CS-5984 Group Project: Project Plan for the Software Project Benchmark System

Submitted in Partial Fulfillment of the Requirements of
CS-5984, SS: Software Project Management

Prepared by:

(Team 3 - *SIMTRIX WEB WEAVERS*)
Homer Simpson
Marge Simpson
Bart Simpson

Table of Contents

1 Revision History

Version	*Date*	*Description*	*Author(s)*
V15	12/07/04	Final, minor syntactic and consistency edits.	Homer Simpson Marge Simpson Bart Simpson
V14	12/07/04	Minor edits for consistency.	Homer Simpson
V13	12/07/04	Wrote project schedule, HW & SW requirements, estimation results. Updated assumptions.	Homer Simpson Marge Simpson Bart Simpson
V12	12/07/04	Revised process-base estimate to categorize each activity (e.g., CC, planning, risk analysis) into a RUP phase (e.g., inception, elaboration).	Homer Simpson Marge Simpson Bart Simpson
V11	12/06/04	Wrote environment section.	Homer Simpson
V10	12/06/04	Polished architecture section & incorporated description of Java Page Flows.	Marge Simpson
V9	12/06/04	Wrote introduction. Revised process-based estimate. Created risk information sheets for risk management section.	Homer Simpson Marge Simpson Bart Simpson
V8	12/06/04	Wrote risk management section.	Homer Simpson
V7	11/29/04	Fleshed out WEBMO estimate.	Marge Simpson
V6	11/28/04	Wrote architecture section.	Bart Simpson
V5	11/21/04	Wrote process-based estimate. Updated company profile.	Homer Simpson Marge Simpson Bart Simpson
V4	11/20/04	Wrote historical data section. Wrote development lifecycle section. Assigned features to iterations.	Homer Simpson Marge Simpson Bart Simpson
V3	11/20/04	Wrote SIMTRIX WEB WEAVERS company overview. Fleshed out objectives. Wrote user profile section. Defined all features.	Homer Simpson Marge Simpson Bart Simpson
V2	11/10/04	Wrote assumptions, system context, constraints. Outlined project overview. Outlined risk management.	Homer Simpson Marge Simpson Bart Simpson
V1	11/08/04	Revised formatting and finalized outline. Wrote glossary, project overview.	Marge Simpson Bart Simpson
Inception draft	11/06/04	Created first draft.	Homer Simpson Marge Simpson Bart Simpson

2 Introduction

The purpose of this project was to create a plan for the development of a Software Project Benchmark System (SPBS), a Web-centric system for storing and analyzing metrics on software projects. Our team was provided with the following documentation on the SPBS:[1]

- A high-level overview.
- A basic usage model, including user profiles
- A set of requirements
- An architecture
- A partial threat model, including a sampling of the associated countermeasures
- Some project planning templates, such as a template for the project plan itself and a spreadsheet containing an example Gantt chart.

Our team created a fictional organization, **SIMTRIX[2] WEB WEAVERS**, a relatively small but successful Web development company, and structured this plan as if it was written by **SIMTRIX WEB WEAVERS** personnel. This company is described in Section 2.1. A team of five people from **SIMTRIX WEB WEAVERS** will construct the SPBS, as described in Section 3.5.1 on page 54. This team will employ an iterative, incremental, agile-like development lifecycle characterized by test-as-you-go unit testing and bottom-up integration testing, as described in Section 3.2 on page 30. Also, because the system, including all source code, will be delivered to the company for maintenance, all source code shall be thoroughly documented using JavaDoc[3] Moreover, this team will develop the following artifacts and will ensure that these artifacts are current at the end of each development iteration:

- Product Overview
- Software Requirement Specification
- Software Architecture Document
- Software Design Document
- User's Manual

The remainder of this section, Section 2, contains a glossary of terms and provides an overview of the SPBS, project-level assumptions made by our team, a statement of scope (containing requirements, user profiles, features and the architecture), the system context, major constraints, and a high-level overview of our team's project plan, which is detailed in the remainder of this document.

One of the first activities undertaken by our team was to estimate the cost of the system. In an attempt to increase the accuracy of our estimate, our team employed two estimation

[1] This information was provided by Dr. Shawn A. Bohner, Ph.D.
[2] The name, "Simtrix," was taken from http://www.duffzone.co.uk/desktops/simptrix/simptrix10x7.jpg.
[3] http://java.sun.com/j2se/javadoc/

techniques (process-based and a parametric). Our team also ensured that different people developed each estimate and that the estimates remained independent. Section 3 details the estimation techniques applied, the projected cost of the system, and the resources our organization plans to employ to build the system.

Section 4 specifies the risk management strategy our team plans to employ, Section 5 describes the project schedule, including the expected deliverables and milestones, Section 6 describes the specific organization of the team that will be developing the SPBS, and Section 7 describes the development environment we plan to employ to implement the SPBS.

2.1 SIMTRIX WEB WEAVERS Company Overview

For the purposes of this project, our team created a fictional organization, **SIMTRIX WEB WEAVERS**. The names of the employees of the company were taken from the popular animated series, *The Simpsons* (hence, the company name). [14] The remainder of this section contains an overview of **SIMTRIX WEB WEAVERS**.

SIMTRIX WEB WEAVERS was founded in 1999 by two people, Marge (left) and Homer (right) Simpson,[4] in the Washington, D.C. area. The company creates custom Web applications, typically using the J2EE architecture. Currently at 23 people and growing, the company's growth is testament to their success. The company consists mainly of senior developers with 5+ years of experience in general software development and a variety of specializations, from front-end to database developers. Note, however, that while individuals may have more experience in one area than another, virtually everyone is versatile and capable of playing multiple roles (i.e., capable of "wearing many different hats"), as is typical in small organizations.

One of the original founders, Marge, has a background in metrics and process improvement, and due to this fact the organization collects metrics on every project. As a result, the organization has almost 5 years of project metrics to aid in estimating the cost and schedule for future projects.

2.2 Glossary

The terms used throughout this document are described below in alphabetical order:

- **Benchmark report** – The main product provided by the system is benchmarking data, which compares metrics retrieved from a set of projects. At a user's request, this data is provided in the form of a report, termed a benchmark report.

[4] Graphics taken from http://www.duffzone.co.uk/desktops/simptrix/simptrix10x7.jpg and modified by our group.

- **Customer** – For the purposes of this project, the customer refers to the company.
- **Java Page Flow (JPF)** – A visual layer built on top of Struts to show the flow through a Web site. It shows graphically what Web pages connect to what Actions and where each Action leads.
- **Membership administrator** – A member of an organization that has been designated with the role of managing a SPBS membership.
- **Model-View-Controller (MVC)** – an architectural pattern where the presentation logic (the view) and business logic (the model) are separated and communication between them is coordinated by a controller.
- **Redundancy – "**The existence of more than one piece of equipment, any of which could perform a given function. These multiple pieces of equipment are used to help improve the reliability and availability of the system. Redundancy is generally expressed as the number of pieces of equipment required and the total number available. For example, 2 out of 3 means that there are a total of 3 pieces of equipment and 2 pieces are required for proper operation of the system." [http://www.relexsoftware.com/resources/glossary.asp]
- **Registered User** – An individual who has an account with the SPBS and has logged into the system.
- **Site administrator** – See SPBS administrator.
- **SPBS** – Software Project Benchmark System. The SPBS provides customers with the following two high-level capabilities: (1) the ability to add project metrics to the SPBS and (2) the ability to obtain benchmarks that compare the metrics of a given project against the metrics of other projects in the SPBS database.
- **SPBS administrator** – The person who is responsible for administration of the SPBS. This person is a highly trusted member of the SPBS development team and has signed a non-disclosure agreement.
- **SPBS developer** – An individual who plays some role in the development or maintenance of the SPBS. All SPBS developers must sign non-disclosure agreements to ensure that sensitive information is kept secure.
- **SPBS Membership (or simply "Membership")** – Whenever an individual requests an account with the SPBS, a membership is established for the individual's organization. Associated with this membership is a minimum of one "membership administrator" and a set of users, who can utilize the system in a manner provided by the associated membership.
- **System** – The term, System, simply refers to the SPBS software.
- **Visitor** – Individuals who browse the SPBS Site but do not log into the Site.

2.3 Project Overview, Goals and Objectives

Figure 1 graphically depicts the basic usage model for the SPBS from a high level. At its core, the SBPS is a software development metric repository. The SPBS contains data from various organizations on software development, including software management. The SPBS permits authorized organizations (i.e., Customers) to add data to the repository and to request benchmark reports. The SPBS also allows authorized students to access the data for research purposes.

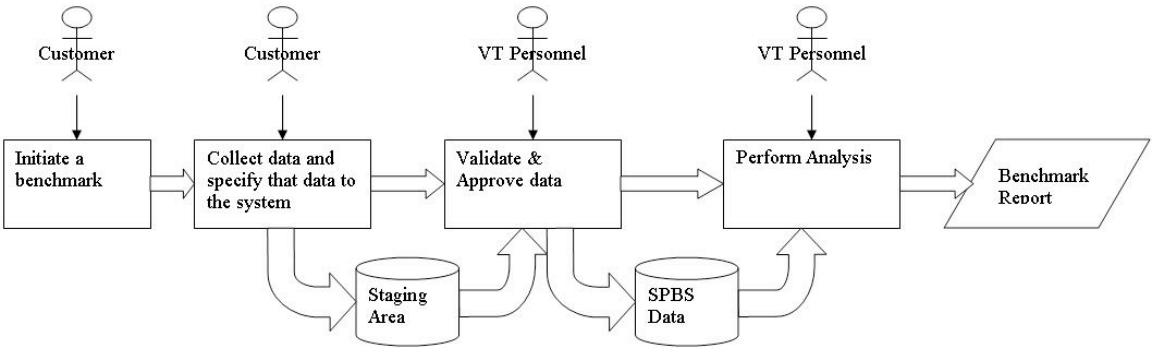


Figure 1. SPBS Basic Usage Model

As one can see from this figure, a customer may decide to create a benchmark. To this end, the customer initially collects a set of metrics data and specifies this data to the SPBS. This data is placed into a staging area, which is separate from the SPBS metrics repository. Upon receipt of this data, an individual at the company validates this data and works with the customer to resolve any anomalies. This data is only added to the SPBS metrics repository (labeled "SPBS Data" in the figure) once it has been successfully validated. This data is subsequently used, along with metrics data from other projects in the SPBS repository, to create benchmark reports.

Customers may obtain two kinds of benchmark reports from the system. The first report is automated in that the Customer may request a standard report to be displayed. This report contains a simple comparison of the Customer's metrics against the corresponding metrics for all projects in the database. The value of this report to an organization is subjective, but is expected to be low, on average. The reason for this is that data from all projects in the database, as opposed to data from projects with similar characteristics, is used. To illustrate this point, consider two software projects; a Web-centric project similar to this one and an embedded application characterized by extreme resource limitations. It can be argued that comparing SLOC per hour is irrelevant because of the difference in project focus. The Web-centric project is typically developed using a variety of productivity enhancement tools (e.g., HTML editors, JSP editors, code generators) and code efficiency is not critical. The embedded application, however, is typically developed without many (if any) productivity enhancement tools and efficiency is critical. As a result, it is expected that the SLOC per hour would be much higher for the embedded application than for the Web-centric application. Comparing the two projects may, therefore, not have much utility.

The second kind of report that may be requested is a manual report. Customers request this report by specifying the characteristics to be included in the report and issuing a request of the system. Upon receipt of this request, an individual at the company collects the requisite data from the database for comparison against the metrics provided by the customer. This individual subsequently provides this report to the customer in a secure manner. Customers are charged for this service.

The primary motivators for this project are:

- Software project managers want to know how their organizations are performing with respect to other organizations, and
- Students, particularly graduate students, want access to project management data for research.

The primary goal of this project is to develop project benchmarking software that allows organizations to add benchmarking data to and obtain benchmark reports from the system.

2.4 Assumptions

The following assumptions were made during while creating this plan:

- The Company has hired SIMTRIX WEB WEAVERS to build the SPBS.
- The Company has $250,000 to spend on the construction of the project, including any additional hardware and software.
- The Company has mandated that open source software be used for deployment to decreasing licensing and maintenance costs. It is assumed that the company has done the necessary research on total cost of ownership to justify this decision.
- The Company will be maintaining the system and as a result, will need a relatively comprehensive set of documentation as well as the source code.
- The Company's existing technology infrastructure (e.g., Internet connection, firewalls, demilitarized zone) will be used.
- The Company has specified neither maintainability nor reliability constraints.
- The Company will obtain licenses for OptimalJ [10] at an education discount for continued management of this project. Because the number of license seats are not known, nor what discount Compuware will provide, this cost cannot be included in this project plan.
- Development licenses for WebLogic Workshop are free.[5]
- The Java2 Enterprise Edition (J2EE) platform has been selected for implementation. Additionally, the use of Struts as a Web application infrastructure has been proscribed.
- A detailed architecture document exists and maps the architecture to J2EE. The summary of this mapping provided in Section 2.5.4 is therefore assumed to be correct and well supported.

2.5 System Statement of Scope

This section defines the scope of this project by providing a brief overview of the system requirements, user profiles and corresponding features and benefits, role-independent benefits, a matrix that traces these benefits back to the requirements, and high-level architecture.

[5] http://dev2dev.bea.com/products/wlworkshop81/index.jsp

2.5.1 Requirements

This section details the requirements of the SPBS. These requirements were originally provided by Dr. Shawn A. Bohner, Ph.D., and have been grouped according into the following categories:

- User Functionality Requirements – These requirements detail the functionality that is available to users of the system. It is important to note, however, that not all functionality specified in these requirements is available to all users. For example, requirement R8 indicates that users have the ability to add new projects to the system. This functionality is, however, only available to project managers, membership administrators, and site administrators, as specified in Section 2.5.2.2 summarized in Table 6 on page 15. Thus, these requirements detail functionality that must characterize the system without concern to roles and access control.
- System Administrator Requirements – These requirements detail the functionality that the system administrator must have.
- System Functional Requirements – These requirements detail the role-independent functionality that must characterize the system.
- Critical Non-functional Requirements – These requirements detail the non-functional requirements that must characterize the system.
- Future Enhancements – These items detail possible future enhancements to the system but do not specify any new requirements. While designing and implementing the system, however, our development team will consider these possible enhancements to ensure that they can be supported.

Our group noticed that some requirements, as evidenced by the use cases provided by Dr. Bohner, were missing from the original requirement specification. For this reason, our group added these requirements to the list. All requirements added have been assigned a numbering scheme whereby the company acronym, SWW, prefixes the number. For example, SWWR71 in Table 1 was added by our team.

2.5.1.1 User Requirements

Table 1 User Functionality Requirements

	Requirement	Priority
R1	A user types in the URL and can see the home page displayed on a remote client. The home page includes options to register new users, login and a link to password lost page.	1
R3	The user logs in with user name and password. The user is authenticated.	1
R4	In case the login is incorrect (invalid or incorrect username and password), an invalid login error message is displayed.	1
R5	If there is a concurrent access (two login requests with same username/password), the last login request is denied and a concurrent access error message is displayed.	1
R8	If the user clicks on 'Add New Project Information', the user is displayed a form that requests data for the demographics of the project.	1

R9	The user has to fill in the demographic information upon which time he/she has a choice of filling seven metrics in any order. The user can fill all the data for a form, submit it and then log off. They then can return (log in after any amount of time) and complete other remaining metric forms according to their convenience. However all data for a particular metric form has to be filed in before logging off (else data is lost).	1
R10	Once the demographics are submitted to the database, the user gets a 'metrics page' that displays the seven metrics parameters – 'Schedule and Progress', 'Resource and Cost', ' Product Quality', 'Process Performance', 'Technology Effectiveness', and 'Customer Satisfaction'	1
R12	If, while entering metric data, the user clicks on submit, the data is validated as per the business logic provided, stored in the database (if there are no errors) and user is taken back to metrics page. In case any data didn't comply with the validations an error message (with specific details of the error) is displayed asking user to rectify those errors and then resubmit.	1
R13	If, while entering metric data, the user clicks on the button "Reset", all data values entered by the user is lost and the page is refreshed.	1
R15	When the user clicks on a project, the user can alter values entered for any of the metrics listed.	1
R16	If the user clicks on 'Delete Project Information', the user is asked for confirmation. Upon confirmation the record is deleted.	2
R17	Only after the users have contributed one complete set of Project Metrics, they are allowed to view the statistical analysis (benchmark results)	1
R18	The user can click on 'View Reports' page. The user is given a variety of options on the report. In the background, the system checks to make sure user has completed at least one set of project metric data (else users are asked to complete one set of project metrics) and that the user has at least one report credit.	3
R19	In case there is no activity from a browser for 30 minutes, the session is timed out and the user has to log in again.	1
R20	In case the browser is accidentally or deliberately shut or there is a browser error, the session is timed out 30 minutes after the last transaction.	2
R22	If a user forgets his/her password, a 'forgot password' page will be displayed. The user will enter their username and email address. The user will be emailed a new password that will expire after 2 days, if the user does not log in. Immediately upon logging in, the user will be forced to change their password.	1
R23	A user account will be locked after 3 unsuccessful login attempts. The user will need to contact PMBA staff to have their account unlocked.	1
R21	If the user does not have an account, he/she clicks on "register User"	2
R61	After successfully logging in, a user is presented with options that	1

	depend on the user's role level. Available options include: 'modify memberships', 'modify user information', 'modify project information', 'view reports', and 'purchase reports'. Please see section 4.1.2 of this document for a further description of user roles.	
R62	If a user clicks on 'modify user information', they will be able to change their password.	1
R63	If a membership administrator user clicks on 'modify memberships', they will be able to add users to a membership, change their user levels, and delete users from a membership.	1
R64	If a user clicks on 'modify project information', they will be able to add, modify or delete projects. They will have the ability to modify project metadata as well as project metrics. They will also be able to view all existing projects and data associated with that user.	1
R65	If a user clicks on 'view reports', they will be able to view selected benchmark reports only if they have available report credits.	1
R66	If a user clicks on 'purchase reports', they will be able to enter the payment system to purchase report credits.	1
R67	If in the 'add project data' or 'modify project data' sections, the user clicks on one of the seven metrics, questions related to the chosen metric are retrieved and displayed to the user so that he/she can take the survey. At the end of the page, the user is given a choice of submitting or resetting the data provided by the user.	1
SWWR71	The system shall provide users who have registered with the system but have not been approved by the system administrator with the ability to check his/her registration status..	1

2.5.1.2 System Administrator Requirements

Table 2 System Administrator Requirements

	Requirement	Priority
R72	Disabling user access to system by removing their login privileges	2
R74	Reminding users after 15 days of inactivity to complete their metrics	4
R75	Backing up the data in server everyday	1
R76	Activating membership administrator account after verification.	1
SWWR77	The system shall provide the system administrator with the ability to move validated metrics data from the SPBS staging database to the SPBS database.	1
SWWR78	The system shall provide authorized graduate students with the ability to query the metrics database. These students must have previously signed non-disclosure agreements in order to ensure the confidentiality of the database.	1

2.5.1.3 System Functional Requirements

Table 3 System Functional Requirements

	Requirement	Priority
SR81	If the user shuts the browser before the commit, the session has to be timed out	1
SR82	Generate log files every 30 days	3
SR83	Delete files older than 3 months	3
SR84	Must be compatible with IE	1
SR85	Must be compatible with Netscape Navigator	2
SR86	Must be compatible with Mosaic	5
SWWSR86	The system must ensure that all user-specific data sent to and received from the system remains secure.	5

2.5.1.4 Critical Non Functional Requirements

Table 4 Non-Functional Requirements

	Requirement	Priority
NR1	The site has to support a minimum of 1000 hits per day	2
NR2	The site has to support a minimum of 100 concurrent users.	2
NR3	The transactions have to be secure since it hosts sensitive classified information.	2
NR4	The database has to support at least 15 GB of data.	3
NR6	The site has to be reliable	1
NR7	User response time should be 30 seconds or better	1
NR8	Application should be platform independent for the user	1
NR9	The system has to be available 23 * 7	1

2.5.1.5 Future Enhancements

Table 5 Future Requirements

	Requirement	Priority
FR1	Add new modules	2
FR2	Add in new languages	4
FR3	Add more reports	3
FR4	Have dynamic reports	4

2.5.2 User Profiles (or roles) and Features/Benefits

This section defines the profiles and beneficial features to the anticipated users of the system. Before examining each profile in detail, however, we provide a brief, high-level overview of the process by which users characterized by these profiles are created.

