Project Number: DCB-0106

Curiouser Browser

A Major Qualifying Project in Computer Science

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor Science

By

Steve T Law

Michael Cen

Brad Goodwin

Date May 3, 2002

Approved:

Professor David C. Brown, Advisor

Professor Mark Claypool, Co-Advisor

Abstract

This project extended a previous MQP, the Curious Browser, to discover the effectiveness of additional implicit indicators of user interest. We researched possible indicators, and then conducted a small study to find which new indicators should be added. Two examples were the quantization of horizontal and vertical mouse movement. Amongst other results we showed that the number of clicks on a page provided a strong prediction of the explicit rating while the amount of mouse wheel use did not.

Table of Contents

INDEX OF FIGURES VI

INDEX OF TABLES VII

CHAPTER 1: INTRODUCTION 1

1.1 OVERVIEW 1
1.2 PREVIOUS PROBLEM 3

CHAPTER 2: RELATED WORK 5

2.1 CURIOUS BROWSER MQP 5
2.1.1 Curious Browser Introduction and Approach 5
2.1.2 Curious Browser Program Design 6
2.1.3 Curious Browser GUI Design and Installation 7
2.1.4 Curious Browser Analysis and Results 9
2.1.5 Curious Browser Summary 11
2.1.6 Relevance To Our MQP 11
2.2 SUMMARY OF PRESENT PUBLICATIONS 12
2.2.1 "Adaptive Web sites: Conceptual Cluster Mining" 12
2.2.2 "Contents or Graphics" 13
2.2.3 "Empirically Validated Web Page Design Metrics" 15

CHAPTER 3: METHODOLOGY 18

3.1 CURIOUS BROWSER MQP 18
3.2 UPDATE THE PREVIOUS RESEARCH 18
3.3 MINI-EXPERIMENT 19
3.4 IMPLEMENTATION 19
3.5 EXPERIMENT 20
3.6 ANALYSIS 20
3.7 CONCLUSION 21

CHAPTER 4 MINI-EXPERIMENT 22

4.1 INTRODUCTION 22
4.2 DESIGN OF EXPERIMENT 23
4.3 IMPLEMENTATION 24
4.4 ANALYSIS 24
4.5 CONCLUSION 27

CHAPTER 5: BROWSER INTERFACE DESIGN 28

5.1 INTERFACE DESIGN 28
5.2 WELCOME INTERFACE 28
5.3 DISCLAIMER INTERFACE 29
5.4 CLASS SELECTION INTERFACE 31
5.5 BROWSER INTERFACE 33
5.6 GUI DIFFERENCES BETWEEN THE THREE BROWSERS 38
5.7 EVALUATION INTERFACE 39

5.8 INSTRUCTION INTERFACE 40
5.9 CLOSING INTERFACE 41

CHAPTER 6: ALGORITHMS/IMPLEMENTATION 43

6.1 MOTIVATION FOR CHANGES 43
6.1.1 Pop-up Windows 44
6.1.2 Site formats 44
6.1.3 Wheel Mouse 45
6.2 EXISTING ALGORITHMS 46
6.2.1 Mouse Activities 47
6.2.2 Keyboard Activities 48
6.2.3 Explicit Rating of Interest 50
6.2.4 URL, User name, Time/Date Stamp 50
6.3 NEW ALGORITHMS 52
6.3.1 Amount of Right Mouse Button Usage 53
6.3.2 Mouse Wheel Activity 53
6.3.3 Mouse Coordinates 54
6.3.4 Status Bar Change 55
6.3.5 Number of Copy Commands 56
6.3.6 Size of the HTML File 57
6.3.7 Triggering the Evaluation Window 59
6.4 IMPLEMENTATION 62
6.4.1 Mouse activities 62
6.4.2 "Browser Window is not in focus" Activities 64
6.4.3 "Mouse Cursor is out of the Browser Window" Activities 64

CHAPTER 7 EXPERIMENT DESIGN 66

7.1 DATABASE DESIGN 66
7.1 STRUCTURE OF SAMPLE.DAT AND SAMPLE2.DAT TEXT FILES 67
7.1.1 Sample.dat 68
7.1.2 Sample2.dat 68
7.2 TARGETED PARTICIPANTS 70
7.3 ADP LAB SETUP 70
7.4 EXPERIMENT 71

CHAPTER 8: EXPERIMENT IMPLEMENTATION 73

8.1 DATA COLLECTION 73
8.2 CRASHING OF THE CURIOUSER BROWSER 73
8.2.1 Crash 73
8.2.2 Extension of the Experiment 75

CHAPTER 9 ANALYSIS 77

9.1 PARSING ALGORITHM 77
9.1.1 Find and Separate the data 78
9.1.2 Problems 78
9.2 TRIMMING FOR VALID DATA 79
9.3 DESCRIPTIVE STATISTICS 81

9.3.1 Time on Page vs. Explicit Interest Rating 83
9.3.2 Time on Horizontal Scroll Bar 86
9.3.3 Time on Vertical Scroll Bar 88
9.3.4 Events on scrolling 90
9.3.5 Time Using Mouse 92
9.3.6 Clicks on window 94
9.3.7 Number of Up Arrow button occurrence 96
9.3.8 Number of status changes 97
9.3.9 Number of mouse wheel scrolls 99
9.3.10 Size of Web page 100
9.3.11 Number of Mouse Cursor Lines going Up 102
9.3.12 Number of vertical mouse cursor lines going down 103
9.3.13 Number of horizontal mouse cursor lines going left to right 106
9.3.14 Number of horizontal mouse cursor lines going right to left 108
9.3.15 Number of total mouse cursor lines 110
9.3.16 Time spent on page versus Total Scrolling 111
9.4 GENMOD AND RESULTS 112
9.5 DRAWN CONCLUSIONS 116

CHAPTER 10 CONCLUSION AND FUTURE WORK 117

10.1 CONCLUSION 117
10.2 FUTURE WORK 119

REFERENCES 121

APPENDIX A GRAMMARS 123

A.1 IMPLICIT ACTION DATABASE: SAMPLE.DAT 123
A.2 COORDINATE DATABASES 123
A.1.1 Sample2.dat grammar (First two weeks of the Experiment) 123
A.1.2 Sample2.dat grammar (updated-last week of the Experiment) 124

APPENDIX B E-MAILS TO CS PROFESSORS 125

B.1 FIRST E-MAIL 125
B.2 SECOND E-MAIL 125

APPENDIX C PARSING ALGORITHM CODE 126

C.1 PARSE.PL 126
C.2 PARSER.PL 128

Index of Figures

FIGURE 2. 1: EVALUATION INTERFACE 7
FIGURE 2. 2: THE GUI OF THE CURIOUS BROWSER 8
FIGURE 2. 3: THE GUI OF INTERNET EXPLORER 5.0 9
FIGURE 4.1: BAR GRAPH OF THE MINI-EXPERIMENT AT ADP LAB 25
FIGURE 4.2: BAR GRAPH OF THE MINI-EXPERIMENT AT THE PC ROOM IN GORDON LIBRARY 26
FIGURE 5.1: WELCOME INTERFACE 29
FIGURE 5.2: DISCLAIMER INTERFACE 30
FIGURE 5.3: COURSE SELECTION WINDOW 32
FIGURE 5.4: CONFIRMATION WINDOW WITH COURSES CHOSEN 33
FIGURE 5.5: CONFIRMATION WINDOW WITH NO COURSES SELECTED 33
FIGURE 5.6: THE GUI OF MICROSOFT INTERNET EXPLORER 35
FIGURE 5.7: THE GUI OF THE CURIOUS BROWSER 36
FIGURE 5.8: THE GUI OF THE NEW CURIOUSER BROWSER 37
FIGURE 5.9: EVALUATION INTERFACE 40
FIGURE 5.10: INSTRUCTION INTERFACE 41
FIGURE 5.11: CLOSING WARNING WINDOW INTERFACE 41
FIGURE 5.12: CLOSING WINDOW 42
FIGURE 6. 1: STATE DIAGRAM FOR CAPTURING MOUSE ACTIVITIES 48
FIGURE 6. 2: STATE DIAGRAM FOR CAPTURING KEYBOARD ACTIVITIES 49
FIGURE 6.3: STATE DIAGRAM FOR CAPTURING THE USER NAME 51
FIGURE 6. 4: SATE DIAGRAM FOR CAPTURING STATUS BAR CHANGES 56
FIGURE 6. 5: STATE DIAGRAM FOR CAPTURING COPY ACTIVITIES 57
FIGURE 6.6: STATE DIAGRAM FOR GETTING THE SIZE OF THE HTML FILE 59
FIGURE 6. 7: STATE DIAGRAM FOR GETTING EXPLICIT INTEREST RATING 61
FIGURE 6. 8: BOUNDARY (IN RED) SETUP FOR THE MOUSE COORDINATE IMPLICIT RATING 65
FIGURE 9.1: BOX PLOT OF TIME ON PAGE VS. EXPLICIT INTEREST RATING 84
FIGURE 9.2: BOX PLOT OF TIME ON HSROLL VS. EXPLICIT INTEREST RATING 87
FIGURE 9.3: BOX PLOT OF TIME ON PAGE VS. EXPLICIT INTEREST RATING 89
FIGURE 9.5: TIME ON MOUSE VS. EXPLICIT INTEREST RATING 93
FIGURE 9.6: BOX PLOT OF CLICKS ON WINDOW VS. EXPLICIT INTEREST RATING 95
FIGURE 9.7: BOX PLOT OF NUMBER OF STATUS BAR CHANGES VS. EXPLICIT INTEREST RATING 98
FIGURE 9.8: BOX PLOT OF NUM OF MOUSE WHEEL SCROLL VS. EXPLICIT INTEREST RATING 100
FIGURE 9.9: LINE GRAPH OF SIZE OF PAGE VS. TIME ON PAGE 101
FIGURE 9.10: BOX PLOT OF NUM OF MOUSE CURSOR MOVING UP VERTICALLY VS. EXPLICIT INTEREST RATING 103
FIGURE 9.11: BOX PLOT OF NUM OF MOUSE CURSOR MOVING DOWN VERTICALLY VS. EXPLICIT INTEREST RATING 105
FIGURE 9.12: BOX PLOT OF NUM OF MOUSE CURSOR MOVING FROM LEFT TO RIGHT VS. EXPLICIT INTEREST RATING 107
FIGURE 9.13: BOX PLOT OF NUM OF MOUSE CURSOR MOVING FROM RIGHT TO LEFT VS. EXPLICIT INTEREST RATING 109

FIGURE 9.14: BOX PLOT OF NUM OF TOTAL MOUSE CURSOR LINES VS. EXPLICIT INTEREST RATING 110
FIGURE 9.15 TIME SPENT ON PAGE VERSUS TOTAL SCROLLING 112
TABLE 9.19: GENMOD TEST - ALL IMPLICIT RATINGS AND EXPLICIT FAMILIARITY RATING VS. EXPLICIT INTEREST RATING WITHOUT "0" RATING 114
TABLE 9.21: GENMOD TEST: ALL IMPLICIT RATINGS (WITHOUT LINES) AND EXPLICIT FAMILIARITY RATING VS. EXPLICIT INTEREST RATING WITHOUT "0" RATING 115

Index of Tables

TABLE 2. 4: THE GUI OF INTERNET EXPLORER 5.0........ 10
TABLE 7.1 INFORMATION OF THE DATA COLLECTED FROM THE EXPERIMENT........ 75
TABLE 9.1: LIST OF THE TOTAL DATA 81
TABLE 9.2: THE DESCRIPTIVE STATISTICS OF THE IMPLICIT RATING, TIME SPENT ON A PAGE. 84
TABLE 9.3: DESCRIPTIVE STATISTICS OF TIME ON HORIZONTAL SCROLL BAR SEPARATED BY INTEREST RATING 86
TABLE 9.4: DESCRIPTIVE STATISTICS OF TIME ON VERTICAL SCROLL BAR SEPARATED BY INTEREST RATING 88
TABLE 9.5: DESCRIPTIVE STATISTICS OF EVENTS ON SCROLLING SEPARATED BY INTEREST RATING 90
FIGURE 9.4: BOX PLOT OF EVENTS ON SCROLL VS. EXPLICIT INTEREST RATING 91
TABLE 9.6: DESCRIPTIVE STATISTICS OF TIME USING MOUSE SEPARATED BY INTEREST RATING 92
TABLE 9.7: DESCRIPTIVE STATISTICS OF CLICKS ON WINDOW SEPARATED BY INTEREST RATING 94
TABLE 9.8: DESCRIPTIVE STATISTICS OF NUMBER OF UP ARROW BUTTON OCCURRENCES SEPARATED BY INTEREST RATING. NOTICE THE "0.00" VALUES OF THE MEDIAN COLUMN, THIS MEANS LOW OCCURRENCE. 96
TABLE 9.9: DESCRIPTIVE STATISTICS OF NUMBER OF STATUS CHANGE SEPARATED BY INTEREST RATING 97
TABLE 9.10: DESCRIPTIVE STATISTICS OF NUMBER OF MOUSE WHEEL SCROLLS SEPARATED BY INTEREST RATING 99
TABLE 9.11: DESCRIPTIVE STATISTICS OF SIZE OF WEB PAGE SEPARATED BY INTEREST RATING 100
TABLE 9.14: DESCRIPTIVE STATISTICS OF NUMBER OF VERTICAL MOUSE CURSOR LINES GOING UP BY INTEREST RATING 102
TABLE 9.15: DESCRIPTIVE STATISTICS OF NUMBER OF VERTICAL MOUSE CURSOR LINES GOING DOWN SEPARATED BY INTEREST RATING 104
TABLE 9.16: DESCRIPTIVE STATISTICS OF NUMBER OF HORIZONTAL MOUSE CURSOR LINES GOING LEFT TO RIGHT SEPARATED BY INTEREST RATING 106
TABLE 9.17: DESCRIPTIVE STATISTICS OF NUMBER OF HORIZONTAL MOUSE CURSOR LINES GOING RIGHT TO LEFT SEPARATED BY INTEREST RATING 108

TABLE 9.18: DESCRIPTIVE STATISTICS OF NUMBER OF TOTAL MOUSE CURSOR LINES SEPARATED BY INTEREST RATING 110

Chapter 1: Introduction

1.1 Overview

With the rapid development of technology within the past decade, the Internet has become one of the best and most convenient ways to access information. In February 2002, Business Week conducted a survey to investigate the total number of people that are able to access the Internet from their home in the United States or Canada. The survey shows that 185.6 million people can access the internet from their home [Boardwatch Magazine, 2002]. The Internet allows information to be delivered to anywhere, and to anybody, within seconds or, at the most, minutes. All major companies have their information posted and sell their products on the Internet. Novels are posted on the Internet, so readers do not have to buy books anymore. Movie reviews also appear on the Internet as do movie trailers.

Information provided by the Internet is almost unlimited, since people upload new knowledge onto the Internet every minute. However, the incredible speed of informational growth of the Internet has made the information available on the Internet more and more disorganized. Users may use a Web search engine to search for information on the Internet. However the results may not be helpful to them. In order for users to improve their chances of finding the information they were looking for, recommendation systems were introduced.

A recommendation system is a tool that is built to recommend personalized interest information to Internet users. Amazon.com is an online store that uses a recommendation system. It allows customers to search, buy, and rate products. The site also provides recommendations to the customer. For example, two customers both bought Book A, B, and C from Amazon.com. Then if the first customer buys Book D, the recommendation system would then recommend Book D to the second customer because of their apparent similar interests.

Recommendation systems help us find the information we need. They successfully predict and recommend what users are looking for. In order to filter accurately, the system needs to know the users' opinion before being able to predict what the users might like. Therefore, typically users supply their opinions through explicit ratings.

In an explicit rating system, users are asked to respond to questionnaires or to evaluate particular objects based on a given scale.   An explicit rating system is simple but it has its limits. Users must rate items according to the given instruction and must be willing to give opinions accurately in order for the rating system to work. User participation level also plays a very important role for explicit rating on the Internet. Users may become irritated that an evaluation window pops up while they are surfing the Internet. If that happens, users may skip the step where they provide the evaluation. To avoid these problems, a different system to collect user's ratings is considered: i.e., implicit rating.

Unlike explicit ratings, implicit ratings do not require users to explicitly rate the pages being viewed. Implicit ratings indirectly imply a preference for an object. Instead we depend on user actions, or user patterns of behavior, to predict their level of interest. User actions providing implicit rating may include time spent on page, movements of the mouse, or copying contents of the page.

1.2 Previous Problem

The previous MQP, called the Curious Browser [Le & Waseda, 2000], was done two years ago. The Curious Browser project implemented a browser, based on Microsoft Internet Explorer 5.0, which collected implicit user actions, such as the number of clicks on a page, as well as asking the user to explicitly rate their interest for each page. They compared these implicit and explicit indicators to find correlations that would allow the user's implicit actions on the Web page to predict the user's explicit interest rating for the Web page.

The conclusion from the Curious Browser project was that there were correlations between some implicit ratings and the explicit interest ratings for Web pages. The user actions that they found to be good implicit indicators of interest were the time spent on a Web page, the time spent reading a Web page and the combination of time spent scrolling with the mouse and the keyboard. They found that the strongest correlation between implicit actions and explicit interest ratings was the time spent on a Web page. Some of the user actions that were not good indicators of interest were the number of mouse clicks on the browser window, the time spent moving the mouse and the amount of separate keyboard scrolling and mouse scrolling.

Our project's methodology to extend a previous MQP, the Curious Browser [Le & Waseda, 2000], is to enhance the Curious Browser so that it captures more relationships between users' explicit rating and their implicit ratings while they are navigating the Web. The results of the previous MQP allowed this new project to explore additional implicit indicators and to find ones that were more meaningful. The Curious

Browser project established a framework for this project, however there were some problems trying to use the Curious Browser with current Web sites.

The many advances in the design of Web pages and input devices may have caused some of the old implicit ratings to be incorrect or insignificant. Also, with the availability of Microsoft Internet Explorer 6.0 as the base for a Curiouser Browser, it may be possible to detect new and better implicit ratings that were unavailable to the Curious Browser project two years ago. With these changes to the Web browsing experience in mind, this project set out to find the significance of a new set of implicit indicators.

In the rest of this report we describe in detail the steps we went through to complete this experiment. We describe related research (Chapter 2), give an overall view of the report in the methodology section (Chapter 3) and write about the mini-experiment that we conducted (Chapter 4) which we used to make design decisions. Then we give a detailed description of the overall design of the Curiouser Browser's interface (Chapter 5) and its implementation (code and algorithms) (Chapter 6). With the browser description completed we describe the experiment design (Chapter 7) and the experiment implementation (Chapter 8). Then we analyze the results of the experiment (Chapter 9) and compare the results to our initial goals to see what conclusions we can draw as well as discussing possible improvements to the research and the report (Chapter 10).

Chapter 2: Related Work

2.1 Curious Browser MQP

This project is a continuation of a previous MQP also done at WPI. Much of the research and design aspects of this project are based on the work already done by the Curious Browser MQP (Le & Waseda, 2000). The following is a brief summary of the Curious Browser report.

2.1.1 Curious Browser Introduction and Approach

The Curious Browser MQP tried to determine if there is a strong correlation between implicit ratings and explicit ratings. The browser built for the MQP asked for the explicit rating of each page visited and also monitored implicit actions such as the time spent on the Web page or the time moving the mouse. The relationships between the explicit and implicit ratings were then tested for significance by using statistical testing.

After some research on prior work that studied implicit ratings, the developers of the original browser decided to monitor these user actions:

- Time spent on a page;
- Time spent reading a page;
- Time spent moving the mouse;

- The number of mouse clicks;
- Time spent scrolling using the mouse;
- Time spent scrolling using the keyboard;
- Various combinations of the above.

The developers chose these implicit indicators to be monitored because research publications suggested that there are strong correlations between them and the users preference for the page. Some were also chosen because through their own observations, the developers hypothesized that there was a correlation.

In order to monitor those occurrences, a monitoring tool was needed. Since there was no satisfactory monitoring tool available, Le and Waseda had to create their own Web browser that correctly interpreted HTML code and also monitored the actions mentioned above.

2.1.2 Curious Browser Program Design

The browser that was created is called the Curious Browser. The design of the browser was quite complex due to the lack of open source Web browsers. The most popular browsers at the time were Netscape and Microsoft Internet Explorer, but neither of them have open source code for easy editing. There was Mozilla, which was open source, but could not perfectly render HTML.

Their final decision was to use Visual Basic and create a new browser. The advantage of creating the Curious Browser in Visual Basic is that the IE 5.0 HTML rendering engine, which can render HTML well, can be embedded in it. Also, Visual

Basic can catch events such as mouse movement and keyboard activity on operating systems developed by Microsoft. Another advantage, for example, is that Visual C++ components can also be included for extra help if needed.

2.1.3 Curious Browser GUI Design and Installation

The creation of the Curious Browser took 6 weeks. The final product detects all the implicit events they proposed to monitor. To record the user's explicit rating, they made a pop up window that asked the user to rate each page when as the user leaves that page. The Evaluation Interface pop-up window is shown here in figure 1.1.

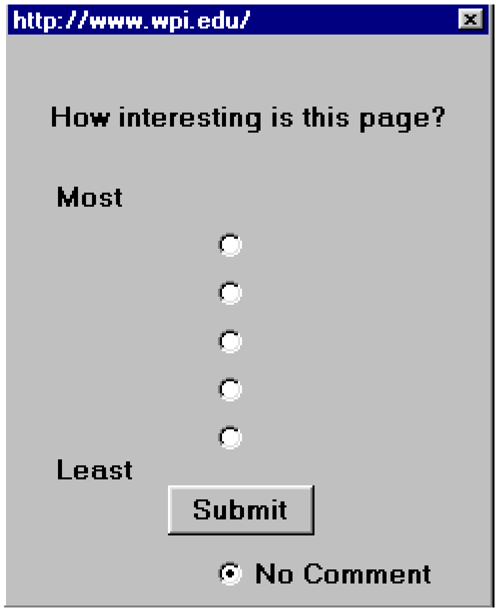
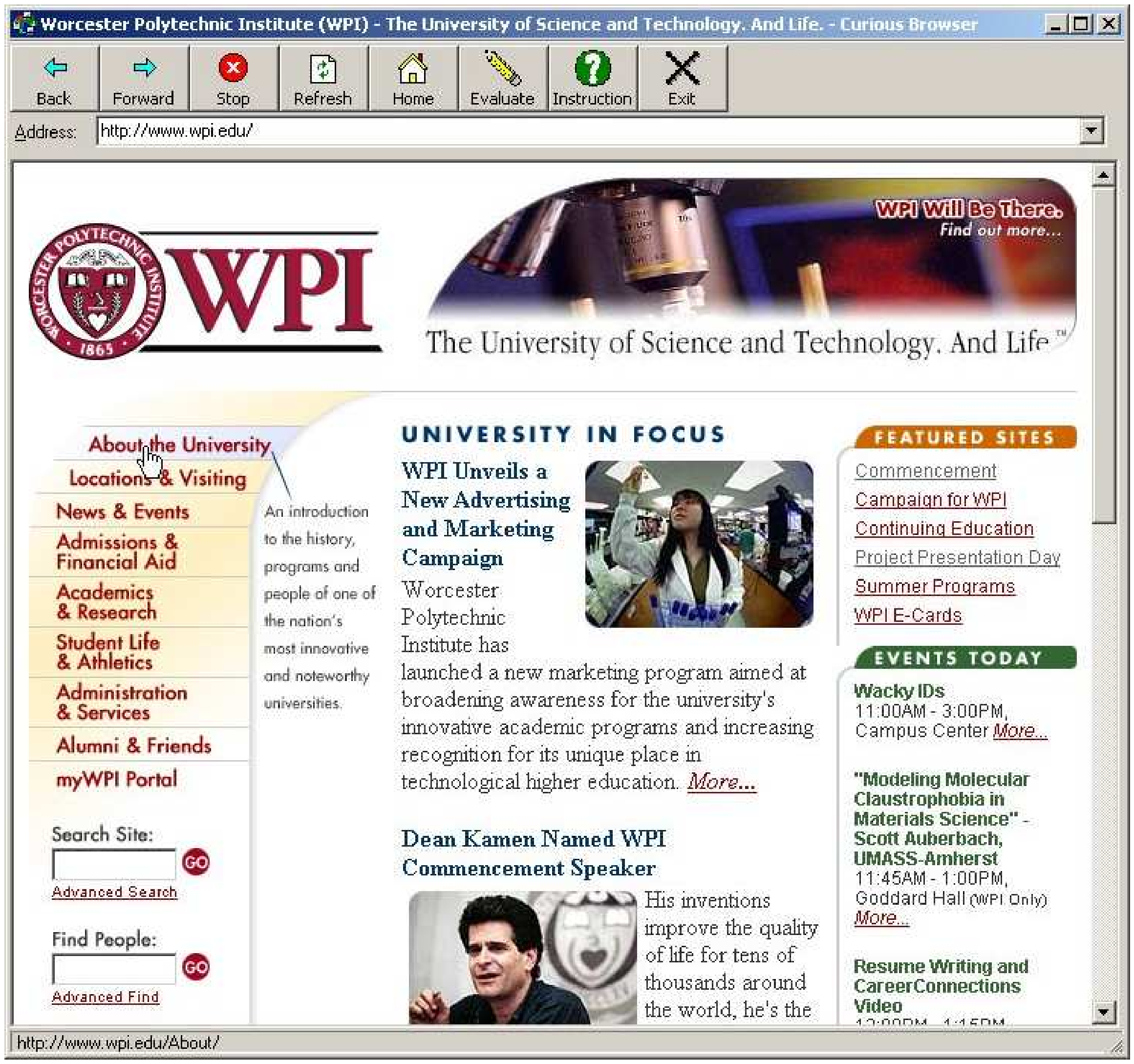


Figure 2. 1: Evaluation Interface

The GUI of the created browser was very similar to the format of the Internet Explorer 5.0, since that was the browser installed for the computers in the lab. Also the IE browser is one of the most popular browsers available for the PC platform. Their interface is shown here in figure 2.2 and the interface for Internet Explorer 5.0 is shown in figure 2.3.



An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

UNIVERSITY IN FOCUS

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. More...

Dean Kamen Named WPI Commencement Speaker

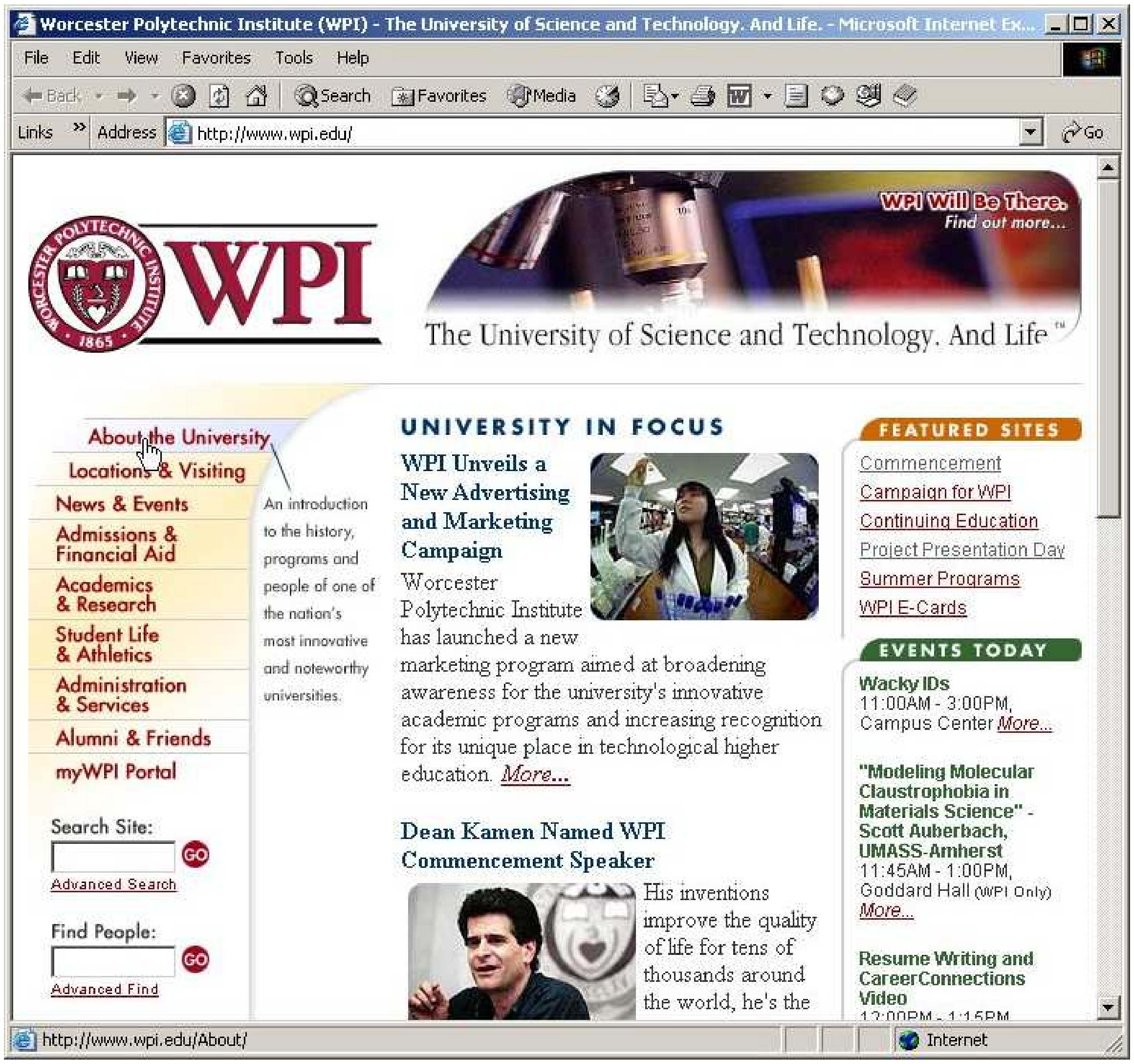
His inventions improve the quality of life for tens of thousands around the world, he's the

Wacky IDs
11:00AM - 3:00PM, Campus Center More...

