	Stopped
	Nav	Stopped
☐	Stop all JVMs	
☐	Perform WARM start	
	Perform COLD start	

GPWS WAS Processes

Execute	Action	Task	Status
☐	Start	server1	Stopped
☐	Start	FsaPlusServer	Stopped
☐	Start	GPWS_Server	Stopped
☐	Start	GPWS_ServerArch	Stopped
☐	Start	ToolBox_Server	Stopped
☐	Start	ActivePortal	Stopped
☐	Start	ActivePortal2	Stopped
☐	Start	ActivePortal3	Stopped
☐		Start all WAS processes	
☐		Stop all WAS processes	

View GPWS WAS Processes Log

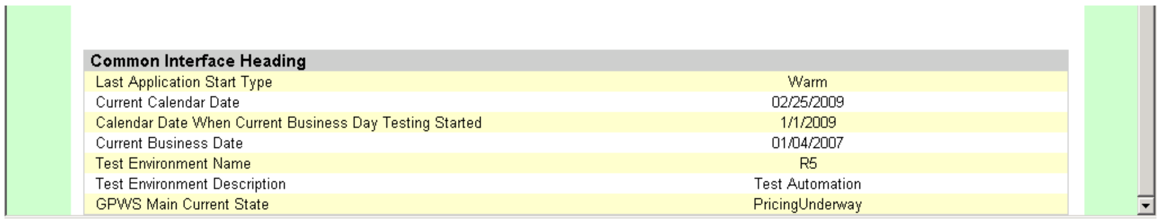
Error Log Rotation

Execute	Log File
☐	REPORTCONTAINER
☐	ONLINECONTAINER
☐	TOOLBOX
☐	HTTPCONTROLLER
☐	HTTPCONTROLLERARCH
☐	BATCHCONTAINER
☐	PROCESSMONITOR
☐	NAVDISTRIBUTIONCONTAINER
☐	NAVENGINE1
☐	NAVENGINE2

GPWS IHS Processes

Execute	Action	Task	Status
☐	Stop	ihs	Started

*Must be in UNIX to restart IHS.

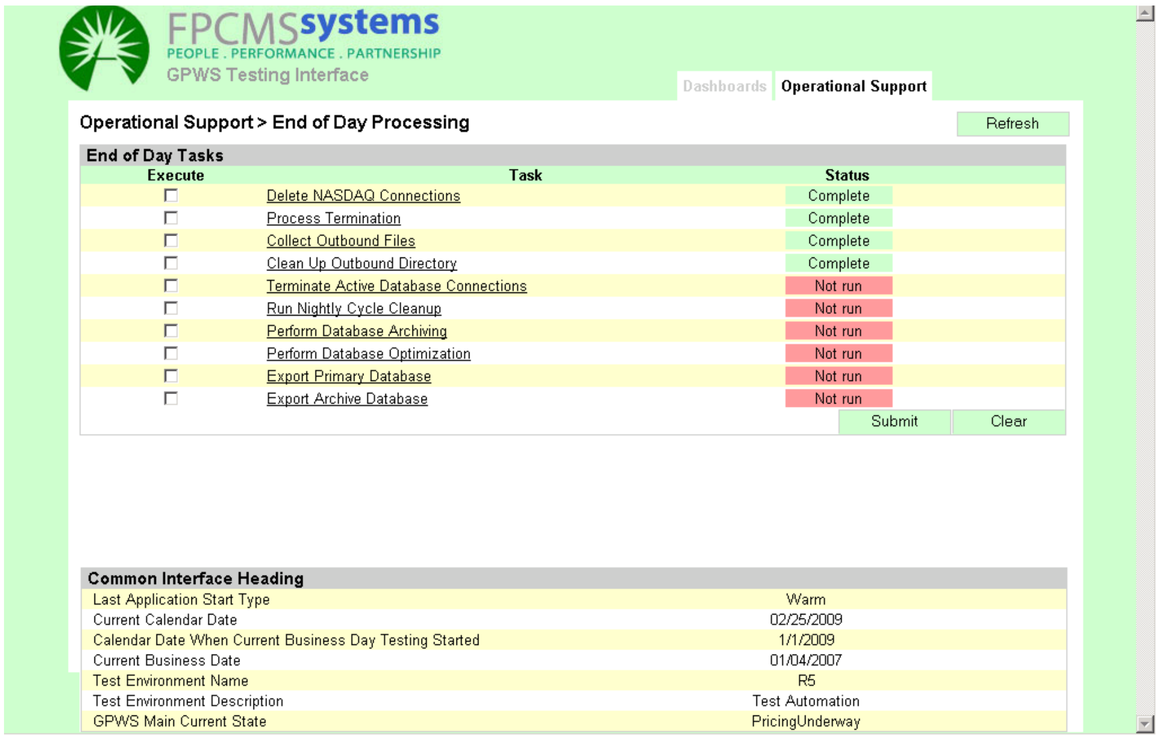


Common Interface Heading

Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

When you click "View" on Process Control, this page will appear. This shows the statuses on the page.

End-of-Day (EOD) Processing



Operational Support > End of Day Processing

End of Day Tasks

Execute	Task	Status
☐	Delete NASDAQ Connections	Complete
☐	Process Termination	Complete
☐	Collect Outbound Files	Complete
☐	Clean Up Outbound Directory	Complete
☐	Terminate Active Database Connections	Not run
☐	Run Nightly Cycle Cleanup	Not run
☐	Perform Database Archiving	Not run
☐	Perform Database Optimization	Not run
☐	Export Primary Database	Not run
☐	Export Archive Database	Not run

Common Interface Heading

Last Application Start Type	Warm
Current Calendar Date	02/25/2009
Calendar Date When Current Business Day Testing Started	1/1/2009
Current Business Date	01/04/2007
Test Environment Name	R5
Test Environment Description	Test Automation
GPWS Main Current State	PricingUnderway

When you click "View" on End-of-Day Processing, this page will appear. This shows the EOD statuses on the page.

D. Financial Analysis

Benefits

Benefits per Test Cycle	**SIT**	**UAT**	**Monthly Prod Patches**	**All Tests**
Tangible Benefits				
Reduced Staff Size Required to Perform Testing	$ 600	$ 600	$ 600	$ 1,800
Reduced Testing Time	$ 10,701	$ 10,701	$ 10,701	$ 32,102
Total Tangible Benefits	**$ 11,301**	**$ 11,301**	**$ 11,301**	**$ 33,902**
Intangible Benefits				
Improved Testing Process Consistency	$ 296	$ 295.88	$ 295.88	$ 888
Improved Ability to Research Test Exceptions	$ 99	$ 98.63	$ 98.63	$ 296
Reduced Training Time for New Resources	$ 16,800	$ 5,600	$ 2,100	$ 1,400
Total Intangible Benefits	**$ 17,195**	**$ 5,995**	**$ 2,495**	**$ 2,584**
Total Benefits	**$ 28,495**	**$ 17,295**	**$ 13,795**	**$ 36,486**

Benefits per Anum	**SIT**	**UAT**	**Monthly Prod Patches**	**All Tests**
Tangible Benefits				
Reduced Staff Size Required to Perform Testing	$ 600	$ 1,800	$ 4,800	$ 7,200
Reduced Testing Time	$ 10,701	$ 32,102	$ 85,607	$ 128,410
Total Tangible Benefits	**$ 11,301**	**$ 33,902**	**$ 90,407**	**$ 135,610**
Intangible Benefits				
Improved Testing Process Consistency	$ 296	$ 888	$ 2,367	$ 3,551
Improved Ability to Research Test Exceptions	$ 99	$ 296	$ 789	$ 1,184
Reduced Training Time for New Resources on Operational Support Tasks	$ 16,800	$ 16,800	$ 16,800	$ 16,800
Total Intangible Benefits	**$ 17,195**	**$ 17,984**	**$ 19,956**	**$ 21,534**
Total Benefits	**$ 28,495**	**$ 51,886**	**$ 110,363**	**$ 157,144**

Costs

Costs per Anum	**Worst**	**Average**	**Best**	**Expected**
Development Costs				
Consultant Fees (WPI)	$ 36,000	$ 30,000	$ 27,000	$ 31,200
Equipment (Laptops)	$ 1,200	$ 1,000	$ 900	$ 1,040
Development Training	$ 60,480	$ 50,400	$ 45,360	$ 52,416
Initial User Training	$ 1,440	$ 1,200	$ 1,080	$ 1,248
Total Development Costs	**$ 99,120**	**$ 82,600**	**$ 74,340**	**$ 85,904**
Operational Costs				
Annual User Training	$ 960	$ 1,200	$ 1,320	$ 1,152
Annual Code Maintenance	$ 9,600	$ 12,000	$ 13,200	$ 11,520
Total Operational Costs	**$ 10,560**	**$ 13,200**	**$ 14,520**	**$ 12,672**
Total Costs	**$ 109,680**	**$ 95,800**	**$ 88,860**	**$ 98,576**

Consultant Cost	
Count	1
Hourly Rate	$ 65
Annual Cost	$ 130,000
Cost Delta ($)	$ 99,840
Cost Delta (%)	317%
NPV Delta (%)	32%

	2008	2009	2010	2011	2012	Total
Benefits per Anum						
Tangible Benefits						
Reduced Staff Size Required to Perform Testing	$ -	$ 7,200	$ 7,632	$ 8,090	$ 8,575	$ 31,497
Reduced Testing Time	$ -	$ 128,410	$ 136,114	$ 144,281	$ 152,938	$ 561,743
Total Tangible Benefits	**$ -**	**$ 135,610**	**$ 143,746**	**$ 152,371**	**$ 161,513**	**$ 593,241**
Intangible Benefits						
Improved Testing Process Consistency	$ -	$ 3,551	$ 3,764	$ 3,989	$ 4,229	$ 15,532
Improved Ability to Research Test Exceptions	$ -	$ 1,184	$ 1,255	$ 1,330	$ 1,410	$ 5,177
Reduced Training Time for New Resources on Operational Support Tasks	$ -	$ 16,800	$ 17,808	$ 18,876	$ 20,009	$ 73,494
Total Intangible Benefits	**$ -**	**$ 21,534**	**$ 22,826**	**$ 24,196**	**$ 25,647**	**$ 94,203**
Total Benefits	**$ -**	**$ 157,144**	**$ 166,572**	**$ 176,567**	**$ 187,161**	**$ 687,444**
Costs per Anum						
Development Costs						
Consultant Fees (WPI)	$ 31,200	$ -	$ -	$ -	$ -	$ 31,200
Equipment (Laptops)	$ 1,040	$ -	$ -	$ -	$ -	$ 1,040
Development Training	$ 52,416	$ -	$ -	$ -	$ -	$ 52,416
Initial User Training	$ 1,248	$ -	$ -	$ -	$ -	$ 1,248
Total Development Costs	**$ 85,904**	**$ -**	**$ -**	**$ -**	**$ -**	**$ 85,904**
Operational Costs						
Annual User Training	$ 1,152	$ 1,221	$ 1,294	$ 1,372	$ 1,454	$ 6,494
Annual Code Maintenance	$ 11,520	$ 12,211	$ 12,944	$ 13,721	$ 14,544	$ 64,939
Total Operational Costs	**$ 12,672**	**$ 13,432**	**$ 14,238**	**$ 15,093**	**$ 15,998**	**$ 71,433**
Total Costs	**$ 98,576**	**$ 13,432**	**$ 14,238**	**$ 15,093**	**$ 15,998**	**$ 157,337**

Total Benefits - Total Costs	$ (98,576)	$ 143,711	$ 152,334	$ 161,474	$ 171,163
Cumulative Net Cash Flow	$ (98,576)	$ 45,135	$ 197,470	$ 358,944	$ 530,106
Return on Investment (ROI)	337%				
Break-Even Point (Years)	0.69				

E. Interview Reports

a. ITAC Interview

Interview Report

Interview Secretary: Liz Carey

Person Interviewed: Donald (Test Manager), Evelyn (Functional Test Engineer), Marie (Operational Testing Resourcing)

Interviewer: Liz Carey, Dan Dahlberg, Mike Diamant, Augustina Mills

Date: 10/1/08

Primary Purpose: To establish knowledge of other testing systems at Fidelity that may benefit from our work.