2.5.2.1 Overview

Everyone who visits the SPBS Site without logging into the system is considered a Visitor. An individual who is interested in registering for a membership with the SPBS can do so via a set of on-line forms.

Once an individual has submitted his or her information to the SPBS, that individual is immediately considered a Registered User. This profile provides the individual with access to a very limited subset of the SPBS. For example, a Registered User will be able to do little more than change his or her password and monitor the status of the requested membership registration.

During this time, the Site Administrator, who receives the information provided by the Registered User, verifies that information by contacting the organization. For example, the Site Administrator shall verify that the individual works for the specified organization and that the individual is who he or she claims to be.

Once the Site Administrator verifies this information, he or she may create a membership with the SPBS on behalf of the Registered User. In this case, the Registered User becomes the Membership Administrator for the organization's membership. The Membership Administrator, among other things defined below, is responsible for managing all user accounts. The Membership Administrator can, for example, create and delete accounts associated with the membership and can also assign different roles to each account.

The process described above is briefly summarized in Figure 2.

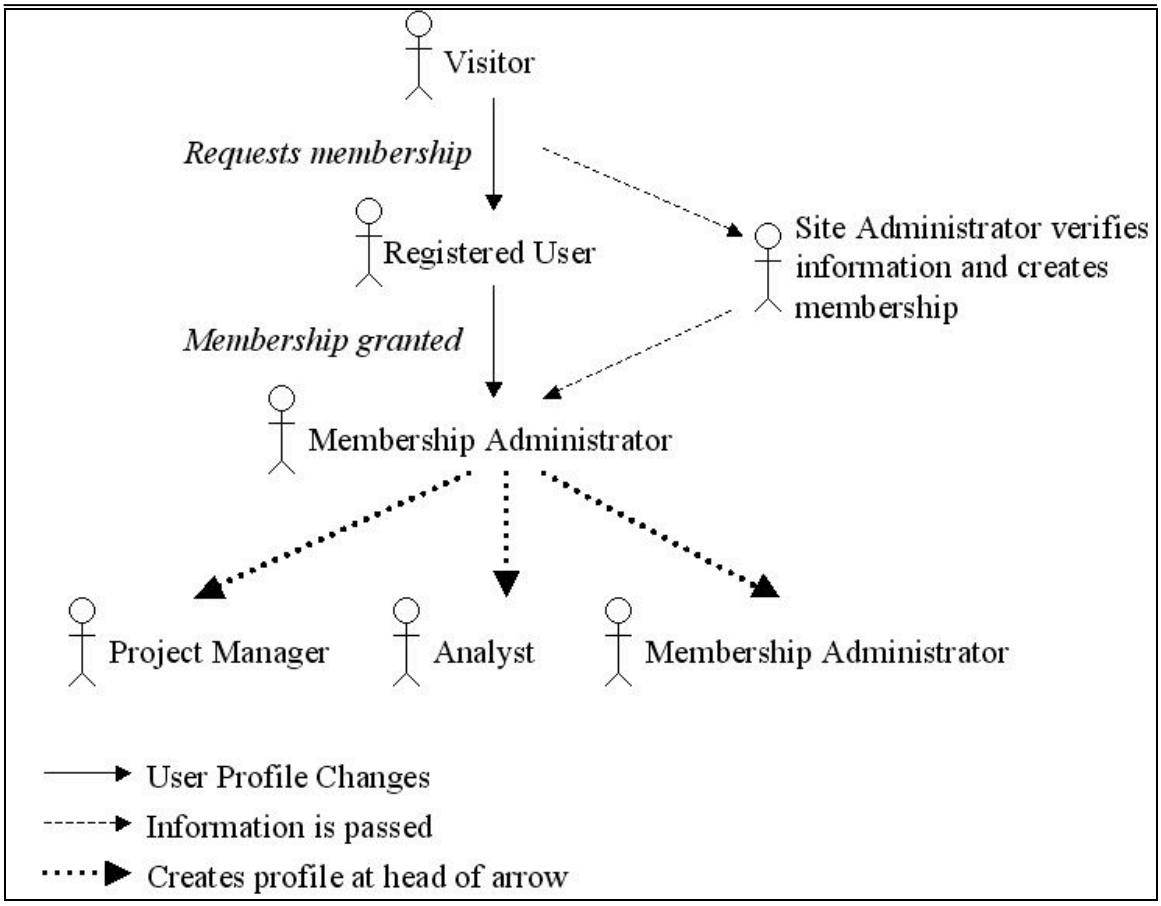


Figure 2. Profile Overview

Not shown in this figure is the fact that graduate students may also query the database for research purposes. The customer has not specified any requirements with respect to a user interface for graduate students. As a result, it is expected that these students will query the database directly. Moreover, these students must, sign a non-disclosure agreement to ensure that the information in the database remains confidential.

2.5.2.2 Roles and Associated Benefits

The system roles and the role-specific beneficial features of the system are described in this section. While there are some exceptions, benefits provided by the system are hierarchical in nature with respect to the system's roles. For example, a Registered User is characterized by the same benefits as a Visitor plus a Registered User may log into the system to check his or her registration status. This is illustrated in Table 6, which provides a summary of the benefits provided to each role. The subsections that follow provide more detail on these benefits.

Table 6. Mapping of Role-Specific System Features to Roles

Feature/ Benefit	*Visitor*	*Registered User*	*Analyst*	*Project Manager*	*Membership Administrator*	*Site Administrator*	*Graduate Student*
Access Site	**X**						

Feature/ Benefit	*Visitor*	*Registered User*	*Analyst*	*Project Manager*	*Membership Administrator*	*Site Administrator*	*Graduate Student*
Register	X						
Log in		X	X	X	X	X	
Check Status of Registration		X				X	
Review company metrics			X	X	X	X	
Request benchmark report			X	X	X		
Input metrics				X	X	X	
Create/modify projects				X	X	X	
Update membership profile					X	X	
Manage users for membership					X	X	
Authorize payment					X		
Approve membership applications						X	
Validate metrics data						X	
Move data to DB						X	
Access DB for research							X

2.5.2.2.1 Visitors

A Visitor is an individual who has not logged into the system. Visitors browse the SPBS Web site for informational purposes. Specifically, Visitors may use the system to:

V1 Access the Site to learn more about the product as well as the organization maintaining the product. For example, Visitors can take a virtual tour of the Web Site.

V2 Register for an account.

2.5.2.2.2 Registered Users

A Registered User is an individual who has provided company registration information and is in the process of verification and validation by the Site Administrator. Once the user and the company credentials are verified, the registered user is provided with Membership Administrator privileges. Thus, Registered Users may use the system to:

RU1 Log into and out of the system.
RU2 Check the status of previously submitted registrations.

2.5.2.2.3 Analysts

An Analyst is an individual who has access to company data and can request benchmark reports but cannot input metrics data. Thus, Analysts may use the system to:

A1 Review company metrics data for projects.
A2 Request custom benchmark reports.

2.5.2.2.4 Project Managers

For the purposes of this project, a Project Manager is an Analyst who can also input metrics data for a given project. Thus, Project Managers may use the system to:

PM1 Input metrics data for a given project.
PM2 Create/modify/delete projects.

2.5.2.2.5 Membership Administrators

A Membership Administrator is an individual, who provides information on behalf of his or her organization, is responsible for creating and managing user accounts within the membership and finally for adding credits to the account. Thus, Membership Administrators may use the system to:

MA1 Specify/modify his/her membership profile (e.g., company information).
MA2 Create and manage user accounts associated with the corresponding membership.
MA3 Authorize payment by adding credits to his/her membership account.

2.5.2.2.6 Site Administrators

The Site Administrator may view, verify and validate data submitted by various companies. The Site Administrator is also responsible for maintaining (modifying/upgrading) the Site. Site Administrators may use the system to:

SA1 Approve new memberships.
SA2 View, verify, and validate data submitted by companies.
SA3 Move a set of metrics data from the SPBS Staging Area to the SPBS database, referenced in Figure 1.

2.5.2.2.7 Graduate Students

Graduate students are not provided with a separate user interface but may directly query the database for research purposes. Thus, the major feature provided to graduate students is the ability to:

GS1 Query the database for research data.

2.5.2.3 Role-Independent Benefits

This section enumerates the beneficial features of the system that do not specifically correspond to at least one user role. Currently, this list consists of only one item, security. Security is absolutely critical to the success of this system. For example, if organizations believe that their data will not remain confidential, those organizations will not use the system.

RI1 Security. The system shall remain secure from all known threats. For this reason, a threat model[6] has been developed and the resultant countermeasures shall be implemented by the system.

2.5.3 Feature Traceability Matrix

The following table maps the features of the system back to the functional requirements:

Table 7. Mapping of Features to Requirements

	Features															
Requirements	V1	V2	RU1	RU2	A1	A2	PM1	PM2	MA1	MA2	MA3	SA1	SA2	SA3	GS1	RI1
R1	X															
R3-R5			X													
R8								X								
R9							X		X							
R10-R13, R67							X									
R15					X											
R16, R64								X								
R17-R18, R65						X										
R19, R20, R22, R23			X													
R21		X														
R61-R62			X													
R63										X						
R66						X					X					
R72, R76												X				
R74													X			

[6] For a description of the threat model, refer to the SPBS architecture document, a separate document held by Dr. Shawn A. Bohner, Ph.D.

	Features															
Requirements	V1	V2	RU1	RU2	A1	A2	PM1	PM2	MA1	MA2	MA3	SA1	SA2	SA3	GS1	RI1
SR84-86	X															
SWWR71				X												
SWWR77														X		
SWWR78															X	
SWWSR86																X

The following requirements were omitted from the above matrix for the following reasons:

- R75 – Backing up the system is an activity that the Site Administrator must do but is not an actual requirement of the system.
- SR81, SR82 – These requirements do not map directly to one of the features of the system. This functionality is part of the system infrastructure, however and will be implemented during the initial phase of system development (discussed in Section 3.4.1).
- SR83 – It is unclear what the original authors meant by this requirement. Were this an actual project, our group would work with the customer in order to clarify this requirement. Indiscriminately deleting files is dangerous, and our group decided to ignore this requirement for the purposes of this project.

2.5.4 High-Level Architecture

The high-level architecture of the SPBS will be discussed briefly in this section. The lower level details of the architecture will be presented only in those areas necessary for this project plan. A more detailed architecture is presented in [17], but this document does not map the architecture to the technology selected for implementation. This section's focus is, therefore, on this mapping.

First, however, a high level introduction to the SPBS architecture is provided. The system utilizes an N-Tiered Client-Server architectural style, as shown in Figure 3. SPBS Overview.

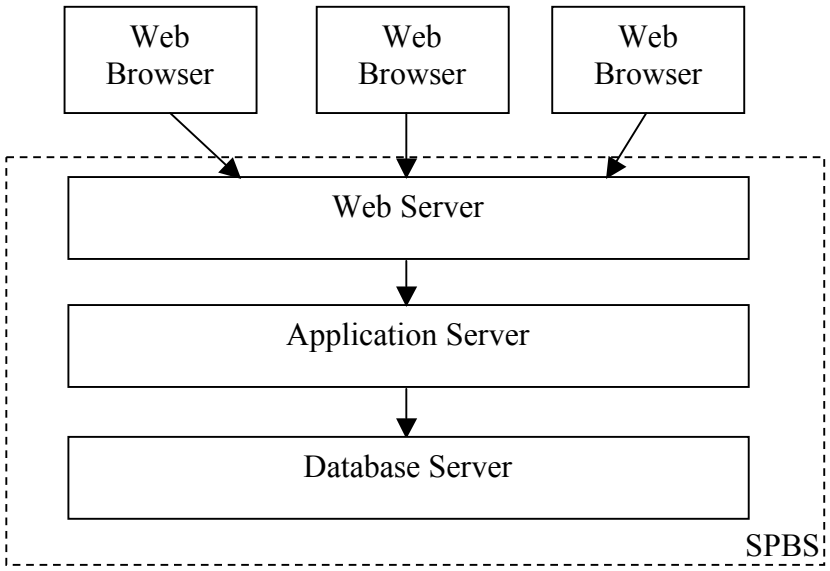


Figure 3. SPBS Overview

As can be seen in the figure above, the SPBS includes three tiers and client browsers, which are not part of the SPBS delivered by **SIMTRIX WEB WEAVERS**, represent a fourth tier. These tiers represent layers in the architecture. Strict layering is enforced such that communication can only occur with adjacent layers. That is, it is not permitted for the Web Server to communicate directly with the Database Server.

Most of the code that needs to be developed will execute within the Application Server. Internally, the architecture at this level is also a layered architecture. This layering is seen in Figure 4. SPBS High Level Architecture

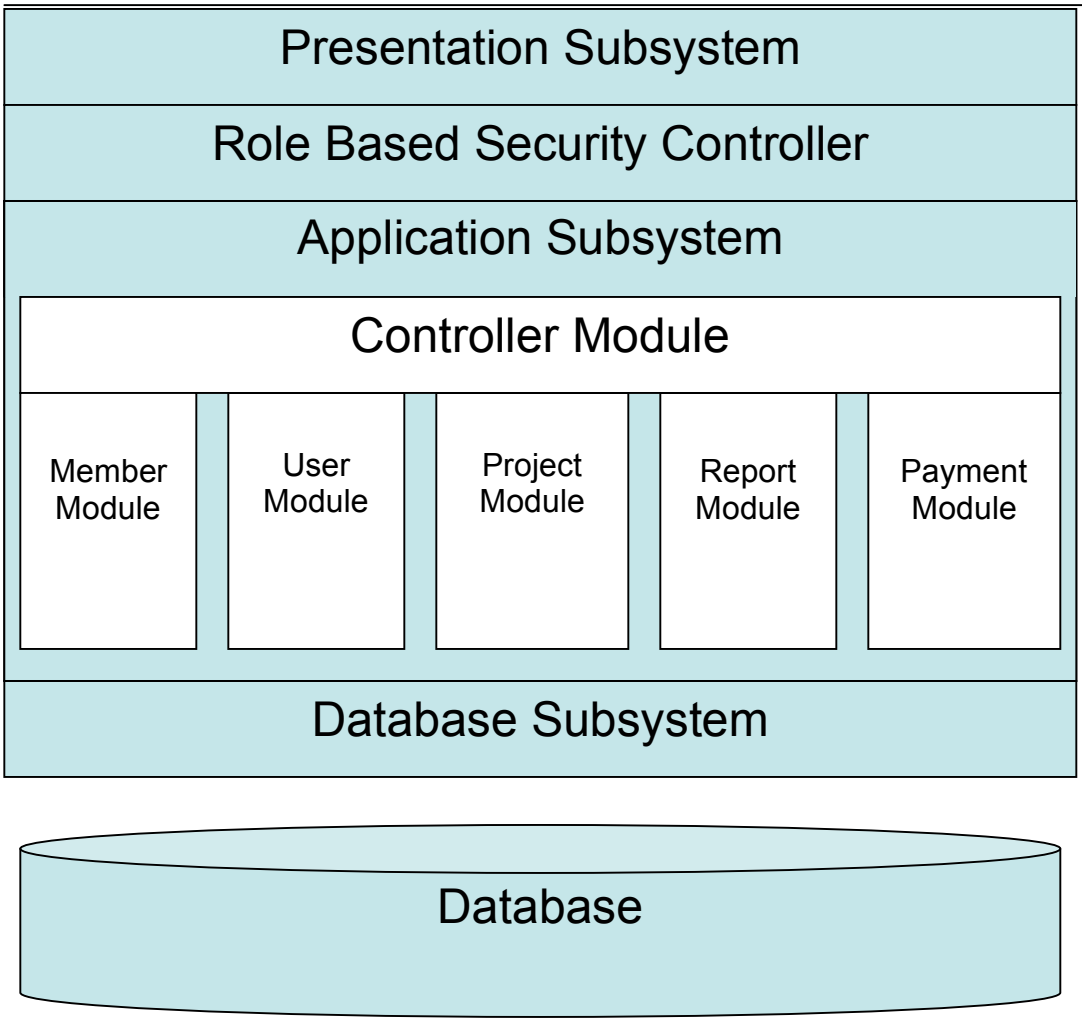


Figure 4. SPBS High Level Architecture

This architecture is an example of the Model-View-Controller (MVC) pattern. In Figure 4, the View is the top layer (Presentation) with the Application Subsystem containing the Controller (the Controller Module) and part of the Model (the other Modules) as a second layer. Finally, a Database Subsystem provides another layer that contains the rest of the Model.

2.5.4.1 Mapping to J2EE

It has been determined that the project shall utilize the Java2 Enterprise Edition (J2EE)[7] application framework as its implementation platform. This decision is very relevant from a project management standpoint as it dictates the choice of language (primarily Java) and specifies tasks that must occur (configuration of deployment descriptors, for example). It also determines the choices of technology available. This section briefly discusses how the diagram presented above is mapped to J2EE components.

[7] http://java.sun.com/j2ee/index.jsp

The Presentation Subsystem, or View in the MVC pattern, consists primarily of Java Server Pages (JSP) and HTML pages in J2EE. These pages are linked together via a Controller, which is typically a Servlet. In this case, Struts[8] has been selected as a framework to implement the MVC pattern for this project as it is integrated well with J2EE and provides an infrastructure for quickly implementing MVC.

The remaining Modules will be written as Enterprise Java Beans. For scalability reasons and ease of implementation, these will be written as Stateless Session Beans. Additionally, this simplifies portability between application servers.[9]

2.5.4.2 Java Page Flows

The following sections provide a graphical view of how a user will navigate through the various sites that compose the SPBS. Java Page Flow (JPF) was used to present this information to the user. JPF is a graphical layout of Struts provided by BEA WebLogic Workshop. The JPF file is a graphical model of the Struts controller and its Action classes. Action classes are integrated into it and annotations in the code link actions to other actions or pages. More information about JPF can be found at http://dev2dev.bea.com/products/wlworkshop81/index.jsp. A further discussion of the use of JPF and WebLogic Workshop is provided in Section 3.4.2.7. The JPFs below were derived from a combination of the requirements and a prototype site.

