Software Project Management Plan

Highland Basic Order Tracking System

Prepared For:	Highland Office Supply
Prepared By:	John Zoltai
	Highland Office Supply
Document ID:	BOTS-SPMP
Version:	1.2a

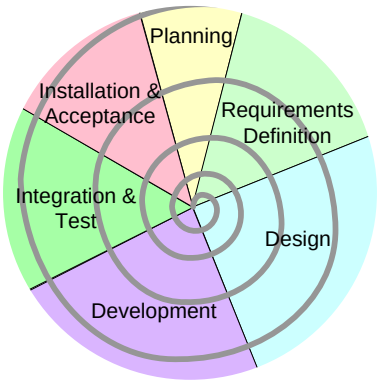


TABLE OF CONTENTS

INTRODUCTION **5**
STANDARDS 6
VISION & SCOPE **7**
BACKGROUND 7
SOLUTION OVERVIEW 8
SCOPE & LIMITATIONS 10
FEASIBILITY AND RISK ANALYSIS **12**
PROJECT SIZE 12
IMPLEMENTATION RISK 13
OPERATIONAL IMPACT 15
CRITICAL SUCCESS FACTORS 16
MANAGEMENT APPROACH **19**
PROJECT STRUCTURE 19
SOFTWARE QUALITY ASSURANCE (SQA) 20
SOFTWARE CONFIGURATION MANAGEMENT (SCM) 20
MEASUREMENT & ANALYSIS 22
PROJECT TEAM TRAINING & QUALIFICATION (PTTQ) 23
TECHNICAL APPROACH **24**
ANTICIPATED IMPLEMENTATION ARCHITECTURE 24
DOCUMENTATION DELIVERY FORMATS 25
COMMUNICATIONS FORMAT 25
PROJECT WEB SITE 25
ANALYSIS LISTINGS **26**

GLOSSARY OF PROJECT-SPECIFIC TERMS	
Glossary of Software Engineering Terms	A standard Glossary of Software Engineering Terms is maintained separately.
Glossary of Project-Specific Terms	A common Glossary of Project-Specific Terms is maintained on the project Web site.

INTRODUCTION

This document serves as the project plan for the Basic Order Tracking System (BOTS) software development effort. The content of this document is divided into four chapters: Vision & Scope, Feasibility & Risk Analysis, Management Approach, and Technical Approach.

The Vision & Scope chapter describes the conditions driving the development of BOTS. It provides an overview of the application as initially envisioned, and describes the scope and limitations of the development effort.

The Feasibility & Risk Analysis chapter addresses the issues of application complexity as well as the anticipated solution domain, database load and project size. Analysis indicates that this is a **Small** size, **Medium** risk, **Medium** impact project.

The Management Approach chapter describes the high-level structure and processes of the project as well as processes for software quality assurance and configuration management.

The Technical Approach chapter describes the anticipated development tools, implementation software, documentation delivery formats and formal communications methods for the project. Although a final determination of suitable implementation tools cannot be made until the design stage is concluded, a "best guess" estimate of tools based on high-level requirements helps to focus the processes of requirements elicitation and systems design.

Support items related to this document include BOTS Use Cases, from which high-level requirements were derived, and a high-level project schedule.

STANDARDS

The following standards were used as guides to develop this project plan. The standards were reviewed and this content tailored to the the specific needs of this project.

- ANSI/IEEE 1058: Standard for Software Project Plans
- ANSI/IEEE 730.1: Standard for Software Quality Assurance Plans
- ANSI/IEEE 828: Standard for Software Configuration Management Plans
- SEI/CMMI: CM, IPM, RSKM, PP, PPQA, PMC, and MA process areas

VISION & SCOPE

The purpose of this chapter is to provide a high-level overview of the BOTS application. This chapter describes the background, the solution overview, and the scope & limitations of this project. This information is used to define and constrain the scope of the subsequent requirements and design stages.