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auberbach, UMASS-Amherst
11:45AM - 1:00PM, Goddard Hall (WPI Only) More...

Resume Writing and CareerConnections Video

Figure 2. 2: The GUI of the Curious Browser


Worcester Polytechnic Institute (WPI) - The University of Science and Technology. And Life. - Microsoft Internet Ex...
File Edit View Favorites Tools Help
Back Search Favorites Media
Links Address http://www.wpi.edu/ Go
WPI Will Be There. Find out more...
WPI
The University of Science and Technology. And Life™
About the University
Locations & Visiting
News & Events
Admissions & Financial Aid
Academics & Research
Student Life & Athletics
Administration & Services
Alumni & Friends
myWPI Portal
Search Site: GO Advanced Search
Find People: GO Advanced Find
UNIVERSITY IN FOCUS
FEATURED SITES
Commencement
Campaign for WPI
Continuing Education
Project Presentation Day
Summer Programs
WPI E-Cards
EVENTS TODAY
http://www.wpi.edu/About/
Internet

An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. *More...*

Dean Kamen Named WPI Commencement Speaker

His inventions improve the quality of life for tens of thousands around the world, he's the

Wacky IDs
11:00AM - 3:00PM, Campus Center *More...*

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auerbach, UMASS-Amherst
11:45AM - 1:00PM, Goddard Hall (WPI Only) *More...*

Resume Writing and CareerConnections Video

Figure 2. 3: The GUI of Internet Explorer 5.0

2.1.4 Curious Browser Analysis and Results

The Curious Browser was installed on 38 computers in the ADP lab and WINE Lab with the permission of the WPI CS dept and the CCC. The experiment was conducted for 2 weeks with WPI students from March 20, 2000 to March 31, 2000. The analysis of the data was done within 2 weeks after the experiments were finished. The accuracy of the implicit ratings was determined by using the explicit ratings as the base.

For the analysis, only data close to the median was looked at. There were many outliers that might have been caused by the different errors that users had.

As part of the analysis, the Kruskal-Wallis test was used to find the degree of independence of the medians among explicit rating groups. These tests decide if the median of each implicit rating has a correlation with the explicit rating group over 95% confidence intervals. The 95% confidence interval is where the result would occur randomly les than 5% of the time. Then basic statistical tests were applied to each rating. After the Kruskal-Wallis tests, combinations of rating were tested, hoping to find greater correlations between implicit ratings and the explicit rating. After all the individual testing and combined testing, they produced the result shown in Table 2.4:

Interest Indicators	**Result:**
Time spent on page	***Good***
Time spent reading a page	***Good***
Time spent moving the mouse (movements and clicks combined)	***Not Good***
Number of left mouse clicks	***Bad***
Time spent scrolling using the mouse	***Not Good***
Time spent scrolling using the keyboard	***Not Good***
Time spent scrolling using the mouse and by the keyboard	***Good***
Time spent reading a page + time spent moving the mouse + scrolling using the keyboard	***Good***

Table 2. 4: The GUI of Internet Explorer 5.0

2.1.5 Curious Browser Summary

They found that there was no significant difference in the accuracy of prediction between the time spent on a page and the combined indicator (the time spent reading a page and the time spent scrolling by the mouse and by the keyboard). Also the two strongest correlations between implicit indicators and explicit indicators occurred with the time spent on page and the combination of time reading a page and the time spent scrolling.

2.1.6 Relevance To Our MQP

The Curious Browser MQP was a very good resource to study since the current MQP is a continuation of that project. Their report contains all the design decisions and their motivation for the design. This project followed the old MQP, but also deleted and added steps where appropriate. The Curious Browser project was conducted two years before this one so many new implicit monitoring techniques have been developed that might be used to improve the old Curious Browser. In the old MQP, the eight types of implicit ratings listed in Table 2.4 were chosen carefully for their possible relevance. Due to the changes of designs of input devices and new technology in making Web sites, these types of implicit ratings need to be reevaluated. This project evaluates the chosen implicit ratings from the Curious Browser MQP and adds new ones.

2.2 Summary of Present Publications

In this section we review recent publications that relate to Implicit Rating, such as mouse movement detection.

2.2.1 "Adaptive Web sites: Conceptual Cluster Mining"

Designing a Web site that yields to every user's demands is very complex and hard. No one design is perfect for everyone, so that is why making a perfect Web site means making it adaptive to users. With that degree of complexity, some Artificial Intelligence (AI) technology would be needed. AI developers responded with the concept of "conceptual cluster mining".

Conceptual cluster mining is defined as: "given a collection of objects, a pair wise similarity measure over those objects, and a conceptual language for describing them, we search for a small set of cohesive clusters that correspond to concepts expressible in the language" (Perkowitz & Etzioni, 2001). The new conceptual cluster mining algorithms have outperformed traditional clustering algorithms. The new algorithm is also called "PageGather" and it was proven to even build a better index page than human Web designers. The reason is that PageGather ignored the human's intuition about Web design and replaced it with a statistical approach.

The results from our project would allow an adaptive site to make modifications to their site without asking the user any questions. They would be able to use implicit ratings to determine what the user found interesting on the site or if the user finds the site interesting at all, and the site could make modifications depending on what they found.

2.2.2 "Contents or Graphics"

"Contents or Graphics" is an article that describes an evaluation of a dataset from a large Web site rating group [Sinha, Hearst, & Ivory, 2001a]. The study was conducted using the Web site dataset of the Webby Awards 2000. This dataset consists of nearly 3000 different Web sites, rated based on *content, structure & navigation, visual design, functionality, interactivity*, and *overall experience.* About 100 judges from the International Academy of Digital Arts & Sciences chose the subset of the 3000 Web sites that made it to the Review category. The 3000 Web sites were separated into 27 different categories based on their content. The judges had six different aspects by which they would measure each site.

Each of the 6 aspects are explained here:

Contents: The content of the Web site needs to be relevant and appropriate to the whole site, and be clear about it. It could relay specific opinions, jokes, or information as long as it makes the reader wanting more.

Structure/Navigation: The organization of the information on the site is, an important factor. The more organized the site is the more efficiently the site will be used. Users should be able to navigate from one thing to the next without getting confused or lost. If the structure is easy for the user to follow then the experience of that user is greatly increased.

Visual Design: The graphics of a Web site should do more than make it look pretty. It should be appropriate for the site and be consistent with the structure of the site. The visual design doesn't have to be cutting edge or trendy, as long as the design is high quality and relevant for the audience.

Functionality: This topic deals with the technology used to make the Web site. Good functionality makes sure the technology being used is operational. Making sure sites load with no errors, regardless of which major browser is being used. Making sure all the links are live and the technology is relevant to the topic of the site.

Interactivity: This aspect rates the way the site lets users do something within the site, such as searching, chatting, gaming, or anything that deals with how the user inputs and receive output. Good interactivity should relay a distinct feeling that the user is not reading a magazine or watching TV.

Overall Experience: This aspect sums up the previous five points and also the feeling of why one would stay or leave this site. The article describes the feeling to be similar to one you might get on a date. Sometimes the user and site just “click” like a successful couple would in a date.

The main focus of this article was to evaluate the judgment techniques employed by Webby Awards. They employed several statistical techniques: correlation coefficients, independent sample t-test, and linear regression. They found the main criteria for contribution to the overall experience was content and the least was visual design. The statistical analysis between the five specific criteria only explains 89% of the variance in the overall ratings in the Review stage. This means there are other factors besides the five criteria that determined the award-winning Web sites.

This article discusses six main aspects that sites can adjust to make users enjoy their sites more. With implicit ratings these sites would be able to tell if changes or modifications that they've made to their site to improve it are actually increasing the amount users enjoy the site. If a Web site is modified and the average implicit rating among users goes up, then they might make more of the same modification.

2.2.3 "Empirically Validated Web Page Design Metrics"

The article discussed how to develop a profile of a good Web site based on the type of the site [Sinha, Hearst, & Ivory, 2001b]. The authors have developed various metrics to describe each different type. They do not want to specify a certain way to develop one good general Web site, because there is no general Web site. There are so many different ways to build a Web site that most guidelines for building a Web site have little in common besides the basics, such as making the content of the Web site viewable. This article contains metrics for measuring the overall design of a site then comparing the sites' score with how well it was ranked in The Webby Awards. They concluded that their metrics ratings were 67% accurate when compared with the ranking made by the Webby Awards.

They developed 11 metrics for their calculation:

Word Count - Total words on a page.

Body Text % - Percentage of words that are body vs. display text.

Emphasized Body Text % - portion of body text that is emphasized.

Text Positioning Count - Changes in text position from flush left.

Text Cluster Count - Text areas highlighted with color, bordered regions, rules or lists.

Link Count - Total links on a page.

Page size - Total bytes for the page as well as elements graphics and style sheets.

Graphic % - Percentage of pages bytes that are for graphics.

Graphics Count - Total graphics on a page.

Color Count - Total colors employed.

Font Count - Total fonts employed.

They also developed a measure of significance based on word count:

Low Word Count

Good pages have slightly more content, smaller page sizes, less graphics and employ more font variations than not-good pages. The smaller page sizes and number of graphics suggests faster download times for these pages (this was corroborated by a download time metric, not discussed in detail here). Correlations between font count and body text suggest that good pages vary fonts used between header and body text.

Medium Word Count

Good pages emphasize less of the body text; if too much text is emphasized, the unintended effect occurs of making the un-emphasize text stands out more than emphasized text. Based on text positioning and text cluster count, medium-sized good pages appear to organize text into clusters (e.g., lists and shaded table areas). The negative correlations between body text and color count suggests that good medium-sized pages use colors to distinguish headers.

High Word Count

High quality pages exhibit a number of differences from low quality pages. Although both groups have comparable word counts in the body text, high quality pa have more headers and text links than low quality pages (this was verified this with handinspection of some pages). As mentioned above, headers are thought to make it easier for someone to scan a page and pull out some information, while generous numbers of links can facilitate information seeking providing they are meaningful and clearly marked [Sinha, Hearst, & Ivory, 2001b].

They concluded by saying that the metrics explored here were simple compared to the advance rating system used by the Webby Awards, but the results were relatively close (67%). However, Metrics are only a small part of the Web site design puzzle.

The metrics discussed in the article are a quantitative method developed to describe a Web page's structure and organization in categories. However, it disregards the factors of user's interaction with and interest in a Web page. If implicit ratings, also a quantitative method, were proved by us to be good indicators of interest, a combination of metrics and implicit ratings can improve the accuracy of the prediction. The metrics measure the structure of the site and then the implicit ratings measure interest, giving two aspects to test the overall success of the Web site.

Chapter 3: Methodology

This chapter is an overview of the process we took in conducting our MQP. We made plans before the actual implementation of the experiments as to what we expected to accomplish with those plans. There were only minor adjustments during the actual implementation.

3.1 Curious Browser MQP

Our MQP was a continuation of a previous one called the Curious Browser [Le & Waseda, 2000]. To learn from the Curious Browser MQP, we obtained a copy of their report and of the source code for the browser they created. From this information we learned about what they did, from the motivation for their project, to the conclusions they drew from the results of their experiment. We also learned about the implicit and explicit ratings analyzed by them and the reasons why those ratings were chosen. They also mentioned in their report the reasons why they built their Curious Browser (CB) with Visual Basic and why the interface of the Browser looks the way it does. Given their plans and design decisions, we have a good general idea of how to conduct our MQP. For further details, please refer to Chapter 2.1.

3.2 Update the previous Research

The Curious Browser MQP was conducted 2 years ago, so the results then may not be the results of now. Since Internet is an ever-growing medium that can change on a daily basis so it was very likely that the research information gathered during the Curious

Browser (CB) MQP on implicit and explicit ratings was outdated. Knowing this we searched for any new research done on implicit ratings since the CB. From the new research, we learned that mouse cursor areas and other mouse related movements were not well researched. We believe that the mouse movements of a user browsing a Web page are a rich area from which to gather implicit ratings. For summaries of articles regarding research on user browsing habits, please refer to Chapter 2.2.

3.3 Mini-Experiment

The Mini-Experiment was a study we did to obtain insight into what our choices for implicit ratings to study were and which ones would be the best to study. First we discussed the possible user actions that can occur while browsing a Web page. We then picked user actions that we are able to record using the tools we have. The experiment was a simple observation experiment conducted on a small random population. Expected outcomes from the Mini-Experiment were insights into which implicit ratings were worth looking into more deeply. For more detail regarding the data collected, please refer to Chapter 4.

3.4 Implementation

Once the research was completed and the implicit ratings to be studied were chosen, we designed and built the Curious***er*** Browser. The design of Curiouser Browser is based on the results from the Mini-Experiment and the GUI design discussed by the previous MQP. The new Curiouser Browser was created using Visual Basic 6.0 due to its compatibility with Application Programming Interface (API) and with Microsoft Windows. With the help of new API calls, we were able to implement the browser to record many implicit actions that the previous MQP could not. To learn more about how API calls could help us record implicit actions, refer to Chapter 6.

3.5 Experiment

The experiment using the newly made Curiouser Browser was conducted in the ADP (Advanced Document Preparation) lab in the Fuller Laboratories building on the WPI campus. The experiment was conducted for three weeks (March 18 to April 5). Every night, data was collected from the computers on which the browser was installed. This was done to ensure that the data was not deleted or tampered with, and to keep more resources like memory and storage space available. The experiment was planned for two weeks but we extended it for an extra week because of technical problems that cut the number of usable results to almost half of the initial amount. To see how we resolved the problem, please refer to Chapter 8.

3.6 Analysis

The data recorded by the Curiouser Browser was analyzed with statistical tests that try to prove that a relationship exists between explicit ratings and implicit ratings. The main test of importance is the GENMOD procedure provided by SAS. SAS is statistical software that is widely available in WPI’s labs, making it convenient for us to access. The idea to use GENMOD was suggested to us by statistic consultants Prof. Joseph Petruccelli, Yongmie Chan, and Jie Liu. For some graphs and further explanation of the GENMOD procedure, please refer to Chapter 9.

3.7 Conclusion

After the analysis of all the data, results were drawn. We have found that some implicit actions have a strong correlation with the explicit interest rating of a page. From these results we were able to show that certain implicit actions do have correlations to the explicit rating. These correlations were found due to the GENMOD procedure and through the analysis of charts and graphs made from the comparison of relevant implicit ratings versus explicit interest rating.

Chapter 4 Mini-Experiment

This chapter discusses a study we conducted to help us decide which Implicit Actions we want to record using our browser. We had a list of possible user actions that could be recorded using the tools we have. We hoped to reduce that list by doing this observation.

4.1 Introduction

Our project required that we implement new and meaningful implicit ratings into the browser. We had to find a way to choose which actions would give us the best and most accurate results out of our list of potential candidates.

- Decision – Time between mouse on link and click on link;
- Searching – Using Web search engines;
- Typing in URL's – Typing URL in the browser's address bar;
- Scroll button – Using mouse wheel button;
- Scroll bars – Using Horizontal and Vertical scroll bars;
- Following list – Reading with the mouse vertically;
- Reading with mouse – Reading with the mouse horizontally;
- Highlighting – Selecting contents of the Web page;
- Circling with Mouse – User's circular motion with the mouse;
- Back button – Using the back button of the browser;
- Right Mouse Clicks – Using the right mouse button.

These candidates were chosen because they were obtainable with the API calls that we have. Also, since our MQP had a limited amount of time, we had to complete the software modification of the browser or we would not be able to run the experiment for all of the indicators, or be able to analyze them as deeply as we would like to.

To learn which user actions we wanted to use as implicit indicators in our study, we conducted a separate, smaller, experiment; a mini-experiment. We used results from this study to separate user actions that occur more frequently from those we found to occur seldom. This allowed us to focus on actions we felt had a stronger chance of returning us meaningful results, because they would occur with reasonable frequency.

4.2 Design of Experiment

To divide the list of user actions into those that could be useful and those that are not, we observed students as they used computers and recorded what actions the students used while browsing the Web. We wanted to observe students in their natural use of a Web browser and record information on their common actions. To do this we decided on the ADP (Advanced Document Preparation) lab as the location where the experiment would be conducted. Many students go there on a regular basis to do work, browse the Web and just to have a computer to use so it seemed like an ideal spot to run the experiment. We were there with a list of actions that we looked for while observing a user. If we observed any of these actions occurring more than three times while observing a user, we would record that the user used that action. With the information gathered we would make decisions about which actions would be implemented and which wouldn't.

4.3 Implementation

When we implemented the experiment we found most students in the ADP to be very cooperative in letting us observe them while they browse the Internet. We would ask a user if he or she would allow us to observe them for a few minutes (usually 5 to 10) and then sit behind them and mark down any of the actions on our list that we observed the user using. We did tell them that we were observing their Internet browsing habits. After observing fifteen users we changed the location of the experiment to the PC Room that resides in the bottom of the Gordon Library and observed fifteen more students. We switched locations to allow the data collected to be more random. Also, after we had observed fifteen users, we thought of one or two more actions we wanted to look for in our observation, so it seemed like the perfect time to switch.

4.4 Analysis

After collecting observations, we graphed the data into the two charts, shown in Figures 4.1 and 4.2. The first chart (Figure 4.1) shows the information gathered in the ADP lab and the second (Figure 4.2) shows the data gathered in the Libraries lab.

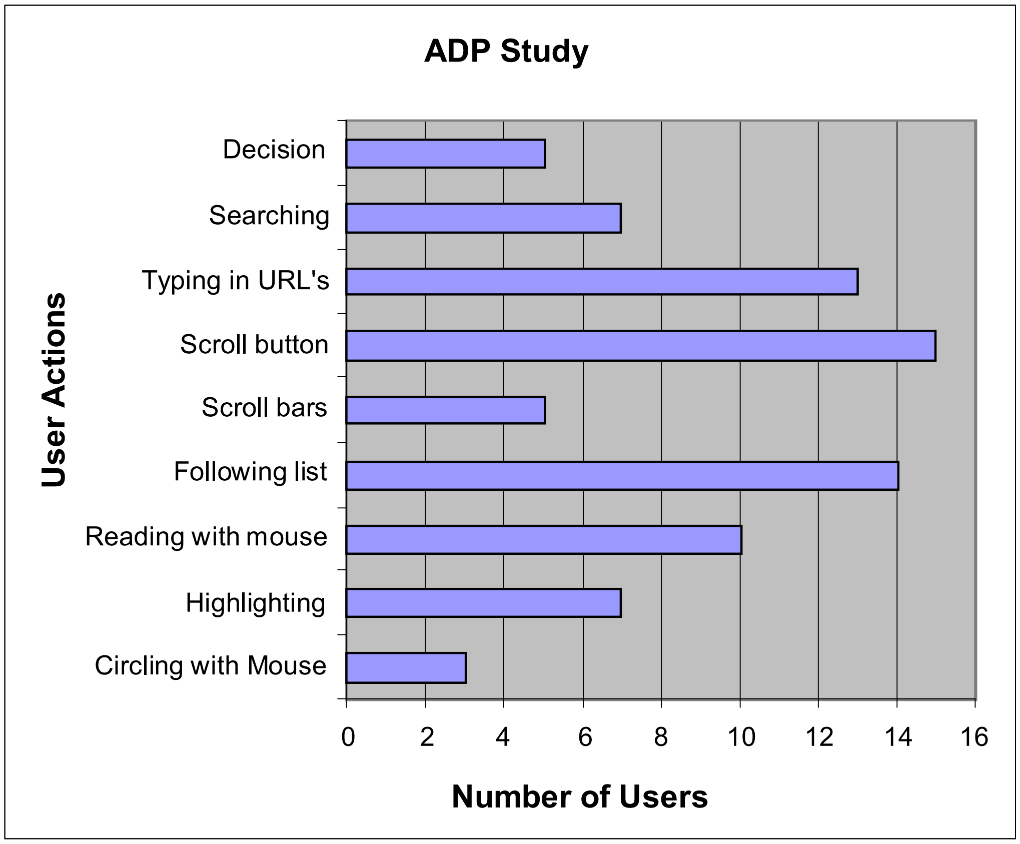


Figure 4.1: Bar Graph of the Mini-experiment at ADP Lab

As you can see in Figure 4.1 the actions that occurred most frequently among different users in the ADP lab were typing in a URL, scrolling with the Scroll button, following lists with the mouse and reading with the mouse. Detection of all of these actions was added to the new browser. The actions that were the least common among users were circling with the mouse and the Decision. Decision is the measure of hesitation time when deciding between links to click. Therefore, we dropped those from the list of actions to be implemented into the browser.

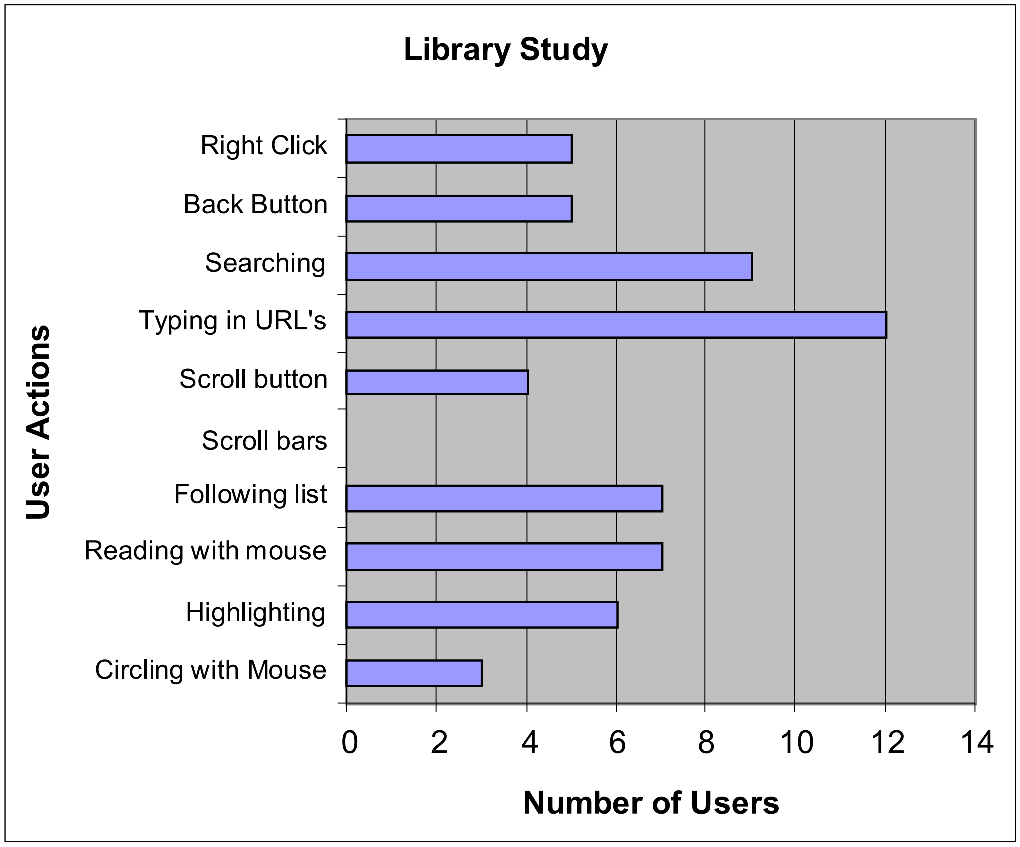


Figure 4.2: Bar Graph of the Mini-experiment at the PC Room in Gordon Library

As you can see in figure 4.2 the actions that occurred the most frequently among different users in the Library lab were: typing in a URL, searching, following lists with the mouse and reading with the mouse. Detection of all of these actions, except searching, was added to the browser.

Before studying any of the users in the library we added two actions to look for while observing (Right clicking and the Back button). Both showed some use but not much. The actions that were the least common among users were circling with the mouse, and the use of the scrollbar. The circling we decided to drop after observing that neither chart showed common use of it. Even though we kept the scrollbar as an indicator we discovered in our results that this too is not commonly used. The decline in the actual scroll bar use may be due to an increase in the mouse scroll button use.

4.5 Conclusion

Initially, we created a list of all the possible implicit actions that we thought could possibly have a strong relationship with a user's explicit rating and were obtainable with our API calls. We were able to remove certain actions that we found are not commonly used and thus would not be good indicators of the explicit rating of a page due to their lack of occurrence. For example, when we came to the conclusion that searching does not have a high frequency of occurrence, we dropped it from the list of actions. We chose a final list of implicit indicators based on the result of the mini-experiment:

- Typing in URL's;
- Scroll button;
- Scroll bars;
- Following list;
- Reading with mouse.

With these implicit ratings added into our new rating list we are now ready to prepare our browser for use and design our experiment.

Chapter 5: Browser Interface Design

This Chapter discusses the GUI (Graphical User Interface) that we designed for our browser. It also discusses the differences between our GUI and those of the Curious Browser MQP [Le & Waseda, 2000], and the Microsoft Internet Explorer 6.0.

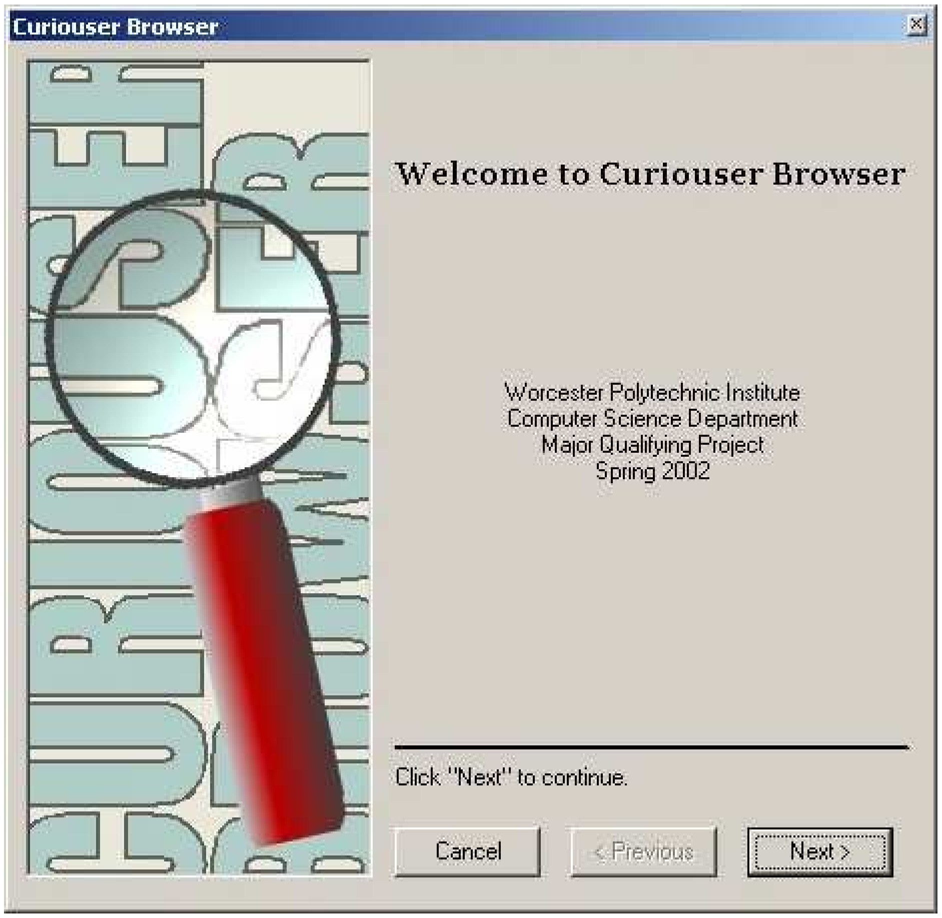
5.1 Interface design

The overall architecture of the Curiouser Browser is similar to the old Curious Browser. The Curiouser Browser is also broken into three parts: Graphical User Interface (GUI), the main Implicit Rating Manager, and the Database. The GUI of the Curiouser Browser is what a user sees and interacts with while using it. The Implicit rating manager captures the specified actions of the users, such as mouse clicks and keyboard buttons, and stores them into memory. After the program closes, the data is then written to the database. The database stores user information including the user name, visited URL, time, date and all the implicit and explicit ratings per URL the user visited. (The Implicit rating manager will be discussed in Chapter 6 and the Database in Chapter 7.)