Summary of Interview:

Donald brought the project group through the basics of ITAC and how they relate to GPWS. He noted the similarities and the differences and stated what the process of ITAC testing is.

Open Items:

The group was told to contact the ITAC group if any questions arise in the upcoming weeks.

Detailed Notes:

- ITAC
  - Testing any particular day at one time
- Automation work
  - Offshore vendor advised by Washesh
  - Crude automation of the test steps
- Comparisons to GPWS
  - Similar things
    - Start of day
    - End of day
    - Stops along the way
    - Lad files
    - Database and GUI is pretty much the same
      - ITAC is web based GPWS is not
  - Difference

        - GPWS has more user intervention; ITAC is less user invasive
            - Application does most of the commands
        - No ctrl m jobs
        - Tells itself info, nothing external interfering with the program
- Operational checklist
    - 1st release checklist
    - Want to have operational automation done by February
- Concerned with the tools that get the back end running
- ITAC is in the UNIX environment (oracle, not DB2)
- Building a web tool for ITAC giving a UI for black screen
    - Make testing less dependent
    - Better using your resources and increase your testing capabilities
- OMC (operational management console)→ similar to GPWS toolbox
    - Less for maintenance, more for checking the state of the environment
    - Override timers
    - Quick view of where the application is
- At a minimum, whatever we develop can be given to ITAC for them to go off of; use to see how Perl and CGI interacts;
    - If they see an example they should be able to replicate with ITAC code pretty easy
- Everything is being done in the same language (Perl)
- ITAC
    - Transfer agents (record keeping of mutual funds)
        - All individual information
    - Outside/downstream that need aggregated information
    - ITAC is an aggregator
        - Takes yesterdays balance, adds in the new stuff, and takes the ending balance and compares it to the downstream
        - Manual adjustments
    - ITAC send feeds out
    - Some special applications for dividends, etc.
    - Takes the data in, manipulates it, sends it out
    - The reconciliation
        - 6-8 days settlement days
        - Not everyday things line up correctly, may take a couple of days
        - Record the settlement differences
    - Stuff coming in from TA is each day (couple times a day) but there are a lot of feeds going out…not just once a day
- Hard to play with code that's a moving target like ITAC

Changing so often that it would be hard to manipulate the code

b. Requirements Interview

Interview Report

Interview Secretary: Liz Carey

Person Interviewed: Rosetin Anwar

Interviewer: Dan Dahlberg

Date: 10/1/08

Primary Purpose: To gather information that is more specific on the high-level requirements.

Summary of Interview:

Dan conducted the interview via Phone and via email with Rosetin Anwar. She was able to provide the group with more detailed information about the testing processes that the group will be dealing with.

Open Items:

Numerous questions went unanswered in this interview including general questions as well as technical questions.

Detailed Notes:

<u>Common Interface Heading</u>

Current Calendar Date = <RA> calendar date = today's date / system date

- Is this date actively written to a database or file? <RA> No, it is not written to a db.
- Should we be aware of any concerns where GPWS thinks it is a different calendar date than the actual calendar date (e.g. EOD was somehow missed)? <RA> GPWS does not care about calendar date. We only have one calendar date.
- Would part of the testing mimic a different calendar date within a testing cycle? <RA> No

Calendar date on which testing for the current business day was started <RA> this is postponed for future release.

- Does GPWS write this date out to a file?
- Is this date encoded into one of the many databases when a new business day begins? If so, how do we retrieve the date?

- Can this information be pulled with an existing script? If so, how do we access the script? Is there permissions tied to the script that we should be aware of?
- How is the date changed from the EOD and Start of a new business day?

Current Business Date

- Is this date also tied to the database? <RA> Yes.
- How can we retrieve this information? Is this something tied to the testing data itself? <RA> Yes. In UNIX command prompt, we type the followings:
  - **sqlp**➔ to connect to the db associated with the test env. Will show result:

    ```
    Database Connection Information

     Database server        = DB2/AIX64 8.2.5
     SQL authorization ID   = PWIMRO3
     Local database alias   = PWSMROI3
    ```
  - **setpwsdbo**➔ to set schema in database to PWSDBO
  - **bd_get*primary***➔ Getting business date from *primary* database. Result:

    ```
    R5 Test Automation:/export/home/pwimro3 $ bd_get primary
    01/04/2007
    ```
- How does the current business day change? How does GPWS know it has changed? <RA> during testing after EOD process complete, op support (1) rolls the environment to the new business date (2) COLD start the env. This is done by running a script on UNIX command prompt: **startup_gpws*yyyy-mm-dd***➔ where yyyy-mm-dd is the new business date

Test Environment Name

- How is this information retrieved? <RA>from **.profile.EXT**. Op support currently set the environment name in the .profile.EXT. It does not have the VAH info currently but I think we can ask the Op Support to add the info there. The entry in the .profile.EXT file should look like this: PS1='"pwswbaimro3: IMRO3 Test Automation" ➔ pwswbaimro3 is the VAH, IMRO3 is the test environment, Test Automation is the name.
- How would an unknown person, without an idea of what the state of GPWS is in, in the testing cycle, be aware of this information? <RA> people who involve in testing MUST know what test environment they are using prior to testing start. The knowledge on what test environment used does not relate to the GPWS state at all.
- How would that person check or verify?
- What is 'UNIX VAH' ?<RA> Virtual App Handle

Test Environment Description

- How should GTI function between different tests? <RA> the GTI function should be the same on ALL test environments. ALL test environments must use the same script/code.
- Is there more important information that should be relevant depending on the type of test that GPWS is currently in?
- How would an outside individual recognize what test is currently in operation? <RA> Outside people whom are not involved in testing should contact Op Support or Env Team before login to test env. They are not allowed to be in test environment (UNIX).
- For Dashboard, since it is open access to everybody, the Test Environment information in the Common Interface Heading display what kind of testing going on in that environment, e.g: pwswbaimro3: IMRO3 Test Automation
- What is 'E2E' ? How does this differentiate between SIT and UAT? <RA> E2E is just a test beside SIT and UAT. SIT and UAT testing are required in all releases prior production installed. Meanwhile E2E is a one-time specific test for the E2E project.

Current states of the GPWS Application

- How can this be figured out manually? <RA> on UNIX command prompt: **get_stateGPWSMain**. Result:
  ```
  R5 Test Automation:/export/home/pwimro3 $ get_stateGPWSMain
  Pricing Underway
  ```
- Is this something that is tracked by hand and the information passed to individuals by 'word of mouth'?<RA> never trust "word of mouth" on testing!☺ The information is stored in database, Op support usually looks at the Toolbox for the info.

Test Environment Dashboard

Type of Last Application Start

- <RA> Whatever process start the environment (maybe new interface??) need to record it. Dashboard just need to retrieve the info

Scenario Under Test

- <RA> Op support or someone needs to record it somewhere, need to be determined. Dashboard just needs to retrieve the info

Enabled Transmissions

- <RA> Dashboard retrieves the info from database. On UNIX command prompt, execute this SQL:
- db2 "select * from T_XTRN_APPL_FILETY where OBND_FILE_ENBL_I <> 'N'"

Enabled Timer Events

- <RA> Dashboard retrieves the info from database. On UNIX command prompt, execute this SQL:

  'db2 "select from t_tmr_evt where dabl_i='N'"

- How do we selectively enable or disable events? <RA> the task is not done in the dashboard, it must be somewhere else (interface/whatever). Dashboard job is only retrieving information.
- Should we be aware of what all possible events are? <RA> don't need to

List of days stored in each database

- What sort of query should be used to pull this information? (see General Questions) <RA> Dashboard retrieves the info from database. On UNIX command prompt, execute this SQL: 'db2 "select distinct val_d from t_val_pnt, t_sec_val_prc where t_sec_val_prc.VAL_PNT_SID = t_val_pnt.val_pnt_sid"

General Questions

- Is the UAT checklist that we received a complete run through of a UAT testing cycle, or are these a checklist of mundane processes that must completed through every UAT test? **<RA> for every testing (SIT/UAT/any)**
- Is there a checklist available for a SIT testing cycle? Do any significant testing cycle procedures overlap between the two? <RA> the checklist is standard for ALL GPWS testing (UAT/SIT/others)
- What information is stored in it? <RA> all GPWS information, transactional and static
- How do the databases for GPWS' data and its operational database differ? <RA> we do not have operational database. When we mentioned database, it is always refer to GPWS database.
- Will we be able to modify tables and views? On one or both? Do you see us needing to? <RA> No. GTI Dashboard only retrieve data , GTI Interface should not write to GPWS testing database except for setting up timer events, enable/disable transmission, change business dates
- After GPWS has ended for a calendar day, what sort of state would the GTI like to remain in? (e.g. an open-non operational function, restricted to limited functions (starting a new business day)) <RA> I am hoping you are not confused on calendar date vs. business date. Calendar date is Today's date (system date). During testing, **we can have one business date for several calendar dates**. After calendar date ends, GTI should retain the GPWS status for the next testing day (tomorrow).
- Do we have access to create tables in the database so we can keep information available for the application? (States of certain procedures, etc).. Is this something that can be more feasibly accessed via globally writable file? <RA> if you need to create table, you need to do it on separate database (new) than GPWS database.

- How do you believe that we should keep track of information that is normally tracked via an excel sheet that is passed around to other individuals? <RA> we should record ALL information needed in a file/table/other, to be available for dashboard to access anytime. Depending on requirement, we may want to send that information to specific users so they can save it somewhere. This need discussion.
- Would the transition of a database to a different state (e.g. current to archive) affect GTI's ability to achieve certain pieces of information? <RA> when you change database on GPWS client (testing) from current to archive, you are switching to look archive database from the primary database. Both databases (primary and archive) are always there for us to retrieve information.