The main site was large enough that it was broken into two pieces – the main site itself and then a sub-site that handles the entering of metrics. This division was done because the size of the main site made it unwieldy as a single Java Page Flow. Thus, a Nested Page Flow was used to split out the set of pages responsible for capturing metrics about a project.

2.5.4.2.1 Main Site

The Main Site is the site used by Visitors, Registered Users, Analysts, and Project Managers. It contains pages for login/logout, registration, viewing of projects, and the requesting of reports. It also contains a sub-site to add new projects, which is discussed in the next section.

[8] http://struts.apache.org/
[9] Additional details about these choices should be presented in an architecture document rather than here. It is assumed that they would be in a real situation.

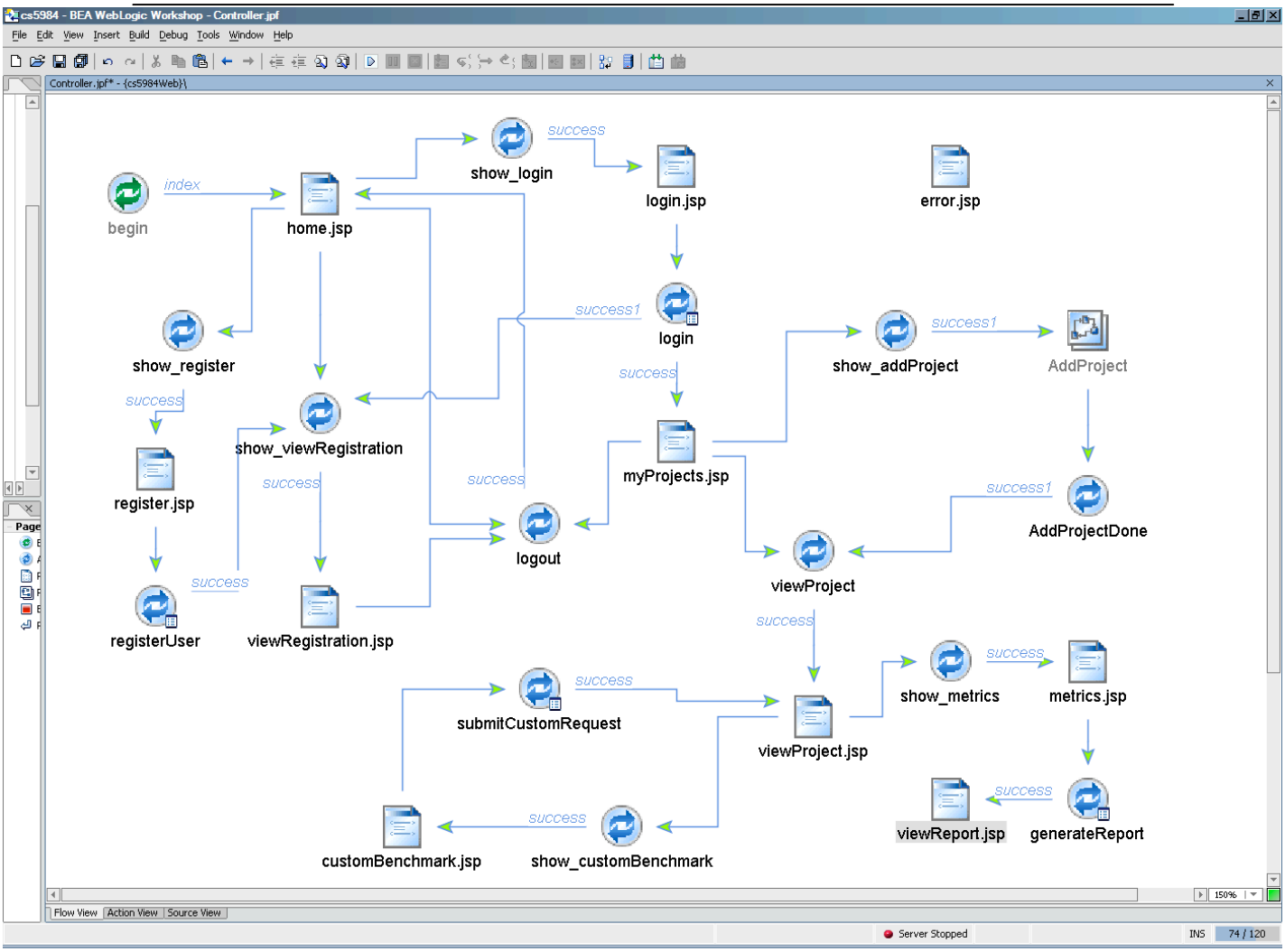


Figure 5. JPF for Main Site

2.5.4.2.2 Add Project Sub-Site

The Add Project Sub-Site is used by Project Managers to input project metrics. It consists of a large number of data entry screens that are accessed sequentially.

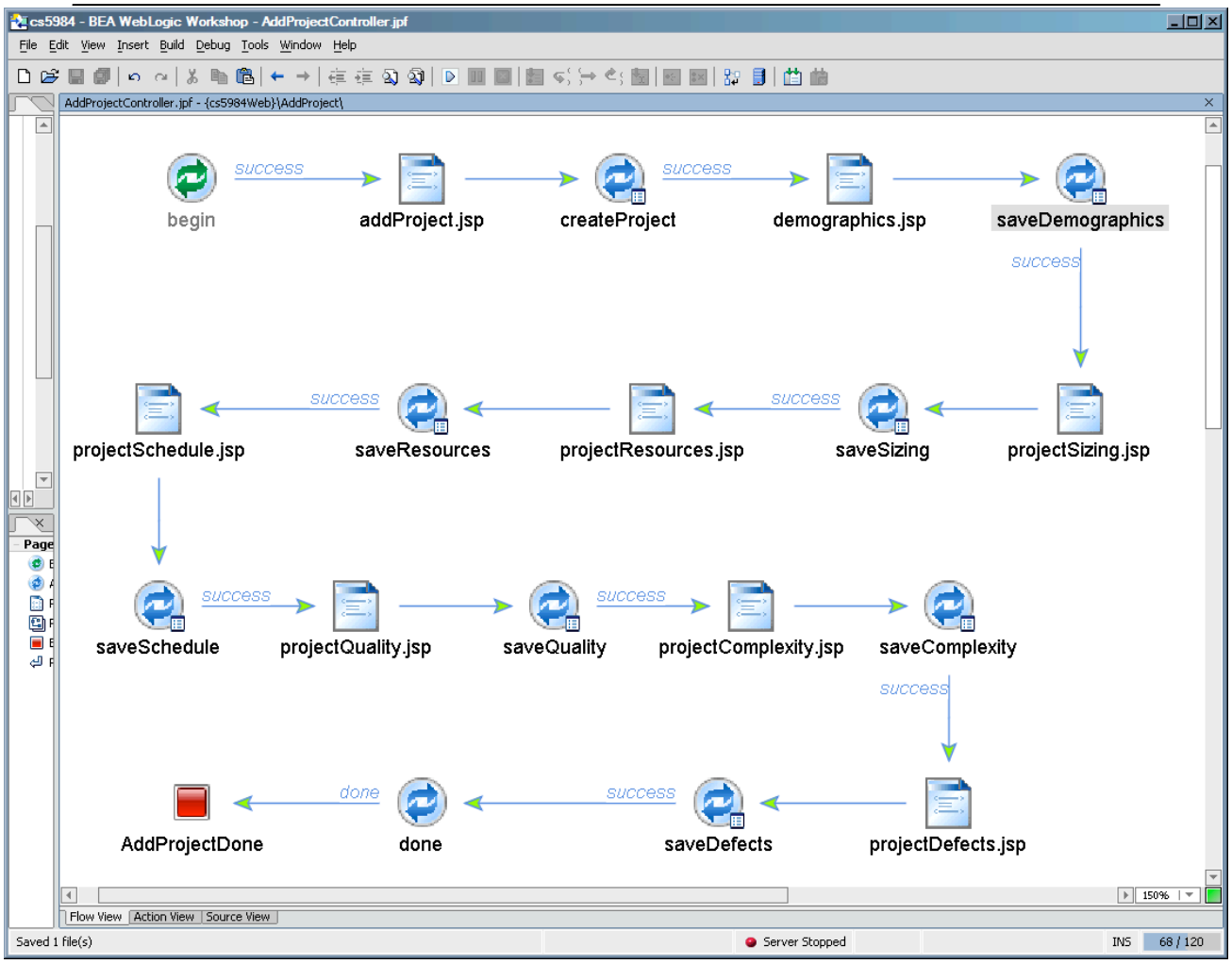
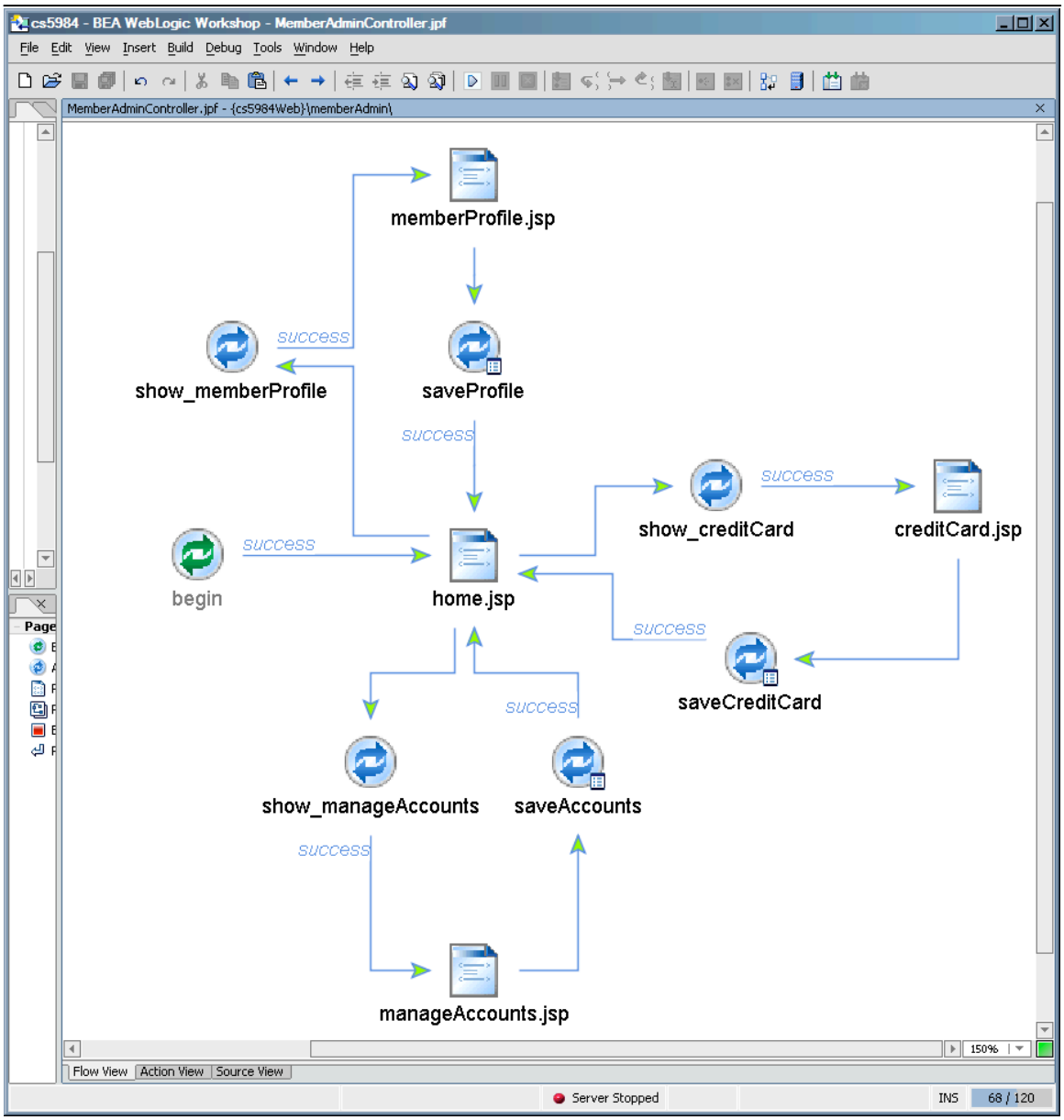


Figure 6. JPF for Add Project Sub-Site

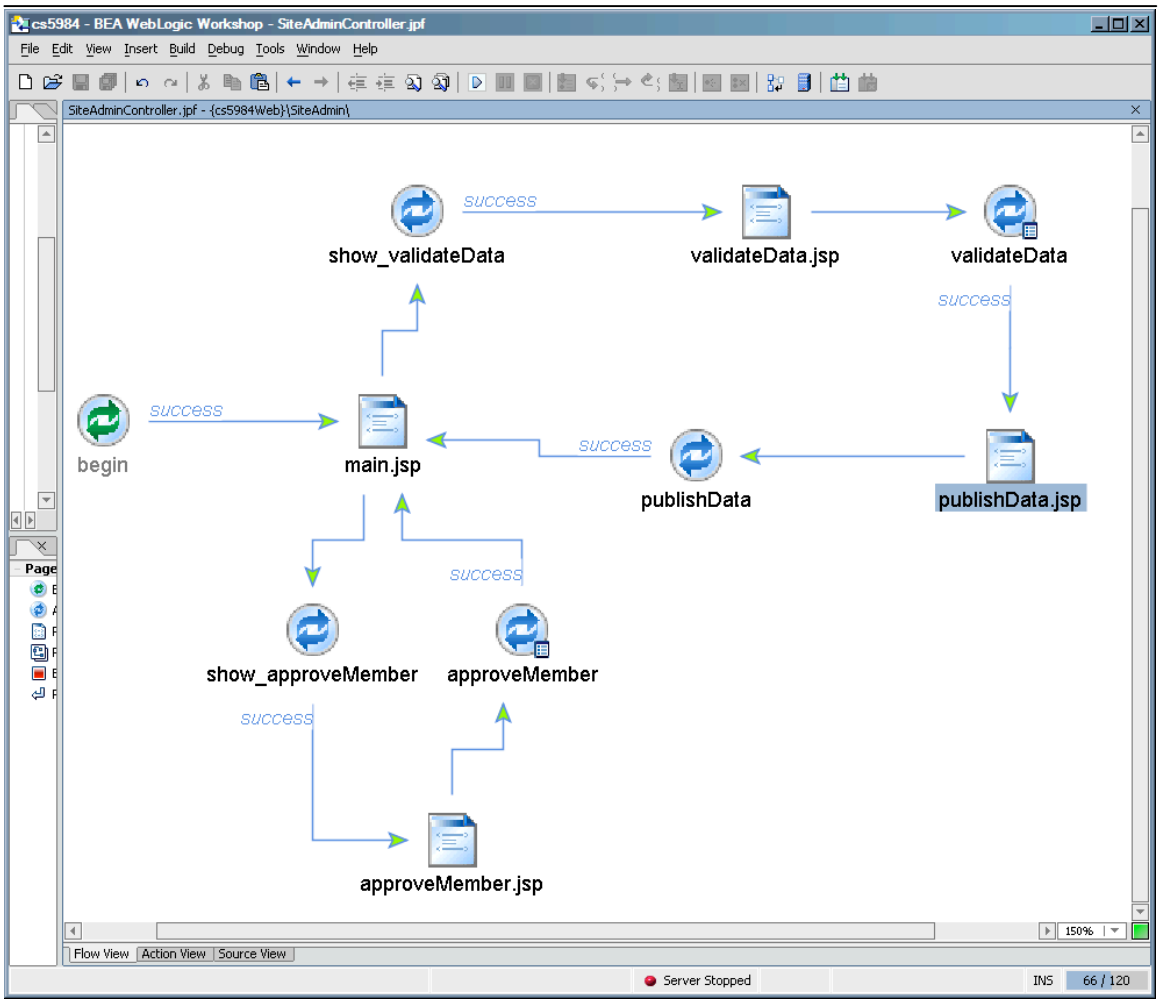
2.5.4.2.3 Membership Administrator Site

The Membership Administrator Site is accessed, obviously, by Membership Administrators. It is used only to make Registered Users into Project Managers and Analysts and to specify credit card information for payment for reports.



2.5.4.2.4 Site Administrator Site

The Site Administrator Site is used by Site Administrators to validate data entered by Project Managers and allow it to be combined with data about other projects for multi-project metrics and reports. It is also the site used by Site Administrators to designate Registered Users as Membership Administrators.



2.6 System Context

The system will be used by software organizations to compare their projects against projects at other organizations. These comparisons will provide insight to their products and processes relative to other organizations, thereby providing a focus (or prioritization) for areas of improvement. For example, if the number of post-delivery defects for an organization's product(s) was extraordinarily high with respect to other organizations, management might concentrate on instituting procedures to reduce the number of post-delivery defects.

Additionally, the system database will be used by the company for research purposes. While the metrics data and organizational characteristics (e.g., size and domain of the organization) may be used, the originating organization(s) will remain anonymous unless written permission is provided by that organization(s). It is for this reason that the system's database will be maintained in a secure room, to which only personnel who have signed non-disclosure agreements have access.

Finally, the system will reside at a secure location to be determined by company personnel. The system will, however, reside in a location that permits the use of the company's existing, secure network infrastructure. This includes the use of the company's connection to the Internet, firewall(s), and demilitarized zone. The proposed intrusion detection system (IDS) is an option that may be exercised by the company, but this is considered a separate effort and has not been included in the estimates in this document.

2.7 Major Constraints

Major project constraints are discussed in detail in the following sections.

2.7.1 Business Constraints

2.7.1.1 Budget

The customer has specified a budget of no more than $250,000.

2.7.1.2 Staffing

SIMTRIX WEB WEAVERS will use a five-member software development team to build the system.

2.7.1.3 Delivery Date

The Customer would like to deploy the System within one year of the start of the project. Thus, this system must be delivered by the close of business, December 1, 2005.

2.7.2 Technical Constraints

Users must have an Internet browser to access the SBPS. Given the limited development budget, the SPBS cannot be built to accommodate all browsers. Instead, the SPBS will support the most widely used browsers: Netscape 7.0 and later; Internet Explorer 6.0 or later; and Mozilla 1.5 or later.

2.7.3 Performance Constraints

2.7.3.1 Availability

The Customer has specified that the System must be available 99% of the time. Internet connectivity, for which the customer is responsible, shall not be considered when determining the availability of the System. The Customer understands that Internet connectivity is not considered part of the System.

2.7.3.2 Response Time

System response time is significant in retaining a user's interest in a Web site. Thus, it is important to minimize the response time to a user's query. There are two measures of performance for the system: static page response and dynamic page response. Each of these performance criteria is explained below.

2.7.3.2.1 Static Pages

Pages that contain only static content shall load in no more than 8 seconds over a 56kb modem.