5.2 Welcome Interface

When a user starts the program, a welcome window appears (See Figure 5.1). It welcomes the user to the program and states that this is a WPI project. There is no minimize or maximize button for this window, so the user must read it or exit the program. At the bottom of the form there are three buttons: “Cancel”, “Previous”, and

"Next". The "Cancel" button is for users to exit the program, so they have the option of not using the Curiouser Browser. When the "Cancel" button is pressed, the program will call a function that will terminate the activities of the program. The "Previous" button is disabled since the welcome window is the very first window of the Curiouser Browser. When the "Next" button is pressed, it displays the next window (Disclaimer window).



Welcome to Curiouser Browser

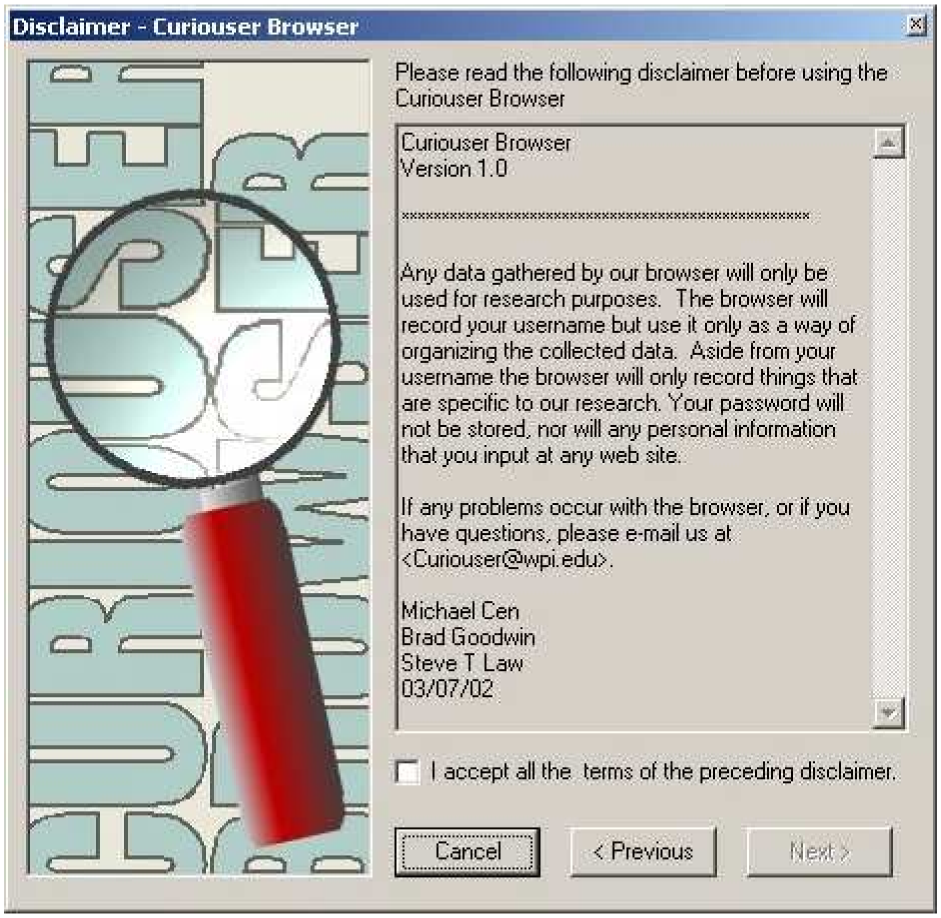
Worcester Polytechnic Institute
Computer Science Department
Major Qualifying Project
Spring 2002

Click "Next" to continue.

Figure 5.1: Welcome Interface

5.3 Disclaimer Interface

In order for us to install the Curiouser Browser on computers in the WPI computer lab, we are required to create a disclaimer that the user must agree to before being able to use the program (See Figure 5.2). It has to give a general explanation of what the Curiouser Browser does and specify that it will store information from the user as he/she browses the Web. It also informs the users that the program does not store any private information, such as passwords. On the right side of the window is the actual disclaimer.


Disclaimer - Curiouser Browser

Please read the following disclaimer before using the Curiouser Browser

Curiouser Browser
Version 1.0

****

Any data gathered by our browser will only be used for research purposes. The browser will record your username but use it only as a way of organizing the collected data. Aside from your username the browser will only record things that are specific to our research. Your password will not be stored, nor will any personal information that you input at any web site.

If any problems occur with the browser, or if you have questions, please e-mail us at <Curiouser@wpi.edu>.

Michael Cen
Brad Goodwin
Steve T Law
03/07/02

☐ I accept all the terms of the preceding disclaimer.

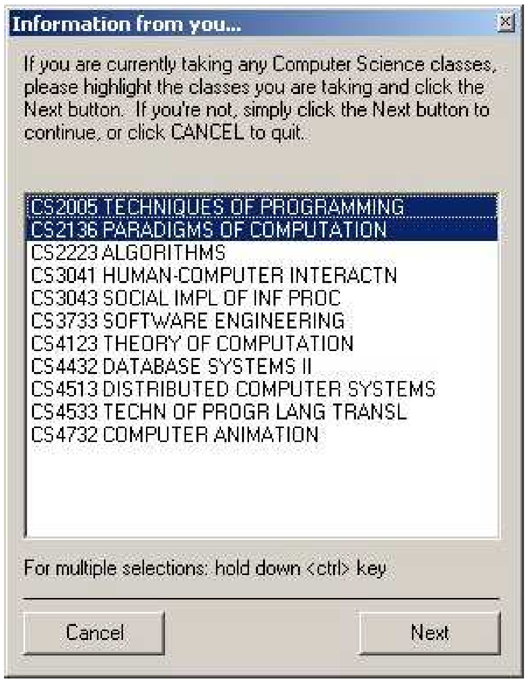
Figure 5.2: Disclaimer Interface

The Disclaimer Interface includes a checkbox for users to click on. This checkbox is a representation of the user agreeing with the disclaimer. The Next button is initially disabled and the checkbox is unchecked. When the checkbox is checked, Next button will be enabled and users can proceed to the next window. By checking the checkbox, the user is agreeing that they have read the disclaimer and understands that some of their information will be stored, but will not be disclosed to the public. As can be seen in Fig. 5.2, the Previous button is enabled. It allows users to go back to the previous window, the welcome screen.

The disclaimer window does not have minimize or maximize buttons, so the user is guaranteed to see this window, and is encouraged to read it. If a user checks the checkbox and then clicks the “Previous” button, the check box will be automatically unchecked and the “Next” button will be disabled again. That way we can make sure that the user should have read the disclaimer before proceeding.

5.4 Class Selection Interface

After a user accepts the disclaimer and clicks the “Next” button, a course selection window will pop up (See Figure 5.3 – 5.5). We asked the professors of all the computer science courses from D term of 2002 to encourage their students to participate in this experiment. Some of the professors even offered their students a bonus point towards the student’s grade for participating. For those professors we added the course selection interface so we could record who are the users and from what current CS course.


Information from you...

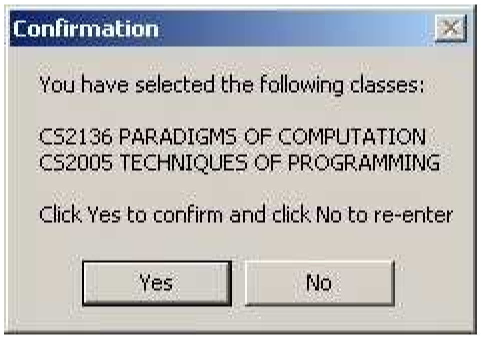
If you are currently taking any Computer Science classes, please highlight the classes you are taking and click the Next button. If you're not, simply click the Next button to continue, or click CANCEL to quit.

CS2005 TECHNIQUES OF PROGRAMMING
CS2136 PARADIGMS OF COMPUTATION
CS2223 ALGORITHMS
CS3041 HUMAN-COMPUTER INTERACTN
CS3043 SOCIAL IMPL OF INF PROC
CS3733 SOFTWARE ENGINEERING
CS4123 THEORY OF COMPUTATION
CS4432 DATABASE SYSTEMS II
CS4513 DISTRIBUTED COMPUTER SYSTEMS
CS4533 TECHN OF PROGR LANG TRANSL
CS4732 COMPUTER ANIMATION

For multiple selections: hold down <ctrl> key

Figure 5.3: Course Selection window

The class selection window lists all the Computer Science (CS) courses that were available during D term of 2002. Since some users might have been taking multiple CS courses at once, the Course Selection Interface allows multiple selections to be made. There are two buttons at the bottom of the window: “Cancel” and “Next”. The “Cancel” button is used to exit the program. The “Next” button is for users to confirm their course selection and checks if the user has selected any courses or not. Before the user can proceed to the next window a confirmation box appears and asks the user to confirm the selected courses and press “Yes” to continue or “No” to re-enter their courses. For example, Figure 5.3 shows that CS2005 and CS2136 are selected. If users click the “Next” button, the message box shown in Figure 5.4, will pop up:


Confirmation

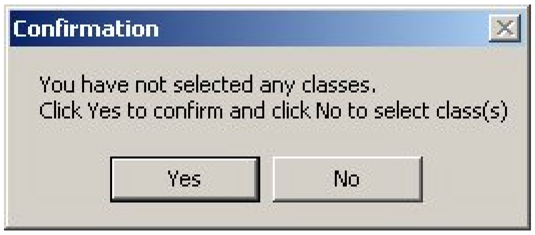
You have selected the following classes:

CS2136 PARADIGMS OF COMPUTATION
CS2005 TECHNIQUES OF PROGRAMMING

Click Yes to confirm and click No to re-enter

Figure 5.4: Confirmation Window with Courses chosen

A confirmation message box is needed so that users can see what they have selected and make sure they did not make any mistake. If users click "Yes", the browser will open and they can start browsing the Web. If the users click "No", the class selection window will reappear. If no class is selected and the user clicks on the "Next" button, the message box shown in Figure 5.5, will pop up:


Confirmation

You have not selected any classes.
Click Yes to confirm and click No to select class(s)

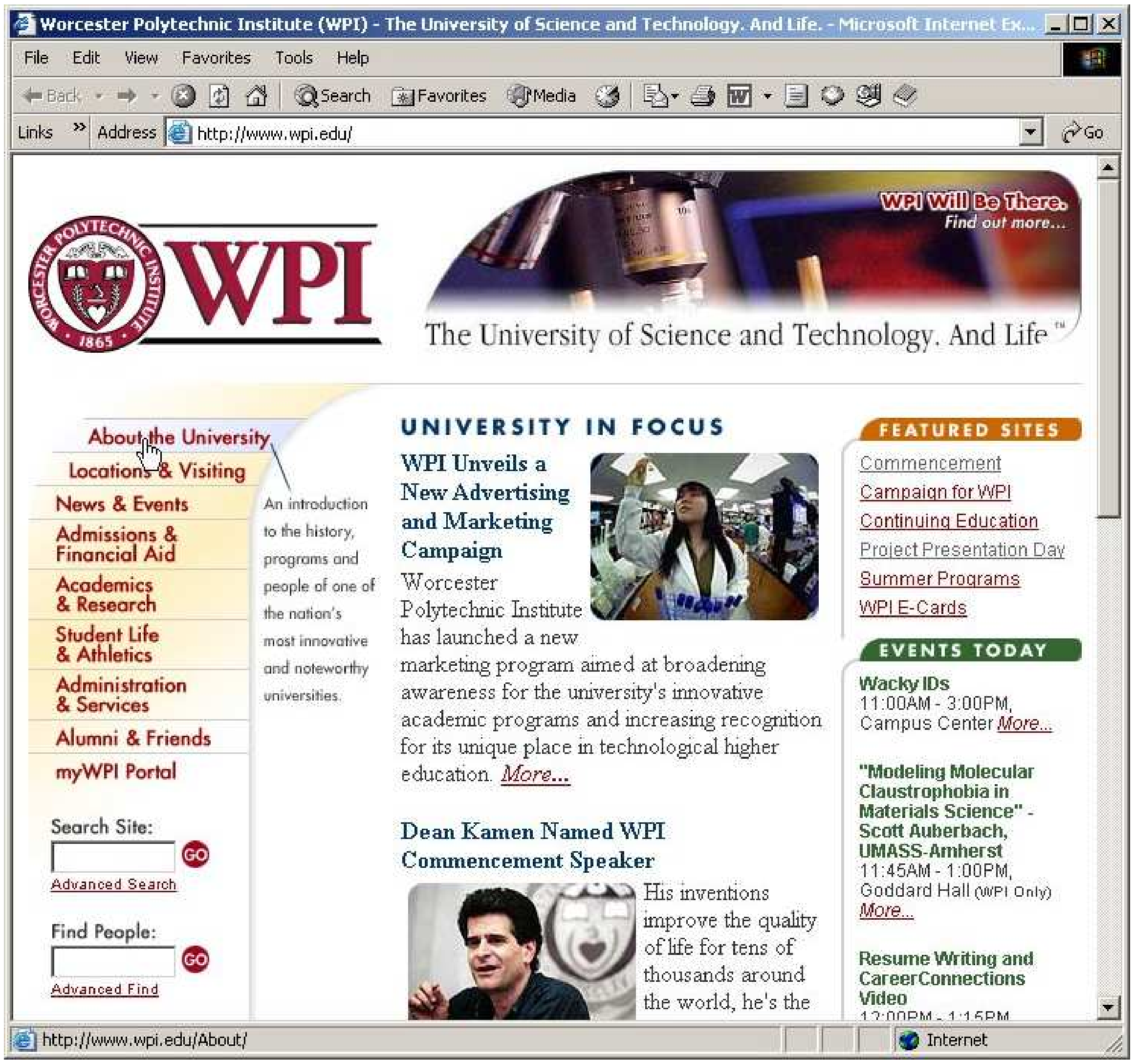
Figure 5.5: Confirmation Window with no Courses selected

If the user clicks "Yes", the browser will open and they can start browsing. If not, the course selection window will reappear allowing the users to select courses again.

5.5 Browser Interface

The main GUI (Graphical User Interface) of the Curiouser Browser has not changed much from the old Curious Browser (See Figure 5.6 – 5.8**)**. We realized that the

GUI would greatly affect users' actions because it is the area where the user interacts with the program. The GUI of the previous project [Le & Waseda, 2000] had its design based on the GUI of Microsoft's Internet Explorer (MSIE) so users will have a familiar GUI with which to interact. If we had significant differences between the GUI of the Curiouser Browser and Internet Explorer, then we might get biased result due to unfamiliarity of Curiouser Browser's GUI. Designing a GUI similar to MSIE will help us eliminate unfamiliarity. For that reason, we also designed the Curiouser Browser based on Microsoft Internet Explorer 6.0 (MSIE 6.0). Figures 5.6 – 5.8 show the GUI of the MSIE 6.0, Curious Browser [Le & Waseda, 2000], and the Curiouser Browser.



The University of Science and Technology. And Life™

An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

UNIVERSITY IN FOCUS

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. More...

Dean Kamen Named WPI Commencement Speaker

His inventions improve the quality of life for tens of thousands around the world, he's the

FEATURED SITES

Commencement
Campaign for WPI
Continuing Education
Project Presentation Day
Summer Programs
WPI E-Cards

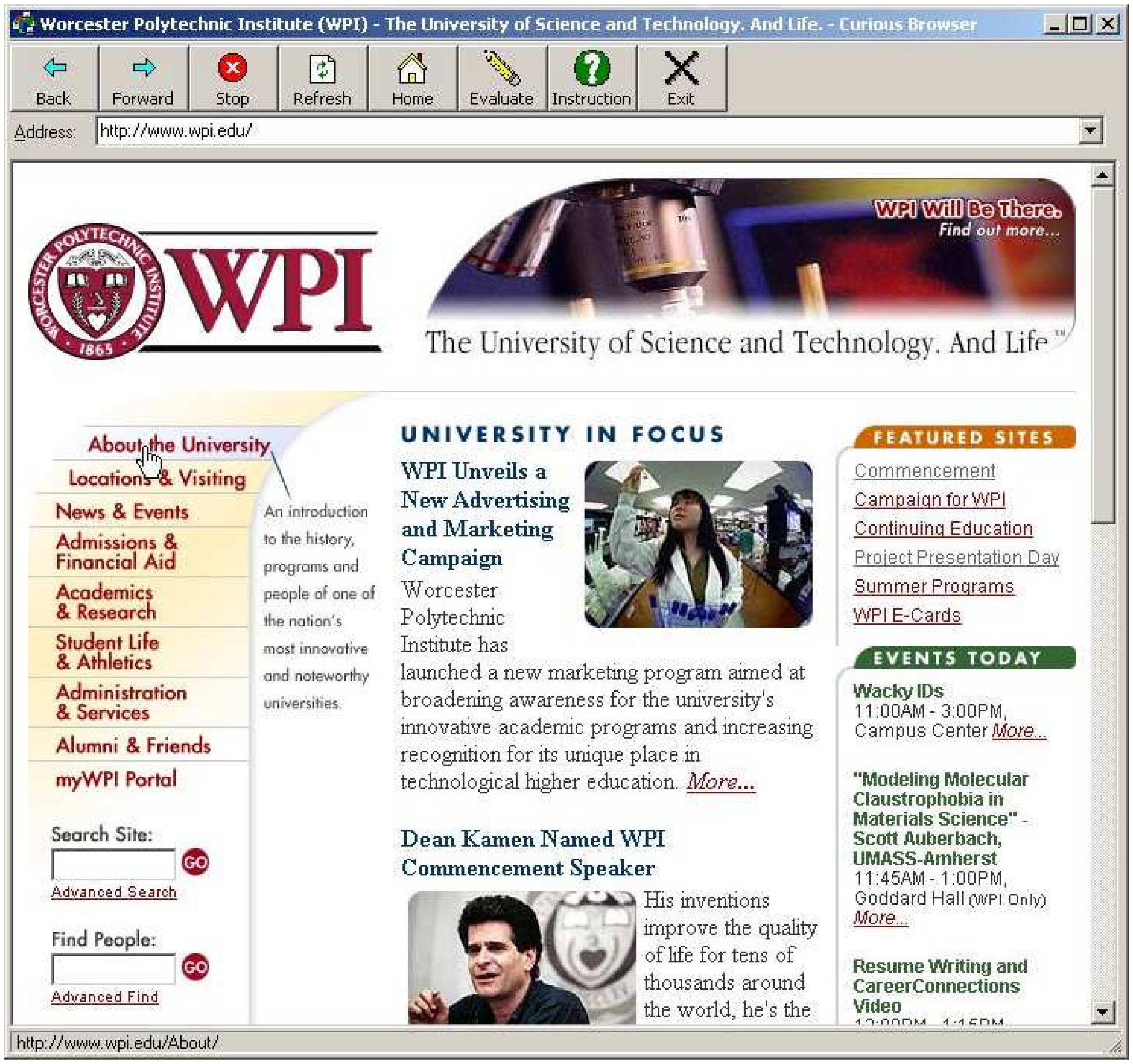
EVENTS TODAY

Wacky IDs
11:00AM - 3:00PM,
Campus Center More...

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auberbach, UMASS-Amherst
11:45AM - 1:00PM,
Goddard Hall (WPI Only)
More...

Resume Writing and CareerConnections Video

Figure 5.6: The GUI of Microsoft Internet Explorer


Worcester Polytechnic Institute (WPI) - The University of Science and Technology. And Life. - Curious Browser

http://www.wpi.edu/

WPI Will Be There. Find out more...

The University of Science and Technology. And Life™

An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

UNIVERSITY IN FOCUS

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. More...

Dean Kamen Named WPI Commencement Speaker

His inventions improve the quality of life for tens of thousands around the world, he's the

FEATURED SITES

Commencement
Campaign for WPI
Continuing Education
Project Presentation Day
Summer Programs
WPI E-Cards

EVENTS TODAY

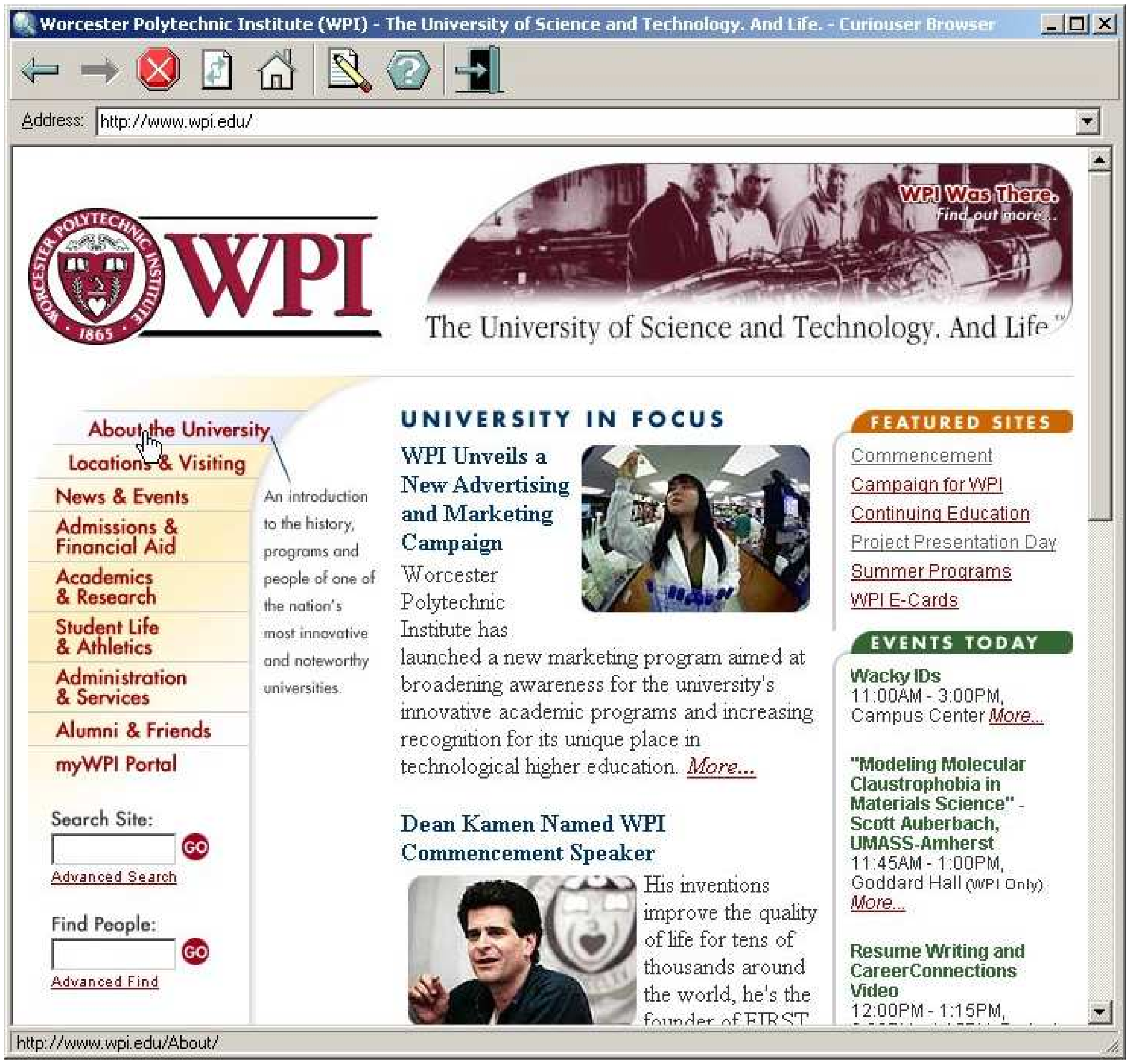
Wacky IDs
11:00AM - 3:00PM, Campus Center More...

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auberbach, UMASS-Amherst
11:45AM - 1:00PM, Goddard Hall (WPI Only) More...

Resume Writing and CareerConnections Video

http://www.wpi.edu/About/

Figure 5.7: The GUI of the Curious Browser


Worcester Polytechnic Institute (WPI) - The University of Science and Technology. And Life. - Curiouser Browser
Address: http://www.wpi.edu/
WPI
WORCESTER POLYTECHNIC INSTITUTE 1865
WPI Was There. Find out more...
The University of Science and Technology. And Life™
About the University
Locations & Visiting
News & Events
Admissions & Financial Aid
Academics & Research
Student Life & Athletics
Administration & Services
Alumni & Friends
myWPI Portal
Search Site:
GO
Advanced Search
Find People:
GO
Advanced Find
UNIVERSITY IN FOCUS
FEATURED SITES
Commencement
Campaign for WPI
Continuing Education
Project Presentation Day
Summer Programs
WPI E-Cards
EVENTS TODAY
http://www.wpi.edu/About/

An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. More...

Dean Kamen Named WPI Commencement Speaker

His inventions improve the quality of life for tens of thousands around the world, he's the founder of FIRST

Wacky IDs
11:00AM - 3:00PM, Campus Center More...

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auberbach, UMASS-Amherst
11:45AM - 1:00PM, Goddard Hall (WPI Only) More...

Resume Writing and CareerConnections Video
12:00PM - 1:15PM,

Figure 5.8: The GUI of the New Curiouser Browser

As described in the previous Curious Browser MQP report [Le & Waseda, 2000], the first five buttons of the Curious Browser from left to right (“Back”, “Forward”, “Stop”, “Refresh”, and “Home”) are identical and in the same order as those of MSIE. The address bar, HTML rendering window, and the status bar are the same to those of MSIE.

The GUI of the Curious Browser was based on the GUI of MSIE 5.0, but that was two years ago. The current version of MSIE is 6.0, but the only significant change from version 5.0 to version 6.0 is not in the GUI, so our basic GUI design does not need to be changed. Keeping the GUI consistent is a good way to eliminate the possibility of users not knowing how to use the browser and becoming a bias factor.

5.6 GUI Differences Between the Three Browsers

The following are differences between the old Curious Browser and our Curiouser Browser. One difference between the two browsers is that the title bar has changed to "Curiouser Browser" from "Curious Browser." Also, the buttons of Curiouser Browser are different from those of Curious Browser. The new buttons are still similar in symbol representation, but the colors and lines of the new icons are different. The images of the buttons are changed because the Curious Browser icons didn't seem to be as noticeable as the new ones we found. The new icons have brighter color and shading, making them more appealing and noticeable. For example, the back and forward arrow has added highlights and shadows. The sizes of the arrows are also bigger.

The "Stop" button is redesigned into an octagon-shaped image. It was first designed to have a word "Stop" inside the octagon so that it would be similar to a stop sign. But we decide to use an X mark instead of "Stop" since the icon on MSIE 6.0 for stop does not have words to it either.

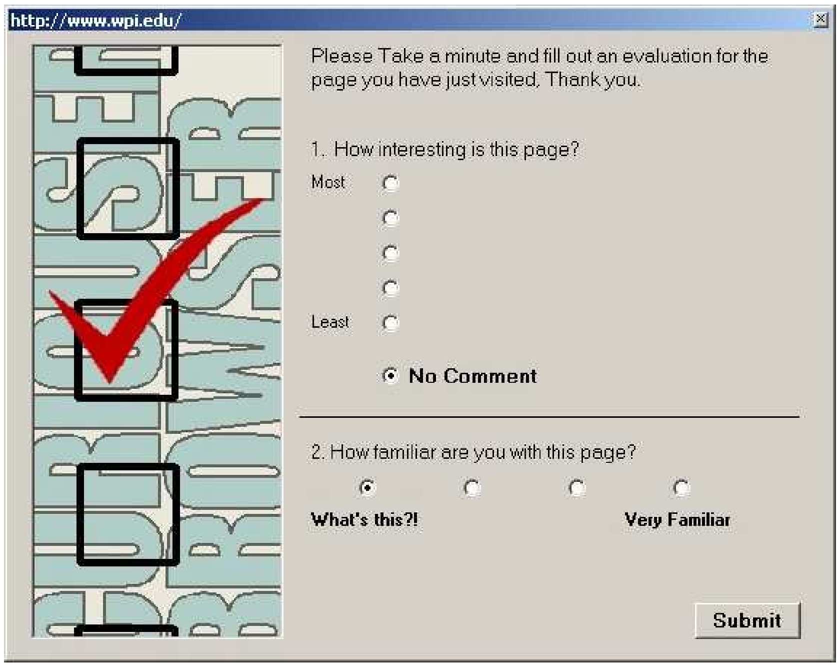
The "Evaluation" button is also redesigned. We chose to have paper and pencil in the image because that is a common symbol to indicate writing. The older icon just has an image of a pencil that more commonly means draw. Having an image that has a pencil and paper might better imply “write something down”, and writing something down is what is needed for an Evaluation.

We changed the shape of the “Instruction” button icon but we kept the question mark inside the shape. By changing the shape of the button and adding edges to it, the button is more noticeable. Users will be able to know where to click if they have a question about the browser since the “question mark” is very well known for its implications.

The “Exit” button is redesigned to be an open doorway with an arrow pointing to the exit. The Exit button from the Curious Browser was just an “X” with the label “EXIT”. This could have been mistaken for stop since “X” also appears in the “Stop” button. The reason why we chose the doorway is that a lot of programs, especially chatting programs, use a door as a symbol to exit makes it easier for users to recognize.

5.7 Evaluation Interface

The GUI of the Evaluation Interface is used to display two explicit ratings, interest rating and the familiarity rating (See Figure 5.9). We have 5-point-scale for the interest rating, with the text “Most” next to the highest rating and “Least” next to the lowest rating. For the familiarity rating, we have a 4 point scale with “Very Familiar” below the highest rating and “What is this??” below the lowest rating to indicate the meaning of the scale. Figure 5.9 depicts the GUI of the Evaluation interface.



