A Usability Problem Diagnosis Tool: Development and Formative Evaluation

Reenal Mahajan

Thesis submitted to the faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Master of Science
in
Computer Science

Advisory Committee:
Dr. H. R. Hartson (Chair)
Dr. Chris North
Dr. Manuel Perez-Quinonez

June 10th, 2003
Blacksburgh, Virginia.

Keywords: Usability problem diagnosis, UAF, UPD, usability evaluation, usability problem record
A Usability Problem Diagnosis Tool: Development and Formative Evaluation

Reenal Mahajan

(Abstract)

Usability evaluation results in several usability problems and the non-UE developer is often not a part of the evaluation as it might deter the participant from reporting all the errors and also, conducting usability evaluation is a usability engineer’s responsibility. Thus the evaluator needs to create unambiguous usability problem reports, which will help the developer fix the usability problems.

This research involves the development and evaluation of the Usability Problem Diagnosis tool, which supports problem diagnosis through analysis and storage in a common database shared between the evaluation and the development team. This tool uses the User Action Framework as an underlying knowledge base to support problem diagnosis.
Acknowledgements

I wish to thank my advisor, Dr. Rex Hartson, for his invaluable help and guidance throughout my thesis. He has always been there to listen to my doubts and encouraged me through hard times. His enthusiasm in my work pushed me to work harder. Without his help I would not have been able to complete my thesis.

I am grateful to all my committee members – Dr. Chris North and Dr. Manuel Perez, for their valuable inputs.

I would like to thank Narayanan Kodiyalam and Vikrant Colaso for helping me in setting up the usability lab for evaluation and for being there whenever I needed moral support and encouragement.

I am very grateful to my parents, brother and family for their immense support and encouragement throughout my graduate life.

Lastly, I would like to thank all my friends and faculty at Virginia Tech for making my graduate life the most memorable time of my life.
Table of Contents

CHAPTER 1. INTRODUCTION AND PROBLEM STATEMENT ------------------- 1

CHAPTER 2. GOALS AND SCOPE --------------------------------------- 3

2.1. GOALS ----------------------------------------------------------- 3

2.2. SCOPE -------------------------------------------------------------- 3

CHAPTER 3. BACKGROUND --------------------------------------------- 5

3.1. OVERVIEW OF THE USABILITY PROBLEM DIAGNOSIS TOOL -------------- 5

3.2. CONTEXT WITHIN USABILITY ENGINEERING LIFE CYCLE ACTIVITIES ----- 5

3.3. ABOUT THE UAF ---------------------------------------------------- 6

3.3.1. UAF content and structure -------------------------------------- 7

3.3.2. How UAF is used by the Usability Problem Diagnosis tool ------- 8

3.4. RELATED WORK ----------------------------------------------------- 8

3.4.1. Usability Problem Extraction ------------------------------------ 8

3.4.2. Usability Problem Classification ------------------------------- 9

3.4.3. Existing UAF work --------------------------------------------- 10

CHAPTER 4. APPROACH ------------------------------------------------ 14

4.1. TOOL DEVELOPMENT --------------------------------------------- 14

4.2. FORMATIVE EVALUATION ------------------------------------------ 14

4.2.1. Iterative design walkthrough ---------------------------------- 15
## Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Usability Engineering Life Cycle</td>
<td>6</td>
</tr>
<tr>
<td>3-2</td>
<td>User Action Framework</td>
<td>11</td>
</tr>
<tr>
<td>3-3</td>
<td>Usability Problem Classifier</td>
<td>12</td>
</tr>
<tr>
<td>3-4</td>
<td>Usability Database Tool</td>
<td>13</td>
</tr>
<tr>
<td>5-1</td>
<td>Usability Problem Diagnosis Tool</td>
<td>18</td>
</tr>
<tr>
<td>5-2</td>
<td>Database Tool</td>
<td>19</td>
</tr>
<tr>
<td>5-3</td>
<td>Flow of actions during problem record creation in earlier tool</td>
<td>21</td>
</tr>
<tr>
<td>5-4</td>
<td>Raw Data Component</td>
<td>24</td>
</tr>
<tr>
<td>5-5</td>
<td>Psychological and Physical Effects</td>
<td>25</td>
</tr>
<tr>
<td>5-6</td>
<td>Perceptual and Task Effects</td>
<td>26</td>
</tr>
<tr>
<td>5-7</td>
<td>The UPD uses the UAF to specify the problem type, sub-type and cause(s)</td>
<td>28</td>
</tr>
<tr>
<td>5-8</td>
<td>The cause(s) are aligned in tree structure to improve readability</td>
<td>29</td>
</tr>
<tr>
<td>5-9</td>
<td>The user can click on any level to start classifying the additional cause</td>
<td>29</td>
</tr>
<tr>
<td>6-1</td>
<td>Relationship of Usability Problem Diagnosis Tool and Usability Problem Database Tool with other usability tools</td>
<td>32</td>
</tr>
<tr>
<td>6-2</td>
<td>Internal working of the Usability Problem Diagnosis Tool</td>
<td>33</td>
</tr>
<tr>
<td>8-1</td>
<td>The wordings of the circled link are unclear</td>
<td>44</td>
</tr>
<tr>
<td>8-2</td>
<td>Incorrect wording of “Enter these values in the current UP record” button confused the user</td>
<td>45</td>
</tr>
</tbody>
</table>
FIGURE 8-3. THE LABEL OF THE BUTTON "REMOVE" SHOULD BE CHANGED TO "REMOVE SELECTED EFFECTS".----------------------------------------------- 46