BACKGROUND

PROBLEM STATEMENT

Highland Office Supply has been growing steadily over the past 10 years. During this time, Highland has incurred significant expense in upgrading their order entry software every few years as their client base and order volume grows. Each time they upgrade, the new vendor has difficulty transferring the old data to the new system, training costs are significant, and the transition requires a great deal of time and effort to minimize and recover from errors incurred during the process.

Highland management has decided that it is time to manage the development of in-house software, using scalable relational database technology. This will allow Highland to benefit from owning their own code and stabilizing the user interface and processes under which they will operate in the future.

BUSINESS RISKS

Failure to complete the development of BOTS will result in continued excess expenditures by management to support upgrades of systems that are not scalable and which do not provide adequate interfaces with other applications. Lack of adequate interfaces will result in continued expenditures of time and effort by staff to duplicate or replicate data outside of the existing system.

PRINCIPAL PROJECT PARTICIPANTS

Software projects by nature include a wide variety of personnel with varying degrees of involvement as the project progresses. Rather than attempt to

document all possible project participants, this project plan identifies those personnel tasked with direct responsibilities for the success of the project.

PROJECT EXECUTIVE SPONSOR

Ted Adams is the Highland executive in charge of approving and funding this project.

PRIMARY END-USER REPRESENTATIVE (PER)

Sara Conner will act as the primary point of contact and interface to BOTS end-users. The PER will work directly with the Primary Developer Representative to identify subject matter experts, coordinate meetings, conduct reviews, and track identified issues, consolidating and coordinating the responses of all end-users into a cohesive set of communications.

PRIMARY DEVELOPER REPRESENTATIVE (PDR)

Joe Smith will act as the primary point of contact and interface to the development team. The PDR will work directly with the Primary End-user Representative to coordinate meetings, conduct reviews, and track identified issues. In addition, the PDR will be responsible for project tracking, customer communications, and insuring that defined standards for software quality assurance and configuration management are met.

QUALITY ASSURANCE (QA) REVIEWER

Jane Jones will conduct periodic reviews of the work products associated with this project as defined in the Software Quality Assurance (SQA) section of the Management chapter of this project plan. The QA reviewer is independent of the development team to ensure objective audits and reviews of the work products and processes of this software development project.

SOLUTION OVERVIEW

VISION STATEMENT

Highland management envisions a database application that will support basic order entry and shipping operations, and transmit fulfilled order data to the Highland billing and accounting systems. Although the initial development effort will be only to support core features that are critically required for successful order entry and fulfillment, the application will be developed using open, standards-based technology that facilitates integration with other systems, additional development, and significant scalability.

HIGH-LEVEL REQUIREMENTS

High-level requirements are the "goals" that the system is to attain. All requirements, their associated design elements, and the subsequent test cases are traceable to one or more of these goals. The high-level use cases from which these goals were derived are available on the project Web site. The high-level requirements for the BOTS application are as follows:

G1-BOTS
550
Client/server RDBMS
Provide a graphical user interface into a Relational Data Base Management System (RDBMS) for BOTS users. The database engine will provide the standard features expected of a RDBMS, including data transfer, storage, bulk modification of data, relationship management and access control. End users will connect to the RDBMS via a graphical user interface client, query and reporting tool, or web browser, as appropriate to the task.

G2-BOTS
125
Access control
Restrict access to BOTS data to specified sets of authorized users.

G3-BOTS
300
Customer Management
Provide the ability to enter and update customer information, including names, addresses, and demographics, as well as allow managers to view graphics summarizing customer demographics.

G4-BOTS
325
Order Management
Provide the ability to enter and update orders, process and approve submitted orders, and allow managers to view graphics and statistics about orders.

G5-BOTS
300
Product Management
Provide the ability to enter and update products and inventory levels, and allow managers to view graphics and statistics about the current inventory.