Please Take a minute and fill out an evaluation for the page you have just visited. Thank you.

1. How interesting is this page?

Most ○ ○ ○ ○ ○ Least

[✓ No Comment]

2. How familiar are you with this page?

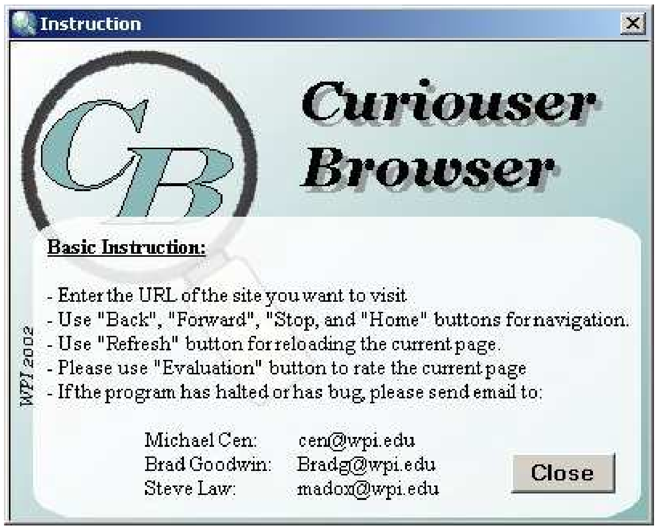
[✓ What's this?!] ○ ○ ○ Very Familiar

Figure 5.9: Evaluation Interface

As shown in Figure 5.9, there are five radio buttons for the explicit interest rating and four radio buttons for the familiarity explicit rating. A “No Comment” radio button is added for users who wish to evaluate the page later or not evaluate one at all. The "No Comment" radio button is the default button for the explicit interest rating. Since we predicted that many of the pages would be skipped, selecting "No Comment" by default make it easier for users to skip the Evaluation window.

5.8 Instruction Interface

The instruction interface contains the same information as that of the Curious Browser (See Figure 5.10). The only difference is that its appearance is more appealing and noticeable. We include each of our email address so that users can contact us if needed. Figure 5.10 below shows the GUI of the Instruction interface.


Instruction

Curiouser Browser

Basic Instruction:

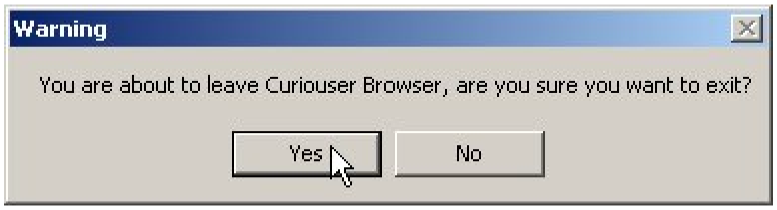
- Enter the URL of the site you want to visit
- Use "Back", "Forward", "Stop, and "Home" buttons for navigation.
- Use "Refresh" button for reloading the current page.
- Please use "Evaluation" button to rate the current page
- If the program has halted or has bug, please send email to:

Michael Cen:	cen@wpi.edu
Brad Goodwin:	Bradg@wpi.edu
Steve Law:	madox@wpi.edu

Figure 5.10: Instruction Interface

5.9 Closing Interface

The closing interface appears whenever a user closes the browser (See Figure 5.11 – 5.12). Since the data is very important to us, we want to make sure that users do not close the Curiouser Browser by accident. To do that, we have a warning window with “yes” and “no” buttons, which is shown below.

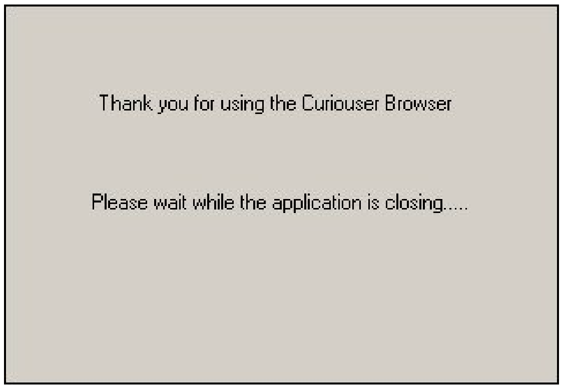

Warning

You are about to leave Curiouser Browser, are you sure you want to exit?

Figure 5.11: Closing Warning Window Interface

If the user clicks “No”, the Curiouser Browser resumes its process. If the user clicks “Yes”, the Curiouser Browser closes and the closing window shown in Figure 5.12 will appear.

During the testing of the Curiouser Browser, we found that the browser seemed to take a while before it completely closes. Since the browser stores data just before the browser closes, we needed to make sure that the browser closed properly. A window appears after the user chooses “yes” in the Closing Warning window, which is shown below. The window is set to stay on the screen for 1000 millisecond (1 second).


Thank you for using the Curiouser Browser

Please wait while the application is closing.....

Figure 5.12: Closing Window

Chapter 6: Algorithms/Implementation

This chapter discusses the changes in Web technology that we had to modify our browser to handle. We also discuss the existing algorithms that were used in the Curious Browser MQP [Le & Waseda, 2000] and the new algorithms that we created to modify the Curious Browser to capture new implicit ratings.

6.1 Motivation for Changes

It has been two years since the Curious Browser experiment was conducted and there have been many changes in Web technology in that time. Web sites that were built two years ago were much less complicated than they are today. At that time most households had slow Internet connections. Currently more areas are getting faster Internet connections installed such as DSL and Cable modem services. This increase in bandwidth also brings a growth in technology that allows Web pages to make their sites more graphical and interactive.

It is because of these changes over the past two years that modifications had to be made to the old Curious Browser. We also had to modify some of the basic functionality of the browser in order to detect the new implicit indicators. We had to ensure that the changes will not affect the browser's performance, which may directly or indirectly affect the experiment.

The following subsections discuss three of the changes on the Internet over the past two years that the Curious Browser was not built to accommodate.

6.1.1 Pop-up Windows

The use of pop-up windows in the past two years had grown exponentially. Two years ago they usually appeared with Web sites that were not frequently being accessed by users. Major Web sites such as weather.com, AOL.com, and NBC.com did not have pop-up windows. Presently, pop-up windows are actually being used mostly for advertisements, but they are also used to emphasize Web site contents. Now visiting one Web page may end up with five new browser windows opened on the screen.

We address this issue in our project by giving users an option to choose whether or not they want to view the pop-up window. Before they choose, we warn them that the pop-up windows are not fully supported by the browser.

6.1.2 Site formats

Framed pages can be very convenient for users, since information, such as menus, can consistently be in front of the users while they are surfing another part of a Web page. A framed Web page potentially contains many HTML pages: the main page is the structural page, which sets up the sizes of the framed pages. The other pages are often the menu page and the main page. Therefore, loading one framed page is actually like loading at least three HTML pages at the same time.

Two years ago, not all Internet browsers support framed pages. In order for every user to receive the same content, framed pages were rarely used in most Web sites. Today, most Web browsers support framed pages, therefore, more Web sites are starting to use framed pages.

This has been a problem for the Curious Browser since it does not recognize framed pages and the Evaluation Window pops up at least three times when a framed page is loaded. To avoid that from happening in this experiment, we modify the algorithm of the Evaluation Window pop up, which is discussed in Section 6.3.7 of this chapter.

6.1.3 Wheel Mouse

A wheel mouse has a wheel-like central button, called the “mouse wheel button”. The mouse wheel button was a new computer interface hardware design, but it has quickly become a very convenient scrolling tool. Users can rotate the mouse wheel forwards and backwards or press down on it for different styles of scrolling. When the wheel is rolled forwards and backwards, the scrolling motion of the page is vertical while the cursor of the mouse remains at the same position relative to the boundaries of the browser window. If the wheel button is pressed down, the user is then able to scroll in all eight possible directions (Up, Down, Up-right, Down-left, etc) on a two dimensional plane. Due to its convenience, the mouse wheel button may substitute for all other scrolling methods, such as scroll bars and arrow buttons.

To address this issue, we created new implicit ratings for all functions related to the mouse wheel button. They are discussed in Section 6.3.2 of this chapter.

The three changes listed above are only some of the major changes to Web design over the past two years. To adapt to these changes, we modified the Curious Browser in order to continue with the experiment. Some algorithms used in the browser were also modified, and new algorithms were designed to accommodate current Web site presentations. After all these modifications, the Curious Browser became the Curious***er*** Browser.

6.2 Existing Algorithms

The Curiouser Browser includes all nine of the implicit ratings used in the Curious Browser. They are listed below:

- Time spent on the Web page – total time spent on surfing the Web page;
- Duration of the vertical and horizontal scroll bars usage – how long each scroll bar is used;
- Duration of mouse usage – how long the mouse is used;
- Number of Mouse clicks – Number of clicks on the Web page;
- Number of Up, Down, Left, and Right Arrow keys usage – Number of times each arrow key is used;
- Duration of Up, Down, Left, and Right Arrows keys usage – how long each arrow key is used;
- Number of Page Up and Page Down keys usage - Number of times each Page key is used;
- Duration of Page Up and Page Down keys usage - how long each Page key is used;
- Explicit Interest Rating – User's rating after surfing a Web page.

The following subsections discuss briefly the algorithms that handle the implicit ratings listed above.

6.2.1 Mouse Activities

The old Curious Browser has implicit ratings that trace mouse activities. It captures the number of left-mouse button clicks and the time spent on moving the mouse. However, it only captures information about the mouse when the browser window is in focus. A window is in focus when that window is selected for use. It also means that the user is currently interacting with the browser window, as opposed to some other window on the screen.

The Curiouser Browser window consists of the HTML rendering area, the vertical scroll bar, and the horizontal scroll bar. No mouse activity is recorded if the mouse cursor is out of the browser window.

In the Curiouser Browser, we enhance the mouse activities' algorithm to get additional implicit ratings from users. The Curiouser Browser can capture right mouse clicks, mouse wheel scrolling, and cursor coordinates. These new implicit ratings are only recorded if the browser is in focus. Figure 6.1 shows the state diagram of mouse activities.

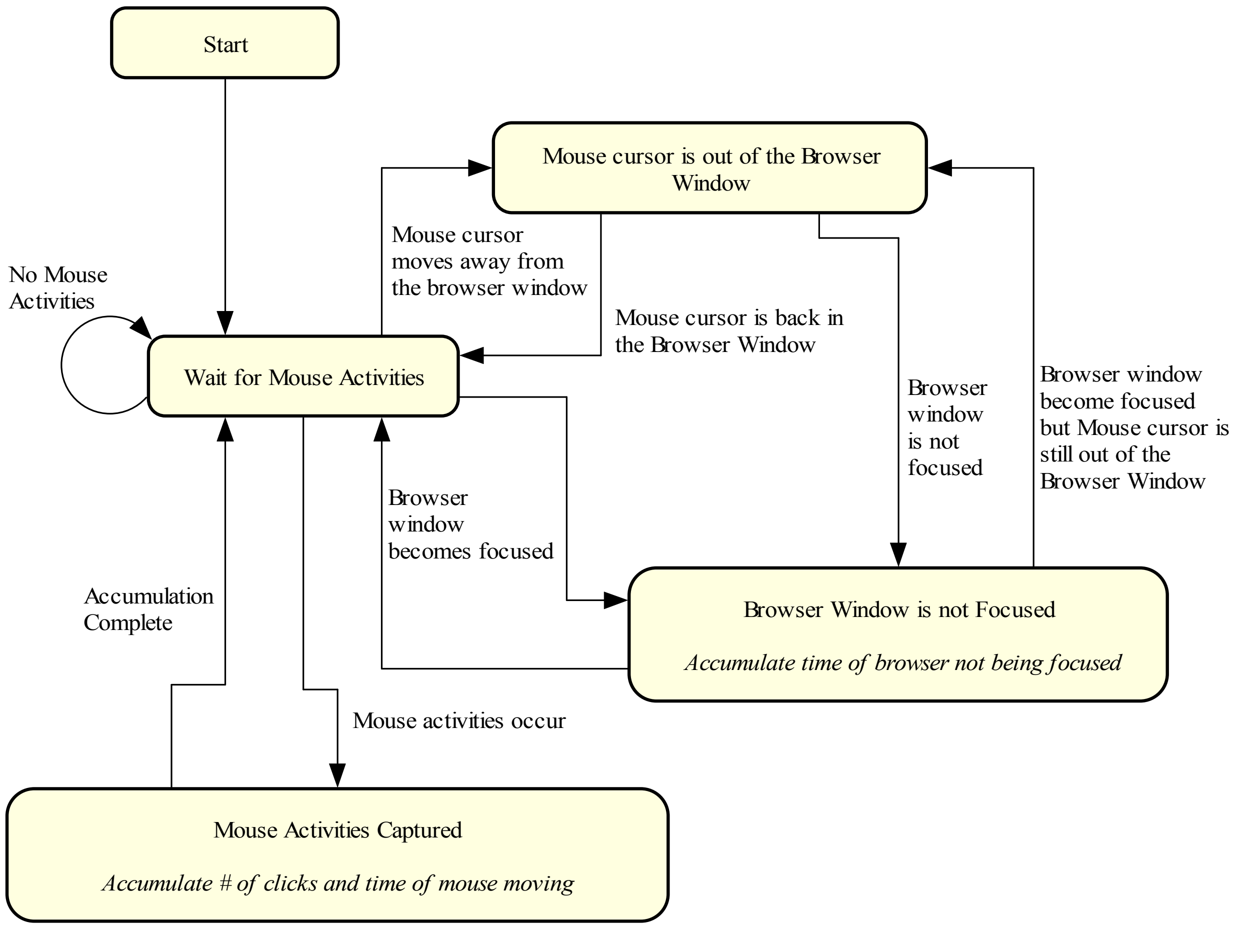
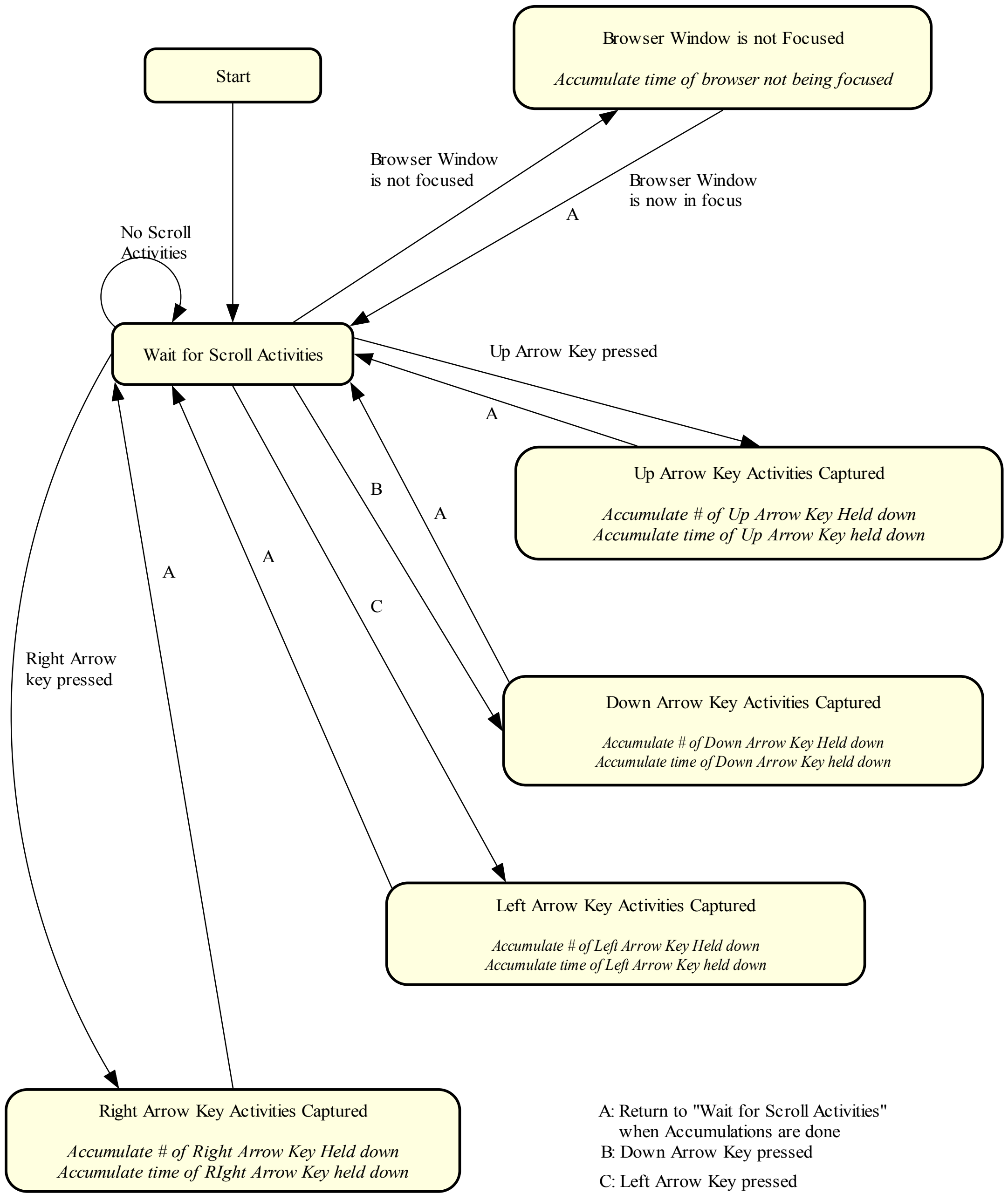

Start
Mouse cursor is out of the Browser Window
Mouse cursor moves away from the browser window
No Mouse Activities
Mouse cursor is back in the Browser Window
Wait for Mouse Activities
Browser window is not focused
Browser window become focused but Mouse cursor is still out of the Browser Window
Browser window becomes focused
Accumulation Complete
Browser Window is not Focused
Accumulate time of browser not being focused
Mouse activities occur
Mouse Activities Captured
Accumulate # of clicks and time of mouse moving

Figure 6. 1: State Diagram for Capturing Mouse Activities

6.2.2 Keyboard Activities

Four keys are monitored by the Curiouser Browser to obtain keyboard-related implicit ratings: Page Up, Page Down, Up Arrow and Down Arrow. Their activities were also detected in the Curious Browser. These four keys are used when users want to scroll without using a mouse.

There are two different types of implicit ratings for each of the four keys. One is the number of times that users press down on the keys, and the other is the amount of time that these keys are held down. The monitored area is similar for those of mouse activities. The Curiouser Browser does not capture anything from the keyboard when the browser window is not in focus. However, it still captures keyboard activities even the mouse cursor is out of the browser window, since users can still scroll using the keyboard. Figure 6.2 shows the state diagram of keyboard activities.



Start
Browser Window is not Focused
Accumulate time of browser not being focused
Browser Window is not focused
Browser Window is now in focus
A
No Scroll Activities
Wait for Scroll Activities
Up Arrow Key pressed
A
Up Arrow Key Activities Captured
Accumulate # of Up Arrow Key Held down
Accumulate time of Up Arrow Key held down
B
A
A
A
C
Right Arrow key pressed
Down Arrow Key Activities Captured
Accumulate # of Down Arrow Key Held down
Accumulate time of Down Arrow Key held down
Left Arrow Key Activities Captured
Accumulate # of Left Arrow Key Held down
Accumulate time of Left Arrow Key held down
Right Arrow Key Activities Captured
Accumulate # of Right Arrow Key Held down
Accumulate time of RIght Arrow Key held down
A: Return to "Wait for Scroll Activities" when Accumulations are done
B: Down Arrow Key pressed
C: Left Arrow Key pressed

Figure 6. 2: State Diagram for Capturing keyboard Activities

6.2.3 Explicit Rating of Interest

The Explicit Interest Rating is used again in the Curiouser Browser. There are two situations that trigger the Evaluation window to pop up. The first is when there is a page change, meaning a change in the URL. The other one is when the "Evaluation" button, located in the tool bar, is pushed. In the Curiouser Browser, we modified the Evaluation window so that the users are asked to rate the page only when they type a URL address directly into the Address Bar and hit the Enter key, or select a link on the Web page.

6.2.4 URL, User name, Time/Date Stamp

The purpose of having an algorithm to retrieve the URL, User's name, Time, and Date stamp is to distinguish between the data entries. This algorithm was already contained in the Curious Browser. The Curiouser Browser uses this algorithm in the same way that the Curious Browser uses it. The user name and the date are captured when the user starts the Curiouser Browser. Every time a new Web page is loaded, the URL and the time (HOURS: MINUTES: SECONDS) are recorded into memory.

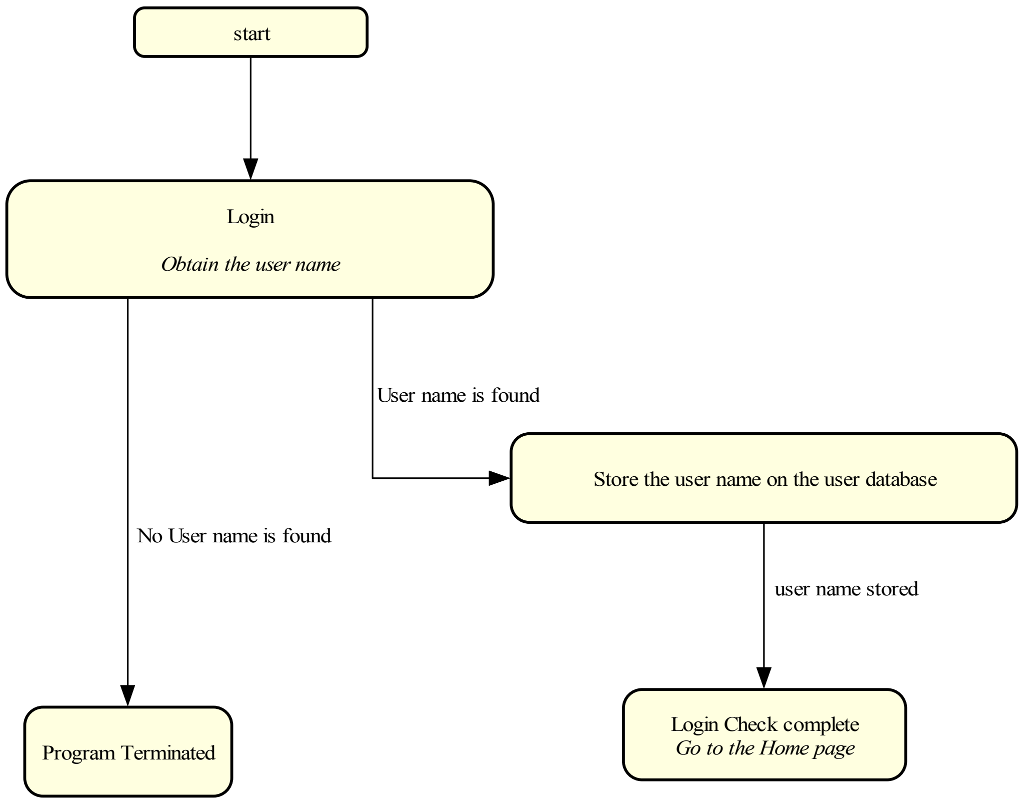


Figure 6.3: State Diagram for Capturing the User Name

As shown in Figure 6.3, if a user name is not found in the system, the Curiouser Browser will terminate. This happens when a user uses the Curiouser Browser without logging in. All computers in the ADP lab use the Microsoft Network. Since users can avoid logging in by pressing the "Escape" button in the login screen and are still able to use the computer, the browser needs to check for the user name in the Microsoft Network to make sure users do login to the computer. If no user name is found, the browser will close. When the user name is captured, the browser also captures the date.

6.3 New Algorithms

Listed below are the eleven new implicit indicators that are added into the Curiouser Browser.

- Number of Right Mouse button clicks – Number of times the right mouse button is used;
- Number of Mouse wheel scrolls – Number of times the mouse wheel is rolled;
- Time spent on scrolling with mouse wheel button held down – Amount of time the mouse wheel is held down;
- Number of Mouse Wheel Button clicks – Number of times the mouse wheel button is used;
- Mouse Coordinates – Mouse's (X, Y) cursor coordinates;
- Number of status bar changes – Number of times the mouse cursor is on link;
- Number of copy command occurrences – Number of times the COPY command is called;
- Size of a HTML file – The size of a HTML file;
- Number of times a URL is entered in address bar – URL is entered in address bar to a new page;
- Number of times a link is selected – Number of times a link is clicked to a new page;
- Explicit Familiarity Rating – User's familiarity to the page in 4-point scale (discussed in Chapter 5.7);

The following subsections discuss briefly on the algorithms of the implicit indicators listed above.

6.3.1 Amount of Right Mouse Button Usage

This implicit rating captures the number of times that the right mouse button is pressed. It is not necessary to capture the time spent using the right mouse button since normally no user would hold the right mouse button down for a long period of time. As mentioned in the section above, the Curiouser Browser does not capture any mouse activity if the mouse is out of the browser window or when the browser window is not in focus. The number of right mouse clicks is accumulated per Web page.

6.3.2 Mouse Wheel Activity

Besides using Page up, Page down, Up Arrow, Down Arrow and dragging the scroll bar to scroll, users can also use the mouse wheel to scroll. As mentioned in Section 6.1.4, the mouse wheel button plays a very important role in Web site surfing because of its convenience in scrolling. The Curiouser Browser captures three activities from the mouse wheel:

6.3.2.1 Number of times the Mouse wheel is rolled

Users may scroll the HTML page up and down by simply rolling the mouse wheel forward or backward.

6.3.2.2 The number of times the mouse wheel is clicked

It records the number of times users clicking the mouse wheel once to scroll the page up and down by moving the mouse forward and backward.

6.3.2.3 The time spent holding down the mouse wheel

It records the amount of time users holding the mouse wheel down to scroll the page up and down by moving the mouse forward and backward.

Similar to all the other mouse activities, no mouse wheel activity is recorded if the mouse is out of the browser window or the browser window is not in focus. The amount of time that the mouse wheel is held down is recorded in milliseconds.

6.3.3 Mouse Coordinates

The Mouse Coordinates Implicit rating captures two kinds of information: the X and Y coordinates of the mouse cursor, and the idle time of the mouse (the amount of time the mouse stays at the same coordinates). Similar to all the other mouse implicit ratings, no data is stored if the mouse is not inside the browser window or the browser window is not in focus.

The Curiouser Browser records the X and Y coordinates of the mouse cursor every millisecond when the mouse is moving. It compares the new coordinates with the old coordinate recorded temporarily 1 millisecond ago to see if the mouse is idle. When the mouse is not moving for 1000 milliseconds, the browser will start calculating the idle time. The unit of idle time of the mouse is milliseconds.

6.3.4 Status Bar Change

Whenever a user points at a link with the mouse cursor, the information about that link shows in the status bar. The status bar at the bottom of the Curiouser Browser works the same as that of a regular Microsoft Internet Explorer. It shows the URL of the link that the mouse cursor is pointing at and tells the user if a Web page is loading. Information changes in the status bar can tell us how many times the cursor is on a link. In order to correctly capture the data about when the cursor is on a link, a timer is used in this algorithm. Whenever the content of the status bar changes, it triggers a 500-millisecond timer. After 500ms, the status-bar-change counter is incremented by one. A 500 millisecond timer is to detect that a user intended to place the mouse cursor on a link. We use 500 milliseconds as the timeframe for the status bar change implicit rating based on the following formulas [Card et al, 1983,p.26]

Eyes movement = 230 | 50 ~ 200 | ms	Speed of eye movement – moving eyes from Web page to the status bar
T_P = 100 | 50 ~ 200 | ms	Speed of perceptual process – focusing onthe status bar
δ_{vis} = 200 | 70 ~ 1000 | ms	Speed of Storing visual image – recognizing the information in working memory provided in the status bar

We added the base numbers together: 200 + 100 + 230 = 530 (ms)

Therefore, 500 millisecond is a good amount of time for users to look at the status bar to view the content of the link, which can mean that they are interested in the link. But if the cursor stays on the link shorter than 500 milliseconds, it can mean that the user only passes by the link with the mouse cursor. Figure 6.4 shows the state diagram of status bar changes.