2.7.3.2.2 Dynamic Pages

Pages that contain dynamic content shall load in no more than 30 seconds over a 56kb modem Dynamic content is content that is not encapsulated in raw text (or HTML), but rather pulled from some dynamic source such as a database, file, Web service, etc.

3 Project Estimates

3.1 Considerations and Initial Assumptions

3.1.1 General Assumptions

We have made the following general assumptions:

- Complex, plentiful graphics are not required by the SPBS, and as a result, very few multi-media files will need to be created by our team. It is for this reason that we will use a relatively junior UI developer.
- Based on the assignment statement, we are assuming that we are in Post Architecture phase. This means that an architecture has been developed, which is part of the initial phase of development. As a result, the project schedule starts in early November to account for the work already done.
- It is assumed that the company has sufficient firewalls, routers, and networking capacity to support the addition of the SPBS without the purchase of any additional hardware (with the exception of the machines required to host the SPBS itself).
- It is assumed that the company has a public-key infrastructure in place and that the company will provide the necessary SSL certificates for the server. Therefore, the cost of obtaining such a certificate or creating a public-key infrastructure need not be considered in this estimate.

3.1.2 Use of the Model-Driven Architecture

3.1.2.1 OptimalJ

In an attempt to increase productivity, the development team will use a tool, OptimalJ,[10] which has not been previously utilized by our organization. OptimalJ fully implements the Object Management Group's Model-Driven Architecture,[9] thereby accelerating "J2EE development by generating working applications directly from visual models. Through the power of patterns and model-driven application design, OptimalJ decreases the need for extensive coding and design skills, and delivers high productivity and consistency."[10] An independent case study cites a 35% reduction in the overall development effort when building a J2EE application using OptimalJ.[2] Note that this study assumed that both teams created relatively detailed design artifacts, which isn't always the case. Because we will be delivering a comprehensive set of documentation to

the company, however, we will need to develop detailed design artifacts. For this reason, this case study applies to our effort.

We chose OptimalJ due to its success and reputation in industry. At least 51 companies are actively using this tool in a wide range of domains, from financial services to manufacturing. Examples include Washington Mutual Bank, Ford Motor Company, General Electric, and Raytheon.[11] Seven independent analysts have submitted very positive reviews,[12] and members of our organization personally know people who have used this tool and have experienced significant productivity improvements. We take the use of this tool into account in our estimates and as a result, reflect this as a risk item that needs to be monitored and managed (Section 4).

We have made the following assumption with regard to this tool:

- The development team will experience a 20% decrease in development effort. The case study performed using OptimalJ cited a 35% decrease in overall development effort.[2] This decrease in development effort included the learning curve associated with the tool and was similar in nature to this project, as previously described. Assuming that our team will experience a similar increase in productivity is a risk. In order to mitigate this risk, however, rather than assume a 35% decrease in development effort, we have assumed that we will experience a 20% decrease in development effort.

3.1.2.2 BEA WebLogic Workshop

OptimalJ is primarily focused on the development of business logic components such as Enterprise Java Beans (EJBs) and Web Services. As such, it does not address rapid development of Web pages. This, however, is an area where BEA WebLogic Workshop can provide assistance. Workshop will only be used for the authoring of Java Server Pages, HTML, and Java Page Flows (JPF). The advantage of limiting the usage of Workshop to these areas is that it generates code that can be easily moved to other application servers. Experience of the team has found that development of Web pages can be reduced by at least 20%.[10] While higher reductions have been seen by the company, like with the use of OptimalJ, a smaller than expected reduction has been selected to mitigate the risk of using this tool.

While Workshop is not touted as a Model Driven Development tool like OptimalJ, it has many of the same characteristics. The portions that will be used within this project include an annotated model for developing Struts applications known as Java Page Flow (JPF). In JPF, the developer graphically lays out how a site will be navigated and specifies Struts actions that separate pages from one another. Workshop then automatically generates appropriate annotations for this graphical model of the site and

[10] Personal experience by Bart Simpson has found that the generation of Web pages can be reduced by as much as 80% for simple sites and by at least 20% even for complex sites due mainly to the generation of Struts configuration files, the integrated testing within WebLogic Workshop, and the visual modeling of the tool.

will generate the necessary Java class files and Struts configuration files for the site to be deployed and accessed. Workshop fully supports two-way development, allowing the annotations to be modified and having this automatically reflected in the graphical model of the site. Most importantly, Workshop supports Struts interoperability so applications built using Workshop can be run in another application server that supports Struts.[11]

3.1.3 Use of Web Model (WEBMO)

In an attempt to increase the accuracy of our estimates, our team has opted to employ the Web Model (WEBMO) estimation technique, an estimation technique that is specifically tailored to Web-centric projects.[3] [4] [5] To this end, we have made the following assumption:

- The development team has sufficient experience with WEBMO in order to use it for estimation of Web projects utilizing the J2EE architecture.
- The site is understood in sufficient detail to allow for estimation with WEBMO, which requires a very detailed, low-level understanding of the required components.

3.2 Software Development Lifecycle

In order to help ensure that the SPBS meets the customer's needs, our team has selected an iterative, incremental, collaborative, agile-like development lifecycle that is, at its core, an instantiation of the WinWin Spiral Model defined by Boehm et al. [1] We do, however, expect some initial customer communication, planning and risk analysis to precede these iterations.

After an initial startup and planning phase, the software will be constructed in three iterations. At the beginning of each iteration, all stakeholders (i.e., the company and SIMTRIX WEB WEAVERS personnel) will meet to discuss the status of the project. The stakeholders will also discuss the goals and requirements for the current iteration. The product of this meeting will be a unified vision regarding the precise functionality that is to be built during the current iteration.

Next, SIMTRIX WEB WEAVERS personnel will build the functionality into the executable baseline. This step will not be done in isolation, as a customer representative will physically visit our site in order to assess progress and provide feedback and direction. Development of the tiers in the architecture will proceed in parallel and developers will unit test the components they build. Integration testing will commence using a bottom-up approach [p. 490 of [13]] in order to minimize scrap.

Finally, each iteration will conclude with a meeting of the key stakeholders in order to demonstrate and evaluate the executable baseline. Issues raised by the customer will be addressed as quickly as possible but may not be resolved until the next iteration. This

[11] See http://e-docs.bea.com/workshop/docs81/doc/en/core/index.html for more details.

will ensure that the product meets the needs of the customer at the end of each iteration, thereby reducing risk and putting the customer at ease.

Section 2.5.2 enumerated all features of the system. During each iteration, an incremental subset of the system's features will be built into the executable baseline, as defined in Table 8.

Table 8. Mapping of Functionality to Development Iteration

Iteration 1	*Iteration 2*	*Iteration 3*
V1 - Access the Site	SA1 - Approve membership applications	SA2 - Validate metrics data
V2 – Register with the system	MA1 - Update membership profile	SA3 - Move data to the DB
RU1 - Log In	MA2 - Manage users for membership	A1 - Review own company's metrics
RU2 - Check the status of registration	MA3 - Authorize payment	A2 - Request benchmark report for company
	PM2 - Create/modify projects	GS1 - Access the DB for research
	PM1 - Input metrics	
RI1 - Security	RI1 – Security	RI1 - Security

During the first iteration, all hardware and COTS components required by the system shall be obtained, configured, and integrated into the system's infrastructure. Additionally, the informational pages[12] shall be developed as well as the capability to register with the system, log into the system, and check the status of one's registration. With the exception of displaying informational pages, all other functionality must be performed in a secure manner. For this reason, the countermeasures required for security of these operations will be developed during this iteration.

During the second iteration, membership capabilities shall be developed. This includes the site administrator's ability to approve membership applications as well as the membership administrator's ability to update his/her company's profile, manage users associated with the membership, and authorize payment for reports. In addition, the abilities to create and modify projects and input metrics associated with those projects shall be developed. Finally, the countermeasures required for security of these operations will be developed during this iteration.

[12] In this context, "informational pages" refers to all pages that provide general information on the product as well as the customer. This includes items such as the initial home (or "welcome) page, the contact us page, etc.

During the third and final iteration, metrics functionality shall be developed. This includes the abilities of analysts to review their organization's project metrics and to request a benchmark report.[13] This also includes the site administrator's abilities to validate metrics data in the staging area and to move this data to the formal SPBS database. In addition, this includes any work that needs to be done to support student queries of the database. Note, however, that no requirements have yet been specified for this ability. As a result, while it has been included in the iteration, this ability represents no additional work. Lastly, the countermeasures required for security of these operations will be developed during this iteration.

Finally, we borrowed the concept of buffer management from Critical Chain Project Management (CCPM),[14] [8] and have inserted a safety buffer at the end of each phase. We plan to use these buffers to absorb any overruns and to monitor the overall health of the project, as is done in CCPM. [8] These buffers are described in greater detail in Section 3.4.1, Process-based Estimation.

3.3 Data Used for Estimates

This section contains a description of historical data used to derive estimates as well as job descriptions and associated salaries, which are used for cost estimates.

3.3.1 Historical Data

One of the two founders of SIMTRIX WEB WEAVERS has a background in software metrics and process improvement. She instated a low-overhead metrics gathering process during the company's first year. As a result, SIMTRIX WEB WEAVERS now employs a five year project metrics database for process improvement and estimation, and this database will be used in conjunction with developer experience and the system characteristics to create the estimates described in this section. The following items represent a sampling of the metrics contained in this database:

- Project size
  - High Order Language (HOL) Lines of Code (LOC)
  - Number of custom-built multi-media files
  - Number of XML files
  - Average number of LOC in the XML files
  - Number of JSP/HTML files
  - Average number of LOC in the JSP/HTML files
- Project duration
  - Number of iterations
  - Duration of each iteration
- Project cost

[13] In this context, benchmark report refers to both the canned, automated benchmark report, which utilizes all data in the database as well as the manually generated benchmark report created by team personnel.
[14] We are not fully using CCPM. Rather, we are simply borrowing the concept of the safety buffer from CCPM.

- Project team size and composition

3.3.2 Job Descriptions

Five roles will characterize the development team from SIMTRIX WEB WEAVERS. Table 9 summarizes the role, ability, and salary for each member of the team. Note that all job descriptions and salary information were obtained from http://www.salary.com. This company conducts extensive research on compensation for a wide variety of positions around the country. The data reported in this section are relevant for the Washington D.C area.

Table 9. Team Composition - Job Responsibilities, Descriptions, and Salaries

Job Title & Project Specific Responsibilities	*Job Description*	*Salary (year)*
Software Engineer III Project Responsibilities: 1) Project Lead 2) Programming	"Designs, modifies, develops, writes and implements software programming applications. Supports and/or installs software applications/operating systems. Participates in the testing process through test review and analysis, test witnessing and certification of software. Requires a bachelor's degree in a related area and 4-6 years of experience in the field or in a related area. Familiar with a variety of the field's concepts, practices, and procedures. Relies on experience and judgment to plan and accomplish goals. Performs a variety of complicated tasks. May lead and direct the work of others. May report directly to a project lead or manager. A wide degree of creativity and latitude is expected." [http://www.salary.com]	**$86,664**
Database Analyst (Database Analyst II) Project Responsibilities: 1) Database Management 2) Programming	"Reviews, evaluates, designs, implements and maintains company database[s}. Identifies data sources, constructs data decomposition diagrams, provides data flow diagrams and documents the process. Writes codes for database access, modifications, and constructions including stored procedures. May require a bachelor's degree in a related area and 2-4 years of experience in the field or in a related area. Familiar with standard concepts, practices, and procedures within a particular field. Relies on limited experience and judgment to plan and accomplish goals. Performs a variety of tasks. Works under general supervision; typically reports to a project leader or manager. A certain degree of creativity and latitude is required." [http://www.salary.com]	**$73,310**
Interface Designer Project Responsibility 1) Web page design 2) Programming	"Designs html prototypes, visual interfaces and interaction of Web-based applications. Designs and evaluates visual human interfaces utilizing user-centered design principles. Implements the user interface design. Works with the product development team to design online user experiences. Ensures user experience is formulated to achieve the goals of the online entity. May require an associate's degree with 0-2 years of experience in the field or in a related area. Has knowledge of commonly-used concepts, practices, and procedures within a particular field. Relies on instructions and pre-established guidelines to perform the functions of the job. Works under immediate supervision.	**$74,061**

Job Title & Project Specific Responsibilities	Job Description	Salary (year)
	Primary job functions do not typically require exercising independent judgment. Typically reports to a manager." [http://www.salary.com]	
Software Engineer II Project Responsibilities 1) J2EE Infrastructure Technical support 2) Software Quality Assurance (SQA) 3) Programming	"Designs, modifies, develops, writes and implements software programming applications. Supports and/or installs software applications/operating systems. Participates in the testing process through test review and analysis, test witnessing and certification of software. Requires a bachelor's degree in a related area and 2-4 years of experience in the field or in a related area. Familiar with standard concepts, practices, and procedures within a particular field. Relies on limited experience and judgment to plan and accomplish goals. Performs a variety of tasks. Works under general supervision; typically reports to a manager. A certain degree of creativity and latitude is required." [http://www.salary.com] **Project Specific: This person is expert in J2EE infrastructure and QA techniques.**	**$71,439**
Programmer II Project Responsibility 1) Programming	"Reviews, analyzes, and modifies programming systems including encoding, testing, debugging and documenting programs. May require an associate's degree in a related area and 2-5 years of experience in the field or in a related area. Familiar with standard concepts, practices, and procedures within a particular field. Relies on limited experience and judgment to plan and accomplish goals. Performs a variety of tasks. Works under general supervision; typically reports to a project leader or manager. A certain degree of creativity and latitude is required." [http://www.salary.com]	**$67,385**
Average Monthly Salary:		**$ 6,214.31**

Assumptions for each role in Table 9 follow:

- Software Engineer III – Due to the fact that this role will be filled by an experienced developer who also has project management experience, we have assumed that this individual will command the highest salary for this job title.
- Software Engineer II – Due to the fact that this role will be filled by an experienced developer who also has J2EE and SQA experience, we have assumed that this individual will command the highest salary for this job title.
- Interface Designer – Due to the fact that this project does not require much sophisticated graphical development, we have assumed that this individual will have formal training in user interface development but will have little to no experience (i.e., an entry-level person).
- Database Analyst II and Programmer II – Due to the fact that these roles will be filled by individuals with greater than average abilities, but that these roles have relatively traditional responsibilities for this project, we assumed that both the database analyst and the programmer command greater than average salaries for their job titles.

3.3.3 Burdened Labor Rate

The fully burdened labor rate for a given team is the total cost of doing business for that team. This rate includes all compensation including salaries, the cost of utilities, mortgage/rent, taxes, etc. Obviously, the burdened labor rate is, therefore, specific to each organization. In our company, we have the burdened labor rate of 50% overhead per employee for average projects. Thus, if an employee makes $10,000 per month, the burdened labor rate for that employee is $10,000 + $10,000 * .5 = $15,000 per month.

The basic monthly labor rate was calculated by averaging the monthly salaries of each team member, as shown below:

($86,664/12 + $73,310/12 + $74,061/12 + $71,439/12 + $67,385/12)/5 = $6,214.31

After applying the aforementioned 50% overhead rate, our burdened labor rate is:

$6,214.31 + $6,214.31*.5 = $9,321.47

3.4 Applied Estimation Techniques and Results

First, a **process-based technique** was employed to estimate the effort involved in developing the SPBS. This estimation is detailed in Section 3.4.1 To help verify the validity of the estimate obtained, a separate technique, the **Web Model** (WEBMO) was used. WEBMO is based on the COCOMO II parametric estimation model and is discussed in detail in Section 3.4.2. Finally, the estimations are combined to create a single estimate, which is presented in Section 3.4.3.

3.4.1 Process-based Estimation

3.4.1.1 Process Decomposition

As described in Section 3.2, our organization has selected an iterative, incremental software development life cycle. Section 3.2 also describes the functionality that is to be built into the product during each iteration. The process-based estimate described in this section follows directly from the development life cycle and the functionality relevant to each iteration. Table 10 summarizes this estimate. All estimates are in terms of staff months and assume 20 business days per month. The table breaks down the various steps in the process into the four life cycle phases defined by Royce in [18] and then further into specific tasks where appropriate. It should be noted that one deviation exists. Typically, both Design and Analysis are included in the Elaboration phase. However, due to the use of Model Driven Development, Design and Code are combined in the construction phase. Creating the models used in MDA represents a design activity, but the models generate code and are therefore more properly part of the construction phase.

Table 10. Process-based Estimation Summary

Phase →	Inception			Elaboration	Construction		Transition		TOTALS
Task →	CC	Planning	Risk Analysis	Analysis	Design	Code	Test	CE	
Function									
Startup & Initial Planning	0.50	0.50	0.25						1.250
Iteration 1	0.0625								0.063
Infrastructure Development	0.05	0.05	0.05	0.05	0.10	0.20	0.05	0	0.550
V1 - Access the Site	0.25	0.10	0	0.15	0.1	0.25	0.1	0.05	1.000
V2 – Register	0	0	0	0	0.05	0.10	0.05	0.05	0.250
RU1 - Log In	0	0	0	0	0.05	0.10	0.05	0.05	0.250
RU2 – Check reg. status	0.10	0.05	0.025	0.10	0.10	0.15	0.10	0.025	0.650
RI1 – Security	0.15	0.20	0.15	0.25	0.50	0.25	0.50	0.15	2.150
									0
Iteration 2	0.364								0.364
SA1 – Approve membership apps.	0.20	0.05	0.10	0.25	0.15	0.25	0.25	0.10	1.350
MA1 - Update membership profile	0.20	.075	0.05	0.10	0.10	0.20	0.15	0.15	1.025
MA2 – Manage users for membership	0.15	0.05	0.05	0.05	0.15	0.20	0.10	0.05	0.800
MA3 - Authorize payment	0.10	0.05	0.05	0.10	0.05	0.20	0.10	0.05	0.700
PM2 - Create/modify projects	0.15	0.05	0.05	0.05	0.15	0.20	0.10	0.05	0.800
PM1 - Input metrics	0.30	0.20	0.10	0.20	0.25	0.35	0.20	0.40	2.000
RI1 – Security	0.15	0.20	0.15	0.25	0.50	0.25	0.50	0.15	2.150
									0
Iteration 3	0.883								0.883
SA2 – Validate metrics data	0.15	0.10	0.05	0.10	0.15	0.20	0.10	0.25	1.100
SA3 - Move data to the DB	0.10	0.05	0.025	0.10	0.05	0.15	0.10	0.025	0.600
A1 - Review own company's metrics	0.25	0.10	0.05	0.20	0.10	0.20	0.15	0.30	1.350
A2 - Request benchmark report	0.35	0.20	0.10	0.30	0.30	0.25	0.20	0.30	2.000
GS1 - Access the DB for research	0.20	0	0	0	0	0	0	0	0.200
RI1 – Security	0.075	0.10	0.075	0.125	0.05	0.125	0.25	0.075	0.875
Project Safety Buffer	1.68								1.680
Totals	6.411	2.125	1.325	2.375	2.9	3.625	3.05	2.225	24.036
% effort	26.67%	8.84%	5.51%	9.88%	12.07%	15.08%	12.69%	9.26%	100.00%

CC – Customer Communication CE – Customer Evaluation
All estimates are in staff months (20 business days)

As one can observe from this table, it is estimated that the entire effort will take approximately 24.036 staff months. As is described in Section 3.1.2, however, we expect to achieve a significant increase in productivity by using the MDA tool, OptimalJ, and by

utilizing BEA WebLogic Workshop. For reasons discussed in the next section, Section 3.4.1.2, we have assumed that we will experience a 20% decrease in development effort. Based on this assumption, it is estimated that the entire effort will take approximately (24.036 * 80%) = **19.23 staff months**.