G6-BOTS
900
Core Features
Provide a uniform interface and application component set to support the following major functions in a consistent manner:

1. Restrict access specified sets of authorized users.
2. Provide the ability for end-users to query the database via pre-built as well as ad-hoc queries.
3. Provide the ability for end-users to view summary lists of query results and drill down from the list into detail data for entry, update, and review.
4. Print standard and ad-hoc reports, and through the reporting mechanism, transfer data to the Highland accounting system as well as Microsoft Excel for other uses and analyses.

SCOPE & LIMITATIONS

This section describes the scope and limitations of the project, not of the software. The scope and limitations of the software under design have been fully described in the solution overview section above.

SCOPE OF CURRENT EFFORT

This project will gather requirements, perform analyses, develop design elements and produce software artifacts for a Phase 1 implementation of BOTS, consisting of all four components implemented at a basic level, but without the graphical displays and statistics dashboards. The dashboards will be implemented as a Phase 2 effort after all basic components have been placed into production.

The goal of this project is to produce a fully functional and acceptable software product that satisfies the high-level requirements described above. The activities associated with this project are described in the associated Software Development Life Cycle document for this project.

The specific activities anticipated for each stage are identified in the high-level project schedule as each previous stage is completed. For example, when the planning stage is completed and this project plan is finalized, the high-level project schedule will contain a detailed listing of the anticipated activities for the requirements stage, but the design stage will only be identified as a single consolidated activity with an estimated duration. The software development team has found that it is very difficult to provide accurate, detailed plans for anything beyond the next stage at any point in the software development life cycle.

SCOPE OF FUTURE EFFORTS

Future efforts may include the addition of new capabilities and enhancements of existing features, as required by Highland personnel.

CONSTRAINTS & LIMITATIONS

CONSTRAINT: SPECIALIZED SOFTWARE

The software under development is intended to be special-purpose software, focused tightly on customer needs. As such, the software is not intended to serve a broad market or multiple customers, and the implementation of the software is not required to be compatible with multiple platforms or interface standards beyond those identified in the application design constraints.

LIMITATION: SOFTWARE PROTOTYPES

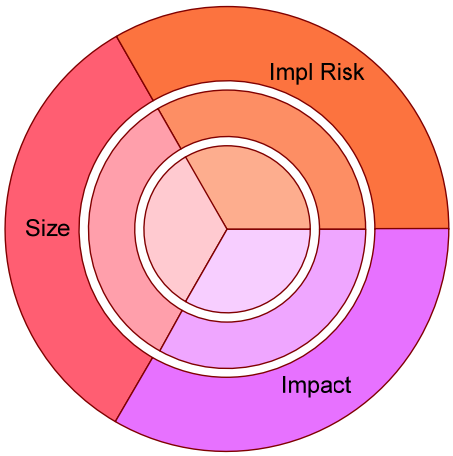
From time to time, the software development team, to clarify requirements and/or design elements, may generate mockups and prototypes of screens, reports, and processes. Although some of the prototypes may appear to be very substantial, they're generally similar to a movie set: everything looks good from the front but there's nothing in the back.

When a prototype is generated, the developer produces the minimum amount of code necessary to clarify the requirements or design elements under consideration. No effort is made to comply with coding standards, provide robust error management, or integrate with other database tables or components. As a result, it is generally more expensive to retrofit a prototype with the necessary elements to produce a production component then it is to develop the component from scratch using the final system design document.

For these reasons, prototypes are never intended for business use, and are generally crippled in one way or another to prevent them from being mistakenly used as production components by end-users.

FEASIBILITY AND RISK ANALYSIS

The feasibility of a software development effort is derived from a combination of three feasibility aspects and their associated mitigation activities. Acceptance of this project plan constitutes acknowledgement that these three aspects are appropriately mitigated to a level acceptable to Highland management.



This chapter evaluates the project size, implementation risk, and operational impact associated with this software development effort, and includes a listing of critical success factors.