FIGURE 8-4. AUTOMATIC SCROLLING-DOWN OF THE PAGE TO THE NEXT EFFECT CATEGORY
------------------------------------------------------------------------ 47

FIGURE 8-5. THESE LINKS ARE NOT IMMEDIATELY OBVIOUS. ------------------------ 48

FIGURE 8-6 THE GREEN FONT IS IGNORED BY THE USER -------------------------- 49

FIGURE 8-7. TOO MUCH OF TEXT ON THIS PAGE WHICH IRRITATED THE USER-------- 50

FIGURE 8-8 AN AFFORDANCE FOR VIEWING PROGRESS IN PROBLEM RECORD CREATION.

HOWEVER, THE USER WAS CONFUSED AND MISTOOK THEM FOR LINKS TO THE RESPECTIVE COMPONENTS.------------------------------- 51

FIGURE 8-9. THE WORDING OF "CONTINUE" BUTTON MUST BE CHANGED TO "CONTINUE TO CREATE PROBLEM RECORD"---------------------------------- 51

FIGURE 8-10 THE LAYOUT OF THE PAGE IS CHANGED TO REFLECT THE GROUPING OF THE RELATED FUNCTIONS AND PLACE THE BUTTONS IN AN ORGANISED WAY. ------- 53

FIGURE 8-11 A WARNING MESSAGE IS DISPLAYED TO THE USER WHEN THEY CLICK ON THE "EDIT EFFECTS ON THE USER" BUTTON.------------------------ 54

FIGURE 8-12 THE USER CAN NOW ADD OR REMOVE MULTIPLE EFFECTS AT THE SAME TIME.-------------------------------------------------- 55
Chapter 1. Introduction and Problem Statement

"Usability has become a competitive necessity for the . . . success of software" [Butler, 1996]. In this competitive world, where there are several products serving the same purpose, the importance of usability has increased. Before the product is shipped, the software developer along with the usability evaluator must expose and fix most of the potential usability errors.

During lab-based usability evaluation, the evaluator calls in several representative user participants to perform a set of tasks using the product under evaluation in a confined/field environment. These tasks are called “benchmark tasks” defined earlier by the evaluator to be representative tasks. Selection of users for the evaluation should be done carefully. They should match the target system user classes and have sufficient domain knowledge.

A number of usability problems are exposed during the evaluation. These problems are reported to the developer to analyze and fix. The developer is, generally, not a part of the usability evaluation as it might deter the participant from reporting all the errors. The problem here is that the software developer might not understand what the problem clearly is and where it is located. Thus the developer is dependent on how well the evaluator makes notes and produces the final problem reports.

To solve this problem, we developed the Usability Problem Diagnosis (UPD) tool, which will allow the evaluator to diagnose and save the usability problem descriptions in a database. The developer can then retrieve the problem using the Usability Database tool and fix it. Problem diagnosis, by problem type and sub-type and by identifying and
isolating the problem cause(s), reduces ambiguity and makes the task of solving the problems easier.

After the development of the UPD tool, we conducted our own formative evaluation to uncover usability problems associated with the tool itself. We analyzed the results of the evaluation, performed cost-importance analysis and fixed few important problems.
Chapter 2. Goals and Scope

2.1. Goals

The goal of my thesis is as follows:

1. Develop the Usability Problem Diagnosis tool to diagnose and save usability problems in a common database, shared by the software development team.