F. Fidelity Documentation

a. GPWS Testing Interface High Level Requirements

Fidelity Investments

FPCMS Testing & Quality Assurance Central Services

WPI MIS Major Qualifying Project

GPWS Testing Interface

Version: **[1.0]** **Date:** **August 4, 2008**

Prepared By:

Barbara Mackey

Contributors:

Rosetin Anwar

Doug Carriger

Meredith Frost

Ben Gedaminski

Glenn Janssen

Washesh Mehra

Erin Molgaard

Sharon Rowe

Revision History

Version number	Date	Originator	Reason for change
1.0	7/29/2008	Barbara Mackey	Initial Draft
1.0	7/31/2008	Barbara Mackey	Initial review by OSTE Glenn Janssen.
1.0	8/4/2008	Barbara Mackey	TDS Review: Rosetin Anwar, Doug Carriger, Meredith Frost, Glenn Janssen, Washesh Mehra
1.0	8/8/2008	Barbara Mackey	Project Team Review: Ben Gedaminski, Washesh Mehra, Erin Molgaard, Sharon Rowe

Introduction

Global Pricing Workstation (GPWS) is a software application used by Fidelity Investments' Accounting and Pricing Operations to collect, validate, and approve the current price of securities held by Fidelity funds. Those security prices are then used to calculate and distribute the net asset values of Fidelity mutual funds. Management teams and analysts in Pricing and Fund Accounting use GPWS to perform all daily pricing functions.

Two key quality assurance activities, which ensure that GPWS releases are ready for production, are System Integration Testing (SIT) and User Acceptance Testing (UAT). GPWS SIT and UAT are the shared responsibility of the Fidelity Reporting, Accounting and Pricing Services Business Analysis team (FRAPS BA) and Test Delivery Services (TDS), part of FPCMS Center of Excellence Testing & Quality Assurance Central Services (FPCMS COE). The Test Delivery Services team focuses on validating that new functionality and procedures conform to requirements and existing functionality and procedures continue to function as before. They are also responsible for the operational support aspects of GPWS SIT / UAT and supporting the operation of the SIT / UAT test environments. The FRAPS BA team focuses on reviewing and testing GPWS business functionality and procedures during GPWS UAT. For many test scenarios, the efforts of both teams are required. Both teams are geographically dispersed across the United States and India.

Test automation is another key initiative of the FPCMS COE. In the spring of 2008, a FPCMS COE project proposal for a testing interface was accepted by the Worcester Polytechnic Institute (WPI) Major Qualifying Projects program. In August 2008, work will begin on the GPWS Testing Interface Project. The goals of this project are to:

Create a testing interface for GPWS. (primary goal)

Propose functional solutions to address the need for operational support tools for GPWS SIT / UAT

Evaluate current GPWS SIT / UAT tools where appropriate

Identify operational support areas of GPWS SIT / UAT testing that could benefit from more or better tools

The purpose of this document is to present high-level requirements for the GPWS Testing Interface. For discussion, the interface is divided into two areas:

Dashboards

Operational support

For actual implementation, the dashboard and operational support areas may be combined if it can be done without negatively affecting usability or increasing risk.

General Requirements

Refer to the project charter (see “Section B – Bibliography”) for a detailed list of project:

objectives

dependencies

scope

success factors

constraints and assumptions

milestones and deliverables

Usage and Access

The following general requirements apply to the GPWS Testing Interface (GTI) access and usage:

GTI may be used by both operational support test engineers and functional test engineers.

GTI may be used in any GPWS test environment.

GTI may be accessed from any Fidelity site.

There must be a means to block access to selected functions, such as database updates or functions that would significantly negatively affect testing if invoked at the wrong time.

Use of GTI does not require a test engineer to have:

UNIX access

A GPWS user profile

Knowledge of the GPWS database schema

Operational

The following general operational requirements apply to the GPWS Testing Interface:

GTI implementation should include a web-based, graphical interface. The interface will allow test engineers to view test environment dashboards and perform operational support functions.

The dashboards and operational support interfaces described in this document may be combined where appropriate.

GTI cannot negatively affect the performance of the test environment in which it is running.

GTI needs to function with existing GPWS testing scripts (Korn Shell, Perl).

GTI must provide a means to monitor a function's progress and report success / failure.

All screen or log error messages must be clearly understandable by test engineers.

All screen or log error messages must be consistently formatted and worded.

All operational support functions must verify that the test engineer wishes to proceed before executing.

Default values must be provided for all options.

Deployment of GTI artifacts must ensure the GPI artifacts cannot be inadvertently deployed into the production environments.

Common Interface Heading

Every dashboard and operational support interface described in the following sections should show the following information:

Description	Value
Current calendar date	
Calendar date on which testing for the current business day was started	
Current business date	Application date. Business date under test.
Test environment name	Based on UNIX VAH

Test environment description	Examples: SIT, UAT, E2E
Current state of GPWS Main	Current state of the application

Standards and Guidelines

The following general standards and guidelines must be followed during GTI development:

GTI code must conform to Fidelity Perl and CGI standards and conventions.

Web-based code must adhere to secure code standards / policies.

GTI cannot require code changes to the GPWS application or the standard GPWS environment configuration.

GTI code and associated artifacts must be stored in standard COE repositories (i.e. ClearCase, Subversion, EDMS, Quality Center)

Dashboards

For the purpose of this document, “dashboard” refers to an interface that provides a test engineer with a way to view the current status, state or value of a component of a GPWS test environment. Such components include:

Application processes

GPWS database values

Application log messages

Environment and system values

Prior user selections

Test Environment Dashboard

This dashboard allows the test engineer to see "at a glance" the following information about the selected GPWS test environment:

Description	Value
Common interface heading	See Section 2.3
Type of last application start	Cold or warm
Scenario under test	Examples: Normal, Early Market Close, US Holiday, Canadian Holiday, 4-to-6 Test
Status of GPWS application processes (JVMs)	Up or down
Status of GPWS WAS processes	Up or down
Status of GPWS IHS processes	Up or down
Status of the GPWS database	Up or down
Enabled transmissions	List of all transmissions that are enabled and which have an

	active IP address
Enabled timer events	List of all GPWS timer events that are currently active
List of days in each database	Show business days stored in GPWS primary / archive / history databases

File Load Dashboard

During a business day, GPWS processes hundreds of mutual fund update and price files. During testing, similar files are grouped and loaded together.

This dashboard allows a test engineer to see the following information about the file loads for the given business day in the selected test environment:

Description	Value
Common interface heading	See Section 2.3
File Group Heading	Section for each file grouping. Examples: Start of day, intraday, fulfillment files, bond vendor files, APT files, ECN files
List of files in the group	
Source location of each file	Example: PROD (saved from production), TEST (created for test effort), AD HOC (created for specific test case)
NO LOAD indicator	Indicates that the file will not be loaded for given business

	day
Load status indicator	For the business day under test, indicates whether: the file was loaded successfully the file was not loaded the file load failed (currently in a “failed” state)
For SOD files only: filename of SOD APT file	

End-of-Day Processing Dashboard

A significant number of operational support tasks occur during GPWS end-of-day (EOD) processing. Several of these tasks are also among the most time-consuming tasks.

This dashboard offers a view at the current status of the major EOD tasks.

Description	Value
Common interface heading	See Section 2.3
Previous EOD Heading	
Calendar date of last EOD processing	
Business date of the last EOD processing	
Calendar date of last database archiving	For each of the GPWS databases
Calendar date of last database optimization	For each of the GPWS databases

Current EOD Heading	
Status of valuation point closure	Have functional test engineers closed the GPWS valuation points yet?
Failed state machine indicator	Are there currently any failed state machines?
Status of “end of day complete”	Has the GPWSMain state machine transitioned to EndofDayComplete?
Status of NASDAQ connections	Deleted?
Status of GPWS application processes (JVMs)	Up / down
Status of outbound files	Have all outbound files been saved?
Status of file cleanup	Have all inbound, testing, error log and outbound files been deleted?
Status of database connections	Are there any active connections to any of the GPWS databases?
Status of nightly cycle cleanup	Run, not run, in-progress
Status of database archiving	
Primary to archive database	Run, not run, in-progress
Archive to history database	Run, not run, in-progress, N/A
Status of database optimization	
Primary database	Run, not run, in-progress
Archive database	Run, not run, in-progress, N/A
History database	Run, not run, in-progress, N/A
Status of database exports	
Primary database	Run, not run, in-progress
Archive database	Run, not run, in-progress

History database	Run, not run, in-progress
Status of EOD statistics	Collected, not collected, in-progress

End-of-Day Statistics Dashboard

The Test Delivery Services team collects statistics on many of the EOD tasks. This helps the Team to set expectations and manage workflow, as well as provide a preliminary trending analysis for this critical application functionality.

This dashboard shows the EOD statistics from the last time EOD processing was run in a selected test environment.

Description	Value
Common interface heading	See Section 2.3
Calendar date of last EOD processing	
Business date of the last EOD processing	
Calendar date of last database archiving	For each of the GPWS databases
Calendar date of last database optimization	For each of the GPWS databases
Value of each statistic collected	See “Appendix B – Bibliography”

Operational Support

The Operational Support interfaces described in this section provide a test engineer with a means to perform many of the tasks needed to support GPWS test activities. Such tasks include:

Test environment preparation

Process control and monitoring

Loading mutual fund update and price files

Initiating end-of-day activities

Performing common GPWS database queries and updates

Currently, access to these tasks is limited to operational support test engineers. These tasks also require knowledge of UNIX, SQL and the GPWS database schema. The intent of creating the operational support interfaces is to expand the ability to perform these tasks to functional test engineers as well.

Test Environment Burn-in

At the start of every GPWS test effort, it is the responsibility of the operational support test engineer to prepare the test environment for test activities. While test preparations may vary widely between tests efforts, Test Delivery Services has defined a set of preparatory steps that must be performed every time, regardless of the test effort. These steps are referred to as “burn-in”.

The critical objective of performing the test environment burn-in is to ensure that there is nothing in the test environment or its associated database that would inadvertently cause an outbound test file to be created and transmitted to another server or application. While an accidental transmission to another test system may be just an inconvenience, an errant transmission to a production system could potentially cause a major business disruption or negatively affect production mutual fund calculations.

The details of the current burn-in procedure can be found in the GPWS burn-in checklist (see “Section B - Bibliography”).

In general, tasks include:

Notification that the burn-in is taking place

Verification of the current business date

Updating the GPWS database to

disable all outbound transmissions

disable all timer events

disable all fund distribution flags

disable all NAV calculation flags

disable transmissions to NASDAQ

Verification that test environment directories are empty where expected

Updating the test environment profile

Enable / disable error log “scrubbing” functionality

This interface allows the test engineer to:

automatically execute the burn-in checklist tasks

override selected tasks where applicable

prevent the override of selected tasks where applicable

determine when a task is completed successfully or fails

execute a single burn-in task as well as a full burn-in

Process Control

The Process Control interface allows a test engineer to start and stop all of the types of GPWS processes for a specific test environment.