PROJECT SIZE

Project size is defined as the level of effort associated with development and/or maintenance of the software. Larger software development efforts are subject to increased risk of failure due to requirements changes driven by changes in the organizational environment. Larger projects are inherently more complex, making maintenance more difficult. Project size is estimated by evaluating six project size factors. Full definitions for each of these project size factors are contained in the Glossary of Software Engineering Terms. In the table below, the estimated ranges are highlighted for each size factor.

Project Size Factors

	Small	Medium	Large
Number of Operational Data Areas	**1-8**	9-15	16+
Number of Support Data Areas	**0-10**	16-30	31+
Number of External Interfaces	**0-2**	3-5	6+
Number of Simultaneous Users	**1-10**	11-40	41+
Initial Data Load (kR = Thousand Records)	**0-10kR**	10-50kR	>50kR
Operational Transaction Load (kTpd = Thousand Transactions Per Day)	**<0.5kTpd**	0.5-3kTpd	>3kTpd

Based on the values calculated for each project size factor, this project is estimated to be a **Small** size development effort.

PROJECT SIZE MITIGATION

Mitigation of project risks associated with the size factors highlighted above include:

1. Segregation of the application into functional components with development of each component addressed through a separate SDLC.
2. Development in compliance with a formally defined architecture to enable consistent, high-reliability information exchange between this application and external systems.
3. Customization of the network infrastructure, bandwidth, and storage mechanisms to support the anticipated number of users.
4. Sizing of the application server and database engine appropriate to the anticipated data volumes.
5. Sizing of the network infrastructure, application server, and database engine appropriate to the anticipated transaction volumes.

IMPLEMENTATION RISK

Implementation risk is defined as a combination of risks that may inhibit successful development and deployment of the software. The higher the risk level, the greater the chance of project failure unless the identified risk factors are mitigated. Implementation risk is estimated by selecting the most appropriate attributes for each of six risk factors. Full definitions for each of these risk factors are contained in a standard Glossary of Software Engineering Terms. In the table below, the selected attributes are highlighted for each risk factor.

Implementation Risk Factors

	Low Risk	Medium Risk	High Risk
Implementation Technology	Well known	**Some new**	Cutting edge
External Interface Complexity	**All simple**	Minority complex	Majority complex
Developer Skills / Resources	Readily available	**Some contention**	Need to be acquired
Customer Resources (UC = Use Case)	>1 SME per UC available	**1 SME per UC available**	Outside SME(s) used
Business Process Maturity	**Standard, stable**	Non-standard, stable	Automating new/unusual process(es)
Business Process Complexity	All simple	**Minority complex**	Majority complex

Based on the attributes selected for each project risk factor, this project is estimated to be a **Medium** risk development effort.

IMPLEMENTATION RISK MITIGATION

Mitigation of the implementation risks highlighted above include:

1. Contention for developer resources will be controlled by extending the project schedule where appropriate, or suspending the project until the necessary talent is available. Certain segments of the project will be timed to coincide with the availability of specific developers. The Orders Component and Product Component tasks and associated subtasks defined in the project schedule address this mitigation effort.
2. Contention for Subject Matter Expert (SME) availability will be controlled by extending the project schedule where appropriate, or suspending the project until the necessary talent is available. Certain segments of the project will be timed to coincide with the availability of specific SMEs. The Orders Component and Product Component task and associated subtasks defined in the project schedule address this mitigation effort.
3. Business processes impose risks of error commensurate with their complexity. To mitigate risks associated with complexity, processes will be simplified where possible. Processes that are still considered to be complex will be subjected to scenario analysis resulting in a set of test scripts and known results in terms of application outputs and internal data. These regression tests will be validated by hand and then automatically run each time the associated components are modified. The Orders Component task and associated subtasks defined in the project schedule address this mitigation effort.

OPERATIONAL IMPACT

Operational impact is defined as the impact on the organization of a worst-case failure of the software during production. Failures of higher impact applications can cause significant damage to the organization.

