Split

2260 Hayward
Ann Arbor, MI 48109

October 23, 2009

To: Rod Johnson, Instructor, University of Michigan

From: Split-Dev
Jim Brusstar, Developer
Michael Lee, Developer
Ben Montgomery, Developer
Robert Steen, Developer

Date: October 23, 2009

Subject: Software Requirements Specification

CC: Elliot Solloway, Instructor, University of Michigan

Version

Version	Date and Time	Sign Off	Changes Made
0.1	5 Oct 2009 4:30pm	mzlee, rdsteen	Initial draft - overview, project management
0.2	8 Oct 2009 4:30 pm	mzlee, rdsteen	Condensed project definition, requirements
0.3	9 Oct 2009 3:30 pm	benmonty, mzlee, jimbru	Added non functional requirements
0.4	11 Oct 2009 7:00 pm	mzlee	Fixed formatting and added project management
0.5	11 Oct 2009 9:00 pm	benmonty	Corrected wording, removed underlines, added comments
1.0	12 Oct 2009 12:40 pm	mzlee	Merged conflicting project definition paragraph
1.1	21 Oct 2009 6:30 pm	mzlee	Start port to LaTeX
1.2	23 Oct 2009 3:00 am	mzlee	Finished port to LaTeX
1.3	23 Oct 2009 3:00 pm	mzlee, rdsteen	Added functional requirements

Overview

Today, most people use tabbed web browsers to access information online. Using tabs increases efficiency and allows a user to view web content more effectively; however, the power of tabbed browsing has reached a limit. The amount of information amassed in a browsing session has become too much for a user to easily control. We are proposing a new style of browsing that allows users to crop their browser windows into exactly what they want to see. This document defines in more detail what our project is and how we will accomplish it. The project definition provides background about the project, objectives, and stakeholders; the technical issues cover an analysis of our user base, stricter definitions of our requirements, and a list of open issues in the form of risks and mitigations.

1 Project Definition

Web browsers have been an ever-evolving form of communication and information distribution. They have evolved from the old days of AOL, with the single box and cascading screens to Internet Explorer, which allowed for many windows in a desktop and now to Firefox, with tabbed browsing. We are now reaching the point of information saturation where none of these models is efficient enough. The biggest problem is that it is not the user who really decides what he or she wants to see; it's the content provider. You may be able choose to look at whatever pages you want, but do you do not have control over the ads, flash, and other distractions. We will create a platform that allows users to decide what they want to see and how they want to see it.

The objective of this project is to create a proof of concept system to demonstrate our technology and its usefulness in the web browser space. It is not to try to actively compete with other web browsers, that market is already saturated. Instead, we wish to show innovation so that we may be able to patent and later develop a better business model. The primary stakeholders of this project are our team members as we will be investing the time, energy, and work into the project. Additionally, the end users will have some stake in our continued development and innovation after our first alpha release. Finally, the University will have a marginal stake in our success as our success leads to furthering the reputation of the University of Michigan.

2 Technical Issues

2.1 User Analysis

Power Users
Power users are our primary demographic. These are the people who want to get as much out of their time as possible when it comes to using a web browser. They will be the people who often want to reorganize their data; the ones who will want a history of clips and positions. They will be our strongest demographic and will be the set of people who will evangelize our product.

Casual Users
Casual users want to just have a web browser and may occasionally use the added functionality of being able to split a page. Usually this person will be introduced to the product by a friend (and power user) who will likely set up defaults. The casual user's experience will entail the use of the defaults to check back to a set of initial pages.

2.2 Requirements Definition

This requirements enumeration has been divided into different groups. The numbering scheme uses the following group labels, and all requirements will fit into one of these categories.

Type of Requirement	Label	Description
User	USR	Requirements affecting how the program appears to the user
Implementation	IMP	Requirements that provide the programs standard functionality
Performance	PERF	Requirements that affect the programs performance

Number: IMP-1 **Name:** Render Mask
Description: Technical Requirement. The browser must be able to define a render mask for clipping purposes.
Verification Method: Test cases will be used to ensure that a render mask can be defined.

Number: IMP-2 **Name:** Incorporate WebKit
Description: Technical Requirement. Incorporate open source technologies to help our development process and not re-implement a widely available technology.
Verification Method: Code base must rely on WebKit.