This interface also includes two functions which are closely related to process control:

“Cold” start function

Error log rotation function

Description	Value
Common interface heading	See Section 2.3
GPWS application processes (JVMs)	May select to start / stop all JVMs or individual JVMs (“warm” start)
GPWS WAS processes	May select to start / stop all WAS processes or individual WAS processes

GPWS IHS processes	May select to start / stop all IHS processes or individual WAS processes
GPWS "Cold" start	Allows the test engineer to start GPWS for a new business day. A cold start performs application initialization.
Error log rotation	Creates a new set of application logs. Renames a predefined number of old logs. Deletes logs specified as no longer needed.

This interface will also:

Verify that a test engineer really wants to stop / start process

Prevent a test engineer from starting a process that is already running (without stopping the process first).

Prevent a test engineer from attempting to stop a process that is already stopped.

For "cold" start:

Allow the user to specify a business date

Verify that GPWS is in the "end of day complete" state before allowing the "cold" start to run

Issue a warning before performing the "cold" start and request additional verification

File Loads

The File Load interface allows a test engineer to:

From the groups of GPWS inbound files, select the groups to load for a given business day

From the groups of GPWS inbound files, select the groups to not load for a given business day

Reverse a “no load” later if needed

Exclude an individual file from a group file load

Initiate an ad hoc file load

Specify the source for a file load (maps to a source directory)

Specify the start-of-day APT file for a given business day

This interface will also:

Enforce the order of file loads where appropriate

Prevent a test engineer from executing a file load before previous loads are complete

Verify that the test engineer really wants to repeat a file load that is already completed

End-of-Day Processing

The details of the current end-of-day processing can be found in the GPWS Operational Day checklist and End-of-Day statistics spreadsheet (see “Section B - Bibliography”).

In general, tasks include:

Verification that business processing is done for the business day

Deletion of the NASDAQ connections

Process termination

Collection of outbound files

Directory cleanup

Termination of active database connections

Running nightly cycle cleanup

Performing database archiving and optimization

Performing database exports

Collection of EOD statistics

This interface allows the test engineer to:

automatically execute the EOD checklist tasks

override selected tasks where applicable

prevent the override of selected tasks where applicable

determine when a task is completed successfully or fails

This interface will also:

Enforce the order of EOD tasks where appropriate

Prevent a test engineer from executing an EOD task before previous tasks are complete

Verify that the test engineer really wants to repeat an EOD task that is already completed

Common Database Queries and Updates

This interface allows a test engineer to perform a predefined set of database queries and updates without requiring direct access to the GPWS database for a given test environment.

The complete list of queries / updates will be defined during the detailed requirements gathering phase. However, the list should include:

Setting EOD timer events

Adding / removing NASDAQ connections

Changes to the GPWS holiday calendar

Setting fund attribute flags (i.e. distribution flags, NAV calculation flags)

Setup for FTP transmissions

This interface will also:

Allow test engineers without knowledge of the GPWS database schema or SQL to query / update the database

For all updates, query the database to show the final results of the changes

Verify that the test engineer really wants to perform a requested update

Automate some of the more time-consuming database update procedures

Ensure database updates are performed accurately and consistently

Appendix A – Requirements Prioritization

A number of criteria were considered during requirements prioritization:

Time it takes to perform a task manually

Frequency of task during a testing business day

Complexity of a task

Risk mitigated by task automation

Benefits added by removing the dependence on operational support test engineers to perform the task

The amount of test automation that currently exists or is in-progress for a task

For the WPI MIS Major Qualifying Project, FPCMS Test Delivery Services prioritizes the GPWS Testing Interface deliverables as follows.

Primary Deliverables

Priority	Dashboard / Operational Support Interface	Section
1	End-of-day Processing Operational Support Interface	4.4

	(excluding end-of-day statistic collection functionality)	
2	End-of-day Processing Dashboard	3.3
3	Test Environment Dashboard	3.1
4	Process Control Operational Support Interface	4.2
5	File Loads Operational Support Interface	4.3
6	File Load Dashboard	3.2

Secondary Deliverables

Priority	Dashboard / Operational Support Interface	Section
1	Test Environment Burn-in Operational Support Interface	4.1
2	End-of-day Statistics collection functionality	4.4
3	End-of-day Statistics Dashboard	3.4
4	Common Database Queries and Updates Operational Interface	4.5

Tertiary Deliverables

Priority	Dashboard / Operational Support Interface	Section
1	Propose functional solutions to address the need for operational support tools for GPWS testing	1.0
2	Evaluate current GPWS testing tools where appropriate	1.0

3	Identify operational support areas of GPWS testing that could benefit from more or better tools	1.0

Appendix B – Bibliography

FPCMS Center of Excellence Testing Framework

http://edms.fmr.com/edms/component/getcontent?objectId=09016fea802ca70f

GPWS Operational Day Checklist (sample from GPWS R5.0 UAT)

http://edms.fmr.com/edms/component/getcontent?objectId=09016fea8043af5a

GPWS R5.0 EOD Statistics - SIT

http://edms.fmr.com/edms/drl/objectId/09016fea804129dd

GPWS R5.0 UAT Burn-in Plan

http://edms.fmr.com/edms/component/getcontent?objectId=09016fea8043afab&amp;current=true

GPWS Technical Guide Website

http://gpwstechnicalguide.fmr.com

Project Charter

<insert link here>

b. File Load Detail Requirements

Fidelity Investments

FPCMS Testing & Quality Assurance Central Services

GPWS Testing Interface

Fileload Detail Requirements

Version: **[1.0]** **Date:** **September 8, 2008**

Prepared By:

Rosetin Anwar

Contributors:

Kim Helbick

Washesh Mehra

Erin Molgaard

Revision History

Version number	Date	Originator	Reason for change
1.0	9/8/2008	Rosetin Anwar	Initial Draft
1.0	9/22/2008	Rosetin Anwar	Initial review by TA team

Introduction

This document is an initial attempt by COE Test Automation team to propose details on GPWS File Load Dashboard and Operation Support File Load Interface based on high level requirement document "GPWS Testing Interface" written for WPI MIS Major Qualifying Project. The GPWS Testing Interface document can be found in this SharePoint site:

http://sharepoint.fmr.com/sites/FPCMSTestingServices/Shared%20Documents/WPI%20MQP%20Program/WPI_HLR_ver01.doc

Requirement details for the other dashboards and operational support interfaces mentioned in the GPWS Testing Interface document will be provided by the WPI MIS team.

The GPWS Testing Interface (GTI) that can be divided into two areas:

Dashboards

Test Environment Dashboard

File Load Dashboard

End-Of-Day Processing Dashboard

End-Of-Day Statistics Dashboard

Operational Support

Test Environment Burn-in

Process Control

File Loads

End-Of-day Processing

Common Database Queries and Updates

The GPWS Testing Interface (GTI) should apply the following general requirements:

GTI may be used by both operational support test engineers and functional test engineers.

GTI may be used in any GPWS test environment.

GTI may be accessed from any Fidelity site.

There must be a means to block access to selected functions, such as database updates or functions that would significantly negatively impact testing if invoked at the wrong time.

Use of GTI does not require a test engineer to have:

UNIX access

A GPWS user profile

Knowledge of the GPWS database schema

The following general operational requirements also apply to the GPWS Testing Interface:

GTI implementation should include a web-based, graphical interface. The interface will allow test engineers to view test environment dashboards and perform operational support functions.

The dashboards and operational support interfaces described in this document may be combined where appropriate.

GTI cannot negatively impact the performance of the test environment in which it is running.

GTI needs to function with existing GPWS testing scripts (Korn Shell, Perl).

GTI must provide a means to monitor a function's progress and report Loaded / failure.

All screen or log error messages must be clearly understandable by test engineers.

All screen or log error messages must be consistently formatted and worded.

All operational support functions must verify that the test engineer wishes to proceed before executing.

Default values must be provided for all options.

Deployment of GTI artifacts must ensure the GPI artifacts cannot be inadvertently deployed into the production environments.

Proposal on Requirement Details

File Load Dashboard

As mentioned in the GTI Testing Interface requirement document, “dashboard” refers to an interface that provides a test engineer with a way to view the current status, state or value of a component of a GPWS test environment.

File Load Dashboard allows a test engineer to see file load information for a given business day in a selected test environment.

User (test engineer or functional tester) access File Load Dashboard via GPWS Testing Interface main screen

It has open access, no password is required to login

It is a web-based interface:

It does not require a test engineer to have a UNIX access, a GPWS user profile, nor the knowledge of the GPWS database schema.

The code to retrieve data is written in Perl and resides on UNIX CGI environment.

Each test environment has the code installed as part of initial test environment deployed by Dave Erickson’s team.

The code will be owned by COE Test Automation Team. Any updates on dashboard functionality should be requested to the COE TA team.

Each GPWS test environment has its own file load dashboard.

The dashboard displays heading and file load status:

Common interface heading:

Today date, which is current calendar date

Current business date

Test information VAH and description

Current GPWS Main state

File Load Status:

GPWS file loads are grouped into Start Of Day, Intraday, APT, Fulfillment, Bond Vendor and ECN

User has an option to display ALL file load status in ALL groups

User clicks the group to get status on files in that group. If there is any file load failure in the group, the dashboard will automatically expand that particular group to show each file status

Source of each file will:

Display ProdSaved, Test or AdHoc if the file has been loaded.

Display blank if the file has not been loaded

Display No Load indicator (ON/blank) on each file:

Display “ON” when the file does not planned to load for given business day

Otherwise display blank

For each file, display one of the file load status: Loaded, Not Loaded, Fail or Blocked

Status Not Loaded when the file has not been loaded

Status Loaded when the file has been successfully loaded

Status Blocked when the No Load indicator is ON

Status Fail when the file load failed. Fail status should be displayed in Red

When the status display Fail, user can click the word Fail to display details for that specific file

For files that come multiple times in a business day e.g. intraday, the dashboard only displays one entry

Display status of fulfillment file load on each hour (0000-1810) from Primary Server only (MBP) for each source (Reuters and S&P)

The dashboard will not refresh automatically. User clicks the ‘Refresh’ button to refresh dashboard

Display these buttons:

Button “GPWS Test Interface” – to go back to the GTI main screen

Button “File Load Interface” – to go to file load interface

Button “Refresh” – to refresh file load dashboard

Open Questions:

For files that come more than one e.g. intraday pg_fie_up / pg_sw_up, how should we show the status when one file load fails and the rest of file load are successful?

How should we display status on dashboard? The list of files can be very long such as Fulfillment files. Two options displaying the file load status:

1st option – to display all of them on file load dashboard main page

2nd option – display only the file load group. User clicks file group to get details of each file load in the group

GPWS Testing Interface document (page 5) includes "Calendar date on which testing for the current business day was started" in the heading to display. Why do we need this information?

Why do we need to display Today's Date on dashboard?