Multiplying this by the burdened labor rate of $9,321.47, results in a **total project cost of:**

19.23 months * $9,321.47 per month = **$179,251.87**

With five team members, the system should take approximately **4 months** (19.23 / 5 = **3.85**) **to develop**. This estimate is used only to level set the project plan. The actual plan will take into consideration project and resource dependencies that cannot be accounted for here.

3.4.1.2 Notes and Assumptions

This section contains notes, assumptions, and rationale for the estimate summarized by Table 10, above. This section has been divided into several subsections, one for each major category in the process-based estimate (i.e., Startup & Initial Planning and Iterations 1-3). There are, however, some notes, assumptions, and rationale that apply to more than one estimation category. These are described below:

- Borrowing from the concepts of CCPM,[15] [8] we have inserted a safety buffer at the end of each phase (or more accurately, at the beginning of each phase except for the first one). These buffers are identified by the blue shaded rows in Table 10. We plan to use these buffers to absorb any overruns and to monitor the overall health of the project, as is done in CCPM. [8] These buffers are dependent upon two characteristics, the duration and the complexity of the previous development phase (5% for simple complexity, 7.5% for medium complexity and 10% for high complexity) and are described in each of the following subsections.
- As described in Section 2, our organization will need to deliver several, detailed documentation artifacts to the customer. The effort required to produce these artifacts is included in the *Startup & Initial Planning* phase and is the primary effort in the category Customer Communication (column CC in Table 10).

3.4.1.2.1 Startup & Initial Planning

This process category covers the kickoff meeting, initial requirements and risk analysis, and project planning. It is expected that during this phase, only two representatives of SIMTRIX WEB WEAVERS will be involved and that the effort will take a little over half a month.

3.4.1.2.2 Iteration 1

Assumptions and justification for each task in this iteration follow:

[15] We are not fully using CCPM. Rather, we are simply borrowing the concept of the project buffer from CCPM.

- Iteration 1 – As described in Section 3.4.1.2, above, a safety buffer has been inserted at the beginning of each iteration to ensure adequate time to resolve any outstanding issues associated with the previous iteration/phases of development. The size of this buffer is only 5% of the previous phase of development due to its simplicity.
- Infrastructure Development – This task includes the purchase, configuration and installation of all required hardware and COTS components. It is expected that only two people will be involved in this phase and that it will take no more than a week, given past experience with this infrastructure.
- Access the Site - This task covers the development of all informational pages, as described in Section 3.2. It is expected that the customer will have relatively clearly defined requirements regarding the content of these pages and that the number of pages does not exceed 5. This being the case, a staff month should be more than adequate.
- Register and Log In – These tasks includes the functionality required to support registration with and logging into the system respectively. This functionality has been componentized by our organization. For this reason, no time has been allocated to customer communication, planning and risk and engineering analyses. Further, very little time (one week) has been allocated to the task itself and it is expected that most time will be spent integrating the functionality with the user interface. Finally, only a day has been allocated for customer evaluation because we don't envision any substantive problems arising.
- Check Registration Status - This task includes the functionality required to permit a user who has registered with the system to check his/her registration status. While this represents new functionality that must be developed, this functionality is simplistic. As a result, the overall effort required to develop this functionality, including time spent in risk analysis, is low.
- Security – This task involves the functionality required to ensure the security of the other functionality and associated data developed during this iteration. Because this is the first iteration, however, basic security mechanisms must be established, such as role-based access, SSL, anti-virus software, auditing, etc. For this reason, significant effort has been allocated to this task. In particular, half a month has been allocated to the design and test of the general and iteration-specific countermeasures associated with this phase. This was done to ensure that the design is comprehensive and that the security mechanisms are adequately verified.

3.4.1.2.3 Iteration 2

Assumptions and justification for each task in this iteration follow:

- Iteration 2 – As described in Section 3.4.1.2, above, a safety buffer has been inserted at the beginning of each iteration to ensure adequate time to resolve any outstanding issues associated with the previous iteration/phases of development. The size of this buffer is only 7.5% of the previous phase of development due to its below average (although not simple) complexity.

- Approve membership applications – This task involves the functionality required to approve a registered user by granting him/her a membership with the system. In order to accomplish this task, the Site Administrator will need to (a) be notified of new registrations, (b) obtain a list of outstanding, unapproved registration requests, and (c) approve registration requests for membership. The effort behind this task is significant and specific to this project, facilitating little reuse. For this reason, significant effort has been allocated to customer communication, engineering analysis, design, coding, and test. The number of interfaces required for this functionality is, however, small, so customer evaluation has been limited to two days.
- Update membership profile – This task involves the functionality required to specify a change(s) to the information associated with one's account. While the information that needs to be changed will be specific to this application, this kind of functionality is common, facilitating significant conceptual and some code reuse. Thus, more effort has been devoted to customer communication than the other categories. Furthermore, additional effort has been devoted to testing this functionality than is typical.
- Manage users for membership - This task involves the functionality required to add, delete and modify users associated with an organization's membership. This functionality is common to many of our products and while it has not been componentized, we expect significant reuse, this reducing the overall effort for this task.
- Authorize Payment - This task involves the functionality required to add credits to one's account (i.e., membership account). The credit payment model relatively simple, is common to many of our company's products, and has been componentized. Thus, little effort has been reserved for this task.
- Create/modify projects - This task involves the functionality required to create/delete and modify projects associated with one's account. This kind of functionality is common to many of our products and while it has not been componentized, we expect significant reuse, this reducing the overall effort for this task.
- Input metrics - This task involves the functionality required to input metrics for a given project. This kind of functionality is custom to this application and is characterized by significant user interface complexity, thus driving the relatively high estimate for this task and the significant amount of time allocated to customer communication and evaluation.
- Security – This task involves the functionality required to ensure the security of the other functionality and associated data developed during this iteration. Based on the fact that the security infrastructure was established during the previous iteration, it is expected that the effort allocated to this task will be approximately half of the previous iteration.

3.4.1.2.4 Iteration 3

Assumptions and justification for each task in this iteration follow:

- Iteration 3 – As described in Section 3.4.1.2, above, a safety buffer has been inserted at the beginning of each iteration to ensure adequate time to resolve any outstanding issues associated with the previous iteration/phases of development. The size of this buffer is 10% of the previous phase of development due to its increased complexity.
- Validate metrics data – This task involves the functionality required to validate metrics that were entered for a given project before those metrics are allowed to enter the SPBS database. The interface required for this task will be relatively simplistic (e.g., it might display each metric associated with a given project and a "validated" field, which the Site Administrator may click to validate a given metric). Furthermore, the processing underlying this interface will be relatively simple as well. While this is true, this task will be inherently user interface-driven, which justifies the relatively high allocation for coding and customer communication and evaluation.
- Move data to the DB – This task involves the functionality required to reflect that a set of metrics data is valid and to move that data to the SPBS database. The interface required for this task will be extremely simple (e.g., a single button to move the data into the database for a given set of metrics data would suffice). Furthermore, the processing underlying this interface will be relatively simple as well, which justifies the small allocation of effort for this task.
- Review own company's metrics – This task involves the functionality required to review one's project metrics (one may have more than one project). The interface required for this task will relatively complex, necessitating the navigation among projects and the display of a project's metrics data in logical, digestible chunks. While this is true, the processing underlying this interface will be relatively simple. Thus, a significant amount of effort was allocated to this task, with more than average being devoted to the customer interface steps.
- Request benchmark report - This task involves the functionality required for a SPBS customer to request an automatically or manually generated report derived from the metrics data in the database. The automatically generated report will compare each metric against the same metric for all other projects in the database while the manually generated report will be performed external to the system by COMPANY personnel. While simple in concept, the automatically generated report will require several pages to display the information in digestible chunks. Further, queries will have to be developed to interface with the database and the information returned from the database will need to be analyzed. There have been no tangible requirements for the manually generated report, however. Because of all of these factors, this task has been allocated a significant amount of effort. Moreover, because it is unclear whether the requirements for the manually generated report have been fully defined, a significant amount of effort was devoted to customer communication and evaluation and engineering analysis.
- Access the DB for research - This task involves the functionality required to allow graduate students to access the database. It is currently envisioned that this task will simply involve allowing graduate students with direct, query access to the database. As a result, no additional system functionality will need to be developed, and the estimate for most of this task is 0. We have reserved a little

time for customer communication to verify this fact. If this turns out not to be the case, it is expected that the additional cost will be absorbed by the project buffer.

- Security – This task involves the functionality required to ensure the security of the other functionality and associated data developed during this iteration. Based on the fact that the security infrastructure and most of the security countermeasures should be in place by this iteration, the effort is roughly halved again, as compared with the estimate for the previous estimation.

3.4.1.2.5 Project Safety Buffer

Due to the medium complexity of this project, a project safety buffer of 7.5% has been included in this estimate. Again, this buffer has been included to (a) absorb any overruns and (b) to provide an indication as to the health of the project as it progresses.

3.4.2 WEBMO Estimation

When estimating Web projects such as the SPBS, SIMTRIX WEB WEAVERS utilizes the WEBMO estimation model. [15] WEBMO is based on the COCOMO II parametric estimation model but accounts for additional cost drivers associated with Web development. In our experience, WEBMO has more accurately predicted the effort required for a Web project. The usage of WEBMO has been extrapolated from [15] and [16].

WEBMO uses the concept of a Web Object as a sizing metric. A Web Object is similar to a Function Point, but it specifically addresses components that make up a Web application. Like Function Points, Web Objects are classified by their complexity (Low, Average, and High) and type with each type and complexity having a different weighting factor. Once the number of Web Objects is determined, a parametric equation is applied to the sizing to obtain effort and duration. Unfortunately, WEBMO is not integrated into any tool currently, and therefore the estimation must be done by hand. This effort is encapsulated in the WEBMO.xls spreadsheet included with this project plan.

Because not all components are Web Objects, WEBMO includes the ability to estimate non-Web components via other mechanisms, such as application objects or function points. In this estimate, function points are used. Within WEBMO, this analysis is performed and then function points are converted to thousands of source lines of code (KLOC) using the appropriate conversion for the language used (Java in this case). This process is described in detail in the following sections.

3.4.2.1 Web Objects Counting

Estimation of Web Objects has two phases. First, traditional Function Point analysis is undertaken with Web application features in mind. Table 11 details the types of Function Points that are counted and provides examples for each.

Table 11 – Web Application Function Points

Function Point Type	Description	Examples
Internal logical files	logical files storing business	HTML files, Java source

Function Point Type	Description	Examples
	logic and presentation code	files
External interface files	interface files to external applications and systems	IDL, WSDL[16] and other interface definitions for external systems
External inputs	input by a user that cases some action to occur	HTML forms and links
External outputs	data provided to the user	HTML pages, e-mail
External inquiries	direct queries by the user	'search' capabilities

It should be noted that the original WEBMO definition described an external inquiry as a "logical, elementary business process that consists of a data 'trigger' followed by a retrieval of data that leaves the application boundary" and suggests browsing of data as an example. [16] Effectively, this represents an external input followed by an external output. Instead of having to struggle with whether a given interaction represents an external inquiry or an input/output combination, the definition of an external inquiry was changed from that presented in [16]. It instead focuses solely on queries executed directly by the user. Thus, a site search function or any other ad-hoc query capability provided by the system would constitute an external inquiry.

The second phase of Web Object analysis is to investigate those components which are not well represented by traditional Function Points. There are four categories that must be considered. These categories are presented below in Table 12.

Table 12 – Web Object Extensions to Function Points

Web Object Type	*Description*	*Examples*
Links	components needed to link applications together	WSRP[17], iframes, use of external interface files
Number of multimedia files	audio, video and images that must be integrated	GIFs, MPEGs, etc.
Number of scripts	components that link internal files and Web building blocks together	JavaScript, Struts configuration files
Number of Web building blocks	major Web components used to provide functionality	Struts Actions, Tag Libraries, Applets

A few comments are necessary to describe ***SIMTRIX WEB WEAVERS'*** utilization of WEBMO. The main issue is that the definitions of the various Web Object types are ill defined and difficult to decipher. Because the documentation for WEBMO was sparse, its application had to be extrapolated and then verified against previous projects. The definition of links and scripts were particularly problematic. Links were defined in [16] as "size predictors developed to take the effort required to link applications, integrate

[16] Web Services Definition Language. Specifies the location and interface definition for a Web service. See http://www.w3.org/TR/wsdl for more details.
[17] Web Services for Remote Portlets – a technology that allows a Portal to host content (a portlet) from remote locations. See http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp for more details.

them together dynamically, and bind them to the database and other applications in a persistent manner" and then later as "logical, persistent entities maintained by the Web application to find links of interest to external applications." These two definitions are clearly different. It is unclear exactly what is meant, especially since the example provided counts logical lines of HTML. SIMTRIX WEB WEAVERS have found that focusing on connectivity to external applications is the most relevant feature for Web applications built using J2EE and Struts. In some architectures, applications are allowed to directly communicate with the database. For example, many applications built using Microsoft Visual Basic communicate with the database using technologies such as ActiveX Data Object (ADO). However, in a Model-View-Controller architecture (as proscribed by the use of Struts), the effort to communicate with the database is part of the model and is therefore accounted for in the traditional function point analysis. Thus, when counting Web objects in this category, only the number of accesses to the external interface files identified in traditional function point analysis need to be considered.

A similar issue arises with Scripts. Scripts were defined in the same reference as "size predictors developed to take the effort required to link html/xml data and generate reports automatically" but then as "logical, persistent entities used by the Web application to link internal files and building blocks together in predefined patterns." It is the second definition that has been selected by SIMTRIX WEB WEAVERS, as it has been found to best represent the complexities in Web applications seen by our company.

3.4.2.1.1 Operands and Operators

The four Web Objects extensions to function points are not only just new categories. Analysis of them requires not only that components of each type be counted but also the actions which they can perform be counted. Thus, for each category, a set of components (operands) and actions (operators) must be determined. For example, a JavaScript script would be an operand and typically has a single operator (such as validate, launch window, etc). Some components can have a large number of operators. For example, an applet may allow the user to do many actions, each of which would be a separate operator. Counting both operators and operands allows the complexity differences between components to be considered at a finer grain.

3.4.2.2 Function Point Counting

Because not all components are Web components, WEBMO includes the ability to count other components via other means such as Object Points or Function Points. WEBMO allows any method to be used as long as it allows for conversion to source lines of code (SLOC). SIMTRIX WEB WEAVERS utilizes Function Points for this layer of analysis. This analysis will be performed on the application tier components (the Modules). Conveniently, Function Point analysis for non-Web components uses the same categories and complexity weights as used in the Function Point analysis that is part of the Web component analysis in WEBMO, which simplifies the process.

Function Point analysis follows the proscribed methods in the COCOMO II documentation (see http://sunset.usc.edu/research/COCOMOII/ for details of this

approach). While Function Points are more traditionally used with COCOMO II's Early Design model, they can be applied in Post Architecture. Since WEBMO already uses Function Points for the Web application, using them for non-Web components is logical as well.

3.4.2.3 Web Objects Weighting

After determining how many types of Web Objects exist and how complex each is, it is necessary to apply a weight to each to come up with a weighted number of total Web Objects. This total becomes the size estimate, which can then be used to calculate effort and duration. The following table details the weights used for each type and complexity.

Table 13. Web Object Complexity Weights

Web Object Type	Complexity		
	Low	Average	High
Internal Logical Files	7	10	15
External Interface Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquires	3	4	6
Multimedia Files	4	5	7
Web Build Blocks	3	4	6
Scripts	2	3	4
Links	3	4	6

The weightings for the extensions to function points, such as Scripts and Links, apply to both operators and operands. It is possible for an operand to have a different complexity from one or more of its operators and for operators on the same operand to have different complexities. This allows for complexity of components to be accounted for more completely.

3.4.2.4 Cost Drivers

Like COCOMO II, WEBMO has cost drivers that effect effort and duration. WEBMO has 9 cost drivers, which compares to COCOMO II's 7 cost drivers in the Early Design model. These cost drivers, their explanations, and ratings are discussed in detail in [15]. WEBMO does not differentiate between Early Design and Post Architecture phases but rather has a single model. Given the amount of low-level detail required for Web Object counting, SIMTRIX WEB WEAVERS has found that WEBMO is only useful in a Post Architecture phase. Even though it lacks the larger number of cost drivers and has no scale factors, as in COCOMO II's Post Architecture model, WEBMO has been found to adequately estimate effort for Web applications.

For each of the 9 cost drivers in WEBMO, a rating of Very Low to Very High must be selected. This selection is based on a combination of the team and the project. The selections for this project are detailed below.

- **Product Complexity (CPLX)**
  The product complexity was determined to be **Nominal** after considering the following factors:

Table 14. Factors that Determine Product Complexity

Factors	*Level*
Control Operations	Nominal (Mostly simple nesting, some inter-module control, flow control)
Computational Operations	Nominal (Mostly simple expressions, some math and statistical operations in generating reports)
Device-dependent Operations	Very Low (No I/O device needed)
Data Management Operations	High (There are multiple data file access and warehouse operations)
User Interface Management Operations	High (Simple input forms but with graphic report generation)

- **Platform Difficulty (PDIF)**
  The platform difficulty was rated as **Low** because the platform is stable and a fast network exists. The platform difficulty was not considered to be Very Low because the amount of data that could be stored could represent a resource constraint.

- **Personnel Capabilities (PERS)**
  SIMTRIX WEB WEAVERS has a very capable team and does not expect substantial delays due to turnover since the team is very cohesive and morale is high. Turnover is a risk, of course, and is documented in Section 4 – Risk Management. SIMTRIX WEB WEAVERS has hired top talent, in the 75th percentile (see Section 3.3.2 for details), which translates to a PERS of **High.**

- **Personnel Experience (PREX)**
  SIMTRIX WEB WEAVERS has a fair amount of personnel experience with Java, J2EE, Struts, databases, and other tools/frameworks that will be used on this project. However, the total experience of the team averages just over 4 years with the tools and languages employed (see Section 3.3.2 for more details on team experience). Therefore, **High** was selected for this cost driver. While the team does not have significant experience with OptimalJ, this was not considered in this estimation. This is because the savings associated with this tool are applied after the estimation is created and thus any impact of this tool should not be considered. Instead, the impact of the lack of experience with OptimalJ is accounted for by reducing the expected savings.

- **Facilities (FCIL)**

All staff are co-located and work well together. Because of the size of the team (5 people), collaboration is very easy and tools to facilitate collaboration are not required. Thus, **Very High** was selected as the rating for facilities.

