Software Development Plan

for

QConBridge

Version 1.1 - Final

Prepared by Richard Brice

WSDOT Bridge and Structures Office

May 9, 2000

Revision History

Version	Author	Description	Date Completed
1.0 - Draft	RAB	Created Document	1/26/2000
1.0 - Final	RAB	Baselined Document	3/14/2000
1.1 - Final	RAB	Added set of hand calculations as a deliverable. Updated Evolution of Plan table with more realistic dates.	5/09/2000

Preface

This document describes how this software development project will be conducted. It maps a course for the project. Committing this plan to writing allows all of the stakeholders to refer to the plan throughout the project. This development plan is a living document and is updated at the end of each stage or phase of development. This plan includes schedules, estimates, and milestones.

Table of Contents

1. Introduction

1.1. Project Overview

The objective of this project is to enhance the LRFD design capability of the Bridge and Structures Office by updating the QConBridge tool to satisfy current needs.

1.2. Project Deliverables

The deliverables for this project are:

Planning Documents

- Software Development Plan (this document)
- Vision and Scope Document
- Software Requirements Specification
- User Interface Design and Prototype
- Software Architecture Document
- Systems Manual which outlines build procedures and other development-related processes.
- Example hand calculations for the complete analysis of one 3-span precast girder bridge.

Software and Related Products

- Source Code
- Executable Code
- End User Documentation including Tutorial, User's Guide and Theoretical Manual
- Web-site for on-line resources, FAQ's, and latest product information

Distribution

- Installation Program(s)
- Web-site for product delivery

Training

- Training Materials
- Delivery of training session(s) for the Bridge Design Sections and department consultants

1.3. Evolution of the Software Development Plan

The Software Development Plan is a living document. While the Revision History at the beginning of this document describes what has been done to this document, the table below lists what is expected to be done. As the QConBridge project proceeds, this document will be updated to capture the development plans for each major milestone.

It is assumed that the project deliverables and organization will be more or less constant. The majority of revisions to this document will be in Section 4.

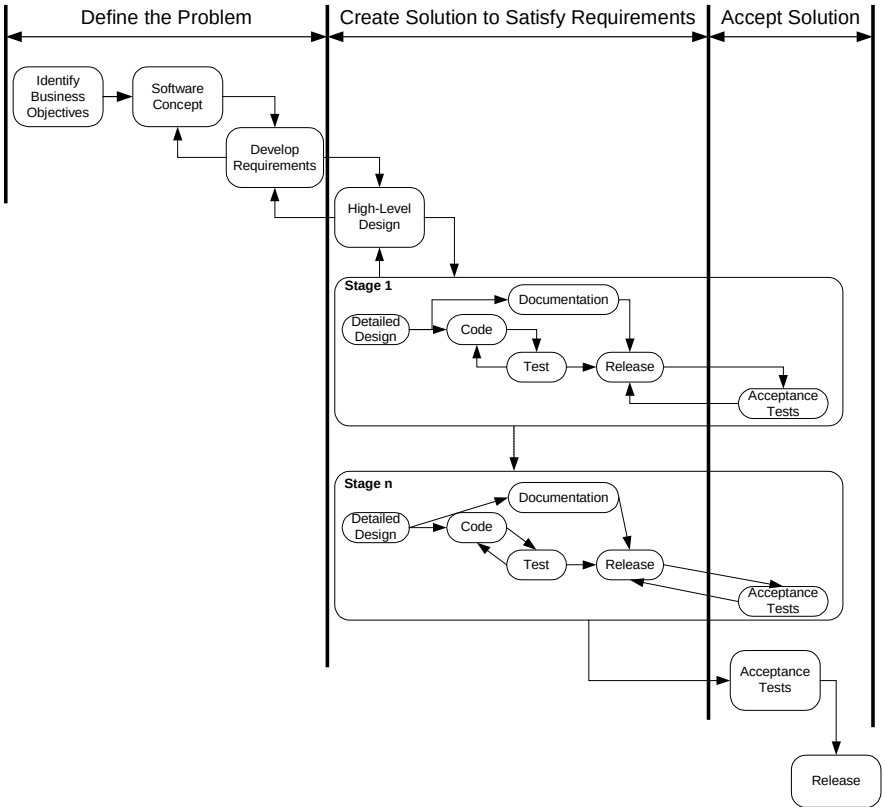
Version	Author	Description	Date Expected
1.2	RAB	Update major milestones based on requirements and high-level design Add schedule for development stages	6/25/2000
1.x	RAB	Add detailed development plans for each stage, including work products, testing, documentation, and delivery	