The following is the file load dashboard display:

Items	Value		
Today date:	09-08-2008		
Current business date:	01-04-2007		
Test environment:	pwswbaimro3: IMRO3 Test Automation		
GPWS Main:	Pricing Underway		
File Load:	Source:	No Load Indicator	Load Status
Display ALL File Load status			
START OF DAY			
(display files below when the			

SOD group is clicked)			
pg_focas_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
pg_mnymkt_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
pg_shaw_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
boi1fund	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
boi1shrs	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
pg_pcf_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
pg_fi_s_noa_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
PG_HH_IN	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
INTRADAY			
(display files below when the			

Intraday group is clicked)			
HOLDCDBW	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
FOI1SHWB	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
SECUCDBW	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
pg_mnymkt_up	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
pg_plfb_in	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
pg_fie_up	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
pg_sw_up	ProdSaved/ Test/ AdHoc/blank	ON	Blocked
APT			
(display files below when the APT group is clicked)			
pg_fi_s_noa_in	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
pg_mm_s_noa_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
pg_io_s_noa_in	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
FULFILLMENT			
(display files below when the Fulfillment group is clicked)			
RTRUSA 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 0000 MBP	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
RTRMUT 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 0000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

RTRCAN 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 0730 MBP	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
SPCFOR2 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 0730 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

SPCUSA 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1000 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1000 MBP	ProdSaved/ Test/ AdHoc/blank		
			Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1100 MBP	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
RTRFWD 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

SPCMUT 1100 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

SPCEXC 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1200 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

RTRFOR2 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1300 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

RTRCAN 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1400 MBP	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
SPCFOR2 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1400 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFWD 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

SPCUSA 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1500 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRCAN 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTREXC 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

RTRFWD 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR1 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRFOR2 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRMUT 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCCAN 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCEXC 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFWD 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR1 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCFOR2 1600 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCMUT 1600 MBP	ProdSaved/ Test/		Loaded/ Not Loaded/

	AdHoc/blank		Blocked/ Fail
RTRUSA 1630 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1630 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1700 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1700 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
RTRUSA 1810 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCUSA 1810 MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
BOND			
(display files below when the Bond group is clicked)			

BEARSABS MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
BEARSCMO MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
BEARSCOR MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
BEARSGOV MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
BEARSMBK MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCABS MB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCCANB	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCCMO	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCCONV	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCCOR	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

IDCEMG	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCFVF	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCGOV	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCJUNK	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCMBK	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCMUN	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCOPP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCTMM	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
IDCTRUST	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
KMUN	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

FRIBOND	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
FMR_Muni	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
FMR_Taxable_Bonds	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPTerm	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
ECN			
(display files below when the ECN group is clicked)			
RTRECN MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail
SPCECN MBP	ProdSaved/ Test/ AdHoc/blank		Loaded/ Not Loaded/ Blocked/ Fail

Op Support File Load Interface

The Operational Support File Load interface described in this section provide a test engineer with a means to perform file load activities and monitor them without login into a UNIX test environment. The intent of creating the operational support interface is to expand the ability to perform these tasks to functional test engineers as well

Based on the high level requirement document, the file load interface allows a test engineer to:

Select one or more groups of GPWS inbound files to load for a given business day

Select one or more groups of GPWS inbound files to not load for a given business day

Select one or more GPWS inbound files to load for a given business day

Select one or more GPWS inbound files to not load for a given business day

Reverse a "no load" indicator later if needed

Exclude an individual file from a group file load

Initiate an ad hoc file load

Specify the source for a file load (maps to a source directory)

Specify the start-of-day APT file for a given business day

Enforce the order of file loads where appropriate

Prevent a test engineer from executing a file load before previous loads are complete

Verify that the test engineer really wants to repeat a file load that is already completed

Proposal for the Op Support File Load Interface:

Operational support access the Op Support File Load Interface via GPWS Testing Interface main screen

It prompts login screen with user ID and password prior to user access the interface. A generic user ID can be setup and used by test engineers. The interface is independent to GPWS user profile setup in FSA Plus Admin

It is a web-based interface:

It does not require a test engineer to have a UNIX access, a GPWS user profile, nor the knowledge of the GPWS database schema.

The code to launch the interface is written in Perl and resides on UNIX CGI environment.

Each test environment will have the code installed as part of initial test environment deployed by Dave Erickson's team.

The code will be owned by COE Test Automation Team. Any updates on interface functionality should be requested to the team.

Each GPWS test environment has its own file load interface. User can not access file load interface on different test environment from one test environment. This should prevent user to accidently loading files on a wrong test environment.

Operation Engineer responsible to provide Test / AdHoc files and place them in appropriate directory

The interface displays heading and file load selections as follows:

Display common interface heading:

Today date, which is current calendar date

Current business date

Test information VAH and description

Current GPWS Main state

File Load Selection:

File load Interface displays the file load groups and their corresponding files:

Display groups of file load (Start of Day, Intraday, APT, Fulfillment, Bond Vendor and ECN). Display source (ProdSaved, Test or AdHoc), No Load indicatorand Status for each group

Display files within each group. Display source (ProdSaved, Test or AdHoc) and No Load indicatorand Status for each file

If user makes selection on a group level, the individual files within the group are disabled to prevent them to be selected.

If user makes selection on the file level, the associated group is disabled

User can make multiple selections on groups and on files in different group

The source field is display asdropdown. User make selection on one source only, multiple source selection is not allowed.

The No Load indicator display as check box. It is checked if the file does not plan to load for the business day. User has ability to reverse the No Load indicator if needed

If user set No Load indicator to ON, then the source will be blank and disabled

User has an option to load Fulfillment files for :

Each hour: 0000, 0730, 1000,…1600, …1810

ALL hours (0000 – 1600) with an exclusion of the 1630, 1700, 1810 file load

The file load interface enforce the order of file load

If prior file load has not been completed when user execute file load, then the interface display error

For the files that come multiple times in a business day e.g. intraday files e.g. pg_sw_up, the interface only displays one entry for the file. But it will load all the files

If user wants to repeat a file load that is already completed, the interface will ask for confirmation

The file load process will abort when having these conditions:

Failure on creating file load log

Failure in input parameters validation

The file to be loaded is missing in source directory

If find multiple files when expected to see only one file in the source directory

Failure when loading any Start of Day file

The file load process will not abort when one or more of multiple file load fails e.g. APT. But the errors are displayed in the return webpage

Display these buttons:

Button “GPWS Test Interface” – to go back to the GTI main screen

Button “File Load Dashboard” – to access file load dashboard

Button “Display Log” – bring user to Log Webpage to view file load activities from GPWS log file

Button “Refresh” – to refresh the file load interface

File Load Status:

Display file load status as Blank, Loaded, Blocked or Fail

Status Blankif the file has not been loaded

Status Loaded if the file has been successfully loaded

Status Blocked when the No Load indicator is ON

Status Fail when the file load failed.

Status is displayed on file level only and not on group level

Fail status is displayed in Red. User can click the word Fail to display details for that specific file

Log Webpage:

Display file load activities based on user selection:

Current business day

File load date timestamp

INFO messages from gpws.log

ERROR messages from gpws.log

Status of file load

Open Questions:

Since the list of files can be very long such as Fulfillment files, there are two options displaying the file load selection:

1st option – to display all on the interface

2nd option – to display file load groups on the interface. If user wants to make selection of the file level, click the group to get to the file selection

The interface enforces the order of file load. Please provide the order and what is required for each load.

What other information user want to see in the Log Webpage?

The following is a proposed interface display:

Items	Value		
Today date:	09-08-2008		
Current business date:	01-04-2007		
Test environment:	pwswbaimro3: IMRO3 Test Automation		
GPWS Main:	Pricing Underway		
File Load:	Source (dropdown):	No Load Indicator (check	Load Status

		box)	
START OF DAY:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select SOD group file load)			
pg_focas_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_mnymkt_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_shaw_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
boi1fund	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
boi1shrs	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_pcf_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_fi_s_noa_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
PG_HH_IN	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
INTRADAY:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select Intraday group file load)			
HOLDCDBW	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

FOI1SHWB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SECUCDBW	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_mnymkt_up	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_plfb_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_fie_up	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_sw_up	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
APT	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select APT group file load)			
pg_fi_s_noa_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_mm_s_noa_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
pg_io_s_noa_in	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT All Hours (0000 to 1600)	ProdSaved/ Test/ AdHoc	□	
(the group selection from 0000 to 1600 below will be disabled if			

user select ALL Hours file load)			
FULFILLMENT 0000	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 0000 group file load)			
RTRUSA 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCMUT 0000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 0730:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 0730 group file load)			
RTRUSA 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCFOR2 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 0730 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1000:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1000 group file load)			
RTRUSA 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCFOR1 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1000 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1100:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1100 group file load)			
RTRUSA 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCFWD 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1100 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1200:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1200 group file load)			
RTRUSA 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCEXC 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1200 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
		□	
FULFILLMENT 1300:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1300 group file load)			Blank/ Loaded/ Blocked/ Fail
RTRUSA 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCCAN 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1300 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1400:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1400 group file load)			
RTRUSA 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

SPCUSA 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1400 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1500:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1500 group file load)			
RTRUSA 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR2 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

RTRMUT 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1500 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1600:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select ALL Hours or 1600 group file load)			
RTRUSA 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRCAN 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTREXC 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFWD 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRFOR1 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

RTRFOR2 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
RTRMUT 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCCAN 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCEXC 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFWD 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR1 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCFOR2 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCMUT 1600 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1630:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select 1630 group file load)			
RTRUSA 1630 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1630 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1700:	ProdSaved/ Test/ AdHoc	□	

(the file selection below will be disabled if user select 1700 group file load)			
RTRUSA 1700 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1700 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FULFILLMENT 1810:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select 1800 group file load)			
RTRUSA 1810 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCUSA 1810 MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BOND All Vendors	ProdSaved/ Test/ AdHoc	□	
(the bond group selection below will be disabled if user select ALL Vendors file load)			
BOND BEARS	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select the BEAR			

group file load)			
BEARSABS MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BEARSCMO MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BEARSCOR MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BEARSGOV MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BEARSMBK MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BOND IDC:	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select the IDC group file load)			
IDCABS MB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCCANB	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCCMO	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCCONV	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCCOR	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCEMG	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCFVF	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

IDCGOV	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCJUNK	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCMBK	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCMUN	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCOPP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCTMM	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
IDCTRUST	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
BOND Others	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select the Bond Others group file load)			
KMUN	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FRIBOND	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FMR_Muni	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
FMR_Taxable_Bonds	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPTerm	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

ECN	ProdSaved/ Test/ AdHoc	□	
(the file selection below will be disabled if user select the ECN group file load)			
RTRECN MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail
SPCECN MBP	ProdSaved/ Test/ AdHoc	□	Blank/ Loaded/ Blocked/ Fail

G. System Request

Project Name: Fidelity GPWS Testing Interface

Project Sponsor: Name: Washesh Mehra

Organization: Fidelity Investments - FPCMS

Phone: 617-3782185 E-mail: Washesh.Mehra@FMR.COM

Business Need: This project has been initiated to improve the efficiency of Fidelity Investment's fund price verification process through business process automation and user-friendly interface development.