- **Schedule Constraints (SCED)**
  Because this project is being developed for an outside company, it is unlikely that the schedule will be able to be relaxed. However, it is also unlikely that the schedule will need to be shorted. Thus **Nominal** was selected as the rating for this cost driver.

- **Degree of Planned Reuse (RUSE)**
  There are currently no plans to reuse this project as part of a product line. Even if there were, there is no guarantee that SIMTRIX WEB WEAVERS would receive the contract. Thus, since reuse is not part of the requirements for this project, it was not considered. Thus, the rating of **Nominal** was selected. Under WEBMO, Nominal is the lowest rating for this cost driver.

- **Teamwork (TEAM)**
  SIMTRIX WEB WEAVERS has excellent team cohesion. However, as a consulting company, there is less of a feeling of personal ownership with a project. Thus, while there is a shared vision among the team, it is not as strong as would be the case with a small company developing software for its own usage. As a result, a rating of **High** was selected.

- **Process Efficiency (PEFF)**
  The development lifecycle selected for the development of the SPBS is a very efficient, agile-like one. Our team will develop informal artifacts for communication within the organization, thus eliminating the number of artifacts that do not directly contribute to the product. Additionally, our team will be eliminating the number of stubs that need to be developed for testing by conducting bottom-up integration testing. While this is true, our team will need to develop several formal documentation artifacts that need to be delivered to the customer because they will be maintaining the product. Moreover, all of the code will need to be well documented for the same reason. Thus, process efficiency was rated as **Nominal**.

The ratings for the WEBMO cost drivers are summarized in the table below, which also shows the values for each cost driver. These values are used in the Effort equation discussed in Section ??.

Table 15 – Summary of Cost Driver Values

Cost Driver	Rating	Value
Product Reliability and Complexity (CPLX)	Nominal	1.00
Platform Difficulty (PDIF)	Low	0.87
Personnel Capabilities (PERS)	Nominal	1.00
Personnel Experience (PREX)	High	0.87
Facilities (FCIL)	Very High	0.68
Schedule Constraints (SCED)	Nominal	1.00

Cost Driver	Rating	Value
Degree of Planned Reuse (RUSE)	Nominal	1.00
Teamwork (TEAM)	High	0.75
Process Efficiency (PEFF)	Nominal	1.00

3.4.2.5 Effort and Duration Equations

After calculating the total number of weighted Web Objects, effort and duration can be calculated via WEBMO's parametric equations. These equations are as follows:

$$\text{Effort} = A \left(\prod_{i=1}^{9} cd_i \right) (\text{Size})^{P1}$$

$$\text{Duration} = B \, (\text{Effort})^{P2}$$

where Size is the number of thousands of lines of source code (KLOC). WEBMO backfires from Web Objects to KLOC so the same formulas used in COCOMO II can be utilized. This means that Web Objects must be multiplied by a conversion factor specific to the language. Suggested conversion factors are provided in [15]. SIMTRIX WEB WEAVERS has found that a conversion factor of 32 is appropriate for a J2EE based the Web application. This is driven by the fact that most development is in Java (whose conversion factor is 32). While other languages are employed, including HTML, XML, and JavaScript, they represent a fraction of the effort. While there is obviously substantial HTML, this is embedded within JSP pages where it is better counted as Java code than HTML. HTML has a conversion factor of 15, and it is quite typical for a JSP file to be at least twice as large as the HTML it generates. Therefore, using SIMTRIX WEB WEAVERS' experience with developing Web applications and using WEBMO, a conversion factor of 32 was found to be appropriate.[18]

A, B, P1, and P2 in the formulae are constants. It is here that WEBMO differs from COCOMO II as P1 and P2 are power laws that are constant in WEBMO but are variable in COCOMO II. The values of A, B, P1, and P2 are determined by the type of application and the number of Web Objects. The following table lists values used by SIMTRIX WEB WEAVERS, which were obtained from [15].

Table 16 – Constant Values for Different Application Types

Application Domain	A	B	P1	P2
Web-based electronic commerce	2.3	2.0	1.03	0.5 or 0.32
Financial/trading applications	2.7	2.2	1.05	0.5 or 0.32
Business-to-business applications	2.0	1.5	1.00	0.5 or 0.32
Web-based portals	2.1	1.8	1.00	0.5 or 0.32

[18] In some ways, 32 represents an arbitrary number. However, the difficulty of breaking down the architecture into what Web objects are implemented in what language and then coming up with appropriate conversion factors for JavaScript and XML was considered too much for this assignment. It would have easily tripled the amount of documentation required to estimate the effort using WEBMO and required even more assumptions.

Application Domain	A	B	P1	P2
Web-based information utilities	2.1	2.0	1.00	0.5 or 0.32

The SPBS best fits the Web-based information utilities domain since it provides information in the form of metrics and reports to users. Thus, these values for A, B, P1, and P2 will be used. The exact value for P2 will be selected depending on the number of Web Objects counted. If it is greater than 300, then 0.32 is used. Less than 300 means that 0.5 will be used. 300 represents a very small Web application and this is not a small application, so 0.32 will invariably be used.

The Duration formula is not particularly of interest except that it can be used to validate the team size. COCOMO II and WEBMO both determine staff levels and use this to determine a duration for the project (effort divided by staff). This can be used as a rough guide to the number of calendar months a project will take. Since a project plan is being developed, this duration estimate can be used to validate the plan.

It should be noted that the formula published in [15] has the effort product listed as from i=1 to 8. However, there are nine cost drivers. Therefore, the formula was changed to ensure that all 9 cost drivers are considered in the effort calculation.

3.4.2.6 Web Object Analysis

Web Object analysis requires a relatively deep understanding of the pages and components that will be used by those pages. Luckily, a prototype of the SPBS exists and was used to count Web Objects. This prototype is available. The prototype was used to get an appreciation for the complexity of the user interface and components that might exist on it. This was combined with the requirements to develop a set of representative sites, including Web pages and actions, to generate a complete enough vision of the SPBS to estimate Web Objects. WEBMO relies on rather detailed knowledge of the Web components and how they are built, so this effort was required and is detailed in Section 2.5.4.2.

Starting with these sites, the following rules were applied to determine the number of Web Objects:

1) Every major page represents a logical file. Most pages are considered to be simple. There are a few that are not, however. These pages are concerned with either gathering large amounts of data (i.e. metrics) or displaying data (reports and validation. The following pages are therefore considered Average in complexity: validateData.jsp, viewReport.jsp, demographics.jsp, projectSizing.jsp, projectResources.jsp, projectSchedule.jsp, projectQuality.jsp, projectComplexity.jsp, and projectDefects.jsp
2) Every HTML form represents an External input. The presence of an HTML form can be seen in the Page Flow diagrams by the existence of a FormBean (a mini table in the lower right of an Action). The number of fields determines whether a Form is Low or Average complexity. Only those forms in the Add Project portion of the Main Site and the forms associated with member registration and

profile management (Main and Member Administration respectively) are considered Average. The rest of the forms are Low.

3) Every HTML form has 2 JavaScript operands associated with it. One is used for client-side validation and has a single operator – validate. The other is used to reset the form and also has a single operator – reset. The complexity of the script mirrors the complexity of the form.
4) Every Action represents a Web Building Block operand. Any Action with a FormBean is considered of Average complexity while those with no FormBeans are considered Low. The rationale for this is that FormBeans will require some validation but will delegate any complex behavior to the application tier. All Actions have a single operator – navigate.

Additionally, to account for the complexity of displaying metrics reports for projects (feature A1) and presenting data to be verified and validated (feature SA2), these two pages have explicit External Outputs. These are considered Average mainly because in this release presentation is not a high premium but rather functionality. To account for communication with the Third Party Payment system, a single External Interface file is listed for the Member Administrator site. Additionally, a Web Building Block is associated with the usage of the External Interface. This represents a component (JavaBean or Tag Library) responsible for communicating via the External Interface. This Web Build Block operand has three operators – add card, remove card, validate card. The actual charging of the card is performed within the application tier and is accounted for in that portion of the estimate. Due to the security concerns with credit cards, both the interface and the Web building block that uses it are considered to have High complexity. Finally, the site was considered to have 5 images (operands), each having a single operator of render. These images are large and small company and SPBS logos and an image for the header.

The summary of all Web Objects is presented in Table 17. A breakdown of Web Objects for each site and sub-site is provided in the associated Excel spreadsheet (WEBMO.xls).

Table 17 – Web Objects

	Weighted Web Objects			
Web Tier	**Low**	**Average**	**High**	**Total**
Internal Logical Files	17	9	0	209
External Interface Files	0	0	1	10
External Inputs	8	10	0	64
External Outputs	0	1	0	5
External Inquires	0	0	0	0
Multimedia Files	10	0	0	40
Web Build Blocks	38	36	4	282
Scripts	16	20	0	92
Links	0	0	0	0
Total Web Objects				702

3.4.2.7 Function Point Analysis

In addition to this analysis of the Web tier, the complexity of the application tier was taken into consideration. This was done via traditional Function Point analysis. WEBMO uses the same complexity values for the traditional Function Point features of Input, Output, Logical Files, Interfaces, and Inquiries but simply treats them with a Web application slant. That slant does not need to be taken here. The number of unadjusted Function Points can be calculated and treated as another type of Web Object in WEBMO, as WEBMO considers the impact of such components within its model.

The application tier consists of the Controller Module, Member Module, User Module, Project Module, Report Module, Payment Module, and Data Access Model. As discussed in Section 2.5.4.1, when this architecture is mapped onto J2EE and Struts, the Controller Module becomes a Servlet with the other modules becoming Session Beans.

Since the WebLogic Server has been selected as the application server, the Controller Modules are Java Page Flow (JPF) files. The complexity of a JPF really resides in the Action classes, but these are already accounted for in the Web application analysis (they are Web Building Blocks). Therefore, only the page layout and general framework needs to be addressed. This means that the Controller model is of Low complexity. There are, however, four JPFs in the system (the Main site, the Add Project sub-site of the Main site, the Site Administrator Site, and the Member Administration Site). Therefore, the Controller is represented by four Logical Files, each of which has Low complexity. All other complexity is encapsulated in the Action classes and is therefore already counted. One advantage of using WebLogic Server is that the Struts configuration files are automatically created by using JPF, reducing the amount of effort to develop Web applications.

The Member Module, User Module, Project Module, Report Module, Payment Module, and Data Access Module however must be addressed in greater detail since they represent a fair degree of complexity. As Session Beans, there are three Logical Files per module – the bean implementation class and two deployment descriptors. There is also one Interface File.

As far as implementation classes go, the security aspects of the Member, User, and Payment Modules make these components Highly complex. The Project, Report, and Data Access Modules are counted having Average complexity since they have substantial business logic that must be implemented but lack the significant security aspects of the other modules. Thus, there are 3 High and 3 Average Logical Files represented by the bean implementation classes.

The bean interface classes and deployment descriptors are considered to be Low in complexity. Interfaces are simplistic with EJBs since they just list the methods available in the bean's implementation class. The deployment descriptors can be mostly generated by any decent tool and even if written by hand are usually copied from an example and modified slightly. Thus, none of these files represents a significant degree of complexity, leading to 6 Low complexity Interface Files, and 12 Low complexity Logical Files.

Inputs, Outputs, and Inquires must be counted on a per module basis. This information is obtained from the architecture descriptions which is presented in **[17]**. This information is summarized in Table 18, where each module is broken down into Inputs, Outputs, and Inquires. The complexity of each function point is shown in parentheses. Function Points that communicate directly with the third party payment system (Payment Module) and handle the input, validation, and viewing of metrics are considered high while the rest are considered average. Note that in some cases, actions are delegated to lower levels and the complexity is accounted for at this lower level. This is the case with the Member Module in particular.

Table 18 – Function Point Analysis for Modules

Module	*Inputs*	*Outputs*	*Inquires*
Member Module	• add user (avg) • delete user (avg) • modify user (avg) • add project (avg) • modify project (avg) • purchase report (avg) • run report (avg)	None	None
User Module	• create new users (avg) • check login credentials (avg) • assign roles (avg)	None	None
Project Module	• input metrics (high)	• retrieve metrics (high)	• validate metrics (high)
Report Module	None	• format report (high)	• request report (high)
Payment Module	• charge credit card (high)	None	None
Data Access Module	None	None	• retrieve user information (high) • retrieve project information (high) • retrieve report information (high)

It should be noted that in the architecture the Member Module delegates all behavior to other modules. It does not itself implement behavior. It therefore represents a façade class. The need for this façade is not apparent from the architecture, but is likely to allow for centralized auditing or other security features.

In the Project Module, metrics validation is considered an inquiry because it represents an action that combines an Input and an Output (mainly, metrics for a project are retrieved, displayed, and then potentially moved to the new database). As a result, it is a compound

action that should be treated as an Inquiry. Similarly, requesting a report in the Report Module represents an Input (which report and project) and an Output (the report) and is therefore considered an Inquiry. All Data Access Module functions are also considered Inquiries since they represent queries that return specific data.

Also included in this estimate are three additional Average complexity logical files. These three files are the deployment descriptors for the Enterprise Application, Web Application, and EJB Jar files. These are listed explicitly because they must be modified to account for the appropriate security constraints on the system, which is also why they are Average.

The Weighted Function Points for the Application Tier are summarized in Table 19, below.

Table 19 – Application Tier Weighted Function Points

	Weighted Function Points			
Function Point Type	**Low**	**Average**	**High**	**Total**
Logical Files	16	6	3	217
Interface Files	6	0	0	30
Inputs	0	9	2	48
Outputs	0	0	2	14
Inquires	0	0	5	30
Total Function Points				339

3.4.2.8 Estimations

Now that the process has been discussed and the analysis completed, the estimates can be calculated. The first step in this calculation is to calculate the total number of weighted function points and Web objects. Within WEBMO, these two can simply be added together. Thus:

Total Function Points/Web Objects = 339 + 702 = **1041 FP/WO**

Now, these objects can be converted to source lines of code (SLOC) as the measure of size. The formula for effort actually requires thousands of source code lines (KLOC), so SLOC is divided by 1000. SLOC is calculated by multiplying the total number of function points/Web objects by a conversion factor for the language. As discussed in Section 3.4.2.5, a conversion factor of 32 was selected.

KLOC = 1041 * 32 / 1000 = 33.312

Using KLOC as the measure of size, the effort can be calculated by using the formula and the constants for Web-based Information Utilities presented in Section 3.4.2.5 and the cost drivers detailed in Section 3.4.2.4.

$$\text{Effort} = A \left(\prod^{9} cd_i\right) (\text{Size})^{P1} = 2.1 * (0.28951425) * (33.312)^{1} = \mathbf{20.25\ months}$$

i=1

As in the process based estimate, a safety buffer was added to the estimate. This buffer was set to 7.5% as the site is considered to be moderately complex. However, here it is applied to the entire estimate rather than built into the estimate. This buffer was deemed advisable as WEBMO is an aggressive estimation tool and requires a large amount of detailed estimation. To offset this risk, the safety buffer was added. This risk and mitigation is described in Section 4.2.1 (risk #RS3)). This leads to the following buffered effort estimate:

Buffered Effort = Effort * 1.075 = **21.77 months**

These calculations do not consider the cost savings associated with using Model Driven Development with OptimalJ and WebLogic Workshop. As discussed in Section 3.1.2 and 3.1.2.2, this accounts for an estimated 20% cost savings. Taking this savings into consideration, the following Effort estimate is obtained:

Adjusted Effort = Effort (1 – Cost Savings) = 21.77 * .80 = **17.42 months**

Multiplying this by the burdened labor rate of $9,321.47, results in a **total project cost of:**

17.2 months * $9,321.47 per month = **$162,380.01**

Using this Effort estimation, the Duration can be calculated. Duration is useful mostly as a check against the schedule. The schedule can consider additional factors such as availability and skill levels that this equation does not.

Duration = B $(\text{Effort})^{P2} = 2 * (17.42)^{0.32}$ = **4.99 months**

This duration assumes a staffing level on average of 3.49 developers. This means that the actual duration will either be less than 4.99 months or that some developers will be idle. Any idle time will allow for more thorough testing, especially of security aspects.

3.4.2.9 Notes and Assumptions

The following section details the assumptions made while performing the WEBMO estimate.

- It is assumed that the Java Page Flows detailed in Section 2.5.4.2 are accurate and complete.
- It is assumed that the 5 images identified in Section 3.4.2.6 are all the images required for the site.

3.4.3 Estimation Summary and Results

The process-based technique yielded an estimate of 19.23 staff months to develop the System and the WEBMO technique yielded an estimate of 17.42 staff months.

Averaging the two methods resulted in an estimate of 18.33 staff months and a total development cost of 18.33 * $9,321.47 ≈ $170,900.00

In addition to the software development costs, the costs of all hardware and software required for deployment must be included. Note that hardware and software used for development is included in the burdened labor rate and is therefore not included here. All open source software was used for deployment, but hardware was required to be purchased. These decisions are documented in Section 3.5.2. The total cost for hardware and software for deployment was calculated to be $6,886.

Thus, the total cost for building and deploying this system is approximately

$170,900.00 + $6,886 ≈ **$177,800**

3.5 Project Resources

3.5.1 People

The development team consists of five members, each playing one of the following roles:

- Project Lead (Software Engineer III)
- Database Analyst II
- Interface Designer
- Software Engineer II
- Programmer II

Because of the relatively small budget and tight schedule, a controlled team organization has been selected. In addition, to facilitate cooperation and creative solutions to problems, a decentralized team organization has been selected.[19] Thus, the Project Lead shall coordinate the efforts of the other team members while difficult problems shall be solved as a group (at the discretion of the Project Lead). This team structure is agile-like in that most of the time. The team will function independently in a collaborative manner. The Project Lead will only unilaterally make decisions when the team encounters a stumbling block. Additionally, while the Project Lead shall serve as primary liaison to the customer, other team members will be involved in consultations with the customer, as necessary.

For more details on the organization of the team, refer to Section 6 on page 69.

3.5.2 Hardware/Software Requirements

This section describes the hardware and software requirements for the deployed system. It does not discuss the hardware and software required for development, which is covered instead in Section 7.1.

[19] Refer to Pressman's book for more detail on controlled-decentralized team organization. [p. 61 of [13]]

3.5.2.1 Minimum Hardware Requirements

The Physical Architecture specified in [17] identifies the need for three machines – a Web Server residing in a demilitarized zone (DMZ) and two application server/database server machines residing in a protected network. Also shown is a proposed intrusion detection system, but this system was not considered for this release. Any intrusion detection system should be considered at a organization-wide level and not for a specific project. Thus, the company may already have or wish to put in place a university wide intrusion detection system.

It is also assumed that the company has sufficient firewalls, routers, and networking capacity to support the addition of the SPBS without the purchase of any additional hardware.

Thus, the only required purchases are for three machines. As discussed in the next section, Linux has been selected as the operating system and will be installed separately. Thus, only hardware is required. This hardware will reside in a network operations center and therefore must be rack mountable. The Web Server machine will not have significant CPU or disk utilization (it is mainly a security component) and therefore can be a lower capability machine. Additionally, it does not need RAID support for data recovery since it will contain no user data and can be re-built from the software release. The other two machines, however, will contain user data and therefore must have RAID 1 support. They also much be a better class of machine in terms of CPU, memory, and disk space. The following table documents the selections for these machines and the cost. Dell was chosen as the vendor due to prior positive experience with the company in terms of price and support. Prices are as of 12/05/2004.