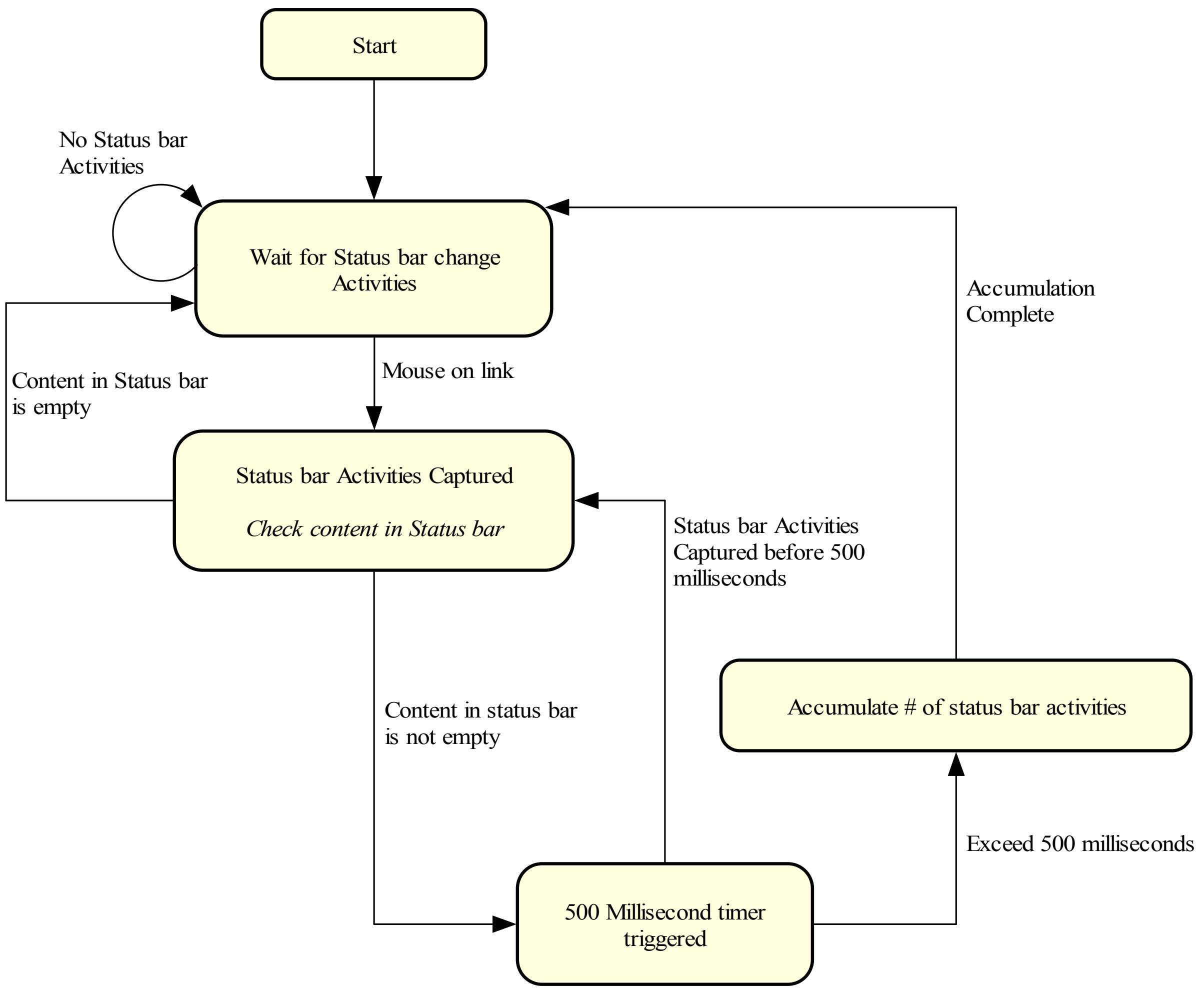

Start
No Status bar Activities
Wait for Status bar change Activities
Accumulation Complete
Mouse on link
Content in Status bar is empty
Status bar Activities Captured
Check content in Status bar
Status bar Activities Captured before 500 milliseconds
Accumulate # of status bar activities
Content in status bar is not empty
Exceed 500 milliseconds
500 Millisecond timer triggered

Figure 6. 4: Sate Diagram for Capturing Status Bar Changes

6.3.5 Number of Copy Commands

There is an embedded user control (a stand-alone program that can only be used by the program that hosts it) called the Clipboard viewer inside the Curiouser Browser. Whenever information is copied, it is saved in an area of memory called a clipboard in the system. The clipboard viewer allows us to view the copied contents. In order to capture a number that shows the correct amount of copy usage, the copy counter will only increment when the copy system call is triggered while the browser window is focus. Although we do not have the copy option available in the menu bar, users can still copy page contents by pressing Ctrl-C, Ctrl-insert, and choosing the copy option in the right-clicked menu bar. Figure 6.5 shows the state diagram of copy activities.

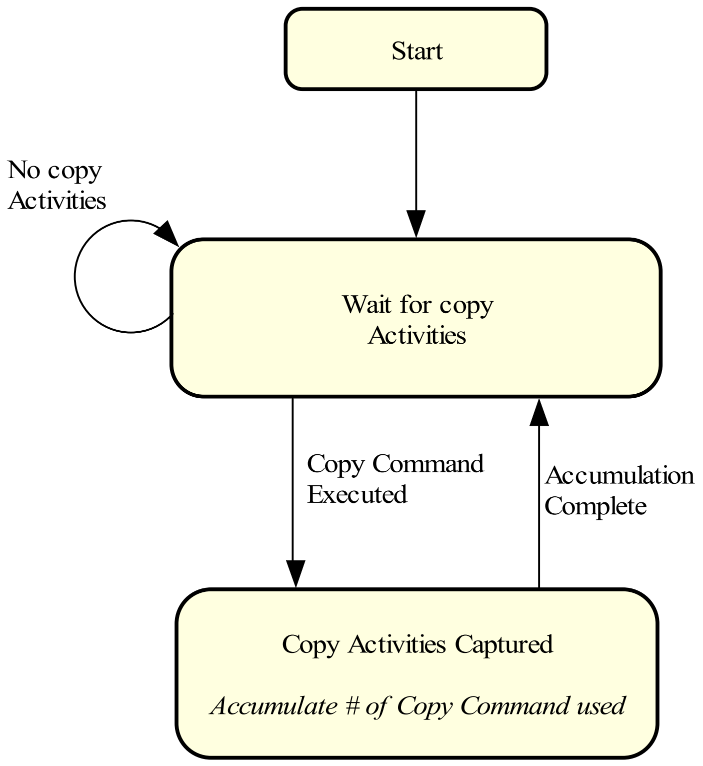


Figure 6. 5: State Diagram for Capturing Copy Activities

6.3.6 Size of the HTML File

This indicator is used to record the size of an HTML file after the page is finished loading. The size of an HTML file is the total size of all the contents that appear in the browser window including text and style sheets, but not graphical files. We hope to establish correlations with other implicit ratings such as the amount of scrolling and the amount of time spent on a page.

In order for us to retrieve the size of the HTML file, we need to wait until the page has loaded. When a page has finished loading, the event "NavigateComplete()" is called. However, since the browser control does not have the ability to get the size of the HTML file directly, we need to send the file to an IHTMLDocument2 interface. This type of interface allows us to retrieve information about the document [Microsoft, 2002]. After the HTML file is sent to an IHTMLDocument2 in "NavigateComplete()", the Curiouser Browser retrieves the size of the file from IHTMLDocument2 and stores it in memory. IHTMLDocument2 has to be empty before a new page is sent to it. When the event “beforeNavigate2()”, the event that is triggered every time before the browser starts to navigate to a page, is called, IHTMLDocument2 is reset to “EMPTY” so that a new page can be sent to it when the event “NavigateComplete()” is called. Figure 6.6 shows the state diagram of the size of a HTML file rating.

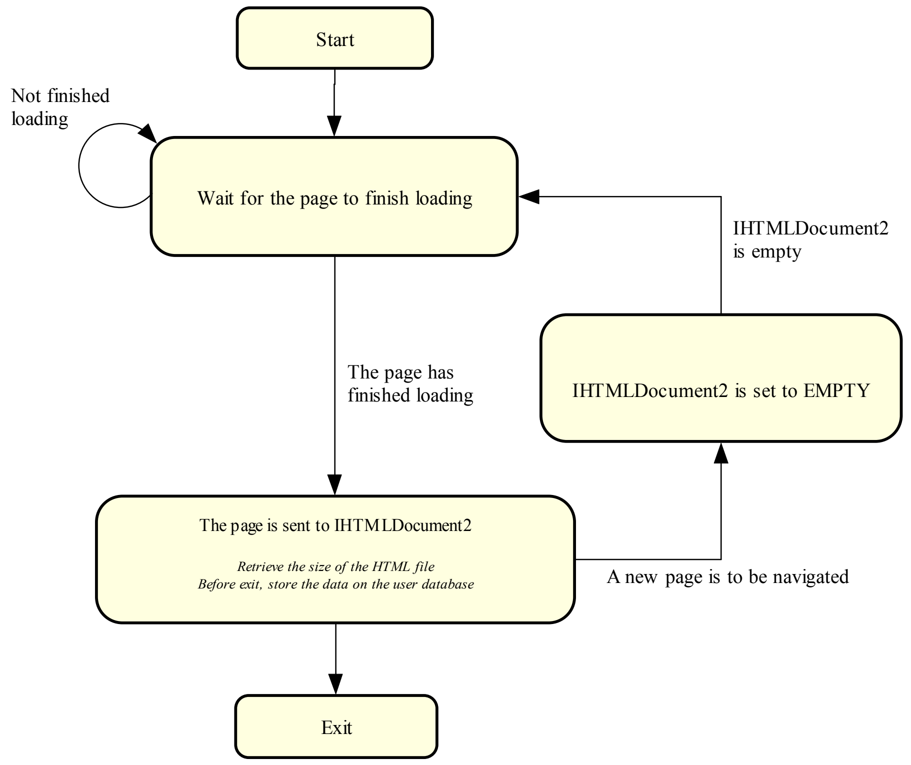

Start
Not finished loading
Wait for the page to finish loading
IHTMLDocument2 is empty
The page has finished loading
IHTMLDocument2 is set to EMPTY
The page is sent to IHTMLDocument2
Retrieve the size of the HTML file
Before exit, store the data on the user database
A new page is to be navigated
Exit

Figure 6.6: State Diagram for getting the size of the HTML file

6.3.7 Triggering the Evaluation Window

In the Curious Browser, the evaluation window pops up whenever the browser is going to navigate to a new URL. That will cause the evaluation window to pop up numerous times when surfing a Web page that is made up of many URL's. For example, a two-framed page consists of three pages: the frame page, menu page, and the main page, but to users, they are only viewing one page. For other pages, the evaluation window will also be triggered if the pages want to load information from other URL's. A new algorithm had to be designed to counter these problems.

Users only have to rate the page that they intend to visit. Whenever the users select a link or enter a URL in the address bar in order to go to another Web page, this is considered as an “intent to visit.” In order to capture the correct amount of data, flags are needed in the Evaluation window algorithm. The flag “enterURL” is turned on when users press enter when the address bar is in focus. The flag “LMOUSEup” is turned on when the left mouse button is up on a link. When the “beforeNavigate2()” event is triggered, the LMOUSEup flag and the enterURL flag are checked to see if they are on or not. If one of the flags is on, the click-URL counter or the URL entry counter will be incremented respectively, a flag called “firstNavigate” will be turned on, and the LMOUSEup flag and the enterURL flag will be turned off. When the “beforeNavigate2()” event is triggered, it checks if the firstNavigate flag is on or not. If it is on, that indicates that data now needs to be stored into memory, and the evaluation window should pop up. If it is off, that indicates that the page to be navigated is not intended by the user, and evaluation window should not pop up. Figure 5.18 shows the state diagram of the explicit interest rating.

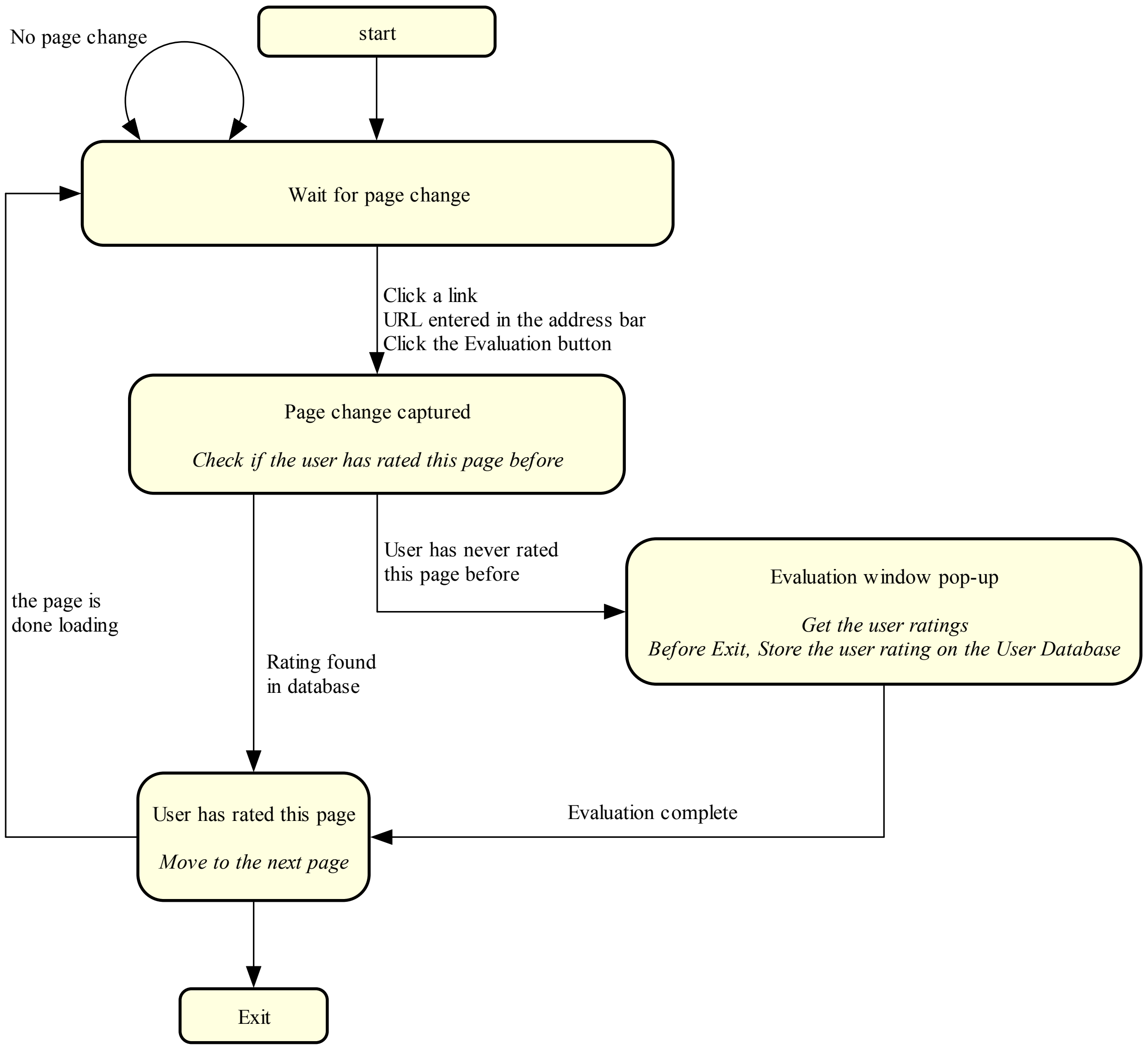


Figure 6. 7: State Diagram for getting Explicit Interest Rating

6.4 Implementation

The Curiouser Browser was developed in Visual Basic 6.0. In order to explain the implementation, we need to explain it by its main activities since Visual Basic is an Event-Driven Language. Some of the activities mentioned in the algorithm sections had been done by the Curious Browser, and a few of them are minor modifications of the implicit ratings used by the Curious Browser. Therefore, only the two implementations and other related implementations are included in this section. The new implementations are Mouse activities, which include Mouse Coordinates and Mouse Button activities. The related implementations are detecting when the browser is not in focus, and detecting when the mouse cursor is out of the browser window.

6.4.1 Mouse activities

All the new mouse activities are captured in the same event procedure and the same function, although there is a similar function already available in the Curious Browser. To avoid complexities and possible bugs, all new procedures were created as new functions, if possible.

6.4.1.1 Mouse Coordinates

The new timer "timer1_Timer()" is a event created for all the new mouse activities and other new modifications. It is mainly used to capture mouse coordinates. The pseudocode for the Mouse Coordinates in the "timer1_Timer()" is:

```
Private Sub timer1_Timer()
```

```
    This object is called every one millisecond.
    Do if “browser window is active” is true and “the mouse is out of the
    browser window” is false
    1) Store the current X and Y into rX and rY, then compare rX and rY with
    the temporary variables, tX and tY
    2a) if rX is equal to  tX AND rY is equal to tY then a 1000-millisecond timer
    “IdleTimer()” is activated
    if “Idletimer()” exceeded 1000 milliseconds,  program starts recording the
    idle time of the mouse until the mouse moves again, if not, idletimer() is
    deactivated and jump back to 1).
    2b) if rX is not equal to tX or rY is not equal to tY then rX and rY are stored
    into the temporary database for the mouse coordinates, and then
    rX is set to equal to tX and rY is set to equal to tY
```

6.4.1.2 Mouse right click and mouse wheel

In order to capture mouse right button and mouse wheel clicks, the existing function named “MouseProc()” was modified. In the Curious Browser, all mouse click activities and Scroll event activities were done in this function. MouseProc is a hook in the Window API for observing mouse portions of the window message stream. The new portion of the pseudocode inside “MouseProc()” is:

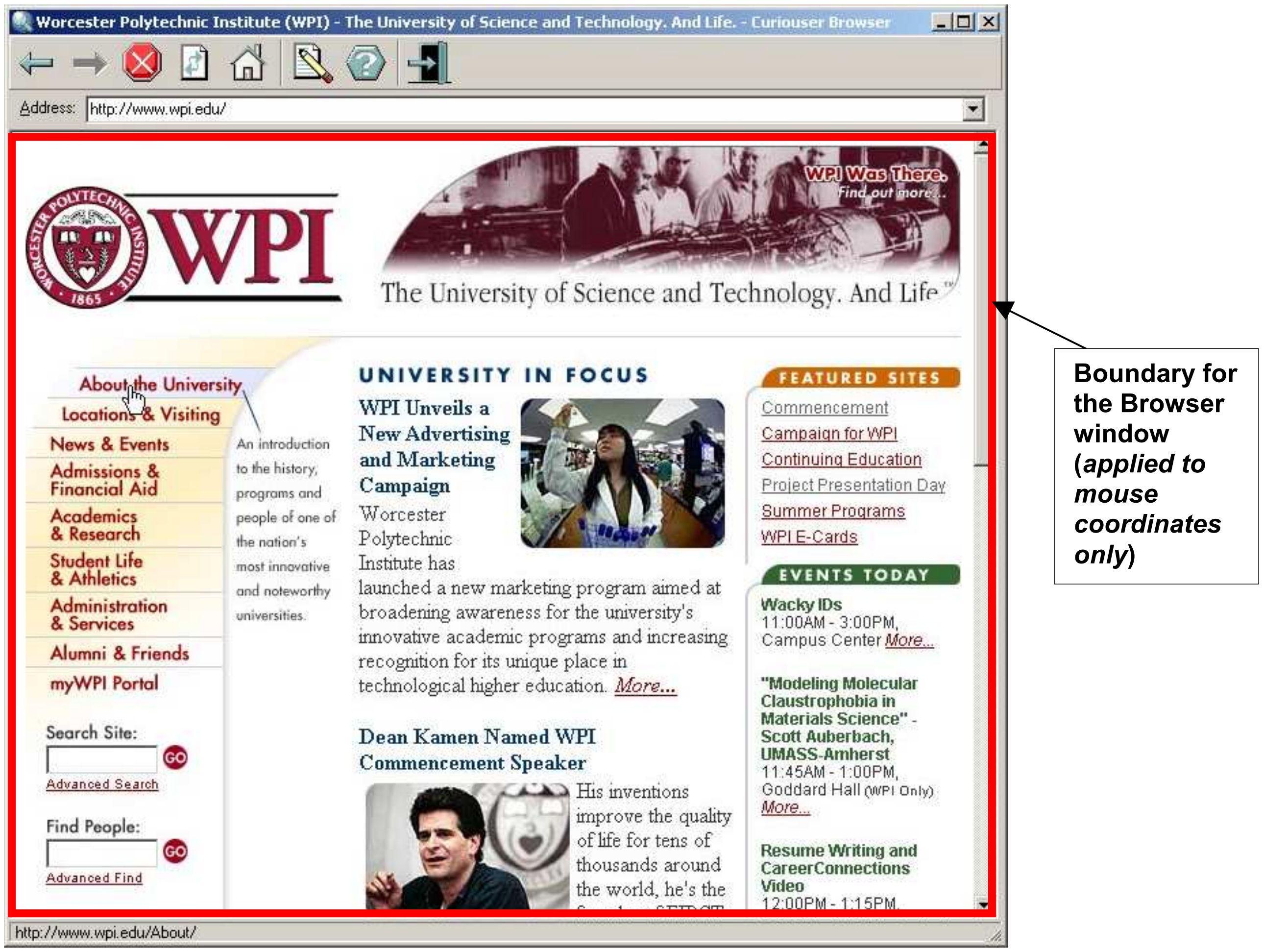
```
Public Function MouseProc()
    //This function is called whenever a message from mouse is captured by
    the program
    Do if “Browser window is not in focus” is false
    1) Accumulate the number of right mouse clicks if the right mouse button
    is released inside the browser window
    2) Accumulate the number of mouse wheel clicks if the mouse wheel
    button is released inside the browser window
    2) Accumulate the number of mouse wheel roll, if the mouse wheel button
    is rolled inside the browser window
    3) Accumulate the time of mouse wheel clicks, if the mouse wheel button
    is pressed down inside the browser window
```

6.4.2 " Browser Window is not in focus" Activities

"Browser Window is not in focus" activities are captured using the function named "*GetActiveWindow*". "GetActiveWindow" is also a Window API and it has already been used in the Curious Browser. This function retrieves the window handle of the active window associated with the calling thread's message queue, which is the Curiouser Browser.

6.4.3 " Mouse Cursor is out of the Browser Window" Acitvities

In order for "Mouse is out of the Browser Window" Activities to be captured, a boundary is needed around the browser window. Thus, if the mouse moves out of the browser window, the activity is false.


Worcester Polytechnic Institute (WPI) - The University of Science and Technology. And Life. - Curiouser Browser
Address: http://www.wpi.edu/
WPI Was There. Find out more...
The University of Science and Technology. And Life™
About the University
Locations & Visiting
News & Events
Admissions & Financial Aid
Academics & Research
Student Life & Athletics
Administration & Services
Alumni & Friends
myWPI Portal
Search Site:
Advanced Search
Find People:
Advanced Find
UNIVERSITY IN FOCUS
FEATURED SITES
Commencement
Campaign for WPI
Continuing Education
Project Presentation Day
Summer Programs
WPI E-Cards
EVENTS TODAY
http://www.wpi.edu/About/
Boundary for the Browser window (*applied to mouse coordinates only*)

An introduction to the history, programs and people of one of the nation's most innovative and noteworthy universities.

WPI Unveils a New Advertising and Marketing Campaign

Worcester Polytechnic Institute has launched a new marketing program aimed at broadening awareness for the university's innovative academic programs and increasing recognition for its unique place in technological higher education. More...

Dean Kamen Named WPI Commencement Speaker

His inventions improve the quality of life for tens of thousands around the world, he's the

Wacky IDs
11:00AM - 3:00PM, Campus Center More...

"Modeling Molecular Claustrophobia in Materials Science" - Scott Auberbach, UMASS-Amherst
11:45AM - 1:00PM, Goddard Hall (WPI Only) More...

Resume Writing and CareerConnections Video
12:00PM - 1:15PM

Figure 6. 8: Boundary (in red) setup for the Mouse Coordinate implicit rating

As shown in Figure 6.7, the red boundary line surrounds the browser window. No data is recorded if the mouse cursor is outside of the red boundary line. Mouse coordinates that are outside of the red boundary line is irrelevant to our research since the browser window is where the HTML page is rendered and that is where the information comes from.

Chapter 7 Experiment Design

In this chapter, we will discuss how the experiment was conducted and reasons for some of the decisions we have made for the Curiouser Browser experiment. First, we discuss what data was being recorded and how the data was stored.

7.1 Database Design

The data we collected are a combination of old implicit/explicit ratings and the new implicit/explicit ratings that we have discussed in the previous Chapter. The total list of implicit indicators collected by the browser is:

1. URL of Web page
2. Username
3. Date
4. Time when Web page was visited
5. Time spent on the Web page
6. Duration of Vertical Scroll Bar usage
7. Amount of Scroll Bar usage
8. Duration of Mouse usage
9. Mouse Clicks on the Web page
10. Duration of Horizontal Scroll Bar usage
11. Number of Up Arrow key usage
12. Number of Down Arrow key usage
13. Duration of Up Arrow key usage
14. Duration of Down Arrow key usage
15. Duration of Page Up button usage
16. Duration of Page Down button usage
17. Number of Page Down button usage
18. Number of Page Up button usage
19. Number of changes of the Status Bar
20. Amount of Mouse Wheel Scrolls
21. Duration of Scrolling with Mouse Wheel button held down
22. Number of Mouse Wheel button usage (Auto Scroll)
23. Size of HTML file
24. Number of Right Mouse button usage
25. Number of Copy Command usage

26. Was the current URL manually typed
27. Was the current URL accessed by clicking a link from previous page
28. Is it a multilink Web page (Part of the test code that caused the crash during the experiment – Chapter 8)
29. Familiarity Rating
30. Interest Rating
31. Vertical Lines (Parsed from raw mouse coordinates - Chapter 9.1)
32. Horizontal Lines (Parsed from raw mouse coordinates - Chapter 9.1)
33. Computer Science classes currently in

Data items 1 to 30 were stored in the sample.dat and the remaining data were stored in sample2.dat. The reason why we chose these indicators to record are discussed in the Chapter 4 and the methods that were used to obtain them are discussed at Chapter 6.

7.1 Structure of sample.dat and sample2.dat text files

Instead of storing the data in a database such as Oracle or Access, two data files in plain text format were used. The data was not large enough to need the structure of a database, so for simplicity, all the data is stored as ASCII text with a delimiter "|" for parsing purposes.

We chose "|" as the delimiter because one of the values we were recording is a URL, which can contain most of the other common delimiters such as semicolon (;), comma (,), period (.) and other symbols. If we had chosen a more common delimiter, the URL could cause our parser to separate the data incorrectly. Consider the URL: http://www.wpi.edu/ as an example for a parser using period (.) as a delimiter. The URL would be separated into "http://www" then "wpi" and last field would become "edu/". The resulting entry will be an incomplete URL and two extra fields of invalid data.

The delimiter was very helpful when we had to convert the text files to Microsoft Excel format. In Microsoft Excel we do some basic database operations such as order all the data by username or order the explicit ratings by value. We also needed the delimiter to help a Perl script that we developed parse the Mouse coordinates for detection of horizontal and vertical lines.

Once all the data for the entire 19 days were collected, we merged all the data in sample.dat into one big worksheet and examined it for any incomplete or faulty data. The resulting data was then analyzed.

7.1.1 Sample.dat

The sample.dat file holds the 30 variables that consist of the two explicit (interest & familiarity) ratings and most of the implicit ratings. This file was then converted to a Microsoft Excel Worksheet format and then input to the SAS software for statistical testing. The Excel Worksheet format was mainly used for the sample.dat file rather than sample2.dat. The data in sample.dat is well organized and has one value per column, so a Worksheet format was suitable. For the grammar of the format of the sample.dat file, please see Appendix A.

7.1.2 Sample2.dat

The sample2.dat file data was separated from the sample.dat data because the former contained mouse coordinates, which were parsed differently than the other implicit values. The sample2.dat file was much larger than sample.dat file because it had to store all the X and Y coordinate for every millisecond the mouse cursor was moving. For example, a user is moving the mouse for just one minute, generates 60 * 1000 * 2 = 120,000 coordinates. The coordinates are stored as Long Variable, making 4 * 120,000 = 480,000 bytes (480KB) for just one minute of continuous browsing. To reduce some of the memory needed, we took into account the probability of the cursor being idle. In this situation we just store the X and Y coordinates and a count of the milliseconds it was idle. For more details regarding the algorithm used to obtain the mouse coordinates, please refer to Chapter 6. The Grammar for the format of the sample2.dat file is located in Appendix A.

Notice that there are two versions of sample2.dat grammar due to the problem of the Browser crashing nearing the end of the second week. Due to the crashing problem (discussed in Chapter 8.2), the program had no way of recording the user's information in the sample.dat file since, unless the program is closed properly, the data is not written to the file. Due to the amount of memory needed to store every active mouse cursor, the sample2.dat data was written while the Browser was still in operation. Sample2.dat's data was still recorded even when the Curiouser Browser crashed. We had to change the sample2.dat format to also store the username so we could keep track of who participated with the experiment and still give them credit for trying to participate in our experiment.

We wrote a Perl script to isolate the occurrences of Horizontal lines and Vertical lines in this file of mouse coordinates. The occurrences of the lines are then analyzed as an implicit rating in Chapter 9.3. For more information on the Perl script, please go to Chapter 9.1 which describes the algorithm for the Perl script parser.

7.2 Targeted Participants

Our main goal was to get as many WPI students as possible to participate in the experiment. Since we assumed that the Curiouser Browser was installed in the College Computer Center Lab (CCC Lab) and/or the Advanced Document Preparation Lab (ADP Lab), focusing on CS students was natural since the two labs are located in Fuller Labs, the location of WPI's Computer Science Department.

7.3 ADP Lab Setup

We sent an email to the Computing & Communication Center (CCC), requesting them to install the Curiouser Browser on all public PCs in Fuller Labs. Then we had a meeting with CCC's Desktop System Administrator, and he stated that we could install the program on the lab computer ONLY IF we did the following:

- Include a disclaimer in the program;
- Create a flyer that can be taped on top of the monitors, indicating that the computer contains the Curiouser Browser.