2. Conduct formative evaluation to uncover usability errors associated with the UPD tool and perform cost-importance analysis.

3. Make a few improvements to the UPD tool on the basis of the results of the evaluation.

2.2. Scope

Work accomplished during the thesis:

- Designed and implemented the Usability Problem Diagnosis tool to diagnose usability problems.

- Formative evaluation of the design.

We used formative evaluation to evaluate the design, as it was part of an even larger iterative cycle including future developers of the tool. Formative evaluation was conducted over the whole life of the cycle of the design and consisted of two kinds of evaluation: Large number of cycles of design walkthroughs [Lewis, Polson, Wharton, & Rieman, 1990; section 5.4] and one final usability testing study (section 7).

  o Iterative design walk-throughs and redesign: “The cognitive walkthrough is a technique for evaluating the design of a user interface, with special attention to how well the interface supports
"exploratory learning," i.e., first-time use without formal training" [Rieman, J., Franzke, M., and Redmiles, D., 1995]. It is typically performed by interface designer and their group [Rieman, J., Franzke, M., and Redmiles, D., 1995]. I periodically met with Dr. Hartson, and we would perform a few representative user tasks identifying the possible flaws in the design. After each walkthrough, I corrected the design and we would repeat the above cycle.

- One final, formal formative usability evaluation study (usability testing) to identify as many remaining usability problems as possible

  - Fixed a small selection of the usability problems found

The following things were not addressed as part of my thesis:

- Summative evaluation of the tool performance as a usability evaluation method (e.g., in comparison to other established methods); we did not test the performance of Usability Problem Diagnosis tool.

- Fix all the problems found (did suggest solutions to be passed on to the next developer in the tool's evolution)

- Report management user interface or functionality, for reading, reviewing, or modifying existing usability problem reports.
Chapter 3. Background

3.1. Overview of the Usability Problem Diagnosis Tool

The practitioners will use the Usability Problem Diagnosis tool to diagnose the usability problems. The “diagnosis” provides the observed effects on user during the evaluation and the cause(s) of the problem. This tool uses the User Action Framework (UAF) [section 3.3] to identify the problem typing causes. Every detail of the problem is captured through a set of six components: Raw Data, Effects on user, Diagnosis, Cost-Importance, Implementer and Management. Each of these components is explained in detail in section 5.4.

3.2. Context Within Usability Engineering Life Cycle Activities

The usability-engineering life cycle [Hix and Hartson, 1993; Mayhew, 1999], shown in Figure 3.2, consists of four major steps: analyze, design, build and evaluate. This cycle is repeated several times before a complete version of the product is built and then this product is tested and summative evaluation is performed. During the development of our tool we performed several iterations. Initially we evaluated our design through regular walkthroughs and finally we conducted a lab-based formative evaluation involving representative users in this domain. This activity yields a list of usability problems, but it's ad hoc, unordered, unanalyzed, and unprioritized. To decide which problems to fix, their problem type and causes, and to help think about potential solutions, we need to diagnose the problems thoroughly. Thus after prioritizing these problems by cost and importance, we fixed a few important problems. This process can now be repeated by the future developers building on our version of the UPD tool.
3.3. About the UAF

Due to the growing importance of usability, considerable resources are devoted to conducting usability evaluations and incorporating usability-engineering practices in software development [Hix and Hartson, 1993]. The effectiveness of the usability development activities is limited by the lack of a unifying framework for the usability tools [Hartson, Andre, Williges, van Rens, 1999].

In answer to this need, UAF is a unifying framework, which integrates tools for analysis, classifying, storage, retrieval, reporting and redesign. “It is a theory-based structure for organizing usability concepts, issues, design features, usability problems,
and design guidelines”. [Hartson, Andre, Williges and Rens, 1999]. The framework is based on the adaptation and extension of Norman’s theory of action [Norman, 1996], called the Interaction cycle, which provides a high level structure for organizing the contents of the Usability Problem Classifier.

“The UAF classifies and “contextualizes” design problems in terms of user’s interaction-based behavior. This allows us to understand the significance of design problems in the totality of cognitive and physical aspects of task performance.” [Hartson, Andre, Williges, van Rens, 1999].