Table 20. Hardware Cost Summary

Usage	*Specification*	*Cost*
Web Server	Power Edge 750, 2.4GHz Celeron processor, 512Mb memory, 40Gb hard drive	$1,210
App/DB Server	Power Edge 750, 2.8GHz Celeron processor, 2Gb memory, 2x73Gb hard drive, RAID 1 controller	$2,838

Since two Application/Database Server machines are required to provide for redundancy, the total cost for hardware is $6,886. This cost includes shipping, rack hardware, and cabling but does not include a monitor, keyboard, mouse or the rack itself. It is assumed that the company has these items within its data center.

3.5.2.2 Minimum Software Requirements

The choices for software packages for deployment were driven by the company's insistence of the use of open source software. There are a number of software pieces required. The table below summarizes these pieces. All of these packages can be obtained for free.

Software Description	Selected Software Package	Additional Information
Operating System	Mandrake Linux, v10.1	http://www.madrakesoft.com
Web Server	Apache HTTP Server, v2.0.52	http://httpd.apache.org
J2EE Web Container	Tomcat, v5.0.28	http://jakarta.apache.org/tomcat/
J2EE EJB Container	JBoss, v3.2	http://www.jboss.org/products/jbossas
Database	mySQL, v4.1	http://www.mysql.com/

Table 21 – Open Source Software Packages

JBoss actually bundles Tomcat. Therefore the application tier is actually a single, integrated package. It is broken out separately since its documentation is separate and it is possible to upgrade independently of the JBoss EJB container. It should also be noted that JBoss is more than an EJB container. It also supports other J2EE functionality such as the Java Messaging Service and Java2 Connector Architecture. When combined with Tomcat, it is a full J2EE application server.

The combination of Apache, Tomcat, and mySQL support the requirement to have two redundant sites for failover purposes. mySQL supports database replication, allowing the redundant site to be kept up-to-date. Failover is accomplished by using the *balancer* Tomcat plug-in into Apache (see http://jakarta.apache.org/tomcat/tomcat-5.0-doc/balancer-howto.html). Note that this failover will not preserve user sessions. The architecture does not support such a failover mode as it would require synchronous, multi-master replication (which would be significantly slower) or an additional machine providing a shared disk such as a Network Appliance file server (which would be a significant expense and is not part of the architecture). Therefore, this type of failover (a hot backup, essentially) was considered the best that could be done given the performance and budget constraints and the agreed upon architecture.

4 Risk Management

This section describes the Risk Mitigation, Monitoring, and Management (RMMM) activities our team plans to undertake during this project. This section additionally identifies the highest priority risks as well as the mitigation strategy for each of these risks.

4.1 Scope and Intent of RMMM Activities

There are two strategies for managing risk, proactive and reactive. [p. 146 of [13]] Our team prefers a proactive approach and as a result, shall weekly identify and rank the project's risk items and then make plans to mitigate and manage those risks. By employing this strategy, our team hopes to successfully avoid project overruns and deliver a high quality System.

4.2 Functional Data Description

This section defines the highest-ranking risks items.

4.2.1 Description of Risks

The highest-ranking risk items are identified, in no particular order, below. For a ranking of these risk items, refer to the next section, Section 4.4 Risk Table.

RS1) **Customer Requirements Change beyond process threshold** – Because we expect some requirements volatility, we have elected to use an agile-like, incremental, iterative development process. While this will accommodate significant changes to requirements, there is a change threshold beyond which the schedule will be impacted. If this threshold is crossed, we will need to review what we have done and the estimated delivery date may be delayed.

RS2) **MDA tools fail to meet expectations** – MDA tools (OptimalJ and Weblogic Workshop) are relatively new tools. In fact, this is the first time our team will use these tools. Although we expect it to maximize our productivity, it might also bring us unexpected technical problems. If these problems can not be solved efficiently, we might lose expected productivity gains.

RS3) **WEBMO estimation is wrong** – WEBMO estimation requires detailed knowledge about how the site will be built, if we miss a component which should be counted into the estimation, our effort/cost estimate maybe significantly inaccurate.

RS4) **Vulnerable security** – The Web site will contain important customer project information and they are supposed to be protected. Possible security problems include: 1) Data are leaked by authorized user through legal channel 2) Data are leaked by external effort through illegal channel 3) Data are leaked by application with improper actions.

RS5) **Poor product quality** – If the system does not meet the requirements (both stated and unstated), the product will fail.

RS6) **Third-party payment system fails** – As a Web site with electronic trades, the availability of the payment system is the key to the customers. Since we are going to introduce a third-party payment system. Their reliability, scalability, performance, security and privacy will definitely affect our system's quality. We are trusting 3rd party to meet requirements. Any failure to do so will jeopardize project and we can't do much about it.

RS7) **Staff turnover** – Any staff turnover can bring significant, negative impact to the project.

RS8) **Open Source software is insecure and/or unreliable** – The security and the reliability of open source software we will use in the project can bring significant impact on the quality and security of our product. Also, if open source are not easy to use, our schedule might be delayed because of unexpected technical problems and lacking of sufficient support.

4.3 Risk Management Calculation

We evaluate risk from two dimensions: the probability of occurrence and the impact of each risk. The overall risk is defined with the following relationship:

Risk = Probability * Impact

The probability of occurrence is determined with a quantitative scale developed by SIMTRIX WEB WEAVERS that with following values:

Table 22. Probability Category

Category	*Probability*
impossible	0%
improbable	1% - 19%
probable	20% - 49%
likely	50% - 69%
highly likely	70% - 99.99%

The impacts of a risk are categorized into four levels: negligible, marginal, critical, catastrophic. We refer to the standard listed in Roger S. Pressman's "*Software Engineering – A Practitioner's Approach*" and tailored it with our situation [13].

Catastrophic

- Fail to meet the requirement and result in project fails
- Failure increases costs significantly and makes financial shortage
- Technical performance becomes unachievable.
- Unachievable IOC

Critical

- Failure to meet the requirement would degrade system performance. Project success becomes questionable.
- Minor delays in software modifications
- Failure results in operational delays and/or increased costs with some shortage of financial resource
- Possible slippage in IOC

Marginal

- Minimal to small reduction in technical performance
- Failure leads to cost increase but the project still have sufficient financial resource
- Schedule slips but still recoverable or achievable

Negligible

- Failure to meet the requirement would create inconvenience or non-operational impact
- Error results in minor cost and/or schedule impact

- Possible budget overrun
- Achievable IOC

4.4 Risk Table

The following table classifies the relative importance (indicated by a "risk value") of each risk item:

Table 23. Sorted Risk Table

Risks	Probability of Occurrence	Impact	Risk Value
RS2) MDA tools fail to meet expectations	30%	4	1.2
RS4) Vulnerable security	30%	4	1.2
RS1) Customer Requirement Change beyond process threshold	20%	3	0.6
RS6) Third-party payment system fails	20%	3	0.6
RS3) WEBMO estimation is wrong	20%	2	0.4
RS7) Staff turnover	10%	3	0.3
RS5) Poor product quality	10%	3	0.3
RS8) Reliance on Open Source software	15%	2	0.3

The probability that a risk item will occur is estimated (second column). If that risk item is realized, the impact that it will have on the project is then categorized (third column) according to the impact values summarized in Table 24. The value categories in this table (e.g., Catastrophic) were taken from Pressman's description of impact values. [p. 151 of [13]]

Table 24. Impact Value Descriptions

Impact Values	*Description*
4	Catastrophic
3	Critical
2	Marginal
1	Negligible

The "risk value" is computed by multiplying the probability of occurrence by the corresponding impact value (fourth column). This value is then used to rank the project risks. As one can see from this table, RS2, RS4, RS8 are currently the most important risk item and RS5 is currently the least important risk item.

4.4.1 Assumptions Regarding Risk Probabilities and Impacts

The following section describes reasons of choosing the probability and the impact of each risk.

RS1) **Customer Requirements Change beyond process threshold** – The probability of occurrence for this risk is getting lower with the progress of the project. While we have already developed the architecture of the system, we can rank the risk as 20% probability now. However, since the budget and the time of

this project is limited, once the customer wants to change requirements in current status, especially when this change is beyond our process threshold, there would have a critical impact on the project because we need to review all we have done and the estimated delivery date will be definitely put off, we might not be able to meet our budget in current situation either.

RS2) **MDA tools fail to meet expectations** – We will meet some predictable problems when using the MDA tools. Sometimes it will generate heavier code than expected; sometimes the code generated may not be what the designer is expecting. We considered these possibilities and estimate a lower productivity increase than what the case study cited (20% instead of 35%). Basically, we believe these tools are reliable productivity-enhancing tools. We also believe that any problems we encounter will be solvable by our team. But if we were to encounter insurmountable problems with these tools late in the project, the impact could be catastrophic because we will have relied these tools for specifying the architecture, designing the system, and generating code. We believe the probability of this catastrophic occurrence is 30%.

RS3) **WEBMO estimation is wrong** – The main reason of failing in WEBMO estimation is the missing of important detail. Considering our current knowledge of the system, we have had a comprehensive understanding of the whole system. The probability of occurrence of this risk is low. We therefore estimate it as 20%. The impact is marginal.

RS4) **Vulnerable security** – Breaches in security could be catastrophic to the Web site. Without effective countermeasures, this probability could be no less than 70%. We researched security threats in detail and designed a comprehensive threat model aimed at addressing these security problems. We believe with the threat model we built in architecture, we can mitigate this risk sufficiently. However, theoretically speaking, the possibility of potential undiscovered threats and the possibility of system leaks always exist, which is why we specified a 30% probability of occurrence for this risk. We will constantly monitor and employ strict risk-based testing to prevent the occurrence of this risk. Over time, as countermeasures are implemented, we expect the probability of this item to become negligible (less than 1%).

RS5) **Poor product quality** – With our experience and development ability and with the past record of our company, the probability of this risk is 10%. The impact of releasing a poor quality product is critical for the project and for our company.

RS6) **Third-party payment system fails** – There are several reliable options of third-party payment system. We need to make a selection through test. The final selection should be compatible to our security metrics and system interface requirement. The probability of failing in this selection is about 20%. While payment system is the component for our customer to get income, it's important and the impact of failing on it is critical.

RS7) **Staff turnover** – We have a tightly-knit, highly-cohesive, jelled, team that has been working together for the past year and a half. We have also spoken with each member on the team and expect no one to leave for the duration of the project. As a result, the probability of this risk is low. While this is true, someone may leave

for reasons we cannot prevent, so the probability of the occurrence of staff turnover risk is about 10%. Since we are a small company, any staff turnover might bring critical impart to the project, especially if this occurs late in the project.

RS8) **Reliance on Open Source software** – The reliance on Open Source software represents a risk to the system. Some Open Source software is not of high quality, well-supported, or sufficiently secure. Issues within open source often must be resolved by the development organization since the Open Source community may not address any issues quickly enough. The probability of this risk occurring is dependent on the Open Source software packages selected. Well used packages were selected, meaning the likelihood of problems existing is low. It is estimated at 15%. Additionally, it is likely that workarounds are already known about the issue and if not, the source code is available. Therefore, the impact is considered to be Marginal. It could have an impact to the schedule, but it all but rare cases, it should only be slight.

4.5 Risk Mitigation, Monitoring and Management Plan

This section contains risk information sheets for every risk items that are characterized in section 4.2.1. Some of the risk information sheets are incomplete in that they do not specify content for the *management/contingency plan/trigger* and *current status* fields. This is the case because these risks are having extremely low risk values and have not been fully investigated at this time.

4.5.1 Risk Information Sheet for RS1

Risk Information Sheet				
Risk ID: RS1	**Date**: 12/01/04	**Prob**: 20%	**Impact**: Critical	**Risk Value**: 0.6
Description: Customer Requirements Change beyond process threshold				
Refinement/Context: The following types of changes are regarded as changes beyond the process threshold: 1) Changes that lead to specific, increased cost 2) Changes to the core function of the system 3) Extraordinary number of changes that cannot be accommodated in a single iteration				
Mitigation/monitoring: **1)** We use Win-win Spiral process. Win-win Spiral's iterative process leaves spaces for the team to accommodate changes. **2)** We use the MDA tool, OptimalJ. Because this tool integrates the process from architecture design to code generating, it can not only improve our productivity, it can also mitigate the effort of changing design.				
Management/contingency plan/trigger: Use Win-win principle, negotiate with our customer to minimize the impact of changing requirements. trigger: when we cannot digest the changes within a single iteration (even with OptimalJ), the effort is so high that it effects our schedule and budget.				

Current status: **12/01/04:** Core requirements are stable. Keep the probability of occurrence as 20% and keep monitoring.

4.5.2 Risk Information Sheet for RS2

Risk Information Sheet				
Risk ID: RS2	**Date**: 12/01/04	**Prob**: 30%	**Impact**: Catastrophic	**Risk Value**: 1.2
Description: OptimalJ fails to meet expectations.				
Refinement/Context: 1. We meet unexpected technical problems in using OptimalJ 2. Productivities can not be able to be achieved as we expect				
Mitigation/monitoring: **1)** The case study of MDA cited 35% decrease in overall efforts when using MDA tools. We assume we will gain 20% decrease, which should be practical enough for our technical experience. **2)** While expected duration of the project from the customer is one year, our current effort estimates are approximately 5 months, which means we would have enough space to accommodate the possible delay OptimalJ may bring to us. **3)** Use CCPM buffers to monitor if we are meeting the productivity expectation.				
Management/contingency plan/trigger: We have enough resources (two experienced programmers and one J2EE expert) to deal with OptimalJ failures. We have 3-4 months of flexible time. Negotiate with customers to minimize our possible lose in cost/effort. trigger: When problems cannot be solved as planned and delay the expected delivery date.				
Current status: **12/01/04:** Keeping the probability of 30%				

4.5.3 Risk Information Sheet for RS3

Risk Information Sheet				
Risk ID: RS3	**Date**: 12/01/04	**Prob**: 20%	**Impact**: Marginal	**Risk Value**: 0.4
Description: WEBMO estimation is wrong.				
Refinement/Context: The estimation results of cost and effort are far from the reality.				

Mitigation/monitoring: **1)** Perform separate independent estimation performed by different individuals to mitigate the possibility of over- or underestimate of using single estimation method. **2)** Incorporate the project safety buffer to absorb overruns, which leaves us more space to accommodate uncertainties **3)** Monitor the plan and review the estimation with the progress of the project
Management/contingency plan/trigger: Re-negotiate with the customer about the budget and schedule. trigger: Once there is sign that the project is going to beyond our budget and the promised delivery date will be impossible to meet.
Current status: **12/01/04:** Two estimation approaches show approximate result. Probability of the occurrence dropped from 30% to 20%

4.5.4 Risk Information Sheet for RS4

Risk Information Sheet				
Risk ID: RS4	**Date**: 12/01/04	**Prob**: 30%	**Impact**: Catastrophic	**Risk Value**: 1.2
Description: Vulnerable security risk				
Refinement/Context: Information is leaked or system is hacked				
Mitigation/monitoring: **1)** Design and apply a comprehensive security scheme: threat model. The model is aimed at threats including network threats, host threats and application threats **2)** Take risk-based testing strategy. Make specific tests on threat models and security problems. **3)** **Recommendation for the company:** Place hardware in a secure room to which only authorized personnel (those who have signed a non-disclosure agreement) have access.				
Management/contingency plan/trigger: Consider third-party security solutions. trigger: in case of threat model cannot pass the security test.				
Current status: **12/01/04:** Threat model has been designed in architecture level, the probability of the occurrence dropped from 70% to 30%. Further design, implementation and testing are needed to mitigate the risk further.				

4.5.5 Risk Information Sheet for RS5

Risk Information Sheet				
Risk ID: RS5	**Date**: 12/01/04	**Prob**: 10%	**Impact**: Critical	**Risk Value**: 0.3

Description: Poor product quality
Refinement/Context: The bug rate is over our quality metric; the performance of the system cannot meet the customer's requirement.
Mitigation/monitoring: **1)** Expert SQA engineer will establish product metrics and testing plan. **2)** Unit tests ensure the reliability of every component. **3)** Integration tests ensure the whole product's quality. **4)** System tests will cover the security and performance of the system. **5)** Customer will participate in establishing the testing plan, building test cases and doing tests. **6)** Use Bugzilla keep tracking on bugs.
Management/contingency plan/trigger:
Current status: **12/01/04:** SQA engineer begin establishing product metrics. The probabilities of occurrence keeps around 10%.

4.5.6 Risk Information Sheet for RS6

Risk Information Sheet				
Risk ID: RS6	**Date**: 12/01/04	**Prob**: 20%	**Impact**: Critical	**Risk Value**: 0.6
Description: Third-party payment system fails				
Refinement/Context: Reliability – If third-party payment system is not stable enough Scalability – If the third-party payment is not scale enough to accommodate our access capacity Performance – If the third-party payment system cannot meet our requirements of transaction time Security – If the third-party payment system is not secure enough during transaction Privacy – If the third-party payment system cannot guard our customer's privacy				
Mitigation/monitoring: **1)** Select reliable third-party payment system which has gained wide reputation. **2)** Test the selected payment system.				
Management/contingency plan/trigger: trigger:				
Current status: **12/01/04:**				

4.5.7 Risk Information Sheet for RS7

Risk Information Sheet				
Risk ID: RS7	**Date**: 12/01/04	**Prob**: 10%	**Impact**: Critical	**Risk Value**: 0.3

Description: Staff turnover
Refinement/Context: Two kind of reasons may lead to staff turnover: 1. Subjective reasons: people are unsatisfied with their working environment. It can be monitored and avoided. 2. Objective reasons: death in the family, disease and other cases which are beyond people's control. This type of turnover is usually unpreventable
Mitigation/monitoring: **1)** Talk with every team member, ensure their individual interests can be achieved as much as possible. **2)** Talk with every team member to determine if he/she is satisfied with his/her job and to ensure that he/she is not planning on leaving for the duration of the project. **3)** Ensure everyone's opinion in the team is respected.
Management/contingency plan/trigger: Continuously monitor the satisfaction and the progress of the team. Frequently ask questions regarding team members' situations. trigger: In case of turnover is unpreventable.
Current status: **12/01/03:** The team is highly-cohesive and jelled. Keep the low probability of 10%.

4.5.8 Risk Information Sheet for RS8

Risk Information Sheet				
Risk ID: RS8	**Date**: 12/01/04	**Prob**: 40%	**Impact**: Critical	**Risk Value**: 1.2
Description: Reliance on Open Source software				
Refinement/Context: Using Open Source software packages can lead to additional risks due to the following factors: 1. Lack of support 2. Lack of quality 3. Lack of documentation				
Mitigation/monitoring: **1)** Choose those widely used open source software, preferably packages supported by commercial vendors. **2)** Develop to standards wherever possible to ensure that, if necessary, system can be ported to a commercial software package. **3)** Use Model Driven Development to allow for rapid redeployment to a different platform. This especially mitigates risks with JBoss/Tomcat.				
Management/contingency plan/trigger: Negotiate with customers; Switch to some commercial software. Trigger: development issues that cannot be overcome				

Current status:
12/01/04: Selected Mandrake Linux, Apache Web Server, mySQL and JBoss/Tomcat. All packages are widely used, well supported, and reputable packages.