Number: IMP-3 **Name:** Adhere to HTML Standards
Description: Technical Requirement. We must adhere to web standards in our render and display. We want to not fragment the web by introducing new HTML tags.
Verification Method: Self enforced. We depend on WebKit to adhere to web standards and will self enforce ourselves to not add new structures.

Number: IMP-4 **Name:** Open a Browser Window
Description: Functional Requirement. Users will be able to open a new default browser window.
Verification Method: Test a mechanism that opens a new browser window.

Number: IMP-5 **Name:** Navigate to a Web Page

Description: Functional Requirement. Users will be able to navigate from one page to another.

Verification Method: Navigate the web browser through a sequence of pages.

Number: IMP-6 **Name:** Basic Web Navigation

Description: Functional Requirement. User should be able to open a new window and navigate to a web page.

Verification Method: Navigate between various webpages and follow a string of test links.

Number: IMP-7 **Name:** Enter Clipping Mode

Description: Functional Requirement. When viewing a default browser window, users will be able to enter clipping mode. Here they will be able to select areas of content which they wish to view.

Verification Method: Have a mechanism that allows entry into clipping mode. Test to make sure selection works.

Number: IMP-8 **Name:** Exit Clipping Mode

Description: Functional Requirement. After making selections in clipping mode, user will be able to hide the rest of the content.

Verification Method: Test to make sure when exiting clipping mode that the visible area is the area selected and the rest is masked.

Number: IMP-9 **Name:** Clip Object Control

Description: Functional Requirement. When a clip is selected, users should be able to maximize to the full page, minimize, close, and move the crop.

Verification Method: Test user inputs to verify functionality exists. Create a set of test cases to check for corner cases.

Number: IMP-10 **Name:** Advanced Web Navigation

Description: Functional Requirement. When viewing a full page, users will be able to navigate backwards and forwards within the sequence of sites they have visited.

Verification Method: Navigate through different webpages and verify that history is preserved.

Number: USR-1 **Name:** Usable Interface

Description: Non-functional Requirement. The interface provided for the web browser must be considered simple to use.

Verification Method: A Usability study will be performed, where an average score of 7 out of 10 is required for this requirement to be met.

Number: USR-2 **Name:** Render Mask Interface

Description: Non-functional Requirement. The browser must have a user-accessible interface for defining a render mask.

Verification Method: A Usability study will be performed, where an average score of 7 out of 10 is required for this requirement to be met.

Number: PERF-1 **Name:** Availability

Description: Non-functional Requirement. The software must be available to users for download.

Verification Method: The software will be able to be downloaded from some server connected to the internet.

Number: PERF-2 **Name:** Demonstrability

Description: Non-functional Requirement. It must be possible to successfully demonstrate the capabilities of the browser to interested users.

Verification Method: The development team will demonstrate the project twice over the course of the semester.

Number: PERF-3 **Name:** Distributable

Description: Non-functional Requirement. The software must be packaged in a way such that it is easy to transport from computer to computer.

Verification Method: The browser and all other components will be combined and possibly compressed into a single file for distribution.

Number: PERF-4 **Name:** Portability

Description: Non-functional Requirement. The browser must be able to run on Microsoft, Apple, and Linux operating systems.

Verification Method: The development team will run the browser on each of the operating systems.

Notes: Both WebKit and QT are cross platform and therefore portable.

Number: PERF-5 **Name:** Responsiveness

Description: Non-functional Requirement. The Split modifications should not slow the browsing experience noticeably in any way.

Verification Method: Test different browsers on the same computer and use a human to detect any reductions in speed.

Number: PERF-6 **Name:** Safety

Description: Non-functional Requirement. Split modifications must not break the browser security model.

Verification Method: Perform a basic security review and ask an external partner for a security review of our product.

Number: PERF-7 **Name:** Simplicity

Description: Non-functional Requirement. The average user must be able to quickly pick up and start using the clipping technology.

Verification Method: Perform a usability study where a user begins with no knowledge of the software and is required to perform a set of clipping tasks.

Number: PERF-8 **Name:** Testability

Description: Non-functional Requirement. Developers must be able to easily write and run tests on the software.