FAILURE SCENARIOS

The worst-case failures of this application consist of the following failures and results:

Failure Scenarios

Failure	Result
Incomplete features.	Slowdown in operations.
Errors in calculations or reports.	Lost sales, lower customer confidence. Additional costs associated with recovery.
Errors during data migration.	Incomplete history, errors in operations.
Access by unauthorized personnel	Release of customer name and address data, but no exposure of customer credit history or credit card information, as this is not maintained in BOTS.

OPERATIONAL IMPACT FACTORS

Operational impact is estimated by selecting the most appropriate attributes for each of six factors. In the table below, the selected attributes are highlighted for each impact factor.

Project Impact Factors

Category	Low Impact	Medium Impact	High Impact
Security	Release of non-sensitive information.	**Release of proprietary or high intellectual property information.**	Release of trade secrets, customer/employee/vendor data, or strategic planning information.
Operations	Continued operations with minor adaptations.	**Restricted operations using manual, paper-based, or alternative business processes.**	Cessation of operations with subsequent restart effort.
Finance	**Recovery costs do not significantly alter the budget of the operating business unit.**	Recovery costs result in significant alteration of the budget of the operating business unit.	Recovery costs result in significant alteration of the budget of the *parent* business unit.
Liability	**Zero, minimal, or acceptable liability is incurred.**	Significant liability resolved by settlement.	Extensive liability resolved by mediation or court action.
Public Image	Zero, temporary, or	**Significant degradation**	Significant degradation of

	acceptable degradation of public image.	**of local or regional public image.**	national public image.
Competitive Edge	Zero to minimal negative impact on competitive edge.	**Significant, but temporary negative impact.**	Substantial loss of competitive edge.

Based on the attributes selected for each project impact factor, this project is estimated to have a **Medium** impact on the organization in the event of a worst-case failure.

PROJECT IMPACT MITIGATION

Mitigation of project risks associated with the impact factors highlighted above include:

1. Access control mechanisms will be established to restrict use of the application to authorized users.
2. The application will encrypt data transferred between the database/application server and the client system.
3. Physical access controls will be established for the database server to prevent unauthorized retrieval, copying, or destruction of operational data.
4. Database and application services will be provided by fault-tolerant, clustered, or redundant servers and file storage mechanisms.
5. Database backups will be conducted daily and transported off-site periodically in accordance with the organization's disaster recovery requirements.
6. The application will be developed using a design controlled methodology, where coding is accomplished after and in strict compliance with the application design.
7. The application will be tested and placed into production via separate, identically configured application and database servers.
8. The application will be tested for functional accuracy, data integrity, data capacity, and transaction load capacity.
9. Changes to application requirements and design elements will be implemented only after verification and validation of each change and the subsequent approval of the project Configuration Control Board (CCB).
10. Development will be conducted in such a manner as to enable retrieval of appropriate source items to enable re-generation of any prior release of the application.

CRITICAL SUCCESS FACTORS

Critical success factors consist of assumptions and dependencies that are generally outside the control of the software development team. If an assumption proves to be false, or a dependency proves to be unreliable, the success of the project is placed at significant risk.

ASSUMPTION: IMPLEMENTATION PLATFORM STABILITY

It is assumed that, once specified, the implementation platforms for client and server partitions of BOTS will not be subject to change in the foreseeable future. This is especially important in terms of the client platform; adding a requirement to support a substantially different client operating system at a later date will threaten the viability of the project.

ASSUMPTION: INFRASTRUCTURE STABILITY

It is assumed that, once specified, the network operating system and the respective operating systems of the fileservers, database servers and Web servers upon which BOTS will be dependent will not be subject to significant change in the foreseeable future. Portions of BOTS will be dependent upon networking protocols, operating system calling parameters and native language support provided by the currently implemented operating systems for the various servers and network. Changes in any of these software infrastructure elements will require significant modification of BOTS and will threaten the viability of the project.