Business Requirements:

Using the system, Fidelity Investments should be able to reduce the time required to validate Fidelity fund pricing data. Fidelity Investments should be able to increase test execution consistency. The system should enable the following:

1. Automate test processes to reduce error and variability in the testing process.
2. Remove dependencies on UNIX access, GPWS user profile, and knowledge of GPWS database schema to enable functional test engineers to perform operational tasks.
3. Provide a web-enabled interface to run testing procedures.

Business Value:

It is expected that Fidelity Investments can reduce costs approximately 45% through process automation and error reduction. Fidelity Investments will be better able to utilize its resources because of reduced testing duration and a user-friendly testing interface. Additionally, Fidelity Investments will minimize non-value added activities, like debugging, with easily accessible testing information. Conservative estimates of annual tangible value to the company include:

1. $814 in cost savings per testing cycle
2. $9,765 in annual cost savings across all testing cycles

H. Use Cases

a. Use Case 1

Use case name:	Cold Start		**ID:**	1	**Importance level:**	High
Primary actor:	Test Engineer					
Short description:	Allows the test engineer to start GPWS for a new business day. A cold start performs application initialization.					
Trigger:	Test engineer begins a cold start test					
Type:	External					
Major Inputs:			**Major Outputs:**			
Description		**Source**	**Description**		**Destination**	
Username		Test Engineer	Access granted/declined		Dashboard log-in verification	
Date		Calendar	Specified business date		Date field	
EOD state verification		EOD field	EOD current state		EOD field	
Submit to start test		Test Engineer	Warning		Additional verification request	
Major Steps Performed					**Information for Steps**	
1. Test engineer enter username and password					<Test Engineer	
1.1 Access granted or denied					>Dashboard log-in verification	
2. Verifies that GPWS is in the “End of day complete” state before allowing the cold start					>EOD field	
2.1 Shows status of the GPWS state					<Date field	
3. Submit to start test					>Test Engineer	
3.1 Warning before executing and performing test					<Additional verification request	

b. Use Case 2

Use case name:	Error log rotation	**ID:**	2	**Importance level:**	High
Primary actor:	Test Engineer				
Short description:	Creates a new set of application logs. Renames a predefined number of old logs. Deletes logs specified as no longer needed.				
Trigger:	New log needed				
Type:	External				

Major Inputs:		**Major Outputs:**	
Description	**Source**	**Description**	**Destination**
Username and Password	Test Engineer	Access granted/declined	Dashboard log-in verification
Creates new log	Error log field	Creates log	New error log

Major Steps Performed	**Information for Steps**
1. Test engineer enter username and password	<Test Engineer
1.1 Access granted or denied	>Dashboard log-in verification
2. Begins to create an error log	>Error log field
2.1Creates error log	<New error log

c. Use Case 3

Use case name:	GPWS processes (JVMs, WAS, IHS)	**ID:**	4	**Importance level:**	High
Primary actor:	Test Engineer				
Short description:	May select to start/stop all or individual JVMs (warm start), WAS, or IHS processes.				
Trigger:	Test engineer begins a JVM, WAS or IHS process.				
Type:	External				

Major Inputs:		**Major Outputs:**	
Description	**Source**	**Description**	**Destination**
Username and Password	Test Engineer	Access granted/declined	Dashboard log-in verification
Verify stop/start process	Interface alert	OK/Cancel alert options	Process page
a. Prevent a start on a running process (without stopping the process first) b. Prevent a stop in a process that is currently stopped	Process Status alert	Running process status check	Process status page

Major Steps Performed	**Information for Steps**
1. Test engineer enter username and password	<Test Engineer
1.1 Access granted or denied	>Dashboard log-in verification
2. Verify stop/start process	>Process page
2.1 OK/Cancel alert options	<Date field
3a.Prevent a start on a running process (without stopping the process first) 3b. Prevent a stop in a process that is currently stopped	>Process Status alert
3.1 Running process status check	<Process status page

d. Use Case 4

Use case name:	EOD Checklist Override		**ID:**	5	**Importance level:** High
Primary actor:	Test Engineer				
Short description:	Execute the EOD checklist tasks				
Trigger:	EOD checklist has tasks that need to be completed				
Type:	External				

Major Inputs:		**Major Outputs:**	
Description	**Source**	**Description**	**Destination**
Username and Password	Test Engineer	Access granted/declined	Dashboard log-in verification
Execute EOD checklist	Interface alert	OK/Cancel Options	EOD checklist page
Override selected tasks	Interface alert	OK/Cancel Options	Task Verification page
Override task verification	Interface alert	Yes/No/Cancel Options	EOD checklist page

Major Steps Performed	**Information for Steps**
1. Test engineer enter username and password	<Test Engineer
1.1 Access granted or denied	> Dashboard log-in verification
2. Execute EOD checklist	> Interface alert
2.1 OK/Cancel Options	< EOD checklist page
3. Override selected tasks	> Interface alert
3.1 OK/Cancel Options	< Task Verification page
4. Override task verification	> Interface alert
4.1 Yes/No/Cancel Options	< EOD checklist page

e. Use Case 5

Use case name:	EOD Checklist task status	**ID:**	6	**Importance level:**	High
Primary actor:	Test Engineer				
Short description:	Checks the status of a task: completed successfully or fail				
Trigger:	Wants to check the status of a task				
Type:	External				

Major Inputs:		**Major Outputs:**	
Description	**Source**	**Description**	**Destination**
Username and Password	Test Engineer	Access granted/declined	Dashboard log-in verification
View processing dashboard for EOD item	Interface alert	OK/Cancel Options	EOD checklist page
Status check on task	Interface alert	OK/Cancel Options	Checklist status page

Major Steps Performed	**Information for Steps**
1. Test engineer enter username and password	<Test Engineer
1.1 Access granted or denied	>Dashboard log-in verification
2. View processing dashboard for EOD item	<Interface alert
2.1OK/Cancel Options	>EOD checklist page
3.Status check on task	<Interface alert
3.1 OK/Cancel Options	>Checklist status page

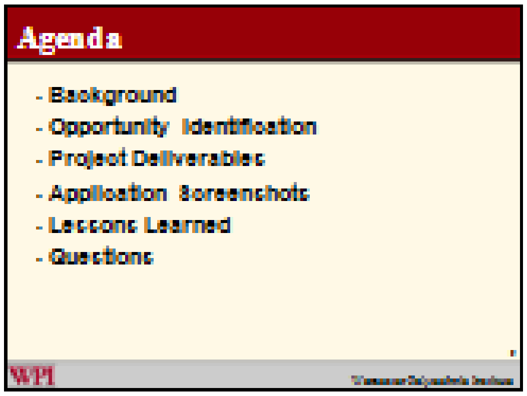
I. Fidelity Presentation

Global Pricing Workstation Testing Interface Project Presentation

Elizabeth Carey
Dan Dahlberg
Michael Diamant
Augustina Mills

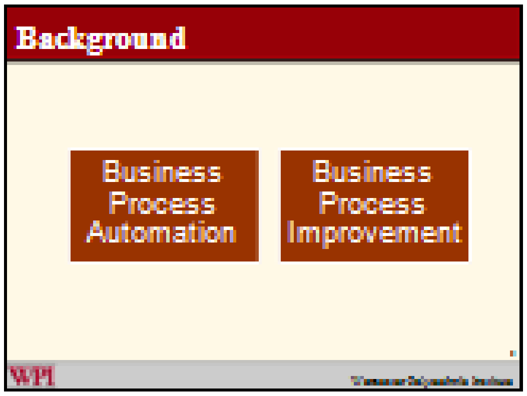
WPI

Worcester Polytechnic Institute

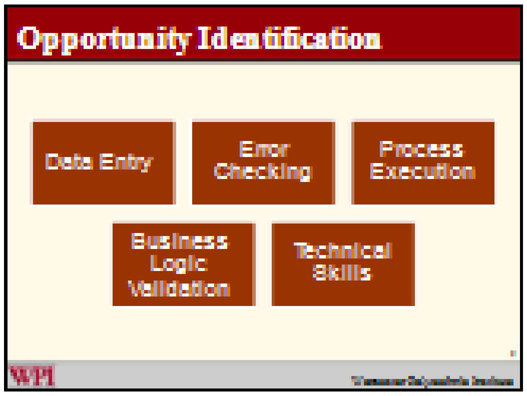


Agenda

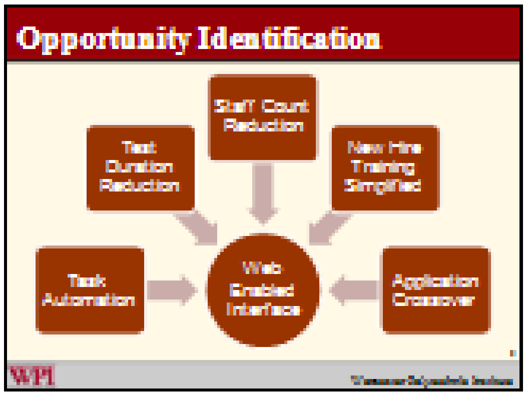
- Background
- Opportunity Identification
- Project Deliverables
- Application Screenshots
- Lessons Learned
- Questions



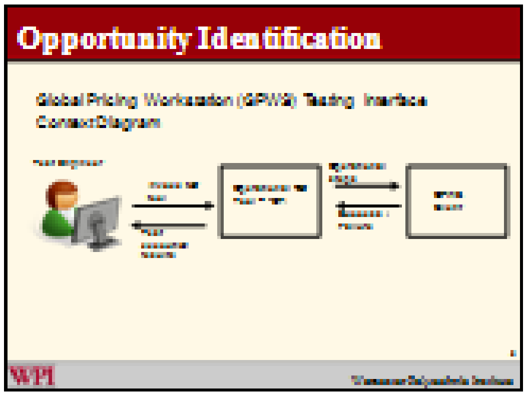
Background



Opportunity Identification

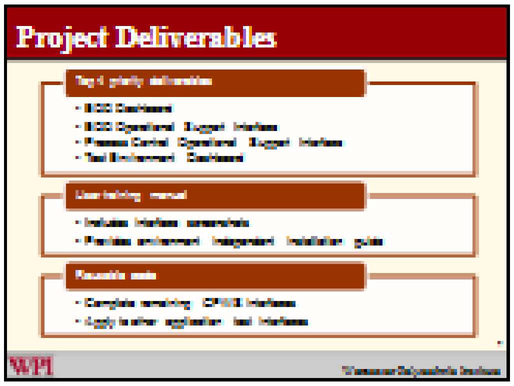


Opportunity Identification



Opportunity Identification

Global Pricing Workstation (GPWS) Testing Interface Context Diagram



Project Deliverables

Top 4 priority deliverables

- ECC Dashboard
- ECC Operational Support Interface
- Process Control Operational Support Interface
- Tool Environment Dashboard