2. Project Organization

2.1. Process Model

On recent development projects, an iterative approach has worked very well. Using a *Release Early/Release Often* approach in the PGSuper project paid many dividends. Stakeholders were engaged throughout the implementation process and provided valuable feedback that shaped the ultimate outcome of the project.

However, a crucial mistake was made in the original QConBridge and PGSuper projects and we intend not to repeat it on this project. The mistake was lack of customer ownership in the product. The consequence of this was failing to meet the needs of the user and costly reworking of the software.



The development process that will be used for this project stresses early involvement by stakeholders and customers coupled with ongoing communications and reviews. The figure to the left provides a high-level view of the development process.

The process begins with the customers, stakeholders, and software developers working together to identify and document the business objectives of the project and to develop a concept for the software product. This information is captured in the Vision and Scope Document. Next, we develop the specific requirements for this project. The requirements include the business and functional requirements as well as the quality attributes of the system. This information is captured in the Software Requirements Specification. After the appropriate analysis and reviews the requirements are "signed off"

and baselined. From this point forward, changes to the requirements are handled with the change control process described in Section 3. These activities define what the problem is, why we need to solve it, and what the required solution must do in order for it to be beneficial.

Next, the development team gets to work on designing a solution that will satisfy the stated requirements. The high-level design consists of the User Interface Design, system architecture design, and development of a Prototype to validate the designs. The User Interface Design and Prototype convey the outwardly observable characteristics of the software. It depicts what the user will see and how they will interact with the system. The system architecture depicts the internal structure of the software and identifies the major components and how they will interact. Before detailed design of the software implementation proceeds, the UI Design will be reviewed by the customer, "signed off", and baselined. The system architecture will also be baselined.

With an overall plan in place, the first stage of work is examined in detail. A detailed design is created, the software is implemented, tested, system and user documentation is created, and the software is released to the customer. The customer provides a critical review of the work products for this stage. If there are problems, the work products are sent back to the developers for re-work.

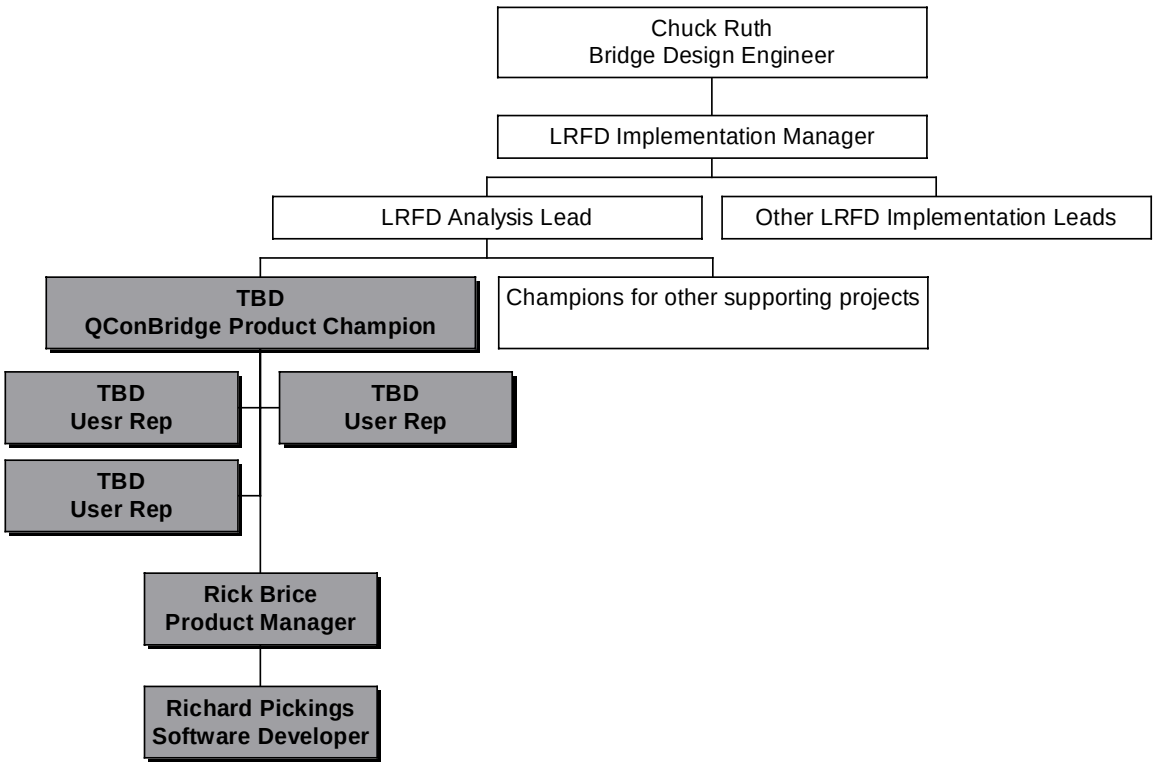
This design, code, test, document, release, acceptance test cycle is carried out for each stage of development.