The reason for this requirement was that CCC was afraid that our program would capture users' information without letting them know, which would violate the privacy of users.

We created a disclaimer (discussed in Chapter 5) to explain that any personal information stored by the Curiouser Browser will only be used for research purposes. The flyer that was on top of the monitors, serves two purposes. The first was to mark the computers in the ADP lab so that users know which ones to use if they wanted to participate in our experiment. The second was to serve as another disclaimer and give basic instructions for the user so they could view important information regarding the browser without having to open it. What follows is the actual content of the flyer:

> ***MQP***: ***Curiouser Browser***- *This computer is part of a Major Qualifying Project to research a users preference for a Web page related to their actions on that page. If you decide to be a part of this experiment simply run the* ***Curiouser Browser*** *from the desktop or start menu and browse the Web. Your username will be stored in our database along with the name of the site you visit. No other personal information will be stored (i.e. password, information entered on Websites). If you have any questions or concerns you can e-mail curiouser@wpi.edu or bms@wpi.edu.*
>
> *Thank you for your participation.*
> *Brad Goodwin, Steve T. Law, Michael Cen*

After all the requirements were met and once the staff had tested the Curiouser Browser, CCC granted our request to conduct the experiment using the ADP computers. The Curiouser Browser was installed on 12 computers in the ADP allowing users in the ADP to have the option of not using a computer that was for our experiment.

7.4 Experiment

Before we started the experiment on March 18, 2002, we sent emails to the professors who were teaching Computer Science classes at the time. It was to notify them about our experiment and persuade them to help us convince their students to participate in it. Also with their permission, we sent out email to all students currently taking their CS classes regarding our experiment and asked them to participate. The email sent to students contained brief information about our experiment and what they needed to do to help us. We did not let them know what exactly we would record for our experiment, so it would not bias the user actions.

To encourage students to participate, we asked the instructors to give extra credit to students who participated in this experiment, if possible. If the instructors agreed to give extra credit, we would provide them with a list of students in their classes who participated in the experiment.

Chapter 8: Experiment Implementation

This chapter discusses the process, problems, and solutions to the problems that happened during the time of the experiment.

8.1 Data Collection

The experiment was intended to run two weeks, from March 18 to March 29. CCC personnel usually reformat and reinstall everything on the computers every two weeks. They also reformat whenever it was necessary in order to keep the computers at their optimal level. But we asked them not to refresh the computers until the experiment ended, because if they did refresh the computers while the experiment was being conducted, data would be lost. All the data was stored in the computer's hard drive and they were retrieved at 11pm on every weekday. The data files to be retrieved from each computer were sample.dat and sample2.dat.

8.2 Crashing of the Curiouser Browser

This section discusses the crashing of the Curiouser Browser and its resolution.

8.2.1 Crash

During the experiment, the Curiouser Browser crashed over 50% of the time when users attempted to use it. That affected the data collection greatly. As mentioned in the design sections, the browser only collects the data when users close the Curiouser Browser. Since the browser was often not closed properly due to the crash, none of the data that should have been stored in sample.dat was written to the file. Sample2.dat, however, was written to a file because its data is collected and written every millisecond of mouse movement. We tried to use the usernames and URLs stored in both data files to match the data together. However, we were unable to match them since some of the data was missing.

The crashing problem never happened during the test runs of the program. However, a number of reasons for the crash were considered when a few users emailed us and complained:

- ADP lab computers can be unstable since users install a variety of programs on them;
- ADP lab computers might have run for days within being reset and they are low on resources, which may have affected the performance of the Curiouser Browser since all the temporary data is stored in memory;
- The complexity of the Web pages that users are visiting is beyond the ones that were used in the test runs;
- There is an unknown bug in the Curiouser Browser that did not occur during the test runs.

After a number of re-tests, we found a bug in the program and we also found out that some Web pages may not work well with our Curiouser Browser. The bug we found in the Curiouser Browser was due a testing procedure that was used in the implementation, and left in the program for possible future use. The test code was used to store all the URLs found in the Web page to prevent the Evaluation window from appearing too many times. However, the testing procedure was not taken out of the system even after the browser was finalized and installed in the ADP computers.

We also realized that some of the pages with many JavaScript and framed pages might not work well with the Curiouser Browser. JavaScript affects the URLs that we store into our database. Framed pages can possibly affect the mouse coordinate data since it can prevent the browser from setting the correct HTML page for data collection.

Although we fixed the bug, 50% of the data collected in the first two weeks could not be used because it was incomplete. That required us to run the test for another week.

8.2.2 Extension of the Experiment

The experiment ran beyond March 29 for one more week. We sent an email to the Desktop System Administrator to explain our situation and requested an extension of the experiment. After we received approval, we sent emails to the professors and students, informing them that the experiment was extended. Although we realized that asking students to redo the experiment was a risky task, we sent emails to those who submitted the incomplete data anyway to apologize and also to ask them to re-do the experiment.

The experiment was concluded at Midnight of April 5, 2002. The data collected during the extension week was more than that collected during the first two weeks.

Table 7.1 shows the amount of information collected from the 3-week experiment.

Data Files	**Size**
Sample.dat in total	0.35MB
Sample2.dat in total	4.40MB
Total	4.75MB

Type of Entries	**Number of Entries**
Complete User Data	88
URL Entries	858

Table 7.1 Information of the data collected from the experiment

By “Complete user data” in the table, we meant that the data is stored in both sample.dat and sample2.dat. For URL Entries, user name and URL combined were counted as one unique URL entry.

With all the raw data in our hands, we put all the data into Microsoft’s Excel worksheets to keep them organized. After all the data are parsed and organized, we start our analysis, which is discussed in Chapter 9.

Chapter 9 Analysis

After collecting and combining all the data, we attempted to interpret the data and determine its meaning. We first used basic statistical tests to make some preliminary observations such as possible patterns of correlation. Correlation could be present between two seemingly independent variables if their values increase or decrease together. We then used a procedure provided from SAS (Statistical Analysis Systems) to test the assumptions made from the preliminary observations.

Before applying statistical tests to all the data, there was some data that needed to be processed first, and that was the raw mouse coordinates. We created a parser to find the patterns of lines from the raw mouse coordinates. By "raw" we mean that the coordinates were just points on a Web page. The following sections will contain detailed information about the steps we took to analyze the data. First we created a parser to extract vertical and horizontal lines from the mouse coordinates. We then did some descriptive statistics to make preliminary observations. Finally, we used SAS to obtain additional results.

9.1 Parsing Algorithm

Out of all the new implicit indicators that we added to the Curiouser Browser, the most interesting to us was the recording of mouse coordinates, since the mouse has become a very important navigation tool in recent years. This gave us the opportunity to trace specific movements of the mouse and to see if those movements might have a possible correlation with the explicit rating of the page. The range of possible cursor movements that we could have studied was large. We decide to keep it simple and search for vertical and horizontal lines. We hoped that the occurrences of the vertical and horizontal lines were indications of scanning through lists and of careful reading, and therefore were possible predictors of interest.

9.1.1 Find and Separate the data

Since the coordinates are stored in a database with information about the user, URL, time and date mixed, we had to separate that data from the coordinate data before continuing. To separate the coordinates from the other data we decided to parse through that data using regular expressions in a PERL script.

To find the lines created from mouse cursor movement per Web page, we needed to devise an algorithm that would allow us to divide the database into groups of coordinates based on Web pages. The coordinates could then be passed to the line-parsing algorithm. The algorithm finds straight lines in four basic directions (up, down, left, and right) and returns the number of lines found for each direction.

We wrote all the data collected to the file "parsed.dat". For each particular user we wrote their username and the URL's of sites they visited. Under each URL we wrote the number of lines found for each direction on that site. This information is separated by "|" delimiters that allow the data to be transferred into Microsoft Excel files.

9.1.2 Problems

The running of our experiment was planned to take about two weeks, but after those two weeks we found that we had an insufficient amount of data to conduct a useful analysis. We decided to let that experiment run for one more week in hopes that after that week we would have gathered a sufficient amount of data. Since part of the reason that we hadn't gathered enough data after two weeks was due to the actual browser crashing frequently, we installed an updated version for the extended week. This updated version of our browser also modified the format in which the coordinates were written to the database. Modifications had to be made to the data parser for it to parse either database (original and modified).

9.2 Trimming for valid data

The experiment was conducted over three weeks, from March 18 to April 5. The size of all the data (including faulty or invalid data) from sample.dat was 350KB and sample2.dat was 4.40MB, which was 4.75MB altogether. Sample2.dat was much larger than sample.dat because sample2.dat contained the raw data of mouse coordinates for every millisecond that the mouse was active.

The total number of data entries collected from the experiment was 1191. However, there were faulty data such as unpredicted user actions and the crashing problem explained in the previous Chapter 8.2. The usable data dropped to 858 unique URL and username entries. There were 141 users that participated in our experiment. After filtering out the invalid data, we could only analyze 85 users' data.

We had to omit the data that had a value "0" for the explicit interest rating. The value "0" meant "No comment", which meant the user did not give us a valid explicit rating to test. Since we had two separate files of data, we had to combine them for the analysis. Unfortunately some data were not present in both files so those had to be discarded. The entries that were present in both files were referred to as “Matched”. Table 9.1 shows the cumulative statistics of the entire data and the faulty data that were omitted.

- First row: Complete user data recorded in sample.dat
- Second row: Complete user data recorded in sample.dat after all “0” implicit interest ratings are taken out
- Third row: Matched data between sample.dat and sample2.dat
- Fourth row: Matched data after all “0” implicit interest ratings are taken out

	URL	**Username**	**Time on page**	**Time on Hscroll**	**Time on Vscroll**	**Events on Scroll**	**Time on mouse**
Total without Lines	1191	88	29834324	57973	1630456	183	4227782
Total without Line w/out 0 rating	690	81	22619896	15651	984582	90	3017456
Matched Data	858	84	26348557	57973	1401793	133	3646447
Matched Data w/out 0 rating	662	82	18088123	15651	891788	65	341830

	Clicks on Window	**Num of UpArrow**	**Num of DownArrow**	**Msec of UpArrow**	**Msec of DownArrow**	**Msec of PageUp**	**Num of PageUp**
Total without Lines	3036	91	330	9502	47990	4571	18
Total without Line w/out 0 rating	1648	86	264	9362	46116	4278	12
Matched Data	2108	66	247	6612	43457	3672	14
Matched Data w/out 0 rating	1232	44	184	3234	28455	2243	2

	Msec of PageDown	**Num of PageDown**	**Num of Statusbarchange**	**Num of mouse wheelscrolls**	**Msec of mousewhell**
Total without Lines	2894	24	6447	7170	0
Total without Line w/out 0 rating	2682	19	3868	4831	0

Matched Data	2333	17	4549	5326	0
Matched Data w/out 0 rating	702	3	2903	3770	0

	Num of Mousewheel	Size of files	Num of RightMouseClicks	Num of Copies
Total without Lines	54	16872393	31	27
Total without Line w/out 0 rating	40	11225671	6	13
Matched Data	49	13316353	8	13
Matched Data w/out 0 rating	36	8988065	4	7

	istypeURL	isClickURL	UP	DOWN	L - R	R - L	TOTAL
Total without Lines	532 True, 419 False	262 True, 239 False					
Total without Line w/out 0 rating	230 True, 460 False	556 True,134 False					
Matched Data	588 True, 270 False	693 True, 165 False	1199	2255	1642	1526	6622
Matched Data w/out 0 rating	125 True, 442 False	442 True, 125 False	843	1607	965	911	4326

Table 9.1: List of the total data

9.3 Descriptive Statistics

In this section, we look at the individual descriptive statistics to make preliminary observations for each implicit rating. Using the Minitab (version 4) statistical software command “Display Descriptive Statistics,” we generated a table that displays the general descriptive values of each implicit rating separated by the interest scale (explicit interest rating). These values were generated from the Minitab table for each rating:

- N – Number of records for each explicit interest rating.
- Mean – The average of the rating.
- Median – The median of the rating.
- TrMean – The mean of the rating, but excluding the top and bottom 5% of the values.
- StDev – The standard deviation, which is a measure of spread.
- SE Mean – Stand error of the mean. The formula is:

$$SEMean = \frac{StDev}{\sqrt{N}}$$

- Minimum – Smallest value of the rating
- Maximum – Largest value of the rating
- Q1 – The first quartile. Formula: (N+1)/4
- Q2 – The third quartile. Formula: 3*(N+1)/4

After obtaining the descriptive statistics table, we then looked at the frequency of occurrence for the implicit indicators. If they did not have enough occurrences, they would not be further analyzed, because implicit indicators that occur infrequently have low significance. We determine low occurrences by observing the median value, since the median regions of the data was buffered well against outliers. For example, Table 9.3 shows the descriptive statistics regarding time spent using the horizontal scroll bar.

Notice that the median column is all zero, meaning there was not enough values for a median to be calculated.

With the low occurrence ratings omitted, we then generated a box plot for each implicit rating versus the explicit interest rating. The box plot was generated using a 95% confidence level, which means that 95% of the total data is within this range. By taking out the outer 5%, we have removed some of the outliers. The box plot shown focuses on the median area of the data. By viewing the median region, we omitted viewing most of the outliers. The outliers of the data range are any value lying 1.5 and 3 times away from the middle 50% of the data [Minitab Windows Help, 2000].

Unfortunately, our data would not allow us to get accurate results by just observing the box plot of the median area. This is because the explicit interest rating was in an Ordinal Multinomial distribution model and not a Linear distribution model. To find correlations from an Ordinal Multinomial Model, we used the SAS procedure GENMOD. Further explanations of this procedure and the meaning of Multinomial Model are discussed in Chapter 9.4.

9.3.1 Time on Page vs. Explicit Interest Rating

Descriptive Statistics: Time on page by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
Time on    1                44      23010      14539      16951      33128
           2                81      26255      14674      22369      30161
           3               154      31262      22068      26270      34268
           4               168      38108      21592      32114      46386
```

```
            5                243      34010      19267      26518      48865

Variable    Rating       SE Mean    Minimum    Maximum         Q1         Q3
Time on     1               4994       2084     181435      10495      21196
            2               3351       1826     188367       7774      31767
            3               2761       2126     268341      13189      36572
            4               3579       1117     330139       9501      45000
            5               3135       1844     495460      10690      38042
```

Table 9.2: The descriptive statistics of the implicit rating, time spent on a page.

As shown in Table 9.2, the median of the time spent on a page steadily increases as interest increases. However, this pattern only appears from rating 1 to 4 of the explicit interest rating, and decreases at rating 5. A correlation between two variables appears when there is a steady increase or decrease of both values of the two variables together. From this table, there may be a correlation, but it is not strong enough to be a significant correlation. For further observation we have generated Figure 9.1 for the above data.

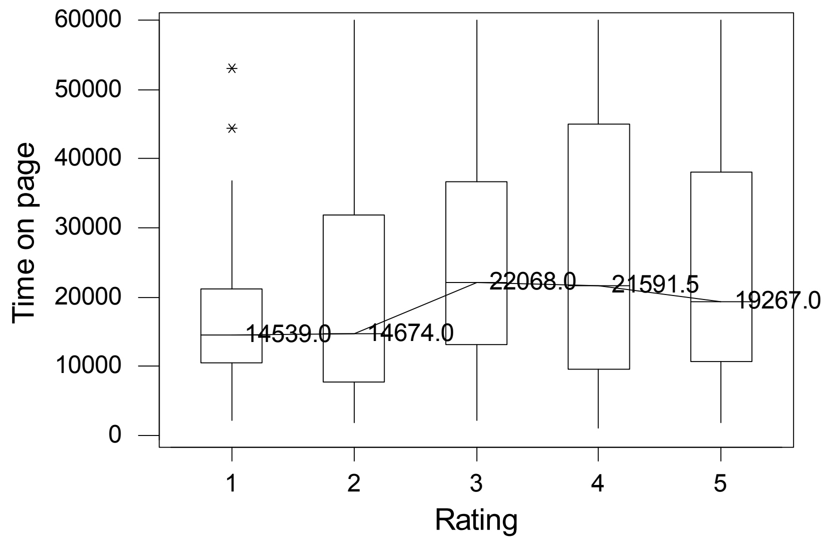


Figure 9.1: Box plot of Time on page vs. Explicit Interest Rating

From Figure 9.1, we see that there was a slight drop in the median value of the time on page indicator. To find whether there was an actual correlation between time spent and interest on a page, we needed to do a more advance statistical test called GENMOD, which we discuss in Chapter 9.4.

9.3.2 Time on Horizontal Scroll Bar

Descriptive Statistics: Time on Hscroll by Rating

```
Variable   Rating           N       Mean     Median     TrMean      StDev
Time on    1               44    0.00000    0.00000    0.00000    0.00000
           2               81    0.00000    0.00000    0.00000    0.00000
           3              154       39.1        0.0        0.0      326.2
           4              168       48.1        0.0        0.0      623.2
           5              243       6.40       0.00       0.00      65.38

Variable   Rating     SE Mean    Minimum    Maximum         Q1         Q3
Time on    1          0.00000    0.00000    0.00000    0.00000    0.00000
           2          0.00000    0.00000    0.00000    0.00000    0.00000
           3             26.3        0.0     3509.0        0.0        0.0
           4             48.1        0.0     8078.0        0.0        0.0
           5             4.19       0.00     910.00       0.00       0.00
```

Table 9.3: Descriptive Statistics of Time on Horizontal Scroll bar separated by interest rating

The table for time using the horizontal scroll bar is shown on Table 9.3. We noticed that the median value of this implicit rating is 0 for all explicit ratings. This means the time spent on the horizontal bar is not frequent enough to have any significance. We went back and counted the occurrence of this implicit rating and found seven occurrences out of 690 entries. Since it has so little significance, we decided to remove it from the next analysis. To further illustrate that the horizontal scrollbar has little significance, we generated this box plot (Figure 9.2) for visual confirmation.

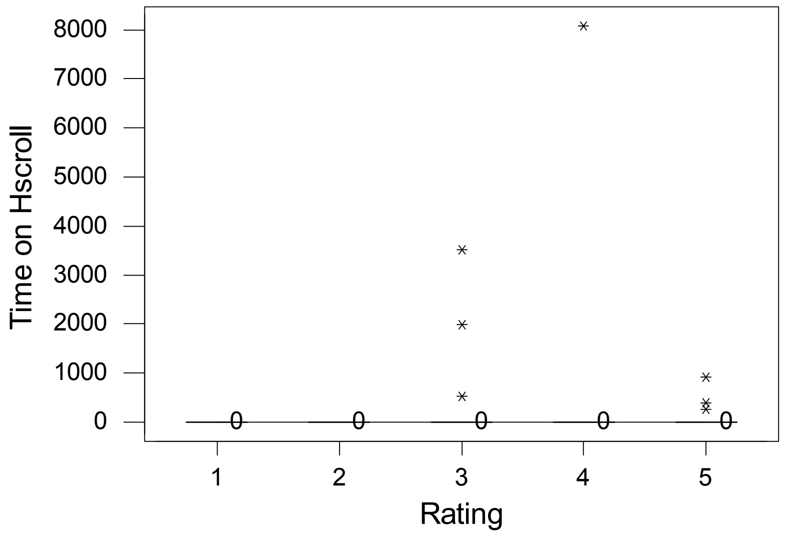


Figure 9.2: Box plot of Time on Hsroll vs. Explicit Interest Rating

The box plot in Figure 9.2 has no box regions, because the median values are all zero. The graph consists of only a horizontal line that connects the median values (zero) and outliers indicated by *. As discussed before, this implicit rating was not further analyzed with the GENMOD procedure.

9.3.3 Time on Vertical Scroll Bar

Descriptive Statistics: Time on Vscroll by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
Time on    1                44        222          0          0       1476
           2                81        929          0         94       4196
           3               154       1133          0        348       3942
           4               168       3273          0        602      14433
           5               243        721          0         45       5671

Variable   Rating      SE Mean    Minimum    Maximum         Q1         Q3
Time on    1               222          0       9789          0          0
           2               466          0      27961          0          0
           3               318          0      27056          0          0
           4              1113          0     116739          0          0
           5               364          0      82625          0          0
```

Table 9.4: Descriptive Statistics of Time on Vertical Scroll bar separated by interest rating

The result of the time spent using the vertical scroll bar is the same as the horizontal scrollbar, namely there are too few occurrences. This can be seen by looking at the median values of this implicit rating, which all have the value zero. To illustrate that the vertical scroll bar is used infrequently, we generated a box plot graph shown in Figure 9.3 to compare with Figure 9.2

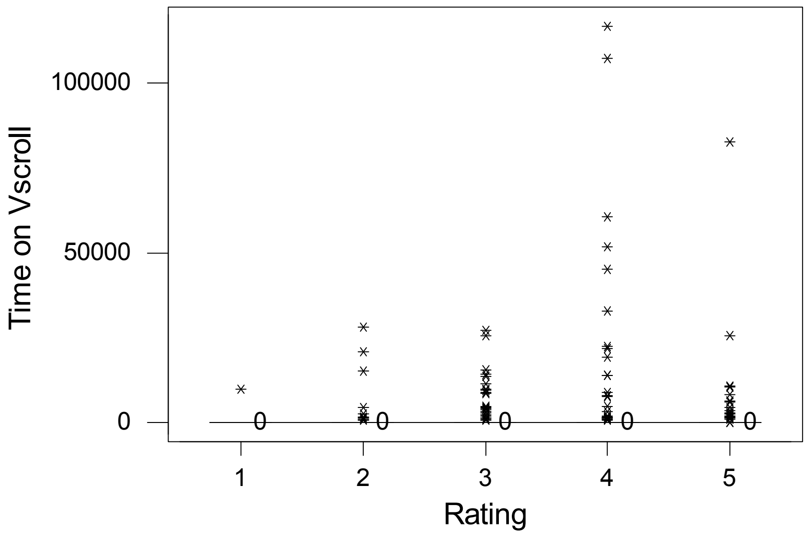


Figure 9.3: Box plot of Time on page vs. Explicit Interest Rating

As shown in Figure 9.3, the time spent using the vertical scroll is almost the same as the time spent using the horizontal scroll. The difference between the two box plots is the number of outliers. Both box plots have the same median connection line at value zero, but the time using vertical scroll has more outliers.

9.3.4 Events on scrolling

Descriptive Statistics: Events on Scroll by Rating

```
Variable   Rating          N      Mean    Median    TrMean     StDev
Events o   1              44   0.00000   0.00000   0.00000   0.00000
           2              81    0.0988    0.0000    0.0274    0.3743
           3             154    0.1558    0.0000    0.0942    0.4143
           4             168    0.1845    0.0000    0.0987    0.5321
           5             243    0.1111    0.0000    0.0365    0.4545

Variable   Rating    SE Mean   Minimum   Maximum        Q1        Q3
Events o   1         0.00000   0.00000   0.00000   0.00000   0.00000
           2          0.0416    0.0000    2.0000    0.0000    0.0000
           3          0.0334    0.0000    2.0000    0.0000    0.0000
           4          0.0411    0.0000    3.0000    0.0000    0.0000
           5          0.0292    0.0000    5.0000    0.0000    0.0000
```

Table 9.5: Descriptive Statistics of events on scrolling separated by interest rating

The scrolling event is the implicit rating that counts the occurrences of both vertical and horizontal scroll bar usage. From the median values, once again we have an implicit rating that does not occur often. Since we have graphed the data of the time spent on vertical and horizontal scroll bars, we graphed scrolling event in Figure 9.4.

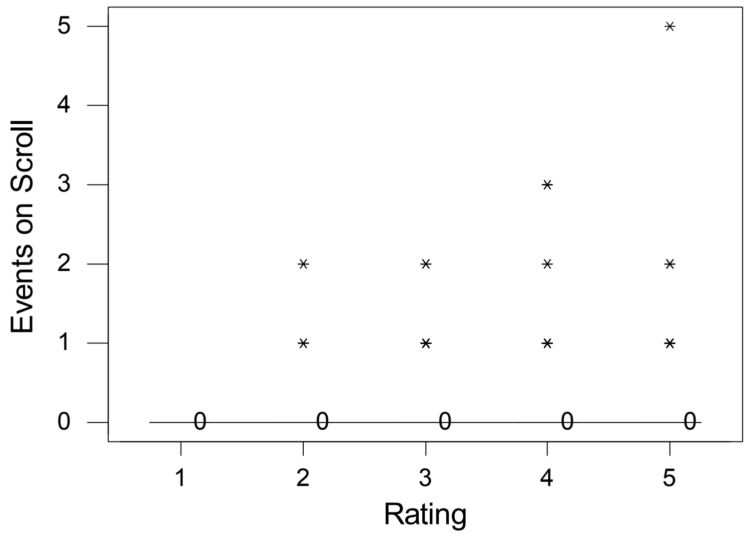


Figure 9.4: Box plot of Events on Scroll vs. Explicit Interest Rating

The box plot has the same format as its subordinates, and also shows that this implicit rating is also infrequent. As we did with the other two ratings, we omitted this implicit rating from further analysis.

9.3.5 Time Using Mouse

Descriptive Statistics: Time using mouse by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
Time on    1                44       3240       2125       2785       3372
           2                81       4025       2645       3574       3918
           3               154       4050       2616       3346       4919
           4               168       4752       2565       3762       7343
           5               243       4637       2985       3968       5041

Variable   Rating      SE Mean    Minimum    Maximum         Q1         Q3
Time on    1               508        152      18498       1317       4359
           2               435        542      18381       1368       5262
           3               396          0      42035       1346       4903
           4               567          0      74423       1380       5549
           5               323          0      36852       1476       5705
```

Table 9.6: Descriptive Statistics of Time using mouse separated by interest rating

This is an implicit rating that has an occurrence high enough to be called significant. This is because the median values are not zero and there appears to be a correlation between the median values and the interest values. However, this only happens between the explicit values of one and two. The next few intervals of explicit interest values actually have a decrease in the median values.

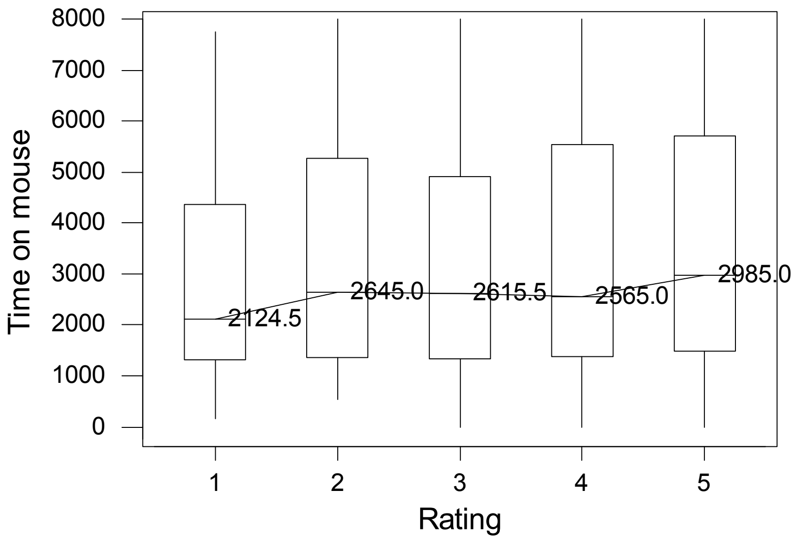


Figure 9.5: Time on mouse vs. Explicit Interest Rating

Notice from the median values and median connection line, there is a slight increase of values for both ratings. Then there is a slight decrease of median values while interest values still increase. The median values go from 2645, 2616, and then 2565 between the range of 2 and 4 on the explicit interest rating. The median value then increases again to 2985. The variation of these four values might be small enough to allow some sort of correlation. For further testing, we use the GENMOD procedure in Chapter 9.4.

9.3.6 Clicks on window

Descriptive Statistics: Clicks on Window by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
Clicks o   1                44      1.705      1.000      1.550      1.608
           2                81      2.185      1.000      1.890      2.414
           3               154      2.266      1.000      1.732      3.466
           4               168      3.107      1.000      2.092      5.839
           5               243      2.160      1.000      1.767      2.732

Variable   Rating      SE Mean    Minimum    Maximum         Q1         Q3
Clicks o   1             0.242      0.000      7.000      1.000      2.000
           2             0.268      0.000     15.000      1.000      3.000
           3             0.279      0.000     29.000      1.000      2.000
           4             0.451      0.000     41.000      1.000      2.750
           5             0.175      0.000     26.000      1.000      2.000
```

Table 9.7: Descriptive Statistics of Clicks on window separated by interest rating

The median value of Clicks on window is all one. That means there is no correlation between clicks on the mouse and interest on the page. The occurrence of clicks on windows does suggest that it has influence on the total data, so we decided to pass it to GENMOD for further studies. The box plot is shown in Figure 9.6. The straight horizontal line is the line that connects the medians. That line is on every box plot and is there for observation of possible correlation. See Figure 9.6 for the graph.