### 3.3.1. UAF content and structure

The unified framework consists of the following tools:

- **UAF Explorer** – for teaching usability concepts.
- **Usability Problem Diagnosis tool** – for extracting, analyzing, diagnosing, and reporting usability problems by problem type and by causes.
- **Usability Problem Inspector tool** – for conducting focused usability inspections, guided by the categories and sub-categories of the UAF.
- **Usability Database tool** – for maintaining a life history record of each problem within a project and for supporting aggregate data analysis such as cost-importance analysis [Hix & Hartson, 1993] and usability data visualization;
- **Usability Design Guidelines tool** for organizing and applying usability design guidelines in a systematic way.

These tools are independent of the UAF framework but use UAF as a framework to evaluate and analyze an interaction design and the associated usability problems and their causes.
3.3.2. How UAF is used by the Usability Problem Diagnosis tool

The UAF is a hierarchically structured knowledge base of usability concepts. It has become a common conceptual framework, supporting a number of usability tools like UPD and UPI. While the meaning and the structure are retained, the concepts are expressed in a way suitable for the purpose of the tool. [Hartson, Andre, Williges, van Rens, 1999].

For the purpose the UPD, the expression is in terms of the cause(s) of the usability problems. It allows the practitioner to document problems found i.e. provide a complete description of the problem type and subtype [Hartson, Andre, Williges, van Rens, 1999].

3.4. Related work

The workshop at the annual CHI conference in 1994 addressed the topic of what to do after the usability data has been collected [Nayak, Mrazek, & Smith, 1995]. There was a need for a set of tools to convert the raw observational data, collected during usability evaluations, to meaningful conclusions and inferences, which can help in fixing the usability problems discovered.

3.4.1. Usability Problem Extraction

During a usability evaluation several usability problems are discovered and each problem has different cause and a different solution. Hence, it is very important to extract these problems, find the root cause, and report them separately enabling the developer to come to an efficient solution to the problem.

In order to separate individual usability problems Cockton and Lavery [1999] have developed a framework for usability problem extraction called Structured Usability
Problem Extraction (SUPEX). Using this tool the evaluators can reliably convert the usability data into sets of individual problems. An advantage of this tool is that it allows the user to identify the usability problems by isolating and analyzing parts of interaction at a time, thus all the problems are identified and reported. Though this tool is helpful in problem extraction, a more unified approach is needed for problem diagnosis, usability data management and visualization. Extraction is part of the overall process of usability problem analysis. The complete problem analysis consists of the problem extraction and diagnosis. Our Usability Problem Diagnosis tool will provide the user with problem extraction and diagnosis and the Usability Database tool will allow the user to manipulate the created usability problem records.

3.4.2. Usability Problem Classification

Design Guidelines [Mayhew, 1992; Shneiderman, 1998] and heuristics [Nielsen & Molich, 1990] provide a basis for high-level classification of the usability problems. Nielsen and Molich [1990] defined a set of heuristics for heuristic evaluation (usability inspection). The usability problems discovered at the end of heuristic evaluation can be explained with reference to these heuristics. As used by Nielsen and Molich, these methods are concentrated on identifying rather than classifying usability problems. However, some used these heuristics as classification categories. Due to small number of these categories, the classification power was limited. Since the Usability Problem Diagnosis tool is based on User Action Framework, it has access to a few hundred categories of the User Action Framework for classification.

Mack and Montaniz (1994) have investigated general classification of usability problems. Others have tried classifying problems in terms of their source and location in


These above approaches to classifying usability problems are adhoc and unstructured.

3.4.3. Existing UAF work

The User Action Framework, as shown in Figure 3.1, was built by Dr. Rex Hartson, Terence Andre, Steven Belz, Faith McCreary and Miranda Capra. The reliability of the framework was tested by Sri Sridharan [Sridharan, 2001]. I used this existing framework to solve the problem diagnosis component of the usability problem record. Each node in the User Action Framework specifies a usability attribute thus a complete path (till the terminal node) in the framework provides a list of usability attributes associated with the usability problem, which assists in building a complete problem report [Sridharan, 2001]. Thus the User Action Framework can be used in diagnosing the problem type, sub-type and cause(s). The result of the diagnosis is a path consisting of several nodes which together completely describe the problem.
The Usability Problem Classifier tool, as shown in Figure 3.2, was the combined effort of the following people: Linda van Rens, Rex Hartson, Pratima Aiyagari, Linh Van, Helena Mentis, Priya Shivakumar and this author. Initially, the design of the tool was such that the user started creating the usability problem record in the Usability Database tool (built by Rex Hartson, Pratima Aiyagari, Linh Van, Helena Mentis, Priya Shivakumar and this author), as shown in Figure 3.3, and shifted to Usability Problem Classifier tool for classifying the problem causes. The design of the previous tools – Usability Problem Classifier and Usability Database was more driven by the database structure and was not a match to the user tasks, which spanned both the tools. I used the basic idea of these tools to build the Usability Problem Diagnosis tool. This tool was
concerned with the creation of a usability problem record. It eliminated the need to shift back and forth from the Database tool to the Classifier tool. Further, the Usability Problem Diagnosis tool greatly expands the breadth of scope and functionality of the older Usability Problem Classifier tool by allowing the user to specify more than one cause for usability problems and providing means to build and modify existing components of the usability problem record. In the older Usability Problem Classifier tool, the usability causes and effects on user were not saved in a database and were lost once the application was shut down.