Verification Method: Create use case tests and automated functionality tests

Notes: Both WebKit and QT have built in test cases we can reuse.

2.3 Risks

Below we have listed our major risks to the project, our detection method, possible mitigations, and our avoidance plans. Because the technology we are trying to implement is largely unexplored, we do not have very many known risks; however the ones we are aware of are potentially project defining.

Description: May not have enough time to complete fully functional browser

Detection: Periodically compare current status to roadmap

Mitigation Plan: Use a publicly available, pre-made web browser

Avoidance: Weekly status meetings

Description: Could compromise browser security model **Detection:** Ask for an external security review **Mitigation Plan:** Create a new security model for our browser **Avoidance:** Do not alter core browser functionality

Description: Technology used is not capable of performing required tasks **Detection:** Exploring the functionality and documentation of Qt and WebKit **Mitigation Plan:** UI frameworks and web rendering engines are interchangeable **Avoidance:** Sought professional advice before beginning

3 Project Management

3.1 Schedule

Our tentative plan is displayed in figure 1. As you can see, we have decided to break down the work items into approximately four-week sprints. We have begun work on our system and are currently on target with our projections.

After our initial research task, we have discovered that there is very little to do in WebKit and most of our time will be spent in UI development using QT. The project breakdown remains the same.

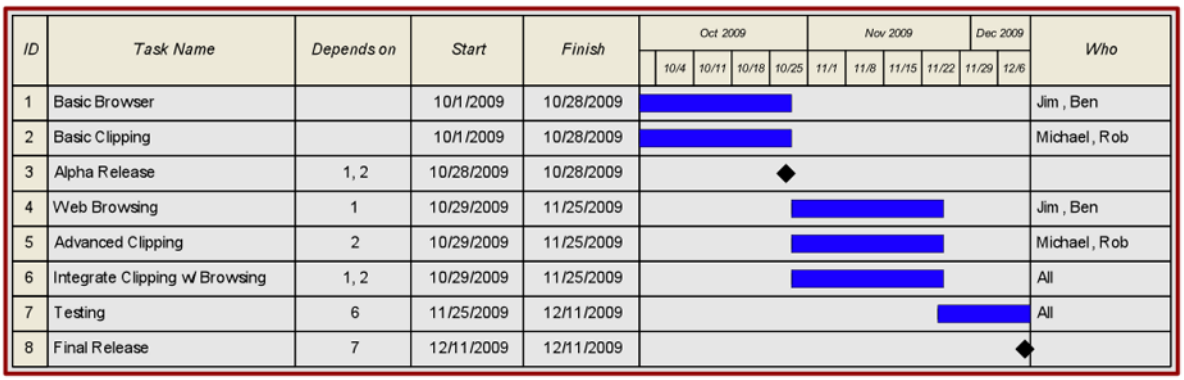

Figure 1: The Tentative Schedule

3.2 Resources

Every member of our team is a skilled programmer and will, in that capacity, be an invaluable asset. Some members of our team have more specialized backgrounds that will be of assistance. Jim Brusstar and Ben Montgomery have done prior work on user interfaces, which is why they are heading up user interface development. Michael Lee and Robert Steen have experience in

testing and will aid in verifying that our software is release-quality.

We will utilize a variety of tools to assist in the development process. Standard communications channels will be used - email most heavily. The source code will be managed with Git, a distributed revision control system. We will set up a Trac website to act as a bug tracker and project management system.

For development, we plan on using open source projects to help accelerate our development rate. WebKit is a great project that should suffice for our HTML rendering engine and has an interface for QT, an open source, cross platform UI design framework. By using these projects, we hope to be able to focus more on our goals and less on the basic functionality.

4 Conclusion

Our project is replacing the current paradigm of browsing. Browsers don't give users enough control over the content they see; users are forced to see the content as the creators intended, but this is not always desirable. By giving our users the ability to split a webpage into separate cropped objects, we provide them with complete customization of the browsing experience. This document explained in detail the main roadmap, work projections, and open issues for us to be able to accomplish this ambitious project. Our project makes heavy use of the open source technologies Qt, a UI framework, and WebKit, a web rendering engine. We have a team of experienced developers and are on track to complete the project by our deadline.