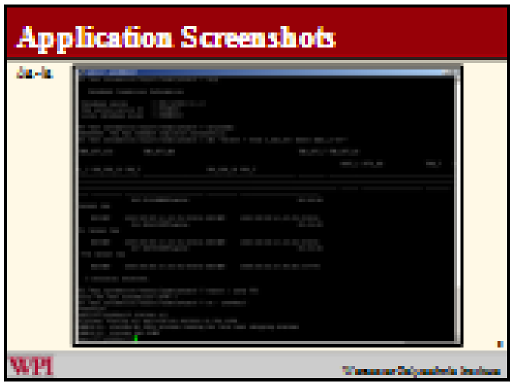
User training manual

- Includes interface screenshots
- Provides environment independent installation guide

Reusable code

- Complete working CPMS interfaces
- Apply to other application tool interfaces

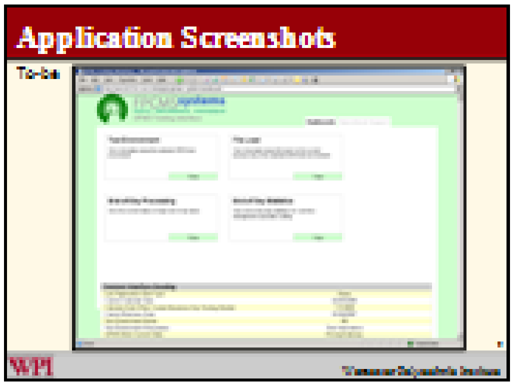
WPI Worcester Polytechnic Institute



Application Screenshots

As-is

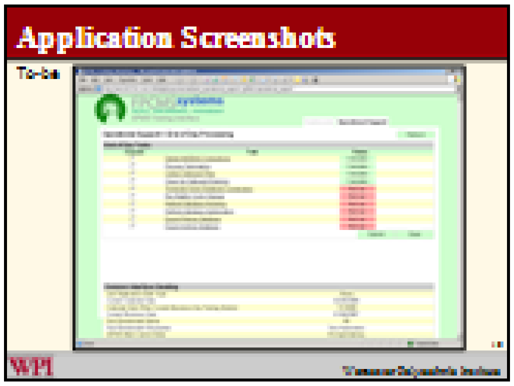
WPI Worcester Polytechnic Institute



Application Screenshots

To-be

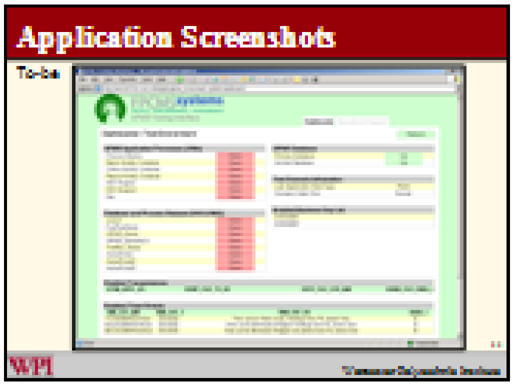
WPI Worcester Polytechnic Institute



Application Screenshots

To-be

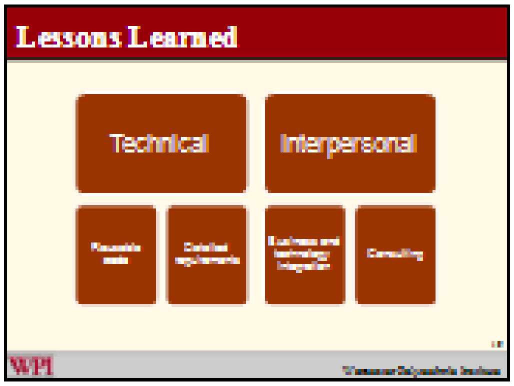
WPI Worcester Polytechnic Institute



Application Screenshots

To-be

WPI Worcester Polytechnic Institute



Lessons Learned

Technical

Interpersonal

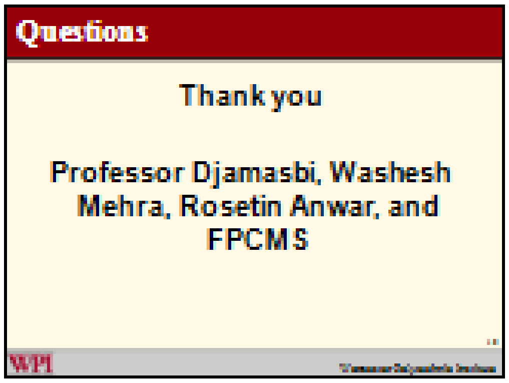
Reusable code

Detailed requirements

Business and technology integration

Consulting

WPI Worcester Polytechnic Institute


Questions

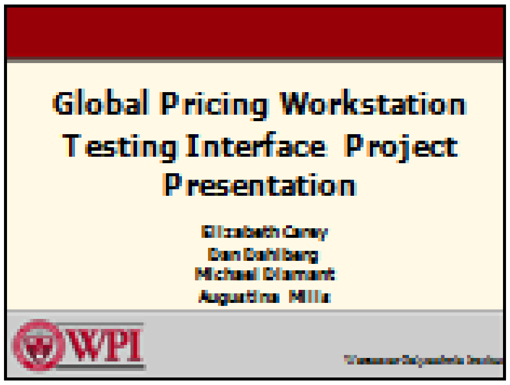
Thank you

Professor Djamasbi, Washesh Mehra, Rosetin Anwar, and FPCMS

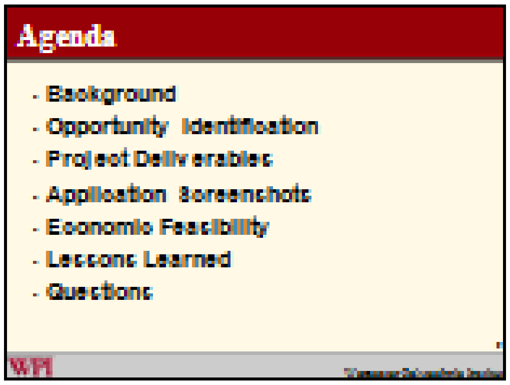
WPI

Worcester Polytechnic Institute

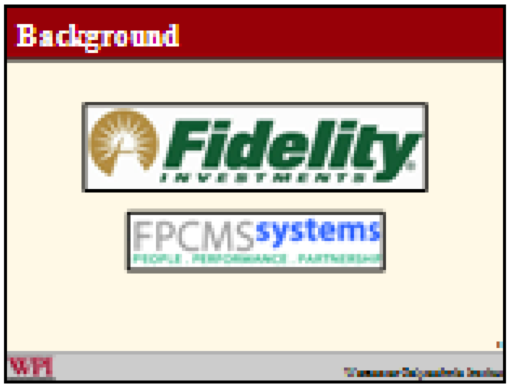
J. WPI Presentation

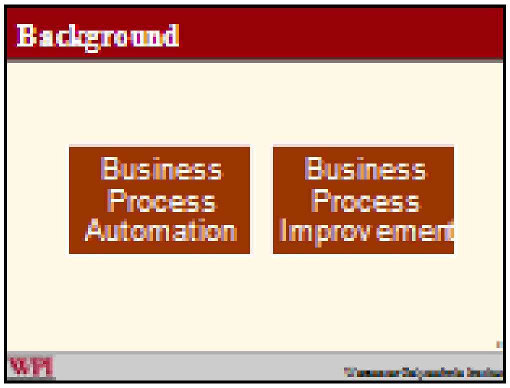

Global Pricing Workstation Testing Interface Project Presentation

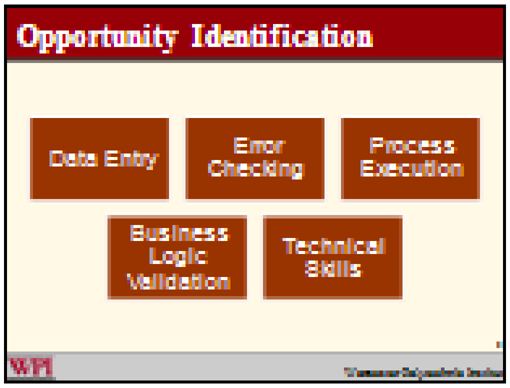
Elizabeth Carey
Dan Dahlberg
Michael Diamant
Augustina Mills

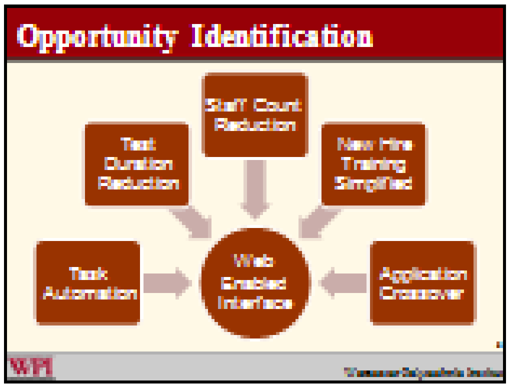

Agenda

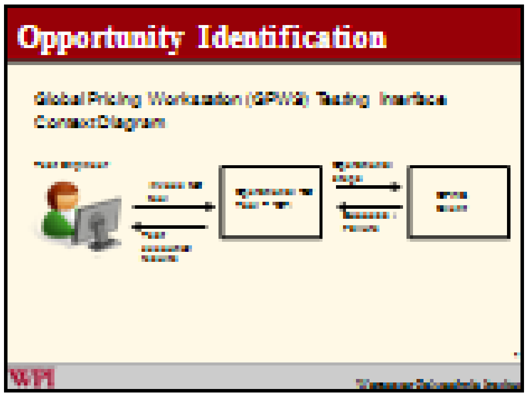
- Background
- Opportunity Identification
- Project Deliverables
- Application Screenshots
- Economic Feasibility
- Lessons Learned
- Questions


Background


Background


Opportunity Identification

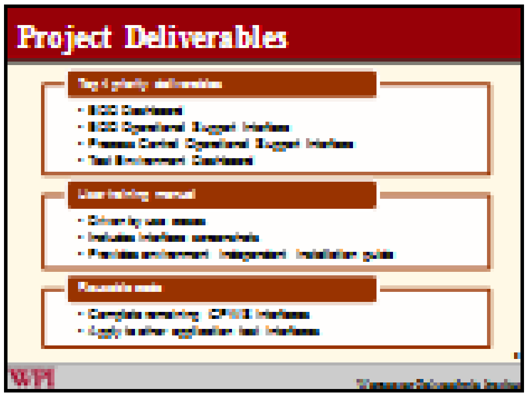

Opportunity Identification



Opportunity Identification

Global Pricing Workstation (GPWS) Testing Interface Context Diagram

Worcester Polytechnic Institute



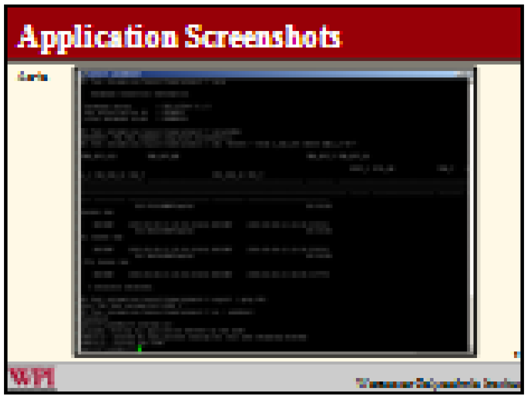
Project Deliverables

Top 4 priority deliverables
- BSS Dashboard
- BSS Operational Support Interface
- Process Control Operational Support Interface
- Test Environment Dashboard