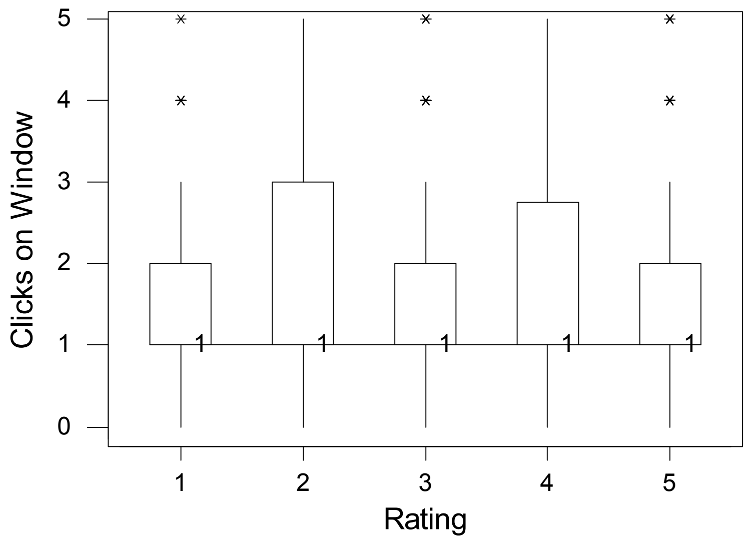


Figure 9.6: Box plot of Clicks on Window vs. Explicit Interest Rating

From the Figure 9.6, it shows a flat horizontal line that connects the median of each box region. Usually a horizontal line means no correlation between the two variables. Since we are looking at the median values of the data set, it might be possible that a correlation is not detectable by visual analysis.

9.3.7 Number of Up Arrow button occurrence

Descriptive Statistics: Num of UpArrow by Rating

```
Variable   Rating           N       Mean     Median     TrMean      StDev
Num of U   1               44    0.00000    0.00000    0.00000    0.00000
           2               81     0.0370     0.0000     0.0000     0.3333
           3              154      0.344      0.000      0.000      2.265
           4              168     0.1726     0.0000     0.0000     1.1477
           5              243    0.00412    0.00000    0.00000    0.06415

Variable   Rating     SE Mean    Minimum    Maximum         Q1         Q3
Num of U   1          0.00000    0.00000    0.00000    0.00000    0.00000
           2           0.0370     0.0000     3.0000     0.0000     0.0000
           3            0.182      0.000     22.000      0.000      0.000
           4           0.0886     0.0000    12.0000     0.0000     0.0000
           5          0.00412    0.00000    1.00000    0.00000    0.00000
```

Table 9.8: Descriptive Statistics of Number of Up Arrow button occurrences separated by Interest Rating. Notice the "0.00" values of the Median column, this means low occurrence.

By looking at the table of descriptive statistics associated with number of Up Arrow button occurrences (Figure 9.7), we noticed that the medians are all zero. That follows the pattern of low occurrence as shown by the implicit ratings time using horizontal scroll, time using vertical scroll and events of scrolling with scroll bars. Therefore, we will not further analyze this implicit rating.

By looking at the analysis done so far, we noticed that only three out of the seven implicit ratings had an occurrence high enough to be significant. That meant more than half of the implicit ratings were insignificant. These implicit ratings are not significant due to low occurrence:

- Number of Down Arrow usage and the duration of its usage.
- Number of Page Up button usage and the duration of its usage.
- Number of Page Down button usage and the duration of its usage.
- Number of Mouse Wheel button usage and the duration of its usage.

- Number of Right Mouse clicks usage
- Number of Copy Command usage

These have the same graph pattern as those of Figure 9.2. The graph has no median interval since its value is zero and the rest of the graph consists of the outliers. The implicit ratings listed above will not be further analyzed since they are not relevant.

9.3.8 Number of status changes

Descriptive Statistics: Num of Statusbarchange by Rating

```
Variable   Rating          N      Mean    Median    TrMean     StDev
Num of S   1              44     4.159     2.000     3.650     4.534
           2              81     4.741     3.000     4.384     4.095
           3             154     5.175     3.000     4.384     5.981
           4             168     6.673     4.000     5.461     8.152
           5             243     5.691     4.000     4.941     5.950

Variable   Rating    SE Mean   Minimum   Maximum        Q1        Q3
Num of S   1           0.683     0.000    21.000     1.000     5.000
           2           0.455     0.000    17.000     2.000     7.000
           3           0.482     0.000    35.000     1.000     7.000
           4           0.629     0.000    55.000     2.000     8.000
           5           0.382     0.000    47.000     2.000     7.000
```

Table 9.9: Descriptive Statistics of Number of status change separated by interest rating

The “status bar change” detects changes of the status bar. The status bar is located at the lower left hand corner of the browser. This is originally a MSIE feature to indicate if the Web page is done loading. Also the bar displays information about the link when the mouse cursor is over it. To see a screen shot of the status bar refer to the lower–left hand side of Figure 5.8.

According to the descriptive statistics, we can see that the median values are not all zero, indicating that this implicit rating has enough occurrences to be relevant. The median values are also increasing as the explicit rating increase, which shows a strong correlation. We graphed the data to see if the correlation was strong enough to appear on the graph as a straight positive sloped line. (Figure 9.7)

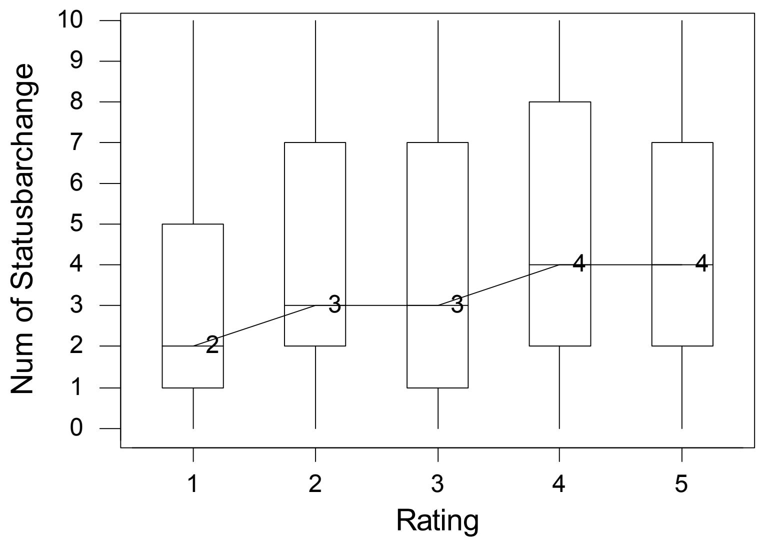


Figure 9.7: Box plot of Number of Status bar changes vs. Explicit Interest Rating

The box plot Figure 9.7 has a slowly increasing line that is connected by the medians of each box region. Although there is a sloped line, the correlation between the Number of Status Bar changes is not strong enough to mean significance between the status bar changes and interest.

There was a slight correlation since status bar detects the links the mouse cursor stops over. It the a user was interested in a link, he or she might move the cursor over it and read the URL to determine if that is the site they want to go.

9.3.9 Number of mouse wheel scrolls

Descriptive Statistics: Num of mouse wheelscrolls by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
Num of m   1                44       5.64       0.00       2.88      15.50
           2                81       7.36       1.00       4.59      16.33
           3               154       7.93       2.00       5.82      13.43
           4               168      6.155      1.000      4.908      9.767
           5               243      7.128      2.000      5.648     10.895

Variable   Rating      SE Mean    Minimum    Maximum         Q1         Q3
Num of m   1              2.34       0.00      68.00       0.00       2.00
           2              1.81       0.00     116.00       0.00       7.00
           3              1.08       0.00      75.00       0.00      10.00
           4             0.754      0.000     66.000      0.000      9.000
           5             0.699      0.000     60.000      0.000     10.000
```

Table 9.10: Descriptive Statistics of Number of mouse wheel scrolls separated by interest rating

Number of mouse wheel scrolls was one of the most hopeful candidates for finding a strong correlation between implicit and explicit ratings. From the median values, we noticed an oscillation of values between one and two. Oscillating data is not a pattern of correlations. From the descriptive test, we guessed that the implicit rating has no correlation, but its occurrence is high enough to pass it to GENMOD for a further analysis.

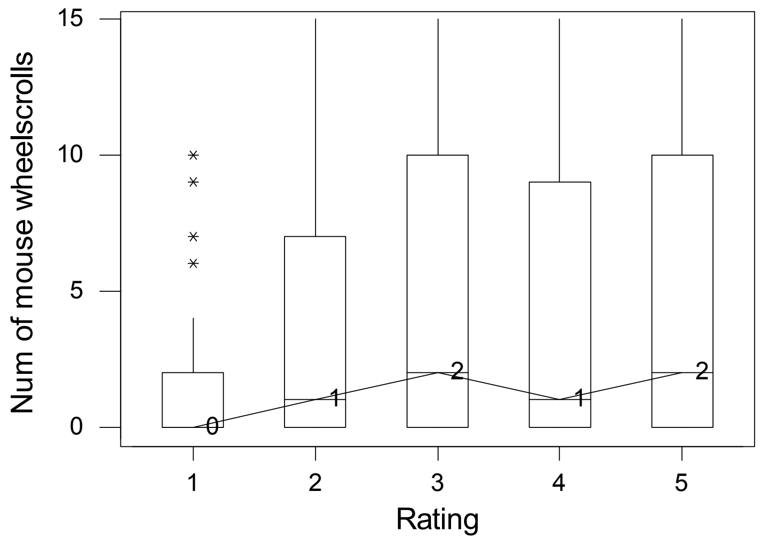


Figure 9.8: Box plot of Num of mouse wheel scroll vs. Explicit Interest Rating

Figure 9.8 is box plot of mouse wheel scrolls. As the median connection line shows, the median value oscillates from value one to two. This visually supports the no correlation assumption.

9.3.10 Size of Web page

Descriptive Statistics: Size of files by Rating

```
Variable   Rating            N      Mean    Median    TrMean     StDev
Size of    1                44     11641      5732      9626     16179
           2                81     20949     14427     18798     23377
           3               154     15116      8247     13094     17790
           4               168     19401      8281     17908     20418
           5               243     14113      8247     11575     18373

Variable   Rating      SE Mean   Minimum   Maximum        Q1        Q3
Size of    1              2439         0     71058       800     15041
           2              2597       119    128220      3195     30168
           3              1434       165     71058      3137     19429
           4              1575         0     92852      2770     31953
           5              1179         0     97199      1997     20593
```

Table 9.11: Descriptive Statistics of Size of Web page separated by interest rating

The size of the Web page is the implicit rating that records the size of the Web page without any graphics. If graphics are not taken into account of the size, then the Size of the Web page could also mean the size of text on the Web page.

From the mean values given in Table 9.11, there is no correlation. The median value increased to its max at interest rating value 2. Then the mean value decreased and increased around the same range (8247 to 8291).

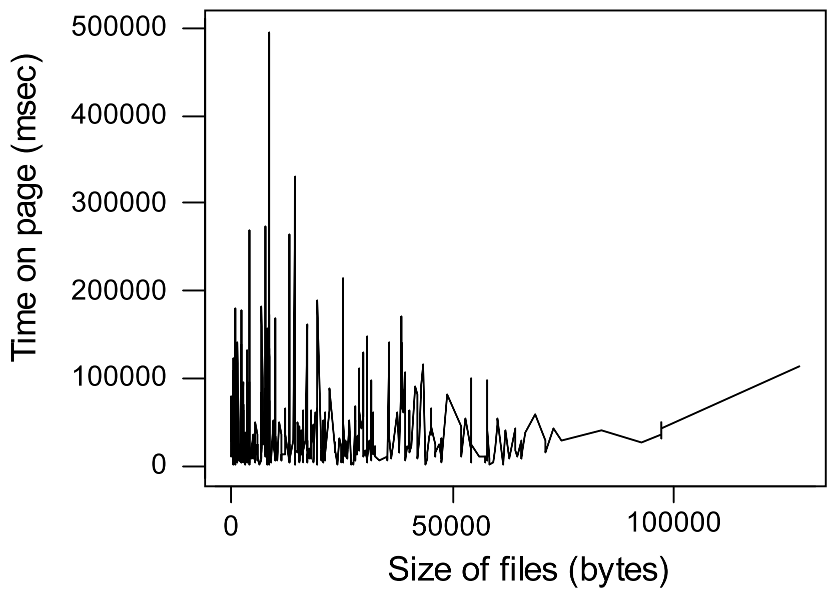


Figure 9.9: Line graph of Size of page vs. time on page

Instead of making a box plot, we created a graph between time on page and size of the Web page to see if there is any correlation. This is to normalize the time on page data. Visual, there is no indication of correlation between time spent on a page and size of file. The points do not create a steadily increasing slope.

9.3.11 Number of Mouse Cursor Lines going Up

Descriptive Statistics: UP by Rating

```
Variable   Rating           N      Mean    Median    TrMean     StDev
UP         1               36     0.972     1.000     0.844     1.276
           2               73     1.575     1.000     1.415     1.755
           3              124     1.540     1.000     1.420     1.553
           4              123     1.715     1.000     1.523     1.720
           5              211     1.379     1.000     1.132     1.786

Variable   Rating     SE Mean   Minimum   Maximum        Q1        Q3
UP         1            0.213     0.000     4.000     0.000     1.000
           2            0.205     0.000     7.000     0.000     2.000
           3            0.139     0.000     6.000     0.000     2.000
           4            0.155     0.000     8.000     0.000     2.000
           5            0.123     0.000     9.000     0.000     2.000
```

Table 9.14: Descriptive Statistics of Number of vertical mouse cursor lines going up by interest rating

The implicit line detection rating has 4 parts: up to down, down to up, left to right, and right to left. They are all detected separately so they are each counted as an individual implicit rating. For this vertical, bottom to top line detection, it is shown in the descriptive statistics that its median is constant. The patterns of correlation usually involve the increase of both independent variables. If the median is constant, that means there is no correlation between the vertical mouse cursor lines and the explicit interest rating.

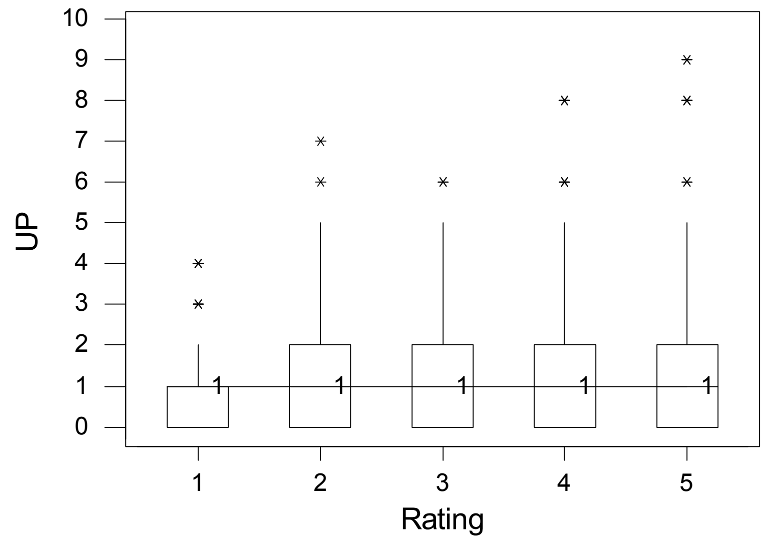


Figure 9.10: Box plot of Num of Mouse cursor moving up vertically vs. Explicit Interest Rating

As shown in Figure 9.10, the plot of the median is horizontal throughout the whole box plot. It suggests that vertical, bottom to top line movements does not have relevance to interest in a page.

9.3.12 Number of vertical mouse cursor lines going down

Descriptive Statistics: DOWN by Rating

```
Variable   Rating           N       Mean     Median     TrMean      StDev
DOWN       1               36      1.417      1.000      1.250      1.574
           2               73      2.411      2.000      2.138      2.374
           3              124      2.726      2.000      2.429      2.676
           4              123      3.610      2.000      3.252      3.454
           5              211      2.834      2.000      2.296      3.506

Variable   Rating     SE Mean    Minimum    Maximum         Q1         Q3
DOWN       1            0.262      0.000      6.000      0.000      2.750
           2            0.278      0.000     12.000      1.000      4.000
           3            0.240      0.000     14.000      1.000      4.000
           4            0.311      0.000     16.000      1.000      5.000
           5            0.241      0.000     20.000      1.000      3.000
```

Table 9.15: Descriptive Statistics of Number of vertical mouse cursor lines going down separated by interest rating

This implicit line rating checks if user moves the mouse cursor only vertically downward. This is also one of the indicators we hope to prove as a good implicit indicator since people read from top to bottom. However, shown in Table 9.15, this implicit line rating has a very slight increase in median. The increase is small and it is hard to tell if the rating is relevant to interest or not.

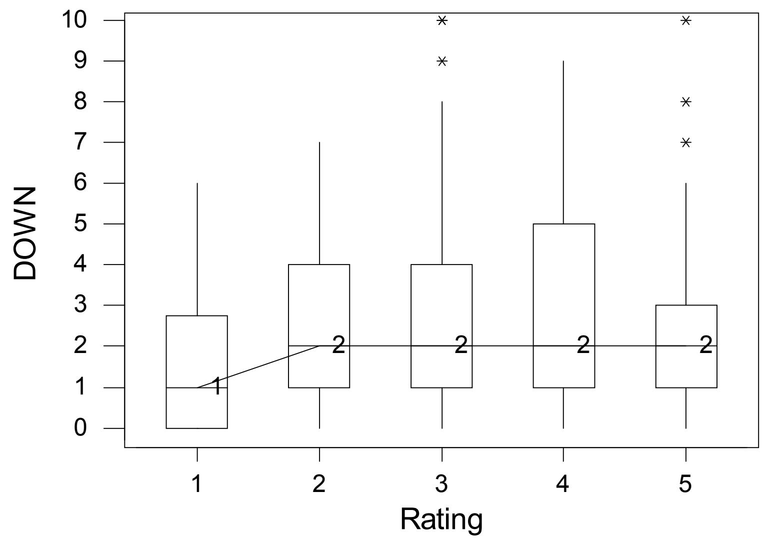


Figure 9.11: Box plot of Num of Mouse cursor moving Down vertically vs. Explicit Interest Rating

The box plot was made to see the increase of median throughout the graph.

Further analysis is needed to see if this implicit vertical line rating is relevant to interest.

9.3.13 Number of horizontal mouse cursor lines going left to right

Descriptive Statistics: L - R by Rating

```
Variable   Rating            N       Mean     Median     TrMean      StDev
L - R      1                36      0.806      0.000      0.594      1.489
           2                73      1.027      1.000      0.877      1.247
           3               124      0.976      0.000      0.679      1.880
           4               123      2.317      1.000      1.604      4.314
           5               211      2.156      1.000      1.381      4.345

Variable   Rating      SE Mean    Minimum    Maximum         Q1         Q3
L - R      1             0.248      0.000      7.000      0.000      1.000
           2             0.146      0.000      6.000      0.000      2.000
           3             0.169      0.000     14.000      0.000      1.000
           4             0.389      0.000     36.000      0.000      2.000
           5             0.299      0.000     34.000      0.000      2.000
```

Table 9.16: Descriptive Statistics of Number of horizontal mouse cursor lines going left to right separated by interest rating

This implicit line rating detects horizontal line from left to right. Since our users usually read from left to right, this implicit rating had a great potential to become a good indicator. Shown in Table 9.16, the median goes up and down. It is very hard to decide if this indicator is relevant to interest. We plotted a graph for this implicit rating (Figure 9.11).

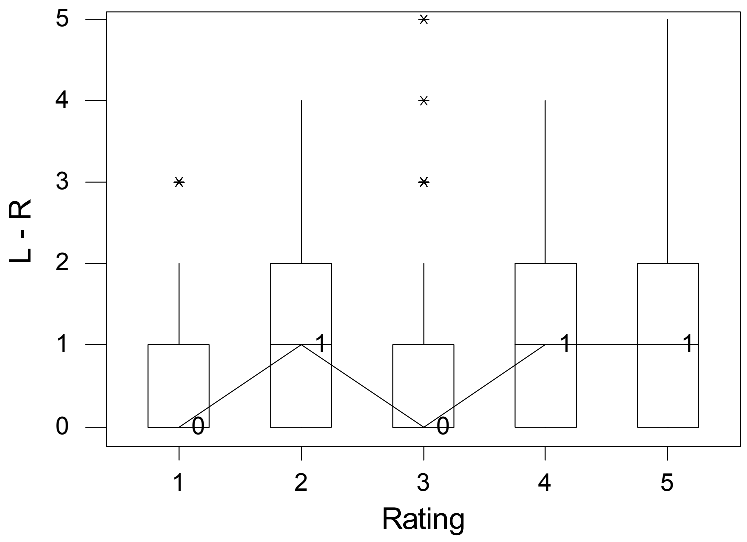


Figure 9.12: Box plot of Num of Mouse cursor moving from left to right vs. Explicit Interest Rating

By looking at the box plot shown in Figure 9.12, the implicit horizontal line rating from left to right does not show any strong correlations either. The median connecting line goes up and down between values 0 and 1. Given that the median values are just two values, further analysis was needed.

9.3.14 Number of horizontal mouse cursor lines going right to left

Descriptive Statistics: R - L by Rating

```
Variable   Rating          N       Mean     Median     TrMean      StDev
R - L      1              36      0.944      1.000      0.781      1.068
           2              73      1.616      1.000      1.523      1.265
           3             124      1.379      1.000      1.188      1.651
           4             123      1.967      1.000      1.532      3.183
           5             211      1.640      1.000      1.090      3.272


Variable   Rating    SE Mean    Minimum    Maximum         Q1         Q3
R - L      1           0.178      0.000      5.000      0.000      1.000
           2           0.148      0.000      6.000      1.000      2.000
           3           0.148      0.000     11.000      0.000      2.000
           4           0.287      0.000     28.000      0.000      2.000
           5           0.225      0.000     27.000      0.000      2.000
```

Table 9.17: Descriptive Statistics of Number of horizontal mouse cursor lines going right to left separated by interest rating

This implicit horizontal line rating from right to left has a constant median, the same as the implicit vertical line rating from bottom to top. As mentioned in Section 9.3.10, the patterns of correlation usually involve the increase of both independent variables. To support the statistics shown in Table 9.17 visually, we came up with Figure 9.13.

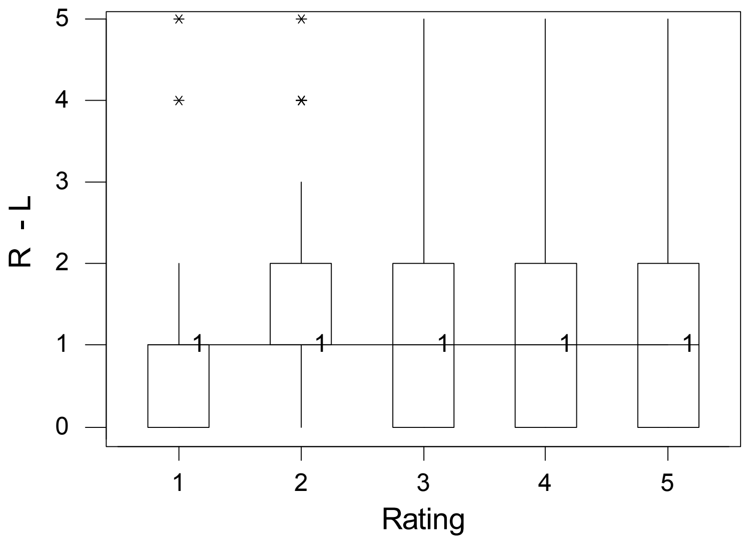


Figure 9.13: Box plot of Num of Mouse cursor moving from right to left vs. Explicit Interest Rating

Figure 9.13 shows clearly that the median is constant for all possible explicit interest ratings. As a result, the descriptive statistics has proved visually that the implicit horizontal line from right to left is not relevant to interest. But since the ranges of the median of this implicit rating is similar to those of the implicit horizontal line rating from left to right, it also needs further analysis.

9.3.15 Number of total mouse cursor lines

Descriptive Statistics: TOTAL by Rating

```
Variable   Rating           N       Mean     Median     TrMean      StDev
TOTAL      1               36      4.139      3.000      3.656      4.141
           2               73      6.630      6.000      6.415      4.405
           3              124      6.621      6.000      6.170      5.493
           4              123      9.610      8.000      8.820      8.635
           5              211      8.009      5.000      6.825      8.925

Variable   Rating     SE Mean    Minimum    Maximum         Q1         Q3
TOTAL      1            0.690      0.000     19.000      2.000      5.500
           2            0.516      0.000     19.000      3.000      9.000
           3            0.493      0.000     26.000      2.250      9.000
           4            0.779      0.000     67.000      4.000     12.000
           5            0.614      0.000     63.000      3.000      9.000
```

Table 9.18: Descriptive Statistics of Number of total mouse cursor lines separated by interest rating

This implicit rating is a collection of all the implicit line ratings. From the median shown in Table 9.18, it shows a slight increase between explicit ratings. However, when the interest rating is 5 the median value drops.

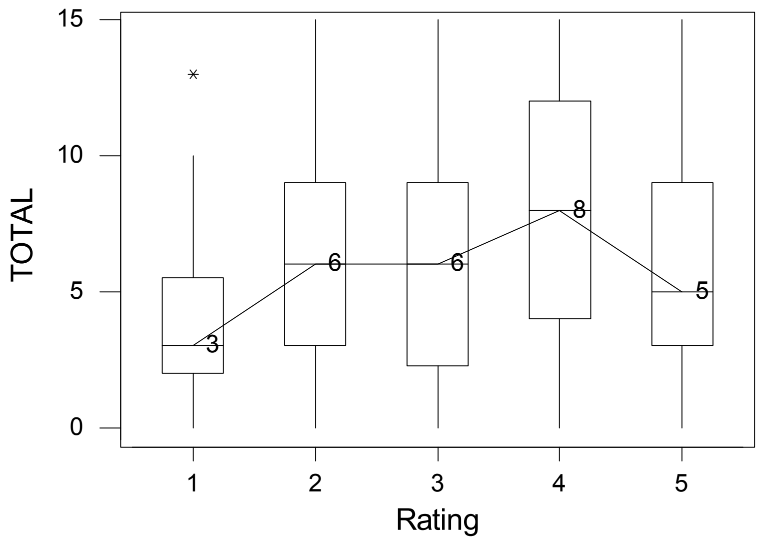


Figure 9.14: Box plot of Num of Total Mouse cursor lines vs. Explicit Interest Rating

Shown in Figure 9.14, the median of the implicit rating total lines, increase slightly throughout the graph. It could be considered as a good implicit indicator but it dropped after rating 4. The percent of the total mouse cursor lines for each direction out of the total number of usable lines were:

Direction	Percent out of Total Number of lines
UP	20
DOWN	37
Left to Right	22
Right to Left	21

Mouse cursor moving down has the highest occurrence. Further tests were also needed for this implicit rating.

9.3.16 Time spent on page versus Total Scrolling

Besides testing correlation between implicit ratings and interest ratings, we wanted to test a correlation between two implicit ratings, time spent on page and total scrolling done on the page. We chose time spent on page because it was a strong indicator of interest in the previous MQP and we chose total scrolling since that was one of the focus areas. Total scrolling is the combination of all the possible scrolling actions such as Mouse Wheel scrolling, Up or Down Arrow scrolling, Scroll bar scrolling and Page Up or Page Down. Figure 9.15 is a line graph of total scrolling and time spent on page.

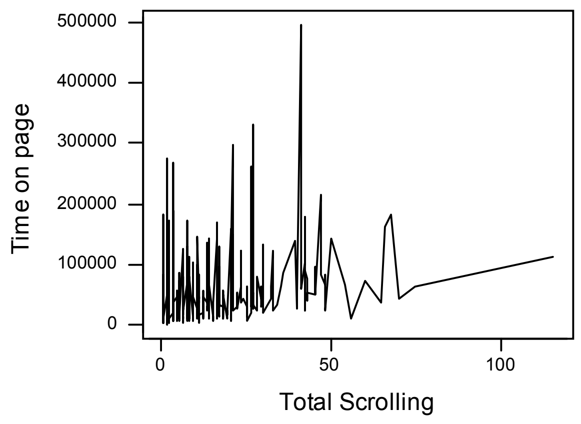


Figure 9.15 Time spent on page versus Total Scrolling

As shown by Figure 9.15 there was no strong correlation that was able to seen visually. There might be a correlation, but maybe due to lack of data, the correlation does not appear.

9.4 GENMOD and Results

GENMOD is a procedure provided by the SAS software. This procedure is to determine the strength of correlation between a variable and another variable that is in an Ordinal Multinomial distribution. Our explicit rating is actually in a multinomial distribution since our interest rating is given in a scale 1 to 5. The scale 1 to 5 is not linear because it is not possible to choose values such as 1.5 or 1.7. The scale used for rating fits the properties of Multinomial models, "where an observation can fall into one of *k* categories" (SAS OnlineDoc Version 8, 1999).

The median range was good for keeping out outliers, but it might hide some of the meaning of the data if the data set was small. That was why GENMOD is used for further testing since it handles all the data accordingly. GENMOD fits a generalized linear model to the data by maximum likelihood estimation of the parameter vector statistics. Beta is the coefficient of the variable, so it can be used to test the null hypothesis. The null hypothesis states that all betas have an equal chance of obtaining a certain value, so the distribution is equal. But when the null hypothesis is rejected, then one of the beta vectors has a different chance of obtaining a certain value; therefore there is a non-random value. If the value is non-random, then it could mean significance. To determine the degree of significance, the *p* value is used. By using the *p* value, we are comparing the statistics computed from the data to a theoretical distribution (in our case multinomial distribution) statistic. The reason for the comparison is to see how closely matched the data is to the model, as if the two distributions are similar, then there is significance.

The implicit ratings that had enough occurrences were passed from the preliminary observations to be analyzed by GENMOD for correlation with explicit interest rating. There were 13 ratings that were passed from the descriptive statistics section. We ran several versions of the GENMOD procedure due to a large elimination of data when we omitted the interest rating value "0", which meant "No comment". For reasons why we had to omit the explicit interest ratings please go to Chapter 9.2. Also due to the discrepancies of sample.dat and sample2.dat, the mouse coordinate lines had to be evaluated with a total usable data entry of 858 as compared to the 1191 entries. When the interest rating value “0” was removed, the usable entries with mouse coordinate lines were 567 while the usable data of the other implicit ratings was left with only 690. For that reason, we had to produce four different GENMOD tests.

The first GENMOD output is of the 858 entries that correspond to the mouse coordinates lines.