Figure 3-3 Usability Problem Classifier
Figure 3-4 Usability Database Tool
Chapter 4.  Approach

My main focus was to develop a tool for diagnosing and storing usability problems and provide the developer with all the required information to fix the usability problem. To develop a "usable" tool, I used the following approach:

4.1. Tool Development

We analyzed the requirements of the tool. Since the tool will be used by the evaluator and the developer, the requirements were assessed for completeness from both the user's point of views. The evaluator will use the tool to enter usability problems into the database and the developer will use the tool to extract the usability problems and fix them. Based on the requirements we designed a normalized database, which provided for efficient querying and data retrieval. Next, we designed the user interface. We used "paper-prototypes" to get the initial design. Once the initial design was finalized we did regular walkthroughs to fine tune the design and uncover any faults. Later we connected the interface to the database and performed walkthroughs to ensure correctness and completeness of the software. We applied formative usability evaluation continuously throughout the project. It was of two types: iterative design walkthroughs and final usability evaluation study.

4.2. Formative evaluation

Formative evaluation is the “evaluation of the interaction design, as it is being developed, for the purpose of improving usability” [Hix and Hartson, 1993].
4.2.1. **Iterative design walkthrough**

These walkthroughs were performed every week for a period of nine months. During these walkthroughs, Dr. Hartson and the author sat together and performed simple representative user tasks testing the functionality developed and noted down the modifications needed or new features to be added to the tool. The required changes were made and the above cycle was repeated every week.

4.2.2. **Lab-based formative usability evaluation study**

Before the evaluation, we prepared a list of “benchmark tasks” (see list in Appendix A), set-up the lab equipment and selected six participants. The criterion for selecting the participants was that they should have had sufficient usability experience and knowledge of the User Action Framework (Refer to section 7.5 for more details). The equipment was setup such that it allowed the evaluator to view the participant’s screen and record their actions. During the evaluation the participant is encouraged to “think aloud” while performing the benchmark tasks. The negative/positive “critical incidents” are recorded. The tasks are not timed and the stress is more on getting the user to complete the task correctly. This study is discussed in detail in section 7. After the evaluation, the data is analyzed and the usability problems noticed are categorized and documented.

4.3. **Iterative redesign**

Following the formative usability study we analyzed the usability problems discovered during the evaluation, performed cost-importance analysis and fixed a few important problems.
These three parts of my approach- tool development, formative evaluation and iterative redesign are each discussed in the following sections.
Chapter 5. Tool Development

5.1. Software Architecture and Implementation

We used Active Server Pages (ASP) to implement the tool. The database was implemented in Microsoft Access and ODBC was used to connect the front-end to the database. We used Macromedia Dreamweaver to design the front-end. The database was organized such that it separated the UAF content and structure into two distinct tables to minimize the access time.

5.2. Usability Problem Diagnosis Tool

As the name suggests this tool is concerned with the “diagnosis” of usability problems. By “diagnosis” we mean, determining the problem type and finding the cause(s) of the problem using the User Action Framework (UAF). In some cases the exact causes may not be represented in the UAF but still the category of the problem can be correctly identified.
The usability evaluator can use this tool, shown in Figure 5.1, to create and store usability problem records. The usability problem record has the following components:

The **Raw Data Component** contains a brief description of the problem based on observations in the usability lab (for example); the **Effects on User Component** contains a description of the effects (Psychological, Physical, Task and Perceptual) on the user while performing the task; and the **Diagnosis Component** records the problem type, subtype and causes of the problem with the help of User Action Framework. Three other components such as **Cost/Importance**, **Management** and **Implemneter Component** have not been implemented yet.