5 Project Schedule

The following section details the project schedule. The schedule is built from the process based estimate provided in Section 3.4.1 and is provided in a Gantt chart provided separately (see SPBS.mpp (Microsoft Project 98) or SPBS.html and gantt.bmp for the resource and task data and the Gantt chart). The schedule has a start date of November 4th, 2004, representing the date that the initial project planning started. This phase ended as of December 8th, 2004, when the first iteration starts. The project has an end date of April 4th, 2004. Considering just the development times for the iterations, which is what the estimation techniques addressed, this represents a 5 month effort, which is exactly what was predicted by the WEBMO estimation technique. It is uses slightly more staff than WEBMO predicted and took slightly longer than the process based estimate predicted due to some periods where the team was not fully allocated. These gaps were unavoidable due to certain tasks such as project planning that could not be shared and had to be completed before subsequent tasks could complete. These gaps are used by SIMTRIX WEB WEAVERS to allow for team members to update their skills.

The schedule was created by taking the tasks in the process based estimate and combining the efforts from the various features into a single value. For the Construction phase, this effort was then divided between the various layers in the architecture (Presentation, Application, and Database). This was necessary to map the development tasks to resources who have specific skills. While the process based estimate considered the effort to implement a feature at a whole, the work breakdown required to do scheduling required a slightly different breakdown. The approach taken was to assume that 50% of the construction effort would be in the Presentation tier (which for purposes of the schedule includes both the View (JSPs and HTML) and the Strut Controller with its associated Action classes. As a primarily Web application, this effort was considered reasonable. The remaining effort was divided evenly between the application and database tier tasks. This approach is not exact, but it is not meant to be. The developers in SIMTRIX WEB WEAVERS are well rounded and can therefore help out in other areas whenever necessary. Thus, it was deemed not to be necessary to provide a more precise division of effort among team members. Such precision would have required a bottom-up estimation process rather than the feature driven estimation approach taken. Given the agile-like development process employed (see Section 3.2), this approach is reasonable. Anything more would be overhead that is not required for the project.

The impact of a small, well-coordinated team is also seen in the inception, elaboration, and transition tasks. These tasks are considered to be group efforts for the most part, though specific individuals are assigned the Project Plan and Risk Plan because of the importance of these documents. The group will coordinate amongst itself to carry out these tasks efficiently and effectively.

5.1 Milestones

Each of the three Iterations represents a major milestone. However, tracking at this course of a level is not sufficient. Therefore, additional milestones were used. Specifically, the four life-cycle anchor points provided in [18] were used:

1. Life-Cycle Objectives (LCO) Milestone – occurs at the end of the Inception phase in the software process and provides details on the schedule, cost, and objectives of the current iterations
2. Life-Cycle Architecture (LCA) Milestone – occurs at the end of the Elaboration phase in the software process and provides an architectural vision for the iteration. In larger projects, it often includes an executable architecture, but this will not be used in this project. Prototypes and mockups may be included, however.
3. Initial Operational Capacity (IOC) Milestone – occurs at the end of the Construction phase in the software development process and releases a version of the system to be fully tested, both by the company and by the customer.
4. Iteration Complete Milestone – This milestone is an adaptation of the Product Release milestone in [18]. It represents an ending point for an iteration where the functionality has been fully tested and accepted by the customer. Unlike the Product Release milestone, software completed at the end of the first two iterations is not deployed in the production environment. They are interim releases to be used for validation and verification of the system.

The following table summarizes the milestones presented in the project schedule. The deliverables that are part of each of these milestones are covered in the next section.

Table 25 – Project Milestones

Phase	Iteration Completion Date	Milestones	Milestone Completion Date
Startup and Initial Planning	12/08/04	Requirements Specification High Level Architecture Project Plan Risk Management Plan	11/23/04 * 11/06/04 * 12/08/04 * 12/08/04 *
Iteration 1	01/06/05	Life-Cycle Objectives Life-Cycle Architecture Initial Operational Capability Iteration Complete	12/17/04 12/21/04 12/30/04 01/06/05
Iteration 2	02/23/05	Life-Cycle Objectives Life-Cycle Architecture Initial Operational Capability Iteration Complete	01/20/05 01/26/05 02/11/05 02/23/05

Phase	Iteration Completion Date	Milestones	Milestone Completion Date
Iteration 3	03/31/05	Life-Cycle Objectives Life-Cycle Architecture Initial Operational Capability Iteration Complete	03/08/05 03/11/05 03/18/05 03/31/05
Project Complete	04/07/05	Project Complete	04/07/05

* - Indicates that the associated milestone has been completed.

5.2 Deliverables

Each milestone has a set of deliverables associated with it. For the Startup and Initial Planning iteration, the deliverable and the milestone are the same. The Project Plan and Risk Management Plan are fulfilled by this document. The Requirements Specification and High Level Architecture were developed and delivered prior to this project plan being developed.

Since this project is to be turned over to the company for maintenance, some additional documentation is required to ease this transition. Thus, there are additional deliverables required as part of the milestones in each iteration. In all cases, the same deliverables are required. As the iterations complete, additional detail is added to the deliverables such that it addresses all functionality added in the iteration. These deliverables are summarized in the table below.

Table 26 – Deliverables for Milestones

Milestone	*Deliverable*
Life-Cycle Objectives	Product Overview Project Plan
Life-Cycle Architecture	Software Architecture Document Software Design Document
Initial Operational Capability	User's Guide
Iteration Complete	Source code
Project Complete	Source code, deployed system

The Product Overview is extended to simply provide more details about the iteration under development and address any changes made for this iteration. Because an agile-like development process has been undertaken (see Section 3.2 for more details), a certain degree of change is permitted. Any changes will also affect the Project Plan. The remaining documents will be updated to provide additional details about the functionality being developed. Thus, while a High Level Architecture document was delivered as part

of the Initial Planning and Setup Iteration, the next level of detail will be added to this document for functionality built as part of an Iteration. Further levels of detail will be added to the Software Design Document. All artifacts will be re-delivered should they have to change for any reason during an iteration.

6 Project Team Organization

The structure of the development team is loosely outlined in Section 3.5.1, People, as a controlled, decentralized team consisting of five members. This section defines the team structure in greater detail.

6.1 Team Structure

The positions of the team associated with this project were defined in Section 3.3.2 and are reiterated in Table 27, below. Table 27 also defines who will be fulfilling each role.

Table 27. Team Members

Name	*Technical Rank Level*	*Project Roles*
Monty Burns[20]	**Software Engineer III**	Project Lead, Programmer
Bart Simpson[20]	**Database Analyst II**	Database manager, Programmer
Ned Flanders[20]	**Interface Designer**	Interface Designer, Programmer
Waylon Smithers[20]	**Software Engineer II**	J2EE Infrastructure Engineer, SQA Engineer, Programmer
Lisa Simpson[20]	**Programmer II**	Programmer

Due to the size of the project and the required skill sets, we chose a team of five people. While each person has an assigned role, all of the team members will play as programmers in the coding activity and be a tester in the testing activity, as is typical in small organizations. The assignment of the primary roles (summarized in Table 27) considered both the capability of each person and the work required of each role.

Monty shall be in charge and all other members of the team shall report directly to him. At the same time, to facilitate team cohesion, cooperation and creative solutions to problems, the group shall solve difficult problems as a group. Thus, Monty shall work to maximize communication and consultation among all group members.

Everybody in the team can be a programmer. Anybody who finishes his/her previous tasks could be assign to help other programmers. While except Lisa Simpson, all have other specific project responsibilities, which will often divert their efforts from programming, Lisa will be the main programmer in the group.

Instead of assigning individual testers, we will take a group effort on testing job. Each programmer will be responsible for creating automated tests for inclusion in the regression test suit and performing unit testing on their implementations.

[20] Character name taken from *The Simpsons*. [14]

Waylon will also work as SQA engineer due to his experience in quality control.

Following descriptions are regarding responsibilities of each type of role rather than individuals.

6.1.1 Project Lead

- Lead the communication between the team and the customer and act as the coordinator between team members.
- Behave as the main constitutor of the plan.
- Monitor and evaluate the implementation of the plan and risks continuously.
- Lead the design of the system.

6.1.2 Database Analyst

- Database management.
- Design and implement the database access module.

6.1.3 Interface Designer

- Be responsible for the appearance of the product.
- Design and implement the entire user interface.

6.1.4 J2EE Infrastructure Engineer

- Help leading the design
- Offer expert J2EE advices to the team in both design and programming.
- Configure the development environment.

6.1.5 SQA Engineer

- Make metrics for the project and the product.
- Design test cases.
- Make internal, informal test plan.
- Monitor the quality of the product.
- Lead integration testing.
- Track failure

6.1.6 Programmer

- Design and implement modules they are assigned.
- Perform unit testing on their implementations
- Creating automated tests for inclusion in the regression test suite.

6.2 Additional Member Responsibilities

Additional responsibilities for all members include the following:

- Customer delegates, although we didn't list above, are treated as part of our team in the whole process of the project.
- All team members shall work together to create the overall design of the system.
- The project manager and the J2EE infrastructure engineer are principals in the architecture design.
- All team members are responsible for reviewing the design and the final product.
- All members shall help perform integration testing.
- Customers are responsible for offering requirement, answering questions from technicians.
- Customers should work with SQA engineer on designing test cases and work with the tester on testing.

7 Environment

The following section describes the development environment used by **SIMTRIX WEB WEAVERS** to develop the SPBS.

7.1 Development Environment

All members of **SIMTRIX WEB WEAVERS** are supplied with top of the line personal computers no more than one year old running Microsoft Window XP Profressional. Each developer is supplied with licenses for OptimalJ. Developers also have BEA WebLogic Workshop installed (licenses are free for developers for this tool) along with all software identified in Section 3.5.2.2. Development will occur using WebLogic Workshop and WebLogic Server for ease of development, but developers will deploy to Tomcat/JBoss for unit, integration, and system testing. This dual environment allows for rapid development using tools well known by the company but still allows for support for the requirement that deployment be on open source. Developers will also have the appropriate tools installed for configuration management and quality control.

7.2 Tracking and Control Mechanisms

7.2.1 Quality Assurance Mechanisms

As discussed above, the development organization has adopted a spiral, incremental development process. During each iteration, a milestone for the Initial Operational Capability (IOC) will be met and software will be released to be tested. Testing for each iteration follows the same general pattern, as shown in Figure 7. This diagram is a general diagram used for all projects performed by **SIMTRIX WEB WEAVERS**.[21] Each project starts in an initial phase (phase 0) and iterates through a number of incremental builds until a full release is ready to be validated and deployed at the customer site. For this project, the phases and increments are represented by Iterations 1 through 3. At the

[21] This test procedure was adapted from a procedure developed for CS 5744, Software Design and Quality, Fall 2004.

beginning of each testing phase, a set of regression tests are run. These regression tests represent all tests created to test functionality from the previous phases (thus, in phase 0 there are no regression tests). Concurrent to regression testing, all new functionality is tested and these tests encapsulated into regression tests for the next phase (iteration).

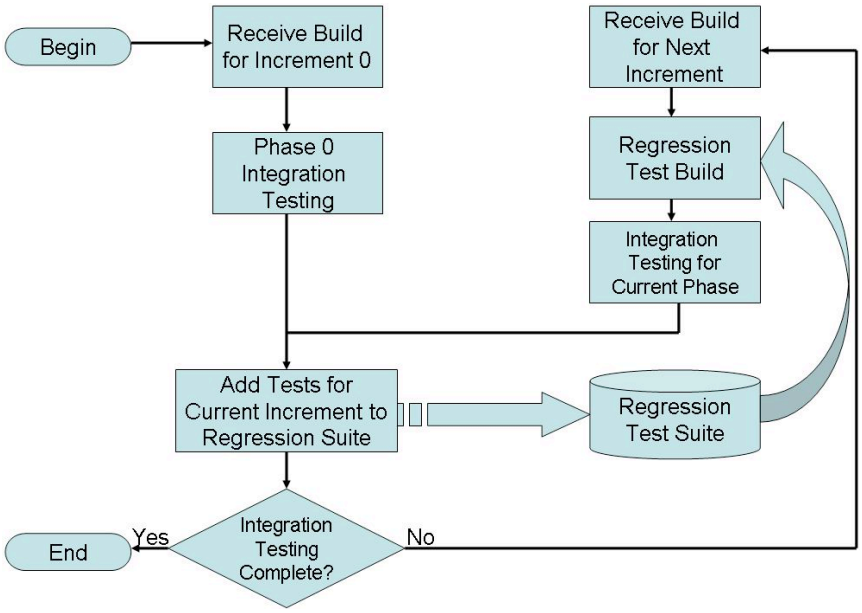


Figure 7. Integration Testing Procedure

Not shown in this figure is the fact that if the build fails to successfully execute against the regression test suite, the build will be rejected and the development organization will have to fix the errors before integration testing can continue (i.e., before the new functionality is tested). Alternatively, our team considered the ramifications of continuing to test new functionality while the broken, existing functionality was repaired by the development organization. The advantage of this approach is that the potential for concurrency exists. Specifically, our organization could continue testing the build at the same time the regression test errors were addressed by the development organization. This could result in an acceleration of the delivery date of the software through the use of concurrency. The disadvantage of this approach is that it is often more expensive because it results in re-testing. For example, suppose that a build fails to pass the regression tests. Next, suppose that our organization continues to test the build for new functionality while the regression test errors are addressed by the development organization. Once those regression test errors are resolved, the development organization will release a new build and all integration testing for the current phase (including regression testing) will need to be performed again. Thus, using this alternative methodology, rather than perform the integration testing for a given phase once, it may have to be performed multiple times. It is for this reason that this methodology was discarded.

To facilitate efficient integration testing as well as regression testing, all integration tests will be developed using the automated testing infrastructures, JUnit [6] and HTTPUnit

[6]. HTTPUnit will be used for tests that involve the presentation layer and JUnit will be used for all other tests. Thus, all tests may be executed independently of user control. The use of these unit testing programs may seem an odd choice for integration testing since they were developed initially for unit testing. However, they can be easily extended to support integration testing by simply testing at a different level. The advantage of such tools is that tests developed with them can be easily run automatically as part of a regression test unit. However, because they are low level testing tools, they are inappropriate for validation testing. Instead, manual verification of all requirements will occur during validation testing before deployment. This is appropriate given that this project is a Web site meant for use by people. Using the site as real users will is therefore essential to fully verify and validate the requirements.

Before executing a set of tests (regression or otherwise), the database will be populated with a set of data, which will be used by the tests. This will eliminate the need for each individual test to populate the database with the data it needs. There will, of course, be situations where a test will modify the content of the database in order to test something. All tests will, however, restore the system, including the database, to its original state. This is true even if the test fails to complete.

7.2.2 Change Management and Control

SIMTRIX WEB WEAVERS has standardized on the usage of WinCVS. WinCVS is a free configuration management system built on top of the well-known CVS system. It adds a graphical user interface for ease of usage. Additionally, **SIMTRIX WEB WEAVERS** has standardized on the utilization of Bugzilla for bug tracking and change control. These two tools are incorporated into a process for change management.

All artifacts generated as part of a development process are placed under configuration management. This includes not only source code, but configuration files, documentation, and even this project plan. Versioning of all artifacts occurs such that a snapshot of the system can be obtained for any point in time in the past.

All code and configuration submissions to the system must be accompanied by an entry into Bugzilla. Bugzilla is therefore used for more than defect tracking, but also for enhancements to the system. All submissions to the system must include a comment that refers to the Bugzilla change request number in a specific format. A validation script is run nightly looking for submissions that lack a change request number and developers are notified (along with their managers). This strict change management is put in place to allow for change to be monitored and to ensure product quality, especially as releases draw near.

8 Bibliography

[1] B. Boehm, A. Egyed, J. Kwan, D. Port, A. Shah, and R. Madachy, "Using the WinWin Spiral Model: A Case Study," *IEEE Computer*, Vol. 31, Iss. 7, pp. 33-34, Jul. 1998.

[2] D. Herst, E. Roman, "Model Driven Development for J2EE Utilizing a Model Driven Architecture (MDA) Approach: A Productivity Analysis", MiddlewareRESEARCH.com, 2003, retrieved 24 Nov. 2004, from http://www.middlewareresearch.com/endeavors/030721CWMDA/endeavor.jsp.

[3] D. Reifer, "Estimating Web Development Costs: There are Differences," *CrossTalk: The Journal of Defense Software Engineering*, Jun. 2002, retrieved on Nov. 24, 2004 from http://www.stsc.hill.af.mil/crosstalk/2002/06/reifer.html.

[4] D. Reifer, "Web Development: Estimating Quick-to-market Software," *IEEE Software*, Vol. 17, Iss. 6, pp. 57-64, Nov/Dec 2000.

[5] D. Reifer, "Web Objects Counting Conventions," retrieved on Nov. 24, 2004 from http://www.reifer.com/documents/WOCounting.doc. Also available at www.info@reifer.com.

[6] "HTTPUnit," 2004, retrieved on 13 Nov. 2004 from http://httpunit.sourceforge.net/index.html.

[7] "JUnit," 2004, retrieved on 13 Nov. 2004 from http://junit.sourceforge.net.

[8] L. P. Leach, "*Critical Chain Project Management Improves Project Performance*," Advanced Project Institute, Idaho Falls, ID, 1997.

[9] Object Management Group, Inc., "OMG Model Driven Architecture", 2003, retrieved Nov. 23, 2004 from http://www.omg.org/mda.

[10] "OptimalJ", CompuwareCorporation, 2004, retrieved on Nov. 24, 2004, from http://www.compuware.com/products/optimalj/default.htm.

[11] "OptimalJ, Customers in Action," CompuwareCorporation, 2004, retrieved on Nov. 24, 2004, from http://www.compuware.com/products/optimalj/1792_ENG_HTML.htm.

[12] "OptimalJ, Press & Analyst Coverage," CompuwareCorporation, 2004, retrieved on Nov. 24, 2004, from http://www.compuware.com/products/optimalj/1791_ENG_HTML.htm#PReviews.

[13] R. Pressman (2001), "*Software Engineering, A Practitioner's Approach, Fifth Edition*," McGraw-Hill, New York, New York, 2001.

[14] *"The Simpsons*," FOX.com, 2004, retrieved on Nov. 23, 2004 from http://www.thesimpsons.com/index.html.

[15] Reifer, Donald J. "Estimating Web Development Costs: There are Differences," *CrossTalk – The Journal of Defense Software Engineering*, June 2002, pp. 13-17.

[16] Reifer, Donald J. "Web Objects Counting Conventions," retrieved on Nov. 24, 2004, from http://www.reifer.com/documents/WOCounting.doc

[17] S. Bohner. Assignment Statement and Slides. CS 5984, Fall 2004.

[18] W. Royce, "*Software Project Management – A Unified Framework*," Addison-Wesley, Boston, MA, 1998.