```
                                   The GENMOD Procedure

                           Analysis Of Initial Parameter Estimates

                                        Standard   Wald 95% Confidence      Chi-
Parameter              DF   Estimate       Error          Limits          Square   Pr > ChiSq

t_on_page               1    -0.0000      0.0000    -0.0000      0.0000     0.61       0.4332
t_on_mouse              1    -0.0000      0.0000    -0.0000      0.0000     0.04       0.8514
clickonwindow           1    -0.0016      0.0224    -0.0455      0.0424     0.00       0.9446
n_status_bar            1    -0.0120      0.0151    -0.0415      0.0176     0.63       0.4266
n_mousewheel            1     0.0059      0.0072    -0.0082      0.0200     0.68       0.4096
filesize                1     0.0000      0.0000    -0.0000      0.0000     1.07       0.3000
isTypeURL       FALSE   1    -0.2770      0.2077    -0.6841      0.1301     1.78       0.1824
isTypeURL       TRUE    0     0.0000      0.0000     0.0000      0.0000      .          .
isClickURL      FALSE   1     0.0839      0.2507    -0.4074      0.5752     0.11       0.7379
isClickURL      TRUE    0     0.0000      0.0000     0.0000      0.0000      .          .
UP                      1     0.1143      0.0554     0.0058      0.2229     4.26       0.0390
DOWN                    1    -0.0767      0.0301    -0.1356     -0.0178     6.51       0.0107
L_R                     1    -0.3098      0.0510    -0.4098     -0.2099    36.88       <.0001
R_L                     1     0.3281      0.0653     0.2002      0.4560    25.27       <.0001
total                   0     0.0000      0.0000     0.0000      0.0000      .          .
```

Table 9.19: GENMOD test - All implicit ratings and explicit familiarity rating vs. Explicit Interest Rating without “0” rating

The value that we were interested is the numbers on the right most columns. These numbers are the *p* values. Since the test is conducted on a 95% confidence level, any p value that is lower than 0.05 is significant. If the *p* value was less than 0.05 then they are whole row is **bold** for better reading. *P* values evaluate whether the obtained data is significant or not, so according to the output of Table 9.18 we have only 4 ratings that are correlated to the explicit interest indicator.

The four are the four directional lines of mouse coordinates. They are cursor lines going up, cursor lines going down, horizontal cursor lines going left to right, and right to left. These lines showed no visual correlation during the preliminary observations, but the output of GENMOD gave *p* values of that were less than 0.05. The reason the preliminary observations did not show any pattern of correlation was because we were looking at the median. The median range was good for keeping out outliers, but it might hide some of the meaning of the data if the data set was small. The lines occur on average one or two times per entry, but that is still enough high occurrences to be counted as relevant to the total data. That was why GENMOD is used for further testing. GENMOD takes in all data entries and handles them accordingly

Now we will examine the data set that was mostly stored in sample.dat and has 690 entries because of the elimination of the "0" rating. We will once again look at which *p* values are lower than 0.05.

```
                                     The GENMOD Procedure

                            Analysis Of Initial Parameter Estimates

                                          Standard     Wald 95% Confidence       Chi-
Parameter              DF     Estimate       Error           Limits            Square    Pr > ChiSq

t_on_page               1      -0.0000      0.0000     -0.0000      0.0000       1.08        0.2984
t_on_mouse              1      -0.0000      0.0000     -0.0000      0.0000       0.35        0.5524
clickonwindow           1       0.0161      0.0205     -0.0241      0.0563       0.62        0.4317
n_status_bar            1      -0.0206      0.0145     -0.0491      0.0078       2.02        0.1552
n_mousewheel            1       0.0099      0.0068     -0.0035      0.0233       2.09        0.1479
filesize                1       0.0000      0.0000     -0.0000      0.0000       0.46        0.4993
Scale                   0       1.0000      0.0000      1.0000      1.0000
```

Table 9.21: GENMOD test: All implicit ratings (without lines) and explicit familiarity rating vs. Explicit Interest Rating without "0" rating

From this table, we realized that none of the implicit ratings are significant with this small data set of 690. We believe that with a larger data set, more implicit ratings will show significance. We made that assumption based on the preliminary observations we did with the descriptive statistics.

9.5 Drawn Conclusions

From our preliminary observation and GENMOD outputs, only the four directional lines of mouse coordinates are significant to the explicit interest rating. The other implicit indicators such as time on page, time using mouse, clicks on window, status bar changes, mouse wheel scrolls, and total directional lines, were not significant. This means that 12 out of the 13 implicit ratings were not relevant to the explicit interest rating. Significance means a strong correlation exist between the two ratings. It could be strong enough so that one rating can predict the other. However, due to the small amount of data left after trimming, some of the implicit and explicit correlations did not have enough data to enforce their correlation. If the data sample were large enough, we believe that other implicit indicators would show strong correlations also. We assume that belief based on our observations made during the descriptive statistics. For example, time spent on page looked like it had possible correlations based on the median values and its box plot. Also time spent on page was found to be a strong indicator of interest in the previous project, Curious Browser. However, GENMOD did not say it was significant. This could be due to the small amount of data.

Chapter 10 Conclusion and Future work

10.1 Conclusion

The goal of our MQP was to update the previous MQP (Curious Browser) in design and methods. The previous project was done two years ago, and it has given time for Web page designers and engineers to develop new user interfaces and interactive Web pages. We wanted to extend the current list of implicit ratings by adding new ratings such as mouse wheel use and mouse coordinates over time. We also wanted to analyze the data using alternative statistical methods and find better correlations between implicit and explicit ratings.

From our data analysis, we discovered that 8 out of the 12 implicit ratings we use do have strong positive relationships with explicit interest rating. However, after the elimination of all the faulty or irrelevant data, some correlations were not strong enough to appear in descriptive statistics or via the GENMOD statistical test. The ones that did show strong positive relationships were mouse cursor lines going up, mouse cursor lines going down, mouse cursor lines going left to right, and mouse cursor lines going right to left.

We concluded that the inadequacy of data originated from users not rating the Web pages they visit. The evaluation form may have irritated the user into not rating the Web page, by continuously popping up and asking for input. When irritated, users tend to take actions that will resolve the irritation the fastest, thus in this case, clicking the default rating is the quickest relief action. Due to this mindset, close to half our data contained the “No comment” value.

We hypothesized that the implicit mouse wheel ratings would make a good indicator of interest. However, our analysis has shown that it was not. This means that there is an equal amount of scrolling with the mouse wheel regardless of interest. In spite of that, not all mouse related implicit ratings were insignificant. According to the GENMOD test, the four types of cursor line obtained from raw mouse coordinates were stronger indicators of interest than was use of the mouse wheel.

By doing this project, we had learned new skills and also refreshed a few old ones. Here is a listing of these skills:

- API usage;
- How to set up a user experiment;
- How to conduct an Experiment;
- Importance of Explicit and Implicit Ratings;
- Statistical Analysis Methods;
- Human Computer Interaction Concepts;
- How to write a Report about an Experiment

10.2 Future work

Our MQP dealt with the ever-changing Internet so we faced many problems during our implementation, and were not able to address all of them. Here are some possible improvements we would make to the browser:

- Able to handle multiple windows;
- Able to handle extremely long URLs;
- Able to handle redirected pages;
- Able to handle framed pages more efficiently.

The concept of implicit indicators has proved powerful enough to support two experiments so far. In this project, the most interesting indicator was cursor movement in straight lines. Clearly there are many more indicators that could be investigated, as well as combinations of indicators. There are several possible new implicit ratings that would be interesting to research:

- Record events when new windows are opened;
- Record events when users select the Back button;
- Record how many times' users go back to a specific page;
- Record events when users type in the browser window.

The experiment in this project was done with a small user population. To increase the size of the user population and to get more diversity of data to counter the randomness of the data, we suggest:

- Conduct experiment in more labs to increase user population;
- Keep lab computers fresh so they would not crash so easily;
- Conduct the experiment with a specific task to minimize random data.

There is much more analysis that can be done from the raw data of the mouse coordinates such as:

- Detecting smaller lines;
- Detecting diagonal lines;
- Measure the speed the cursor was moving;
- Measure the area covered by the mouse movement on a Web page.

References

Le, P and Waseda, M. *A Curious Browser: Implicit Ratings.* MQP, Computer Science Dept., Worcester Polytechnic Institute, 2000.

Mueller, F and Lockerd, A. "Cheese: Tracking Mouse Movement Activity on Websites, a Tool for User Modeling." MIT Media Lab. 18 Dec. 2001, http://www.media.mit.edu/~alockerd/cheese.pdf

Perkowitz, M and Etzioni, O. "Adaptive Web Sites: Conceptual Cluster Mining". University of Washington, Seattle. 11 Nov. 2001, http://wwwis.win.tue.nl/asum99/perkowitz.html

SAS OnlineDoc Version 8. SAS Institute Inc. 12 April 2002, http://www.sas.com/service/library/onlinedoc/v8/whatsnew/

Sinha, R, Hearst, M, and Ivory, M. "Content or Graphics? An Empirical Analysis of Criteria for Award-Winning Websites." Bailando Project: WebTango. Berkeley, California. 15 Nov. 2001, http://Webtango.berkeley.edu/papers/hfw01/hfw01.htm

Sinha, R, Hearst, M, and Ivory, M. "Empirically Validated Web Page Design Metrics." Bailando Project: WebTango. Berkeley, California. 15 Nov. 2001, http://Webtango.berkeley.edu/papers/chi2001/index.html

Microsoft, *MSDN: IHTMLDocument2 Interface*, Microsoft Corporation., 2002, http://msdn.microsoft.com/library/default.asp?url=/workshop/browser/mshtml/reference/ifaces/document2/document2.asp

S. K. Card, T.P. Moran, A. Newell. *The Psychology of Human-Computer Interaction*, LEA., 1983. p. 26.

Boardwatch Magazine. "Graphically Speaking: Total Number of People (in Milions) with Internet Access from Home", Penton Publishing, Inc., Feb. 2002

Appendix A   Grammars

A.1   Implicit Action Database: Sample.dat

sample.dat grammar:

```
<cs_class>     ::= <user_name> "|" <class> "|" {<class> "|"} "|" <newline>

<tuple>        ::= <URL> "|" <user_name> "|" <date> "|" <time> "|" <msec> "|" <rating>
                   { "|" <rating> }

<rating>       ::= /* character string or positive integer */
<date>         ::= <month> "/" <day> "/" <year>
<time>         ::= <hour> ":" <minute> ":" <seconds> " " <part_of_day>
<newline>      ::= /* white space for rest of line */
<user_name>    ::= /* character string */
<URL>          ::= /* character string */
<class>        ::= /* character string */
<hour>         ::= /* zero to twelve */
<minute>       ::= /* zero to fifty-nine */
<seconds>      ::= /* zero to fifty-nine */
<number>       ::= /* positive integer or zero */
<month>        ::= /* one to twelve */
<day>          ::= /* one to thirty-one */
<year>         ::= /* zero to ninety-nine */
<msec>         ::= /* positive integer or zero */
```

A.2   Coordinate Databases

A.1.1 Sample2.dat grammar (First two weeks of the Experiment)

```
<tuple> ::= [ <URL> ] "|" <user_name> "|" <date> "|" [ <time> ] "|" <mouse_coord>

<mouse_coord> ::= "|" <X> "," <Y> { "|" <X> "," <Y> }
                  [ "[" <idle_time> ] { "|" <X> "," <Y> }

<idle_time>    ::= <X> "," <Y> ":" <msec>

<date>         ::= <month> "/" <day> "/" <year>
<X>            ::= <number>
```

```
<Y>            ::= <number>
<time>         ::= <hour> ":" <minute> ":" <seconds> " " <part_of_day>
<part_of_day>::= AM | PM
<user_name>  ::= /* character string */
<URL>          ::= /* character string */
<class>        ::= /* character string */
<hour>         ::= /* zero to twelve */
<minute>       ::= /* zero to fifty-nine */
<seconds>      ::= /* zero to fifty-nine */
<number>       ::= /* positive integer or zero */
<msec>         ::= /* positive integer or zero */
<month>        ::= /* one to twelve */
<day>          ::= /* one to thirty-one */
<year>         ::= /* zero to ninety-nine */
```

A.1.2 Sample2.dat grammar (updated-last week of the Experiment)

```
<tuple> ::= [ <URL> ] "|" <user_name> "|" <date> "|" [ <time> ] "|" <mouse_coord>

<cs_class>     ::= <user_name> "|" <class> "|" {<class> "|"} "|" <newline>

<mouse_coord>::= "|" <X> "," <Y> { "|" <X> "," <Y> }
                   [ "[" <idle_time> ] { "|" <X> "," <Y> }

<idle_time>    ::= <X> "," <Y> ":" <msec>
<date>         ::= <month> "/" <day> "/" <year>
<X>            ::= <number>
<Y>            ::= <number>
<time>         ::= <hour> ":" <minute> ":" <seconds> " " <part_of_day>
<part_of_day>::= AM | PM
<user_name>  ::= /* character string */
<URL>          ::= /* character string */
<class>        ::= /* character string */
<hour>         ::= /* zero to twelve */
<minute>       ::= /* zero to fifty-nine */
<seconds>      ::= /* zero to fifty-nine */
<number>       ::= /* positive integer or zero */
<msec>         ::= /* positive integer or zero */
<month>        ::= /* one to twelve */
<day>          ::= /* one to thirty-one */
<year>         ::= /* zero to ninety-nine */
<newline>      ::= /* white space for rest of line */
```

Appendix B E-mails to CS Professors

B.1 First E-mail

Professor,

For our MQP my partners and I have created a browser (Curiouser Browser) which records specific actions preformed by a user while browsing the web. Some examples of the information that we might record would be the number of times the click the mouse or the amount of time the user spends on a specific page. Along with this information we also ask the user to rate each page as they leave it on a scale from 1 to 5. This allows us to compare the user's natural actions on a page and their personal preference for the page to determine any relationship between the two. The browser will be installed on 12 computers in the ADP lab during the second and third weeks of March (18th - 29th). My partners and I are searching for students to participate in the project and were hoping that you might be able to encourage your students to participate. If possible maybe you could offer them a point or two towards their grade for participating. The students wouldn't have to do more than browse the web and rate pages and participation will give them some insight to he workings a of an MQP. Any necessary help or guidance will be provided by my partners and I. Any help you may be able to provide us with would be greatly appreciated. Thank you for your time.

Sincerely,
Brad Goodwin
Steve T. Law
Michael Cen

P.S. If you wish to know more about the project you can reach us at bms@wpi.edu or curiouser@wpi.edu

B.2 Second E-mail

Professor,

We've decided to continue our experiment for another week in the ADP. If you could inform your students of this we would appreciate it. If you would like one of us to come by and tell your students what the project is about we could do that as well. Also with your permission, if close to half of the students in your class participate in the experiment before Friday at midnight we will supply your class with a CANDY TREAT. We have updated the browser program. Any problems the students had with the browser we did our best to fix. The experiment will end Friday 4/5 (midnight). So any students that can participate should do so by then. The computers are in the ADP lab and information on which students from your class that participated can be provided to you. I have included a previous message we sent you that contains the URL with our instructions for the students and more information on the project.

If you could even just send out an e-mail to your students informing them of the treat and the website we would very appreciative.

Thank You for you time,
Brad Goodwin
Michael Cen
Steve T. Law

Appendix C Parsing Algorithm Code

C.1 Parse.pl

```
#!/usr/local/bin/perl -w
require "parser.pl";
use strict;


# Make sure we read in the whole file.
$/ = undef;

# Make sure a file exists.
if (!defined ($ARGV[0])) {
    die "Usage: copy.pl <FILE>\n";
}

#ARRAYS

my @array;          #holds each element separated by "|"
my @Xcoordinate;    #holds the X coordinates
my @Ycoordinate;    #holds the Y coordinates


#STRINGS

my $temp_user;      #holds the current user name
my $temp_url;       #holds the current url
my $file;           #holds input from file name given at prompt


#REFERENCE

my $Until_Next_URL; #reference to array returned containing
                    #the coordinates for the current url

my @f;

$file = <>;         #Reads from file
if($file =~ /([^.]dat)/xgi){
  = $1
}

foreach(@f){
    print $_,"\n";
}


#This regular expression will parse the DB by the "|" symbol
#giving each division a spot in the array
@array = ($file =~ /\s*([^\|]+)/xgi);


#creates file parsed.dat and filehandler PARSED
#open(PARSED, ">parsed.dat");
#close(PARSED);

open(PARSED, ">>parsed.dat");


#sets temp_user as the initial user in the given file
$temp_user = $array[0];

print PARSED $temp_user,"\n";


#Loop on @array

for(my $k=0;$k<@array;$k++){


    #Find Coodinates until the next Webpage
    ($k,$Until_Next_URL) = Coordinates_URL($k,\@array);

    #Splits the returned input for the url into separate X and Y arrays
    for(my $i=0;$i<@$Until_Next_URL;$i++){
      ($Xcoordinate[$i],$Ycoordinate[$i]) =
split(/,/,$$Until_Next_URL[$i]);
    }


    #call to find vertical lines
    my $Num_Down = Vertical_Down(\@Xcoordinate,\@Ycoordinate);
    my $Num_Up   = Vertical_UP(\@Xcoordinate,\@Ycoordinate);
    my $Num_LR   = Horizontal_LR(\@Xcoordinate,\@Ycoordinate);
    my $Num_RL   = Horizontal_RL(\@Xcoordinate,\@Ycoordinate);

    if(@Xcoordinate>0)
    {
      print PARSED "|",$Num_Up;      #UP
      print PARSED "|",$Num_Down;    #DOWN
      print PARSED "|",$Num_LR;      #Left to right
      print PARSED "|",$Num_RL,"\n"; #Right to Left
    }

    #Checks for if piece of data in the array is the date.
    if($array[$k] =~ m&(\d{1}/\d{1,2}/\d{2})&xi)
    {
      if($temp_user ne $array[$k-1])
      {
          $temp_user = $array[$k-1];
          #Later this line should be written to a file
          print PARSED "\n",$temp_user,"\n";
      }

      #This if statement looks for the next URL in the file
      #and sets if to $temp_url, then increments k

      if($array[$k-2]=~ m&\d+,\d+http([^\|]+)&xgi)
      {
          $temp_url = $1;
          print PARSED"http",$temp_url;

      }
      else
      {
          #sets the temp_url as the Instruction Web site
          $temp_url = "http://www.wpi.edu/~bradg/Instructions.htm";
          print PARSED $temp_url;

      }
    }
}

close(PARSED);
```

C.2 Parser.pl

```
#!/usr/local/bin/perl -w
use strict;

#THIS TEST STATEMENT CAN BE USED IN ANY FUNCTION
#(except Coordinate_URL)TO PRINT THE VALUES FOUND IN THE LINES
#
#           for(my $p=$tempX;$p<$u;$p++) {print
$$Xcoordinate[$p],",",$$Ycoordinate[$p],"|";}
#           print "\nFIRST COOR of line ",$Vertical;
#           print " is
",$$Xcoordinate[$tempX],",",$$Ycoordinate[$tempX];
#           print ",the last is ",$$Xcoordinate[$u-
1],",",$$Ycoordinate[$u-1],"\n";
#           print "The length of the line in is ",$Length_Count;
#           print "coordinates and ",($$Ycoordinate[$u-1] -
$$Ycoordinate[$tempX]), " pixel*15\n\n";





#
#Finds the coordinates until the next URL
#
#
sub Coordinates_URL
{
    my $position = 0;
    my @coordinates;
    my ($k,$temp) = @_;
    my $i;

    for($i=$k; $i <@$temp; $i++)
    {

      if ($$temp[$i] =~ m&(\d{1}/\d{1,2}/\d{2})&xgi)
      {
          return $i,\@coordinates;
      }
      elsif($$temp[$i] =~ m&(\d+,\d+)http([^\|]+)&xi)
      {
          $coordinates[$position] = $1;
          $position++;
          $i++;

      }
      elsif($$temp[$i] =~ m&(\d+,\d+)&xi)
      {
          $coordinates[$position] =  $1;
          $position++;
      }
    }

    #have to return value less then final value of i
    #because array[i] has no value and can't be parsed
    #with a reg exp.
    return $i-1,\@coordinates;
}



#
#
#Finds vertical lines moving down
#
#
```

```
sub Vertical_Down
{

    my $tempX = 0;                          #Position of the initial
coordinates
                                            #of the line in the arrays

    my $Length_Count = 1;                   #Length of lines in the number
of coordinates
                                            #starts at one of the
beginning of each line

    my $Vertical = 0;                       #Counts the number of vertical
lines

    my ($Xcoordinate,$Ycoordinate) = @_; #creates references to the the
                                            #arrays passed into the
function


    #look for vertical line (x should be approximately the same)

    for(my $u= 1; $u<@$Xcoordinate; $u++){

      #Compares to see the X coordinate it constant
      #(within the given range of 100 positive and 100 negative)
      #and that the Y coordinate moves in one direction (Y increases as
it moves down)

      if(($$Xcoordinate[$tempX]-150 <= $$Xcoordinate[$u]) &&
         ($$Xcoordinate[$u] <= $$Xcoordinate[$tempX]+150) &&
         ($$Ycoordinate[$u] >= $$Ycoordinate[$u-1]))
      {

          $Length_Count++;               #increment the number of
coordinates
                                         #recorded in the line

          #If the last coordinate in the array is part of  a string we
want
          #to increment the number of strings and print to the file.

          if(($u+1 == @$Xcoordinate)&&
             (($$Ycoordinate[$u] - $$Ycoordinate[$tempX]) >= 3000))
          {
            $Vertical++;
          }

      }
      elsif(($$Ycoordinate[$u-1] - $$Ycoordinate[$tempX]) >= 3000)
      {




          #Since the line was found and the current coordinate isn't
          #part of the line we can make make the 'tempX' counter equal
to the
          #'u' counter and continue looking for lines (Length set back
to zero
          #since we are looking for a new line after this)

          $Vertical++;           #increment the number of vertical
lines

          $tempX = $u;

          $Length_Count = 1;     #Makes the length to the string count
back to zero

      }
      else
      {
          $tempX++;
          $u = $tempX;
          $Length_Count = 1;
      }

    }
    return $Vertical;


}


#################
#
#
#
#
#

sub Horizontal_LR
{
    my $tempX = 0;                         #Position of the initial
coordinates
                                           #of the line in the arrays

    my $Length_Count = 1;                  #Length of lines in the number
of coordinates
                                           #starts at one of the
beginning of each line

    my $Horizontal = 0;                    #Counts the number of vertical
lines

    my ($Xcoordinate,$Ycoordinate) = @_; #creates references to the the
                                           #arrays passed into the
function


    #look for vertical line (x should be approximately the same)
    for(my $u= 1; $u<@$Xcoordinate; $u++){

      #Compares to see the X coordinate it constant
      #(within the given range of 100 positive and 100 negative)
      #and that the Y coordinate moves in one direction (Y increases as
it moves down)

      if(($$Ycoordinate[$tempX]-150 <= $$Ycoordinate[$u]) &&
         ($$Ycoordinate[$u] <= $$Ycoordinate[$tempX]+150) &&
         ($$Xcoordinate[$u] >= $$Xcoordinate[$u-1]))
      {

#       print $$Xcoordinate[$u],",",$$Ycoordinate[$u],"|";

          $Length_Count++;              #increment the number of
coordinates
                                        #recorded in the line

          #If the last coordinate in the array is part of  a string we
want
          #to increment the number of strings and print to the file.

          if(($u+1 == @$Xcoordinate)&&
             (($$Xcoordinate[$u] - $$Xcoordinate[$tempX]) >= 3000))
          {
            $Horizontal++;
          }
      }
      elsif(($$Xcoordinate[$u-1] - $$Xcoordinate[$tempX]) >= 3000)
      {

          #Since the line was found and the current coordinate isn't
          #part of the line we can make make the 'tempX' counter equal
to the
          #'u' counter and continue looking for lines (Length set back
to zero
          #since we are looking for a new line after this)

          $Horizontal++;           #increment the number of vertical
lines

          $tempX = $u;

          $Length_Count = 1;     #Makes the length to the string count
back to zero

      }
      else
      {
          $tempX++;
          $u = $tempX;
          $Length_Count = 1;
      }

    }

    return $Horizontal;
```

```
}


#
#
#
#
#
#

sub Vertical_UP
{

    my $tempX = 0;                        #Position of the initial
coordinates
                                          #of the line in the arrays

    my $Length_Count = 1;                 #Length of lines in the number
of coordinates
                                          #starts at one of the
beginning of each line

    my $Vertical = 0;                     #Counts the number of vertical
lines

    my ($Xcoordinate,$Ycoordinate) = @_; #creates references to the the
                                          #arrays passed into the
function


    #look for vertical line (x should be approximately the same)

    for(my $u= 1; $u<@$Xcoordinate; $u++){

      #Compares to see the X coordinate it constant
      #(within the given range of 100 positive and 100 negative)
      #and that the Y coordinate moves in one direction (Y increases as
it moves down)

      if(($$Xcoordinate[$tempX]-150 <= $$Xcoordinate[$u]) &&
         ($$Xcoordinate[$u] <= $$Xcoordinate[$tempX]+150) &&
         ($$Ycoordinate[$u] <= $$Ycoordinate[$u-1]))
      {

        #  print $$Xcoordinate[$u],",",$$Ycoordinate[$u],"|";

          $Length_Count++;               #increment the number of
coordinates
                                         #recorded in the line

          #If the last coordinate in the array is part of  a string we
want
          #to increment the number of strings and print to the file.

          if(($u+1 == @$Xcoordinate)&&
             (($$Ycoordinate[$tempX] - $$Ycoordinate[$u]) >= 3000))
          {
            $Vertical++;
          }
      }
      elsif(($$Ycoordinate[$tempX] - $$Ycoordinate[$u-1]) >= 3000)
      {

          #Since the line was found and the current coordinate isn't
          #part of the line we can make make the 'tempX' counter equal
to the
          #'u' counter and continue looking for lines (Length set back
to zero
          #since we are looking for a new line after this)

          $Vertical++;           #increment the number of vertical
lines

          $tempX = $u;

          $Length_Count = 1;     #Makes the length to the string count
back to zero

      }
      else
      {
          $tempX++;
          $u = $tempX;
          $Length_Count = 1;
      }

    }

      return $Vertical;



}


#################
#
#
#
#
#

sub Horizontal_RL
{
    my $tempX = 0;                      #Position of the initial
coordinates
                                        #of the line in the arrays

    my $Length_Count = 1;               #Length of lines in the number
of coordinates
                                        #starts at one of the
beginning of each line
```

```
    my $Horizontal = 0;                  #Counts the number of vertical
lines

    my ($Xcoordinate,$Ycoordinate) = @_; #creates references to the the
                                         #arrays passed into the
function


    #look for vertical line (x should be approximately the same)

    for(my $u= 1; $u<@$Xcoordinate; $u++){

      #Compares to see the X coordinate it constant
      #(within the given range of 100 positive and 100 negative)
      #and that the Y coordinate moves in one direction (Y increases as
it moves down)

      if(($$Ycoordinate[$tempX]-150 <= $$Ycoordinate[$u]) &&
         ($$Ycoordinate[$u] <= $$Ycoordinate[$tempX]+150) &&
         ($$Xcoordinate[$u] <= $$Xcoordinate[$u-1]))
      {

#       print $$Xcoordinate[$u],",",$$Ycoordinate[$u],"|";

          $Length_Count++;              #increment the number of
coordinates
                                        #recorded in the line

          #If the last coordinate in the array is part of  a string we
want
          #to increment the number of strings and print to the file.

          if(($u+1 == @$Xcoordinate)&&
             (($$Xcoordinate[$tempX] - $$Xcoordinate[$u]) >= 3000))
          {
            $Horizontal++;
          }
      }
      elsif(($$Xcoordinate[$tempX] - $$Xcoordinate[$u-1]) >= 3000)
      {

          #Since the line was found and the current coordinate isn't
          #part of the line we can make make the 'tempX' counter equal
to the
          #'u' counter and continue looking for lines (Length set back
to zero
          #since we are looking for a new line after this)

          $Horizontal++;           #increment the number of vertical
lines

          $tempX = $u;

          $Length_Count = 1;     #Makes the length to the string count
back to zero
```

```
      }
      else
      {
          $tempX++;
          $u = $tempX;
          $Length_Count = 1;
      }

    }

      return $Horizontal;


}
1;
```