After all of the stages are complete, the development team finalizes the software product including documentation, installation programs, Internet content, etc, and delivers the software to the customer. The customer then "buys off" on the product by performing their final acceptance tests.

2.2. Organizational Structure

Ideally, the QConBridge project is a supporting activity of an effort to develop the LRFD design capabilities of the Bridge Office. The organizational structure of the QConBridge project is depicted on the org chart below with the gray boxes. This organizational structure is placed in a larger, but fictitious org chart for an LRFD implementation effort. The intent of presenting the organization of the QConBridge

Ideal LRFD Implementation Group with QConBridge Project Team



project this way is to reinforce the notion that the Bridge Design Sections are the immediate customers of this project and Bridge Office Management is the primary stakeholder. It also serves to remind us that this project is being undertaken to support the business objectives of the Bridge and Structures Office.

The Product Champion is supported by two or three other designers who serve as representitives of the users. The job of the User Representitive is to assist in developing the vision, scope, and requirements of the QConBridge project. These individuals are also responsible for verifying that the delivered product satisfies the stated requirements and meets their needs. The Product Champion has the additional responsibility of keeping the Product Manager informed of changes to office practice or policy.

The Product Manager is charged with delivering the product to the customer(s). The Product Manager must coordinate all aspects of the software development including development of requirements, user interface design, system architecture, detailed software design, implementation, testing, documentation, installation, and deployment. He is also responsible for coordinating information between the development group and the design office. This insures that the product delivered integrates with the business objectives of the office. For example, he is responsible for insuring that last-minute changes in the BDM and standard plans are correctly implemented in the software.

2.3. Project Responsibilities

The table below identifies the major activities that must be carried out to ensure a successful project. The person(s) responsible for each activity is identified in the table as well.

Activity	Persons responsible
Overall Project Coordination	Brice and Product Champion
Requirements Development	Brice and Product Champion
Software Design and Implementation	Brice
End-User Documentation	Brice
Software Quality Assurance	Brice
User Acceptance Testing	Product Champion
Installation Programs and Deployment	Brice
Training	Brice

3. Project Management

3.1. Staffing Plan

As the human resources for this project are rather limited, the staffing plan is a bit informal. Given the current budget environment (post-I695) the staff of software developers is fixed. Rick Brice will serve as the Product Manager and will be assisted by Richard Pickings of BridgeSight Software. The staffing for this project from the Bridge Design Sections is more flexible. At minimum, in addition to the two software developers, we will need three core User Representatives that will be assigned to the project for its duration. One of these individuals will assume the role of Product Champion.

During the requirements development phase of this project we will need to bring in approximately three more user representatives. Their job will be to assist in the preparation, analysis, and review of the Software Requirements Specification. After the requirements are signed off and baselined, these additional user representatives will be disbanded.