ASSUMPTION: PROJECT PRIORITY

It is assumed that Highland management will encourage those personnel assigned to the project to place a high priority on tasks associated with the project. Inadequate customer personnel participation is one of the leading causes of software project failures.

ASSUMPTION: SOFTWARE QUALITY ASSURANCE (SQA) PROCESS

The SQA process essentially controls the review and approval cycle for project documents and software artifacts. SQA activities will be initiated by the development team in accordance with the process defined in the Management Approach: SQA portion of this document.

ASSUMPTION: SOFTWARE CONFIGURATION MANAGEMENT (SCM) PROCESS

Configuration management is often referred to as change management. The SCM process essentially controls document and software artifact classification, storage, revision and retrieval. SCM activities will be initiated by the development team in accordance with the process defined in the Management Approach: SCM portion of this document.

DEPENDENCY: SUBJECT MATTER EXPERTS

The primary end-user representative will identify a Subject Matter Expert (SME) for each major component of the application. SMEs will act as primary sources of requirements and will be members of the core review team charged with reviewing documents, providing feedback, and participating in design activities for software components in their domain of expertise.

DEPENDENCY: STABILITY OF END-USER REPRESENTATION

A software development project is highly dependent upon the stability of the population of customer personnel assigned to the project. A constantly shifting set of customer personnel leads to a constantly shifting set of requirements and design elements; this is another leading cause of software project failures.

DEPENDENCY: RAPID REVIEW CYCLE

People work best when information is fresh in their minds. Therefore, it is essential that documentation generated by the software development team be reviewed as quickly as possible, while the discussions that resulted in the documentation are fresh in the minds of customer personnel. A lengthy review cycle typically leads to a project becoming stagnant, and is one of the leading causes of project abandonment.

DEPENDENCY: EXTERNAL SYSTEMS INTERFACE SPECIFICATION

The BOTS development team will need to have access to detailed specifications for the exchange of information between BOTS and other external systems. A "trial and error" approach here will place significant risk on the success of the project.

MANAGEMENT APPROACH

The management of a software project includes five focus areas. The first area focuses on defining the structure, or stages and sequence of activities, of the project. The second area focuses on insuring the development and delivery of high-quality work products through the application of a Software Quality Assurance (SQA) process. The third area focuses on the coordination of changes to current work products through the application of Software Configuration Management (SCM) processes. The fourth area focuses on the acquisition and reporting of project metrics. The fifth area focuses on project team training and qualification.

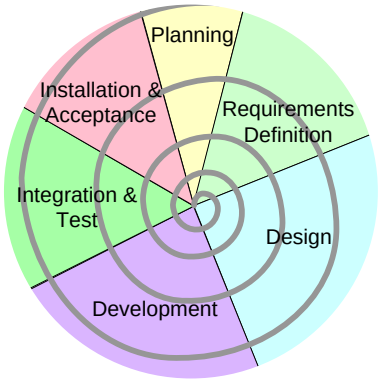
PROJECT STRUCTURE

This section includes an overview of and links to documents describing the software development life cycle, the software quality assurance process, the software configuration management process, and the project team training and qualification requirements for this project.

The project schedule presents a comprehensive listing of the tasks and activities associated with the planning and requirements gathering stages for this project. Subsequent stages are described only at a very high level. In software development, it is very difficult to identify detailed tasks and activities associated with a life cycle stage that is more than one stage beyond the current life cycle stage.

SOFTWARE DEVELOPMENT LIFECYCLE

This development effort utilizes a six stage life cycle:



The Software Development Life-Cycle (SDLC) shown above utilizes the iterative lifecycle with modifications for rapid prototyping. The complete software development lifecycle for this project is described in a separate document, available at:

http://www.shellmethod.com/refs/SDLC.pdf.

Please refer to this document for a description of the structure, inputs to and outputs from each of the stages shown above.

SOFTWARE QUALITY ASSURANCE (SQA)

The software quality assurance process implements periodic reviews of current draft deliverables and selected work products in each stage of development. The primary objective of the SQA process is to ensure the production of high-quality work products according to stated requirements and established standards. SQA incorporates a formal process for evaluating and documenting the quality of the work products produced during each stage of the software development lifecycle.