User training manual
- Driven by use cases
- Includes interface screenshots
- Provides environment independent installation guide

Possible next steps
- Complete remaining GPWS interfaces
- Apply to other application test interfaces

Worcester Polytechnic Institute

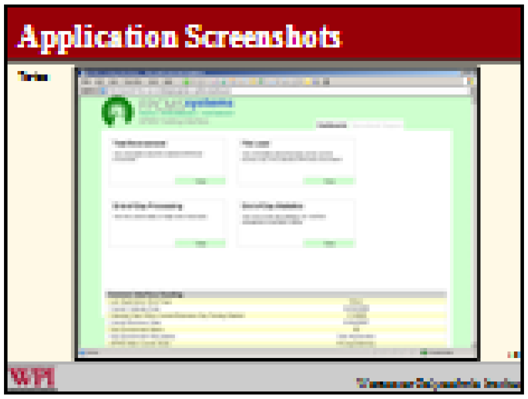


Application Screenshots

Linux

```
[illegible]
```

Worcester Polytechnic Institute

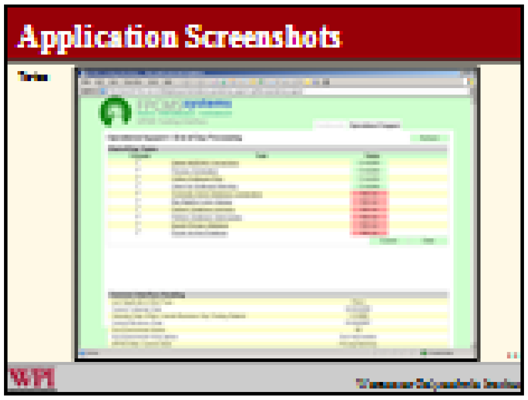


Application Screenshots

Tomcat

[illegible]

Worcester Polytechnic Institute

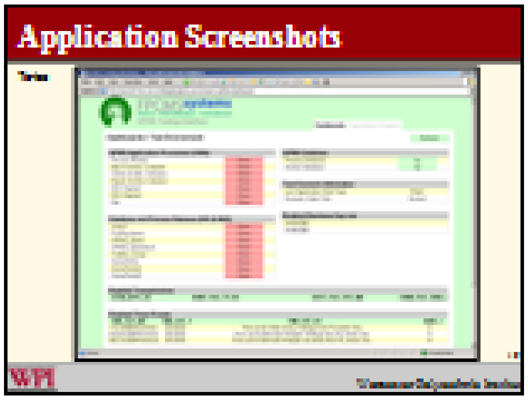


Application Screenshots

Tomcat

[illegible]

Worcester Polytechnic Institute

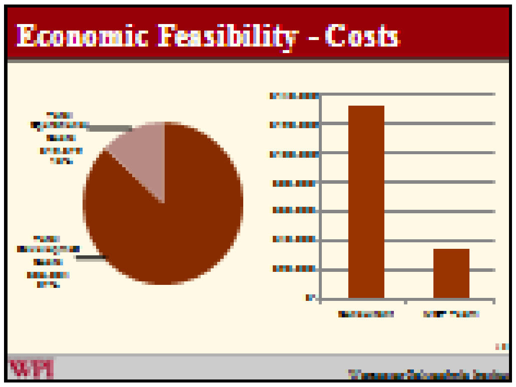


Application Screenshots

Tomcat

[illegible]

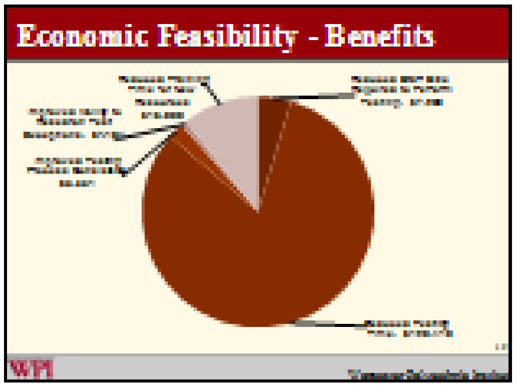
Worcester Polytechnic Institute



Economic Feasibility - Costs

WPI

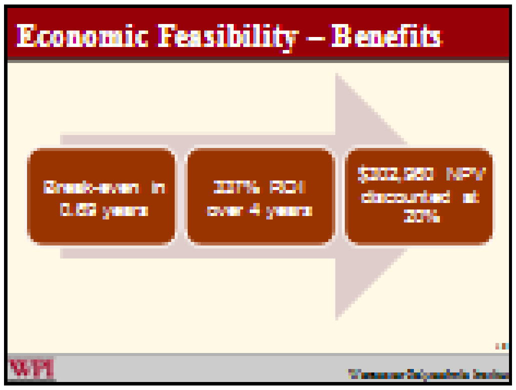
Worcester Polytechnic Institute



Economic Feasibility - Benefits

WPI

Worcester Polytechnic Institute



Economic Feasibility – Benefits

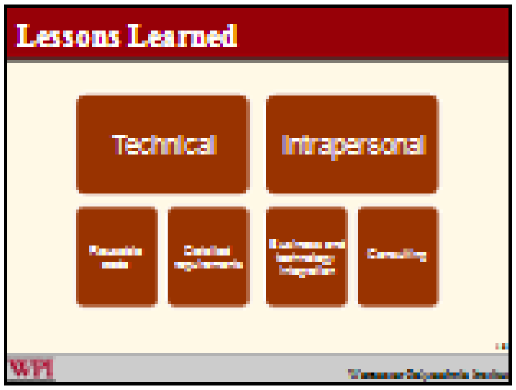
Break-even in 0.69 years

337% ROI over 4 years

$302,980 NPV discounted at 20%

WPI

Worcester Polytechnic Institute



Lessons Learned

Technical

Intrapersonal

Reusable code

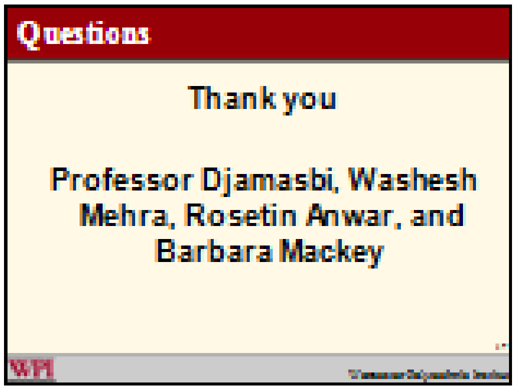
Detailed requirements

Business and technology integration

Consulting

WPI

Worcester Polytechnic Institute



Questions

Thank you

Professor Djamasbi, Washesh Mehra, Rosetin Anwar, and Barbara Mackey

WPI

Worcester Polytechnic Institute

K. WPI Poster

Fidelity Global Pricing Workstation Testing Interface: A Web-Enabled Test Automation Tool

Elizabeth Carey (MIS), Daniel Dahlberg (MIS), Michael Diamant (MIS), Augustina Mills (MIS)
Advisor: Professor Soussan Djamasbi

Abstract

The goal of this project was to create a web-based tool to interface with Global Pricing Workstation (GPWS) test environments on the UNIX platform. Functional solutions to address the need for operational support tools for GPWS System Integration Testing / User Acceptance Testing (UAT) were also proposed. Perl and CGI were used to create web-based interfaces for the four high-priority deliverables for GPWS. The final dashboards that were completed for this project can be used in Fidelity's GPWS with FPCMS and can also be used as a template for other areas of the department.

Background

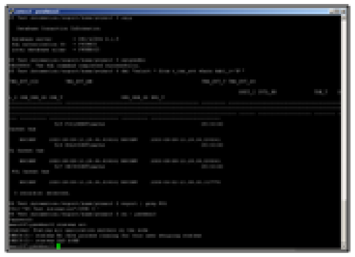
Fidelity Investments is one of the world's largest mutual fund firms. It currently serves more than 23 million individual and institutional clients and manages more than 300 funds with approximately $1.5 trillion of assets under management. The Fidelity Pricing and Cash Management Services (FPCMS) provides accounting and investment management support to Fidelity mutual funds.

The Global Pricing Workstation (GPWS) is the system of record for pricing Fidelity's mutual funds. GPWS was released in August of 2005 and has been in production for the past 3 years. It is rated an AAA application within Fidelity. The AAA rating means it has the highest level of availability, 99.5%, and recoverability, 30 minutes.

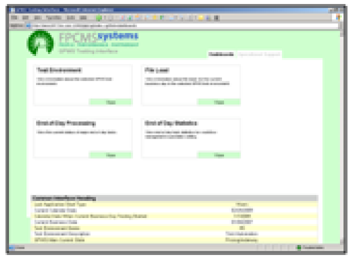
Currently, it is used to price approximately 58,000 securities for 4,554 funds with 275,769 holdings. The user community for GPWS is geographically dispersed and includes employees for pricing analysts, fund accountants, and production support. There are a total of 700 users with 11 production supports. GPWS is a Java-based application with C++ client and Spring operational interface. The database used is DB2 and is runs on AIX servers.

Application Design

The design portion of the project was focused on migrating GPWS to a web-based application. Prior to development by the MQP team, the user would view GPWS as it is in the following screen shot:



The screen shot below is an example of the web-based interface that was developed by the MQP team. The user friendly interface is intuitive and allows the user to perform the necessary functions for the deliverables.



Acknowledgments

Dr. Soussan Djamasbi
Washesh Mehra
Rosetin Anwar
Fidelity Pricing and Cash Management Services

Project Goals

The goal of this project was to create a web-based tool to interface with GPWS test environments on the UNIX platform. Along with the web-based interface, we were to propose functional solutions to address the need for operational support tools for GPWS System Integration Testing (SIT)/User Acceptance Testing (UAT), evaluate current GPWS SIT /UAT tools where appropriate, and identify areas of GPWS SIT / UAT testing that could benefit from more or better tools.

Project Deliverables

Top 4 Priority Deliverables
End-of-Day Dashboard
End-of-Day Operational Support Interface
Process Control Operational Support Interface
Test Environment Dashboard

User Training Manual
Includes interface screenshots
Provides environment independent installation guide

Reusable Code
Complete remaining GPWS interfaces
Apply to other application test interfaces

Conclusions/Recommendations

- Successful creation of a web-based tool for GPWS
- Reusable Code that can be implemented with the final portions of GPWS as well as other application
- Training manual and Implementation Guide to instruct users on how to navigate through the application and how to implement it in various environments
- FPCMS should consider implementing AJAX which will be most helpful in updating status values on dashboards