Through out the project, and especially towards the end, the User Representatives should perform some acceptance testing to verify that the delivered product satisfies the stated requirements and meets their needs. As it is the responsibility of the Product Champion to conduct the acceptance testing, the staffing plan for that activity will be determined at a later date.

3.2. Risk Management

Risk Management is the process of identifying and eliminating or mitigating risks that threaten the success of a project. Once something goes wrong, it is no longer a risk, but rather a problem. It is better to avoid problems (or at least minimize their impact) than to have them catch you by surprise.

Since this project is small, and we have limited resources, a formal risk management plan will not be developed. However, all members of this project will share the responsibility of actively looking for risks.

3.3. Managing Change

It is virtually impossible to hit a moving target. One of the keys to success for any project is controlling changes. Change is inevitable. There is no way to avoid it and still satisfy the needs of the customer. Through out this project we will practice effective change management techniques. This section highlights the change control procedures that we will be using.

1. Initial development work for a work product (such as the SRS) is performed without change control coming into play. During this period, changes can be made freely to the work product.
2. The initial work product is subjected to technical review, which determines whether initial development work on it can be declared complete.
3. When initial development is complete, it is baselined and placed into a revision control system (our system is Microsoft Visual SourceSafe).
4. Further changes to the work product are treated systematically:
    a) Changes are proposed via Change Proposals. A Change Proposal describes the work product in question, the proposed change, and the impact of the change from the point of view of the party requesting the change.
    b) The development team reviews the change proposal to assess the cost and benefits of the proposal.
    c) The change request is either accepted or rejected. If accepted, the development team establishes a priority for the new work.

4. Work Packages and Schedule

Prior to establishing the scope of the project and developing detailed requirements, it is difficult to identify work packages and tasks. For the first version of this document, the high-level work packages will be identified. As the scope and requirements become clear, the work packages will be revised and given more detail.

4.1. Work Packages

Work Package	**Description**	**Tasks**
Vision and Scope Document	Guiding document that describes the long-term vision of the software product and the scope of	Validate business objectives Draft document

Work Package	Description	Tasks
	this phase of development.	Agree on vision and scope Define success for this project and identify success factors
Software Requirements Specification	A detailed description of what the software is supposed to do	Elicit requirements from users by brainstorming, interviews, etc Develop use cases Identify quality attributes Identify constraints Draft functional requirements Develop User Interface Design and Prototype Negotiate with customer Sign-off on requirements Baseline Requirements
Software Architecture	A high-level plan that depicts the major software components and how they interact	Choose technologies Develop system design document. Trace architectural features back to requirements Develop prototype of system architecture to demonstrate worthiness of design
Implement Stage 1	**TBD**	**TBD**
Implement Stage n	**TBD**	**TBD**
Deliver to customer	**TBD**	**TBD**
Customer Acceptance Testing	**TBD**	**TBD**

4.2. Dependencies

Before any implementation can begin, the software requirements and the software architecture must be developed and baselined.

4.3. Schedule

TBD

5. Glossary

Acceptance Testing	Tests performed by the customer to verify that the software satisfies the stated requirements and functions properly
Baseline	The original version of a work product that serves as the basis for all future development work. Once placed under change control, it can be changed only through the systematic change control process.
Sign-Off	A ritual of signing the software requirements specification to establish a baseline. By signing off you assert that "I agree that this document represents our best understanding of the software requirements for this project today. Future changes in this baseline can be made through the project's defined change process. I realize that approved changes might require us to renegotiate the costs, resources, and schedule commitments for this project".
Software Requirements Specification	The document that contains statements of what the software is supposed to do. Requirements can be characterized as business (software will reduce design costs by $100K per year), functional (software shall compute reactions at interior piers due to self-weight dead load), quality attributes (there are non-functional requirements like ease of use, flexibility, and reliability).
User Interface Design	A document that details the various elements of the user interface including menus, dialogs, and views. This document also discusses how a user would perform key tasks.
User Interface Prototype	A mock-up of software under development that is created for the purpose of eliciting user feedback about the software's intended functionality and look and feel.
Version Control Software	Software tool that acts as a librarian for work products. Once checked into the library, documents cannot be modified until check out. The librarian also prevents the same document from being checked out at the same time by two different people.