The complete SQA process for this project is described in a separate Software Quality Assurance Plan. Please refer to this document for a description of the structure and timing of SQA activities for this project.

SOFTWARE CONFIGURATION MANAGEMENT (SCM)

The software configuration management process controls the identification and retention of all the various artifacts (documents, source code, executables, e-mail, etc.) generated during the software development lifecycle. The primary objective of the SCM process is to coordinate the use of software artifacts among the

project participants, making sure everyone is working with the same versions of various artifacts (change control) and making sure that nothing gets lost (retention control).

The complete SCM process for this project is described in a separate Software Configuration Management Plan. Please refer to this document for a description of the structure and timing of SCM activities for this project.

CONFIGURATION CONTROL BOARD (CCB)

The core members of the CCB are the project executive sponsor and all personnel with the role of project-level or component-level PER or PDR. Additional personnel assigned to the CCB for this project include Joe Bolton, Sue Mastiff, and Gene Speed.

CONFIGURATION ITEMS

Project artifacts that are uniquely identified and placed under version control are known as Configuration Items. In this project, configuration items fall into four general classes:

1. Evolving items, such as documents, which are subject to one or more revisions and new releases during the SDLC.
2. Source items, generally source code and object files used to build a production software application, which are generally numerous and frequently changing.
3. Support items, such as operating systems, of which the project requires certain versions for successful operation.
4. Archive items, such as SQA review forms which generally support decisions made during the SDLC, and are stored in electronic format for future reference.

CONFIGURATION TOOLS

Each class of configuration items is handled by a specific tool set and/or storage facility:

1. Evolving items are managed via a Subversion online repository.
2. Stage builds for evolving items are managed via Subversion branches.
3. Source items are managed via the project Subversion repository.
4. Stage builds for source items are managed via Subversion branches.
5. Support items are inventoried physically and stored in Highland's office files.
6. Archive items are stored in the working file share for this project.

ISSUE MANAGEMENT

Enhancement requests and defect reports are known as Issues. Issue validation and management is described in the SCMP. The implementing tool used for issue storage and retrieval is Microsoft Excel.

MEASUREMENT & ANALYSIS

Project metrics are gathered during each iteration of the SDLC to support project monitoring and control. These metrics are useful for cross-project analysis and organizational process improvement activities.

OBJECTIVES & MEASURES

This project gathers, stores, and reports measurement data to satisfy the following objectives:

PROJECT EFFORT & SCHEDULE

- Objective: The project identifies and tracks estimated and actual effort and schedule.
- Measures: Effort and schedule for each stage of an SDLC iteration is first estimated, and then tracked against actuals using MS Project. These estimates and actuals are presented to the project executive sponsor during each stage exit, and maintained in the project archive.
    1. Estimated effort by iteration and stage
    2. Actual effort by iteration and stage
    3. Estimated schedule by iteration and stage
    4. Actual schedule by iteration and stage

PROJECT EXPENDITURES

- Objective: The project tracks budgeted and actual expenditures for labor and other direct costs.
- Measures: The total available funding for the project is monitored during each iteration. Labor costs for each person charging against the project are maintained. Other direct costs against the project are tracked. The total costs for labor and other direct costs as well as remaining funding are reported to the project executive sponsor at the conclusion of each iteration. Funding levels and charges are maintained in the project archive.
    1. Total project funding
    2. Labor charges by iteration
    3. Other charges by iteration

CONFIGURATION CHANGES

- Objective: The project identifies and tracks configuration change issues for production releases.

- Measures: For each iteration of the SDLC, the project tracks the total number of enhancement issues and defects. Defects reported within 30 days of the release of an iteration are identified separately. Additional metrics include the average age of an issue (length of time between validation and closure), as well as the average issue priority. These measures are maintained in the project archive and periodically reported to the CCB.
    1. Enhancement issue count by iteration
    2. Defect issue count by iteration
    3. Defect issues within 30 days of iteration release
    4. Average issue age
    5. Average issue priority

STAGE EXECUTION QUALITY

- Objective: The project tracks the quality of execution for each stage of an SDLC iteration.
- Measures: For stages with only documentation-level deliverables, the project tracks the number of non-concurring reviews for each deliverable. For stages with testing activities, the also project tracks the number of Test Incident Reports (TIRs). Finally, the project tracks Stage Reversions, which are changes in requirements or design elements that force the development team to halt a currently active stage and re-execute work in a previously closed stage. These measures are maintained in the project archive and periodically reported to the CCB.
    1. Non-concurring reviews by iteration and stage
    2. Test Incident Reports by iteration
    3. Stage reversions by iteration

PROJECT TEAM TRAINING & QUALIFICATION (PTTQ)

The project team consists of developers, end users, and quality assurance review personnel. Each of these roles inherit a specific set of responsibilities and impose certain minimum training and qualification requirements.

The training and qualification requirements for this project are described in a separate Project Team Training & Qualification (PTTQ) plan. Please refer to this document for a description of the roles, responsibilities, and qualification requirements for this project.

TECHNICAL APPROACH

This chapter describes the anticipated development tools, implementation software, documentation delivery formats and formal communications methods for the project. Although a final determination of suitable implementation tools cannot be made until the design stage is concluded, a "best guess" estimate of tools based on high-level requirements helps to focus the processes of requirements elicitation and systems design.

ANTICIPATED IMPLEMENTATION ARCHITECTURE

The anticipated BOTS system will use a three-tier application approach, consisting of a database server, an application server, and a browser-based web client. The end user will access the application via a web home page, which provides top-level access to the major features of the application. Individual functions will be provided by normal html pages, active server pages, and visual basic applets, as appropriate. These application components will be stored on the central application server, and delivered to the browser as required.

ANTICIPATED DEVELOPMENT TOOLS

The anticipated development toolset consists of Microsoft SQL server for back-end database storage and retrieval, Microsoft Internet Information Server (IIS) for web and application serving, Seagate Crystal Reports for *ad hoc* and standardized report generation, and Microsoft Internet Explorer for client tasks. Code development will be done via Microsoft Visual Basic. Online help will be delivered in WebHelp format.

The final application will be delivered as a pre-configured SQL Server database and an IIS compatible set of web pages, active server pages, and visual basic applets.

Documentation Delivery Formats

All documentation will be delivered as Adobe Acrobat Portable Document Format (pdf) files, with the exception of the online help system. This will allow all participants to review and annotate documents during the review cycle, as well as fill out online forms and apply digital signatures.

Communications Format

Informal communications between project participants can take place in person, via telephone, or via electronic mail. Minutes from face-to-face and web teleconference meetings may be distributed via e-mail for review and clarification if necessary. For one-on-one meetings or telephone calls between client and development team personnel, the member of the development team may produce an e-mail summary of the conversation and send it to the other party for clarification if necessary.

Formal communications consist of stage deliverable review forms and memoranda of acceptance. These are stored in the project archive.

Project Web site

The development team maintains a Web site for this project. The purpose of the project Web site is to act as a central information repository for the project, allowing all participants direct access to the most recent versions of project artifacts. In addition, the home page contains links to current prototypes, documents under development, and other project support items.

Note that this project site is open to all, but certain documents may be restricted to current project participants via a confidential project password. Unauthorized personnel do not have access to this password. Authorized personnel are identified by the PER and PDR.

ANALYSIS LISTINGS

The following listings are automatically maintained by the word processor for this document. It is of use only to systems analysts when verifying requirements traceability across documents and tracking project size.

G1-BOTS
G2-BOTS
G3-BOTS
G4-BOTS
G5-BOTS
G6-BOTS

550
125
325
300
900