Figure 5-1 Usability Problem Diagnosis Tool
5.3. Database Tool

This tool, shown in Figure 5.2, is concerned with retrieving and manipulating the problem records stored in the database. The full functionality of this tool has not been implemented in our version. Currently the user can retrieve the current usability problem record and edit its fields.

![Database tool](Figure 5-2 Database tool)
5.4. Design

5.4.1. Usability Problem Diagnosis Tool (UPD)

The UPD was earlier known as Usability Problem Classifier (UPC). It was used to enter the effects on the user and the causes of the usability problem during the creation of the usability problem record. The user starts creating the problem record in the Usability Database tool and switches to UPC for the above-mentioned reasons. While doing our regular walkthroughs, we realized that the user was finding it awkward to shift between tools while creating the problem record. The user felt a complete “loss of control” during the shift and found it uncomfortable to hit the browser “back” button to edit the earlier section of the problem record. Also they found it awkward to start creating the problem record from the Usability Database tool. Refer to the Figure 5-3 below for the flow of actions when creating a usability problem record using UPC and Database tool. The user starts the usability problem record creation from the Usability Database tool, shifts to UPC tool to enter the effects on user, returns back to Usability Database tool, shifts again to UPC tool to enter the problem type, sub-type and cause(s) and then finally returns back to Usability Database tool to view the summary of the created problem report.
Figure 5-3. Flow of actions during problem record creation in earlier tool

After clicking on “Diagnose the problem”

After entering the effects on user

After clicking on “Classify the effects on user”

Classifier

Database

Classifier

Database
To avoid the awkwardness of switching back and forth between the Usability Problem Classifier and the Usability Database tool, we decided to have a separate Usability Problem Diagnosis tool (UPD) tool that is concerned with the creation of the usability problem record. The Usability Database tool will only handle the functionality of manipulating and retrieving the problem records from the database.

Initially, we displayed the entire problem record to the user for entering the values. We realized that the user could be intimidated by the amount of data to be entered. Hence, we decided to display the record step by step thus guiding the user through each data component in the process. Thus the “Raw Data” component is initially displayed to the user, followed by the “Effects on User” component, “Diagnosis” Component, “Cost/Importance” component, “Management” component, and “Implementer” component.
5.4.2. The Raw Data Component

The raw data component, shown in Figure 5.4, contains the raw details of the usability problem record and gives a brief overview about the possible cause(s). It is subdivided into three sections- General Information, General Description, and Evaluation Details.

The General Information section contains the name, date, author, and the source of the problem. The General Description section captures the viewpoint of both the evaluator and the designer on the problem. It includes the problem description, as perceived by the evaluator, and the explanation of the designer regarding why the problem occurred, and if it could be avoided. In the Evaluation Details section, the evaluator notes down the comments observed during evaluation, makes suggestions if further evaluation is needed and provides a link to the multimedia link, if available.
5.4.3. The Effects-on-User component

The effects on user component, captures the effects on the user, when the problem occurred, during the usability evaluation. The effects are categorized into four main categories – psychological (e.g. surprised, confused, irritated etc.), physical (e.g. extra unnecessary actions, difficulty with noticing, etc.), perceptual (e.g. difficult discerning text, etc.), and task (e.g. cannot complete task, etc.). The effects are displayed in the order of importance as shown in Figure 5.5 and 5.6. The most commonly observed effects are the psychological effects, followed by the physical, perceptual, and task effects. A list of
effects is displayed in each category for the user to choose from and add the most relevant ones for the current problem. The user may not find any of the effects “relevant” in the given list, hence a provision is provided to the user for creating their own effect and adding it to both the lists – relevant and general. The user is also provided with a way to remove the added effect from the relevant list. This action does not cause the effect to be removed from the general list. Currently, this action is not supported by the tool.

**Figure 5-5 Psychological and Physical effects**
5.4.4. The Diagnosis component

The diagnosis component uses the User Action Framework to identify the problem typing causes for the current problem. The user is allowed to add an additional cause for each problem and can comfortably move back and forth examining node descriptions while trying to identify the problem cause(s). We anticipated that no problem would have more than two causes. We also tried to accommodate both categories of tool users – novice and expert, by providing the tree structure in the left frame for expert users and corresponding detailed node descriptions with physical
affordance to move up and down the tree levels for the novice users, as shown in Figure 5.7.
Figure 5-7 The UPD uses the UAF to specify the problem type, sub-type and cause(s).
The nodes in the cause(s) selected by the user are made clickable and aligned to reflect the tree structure (as shown in Figure 5.8) thus improving the readability and enabling the user to click on any level to start classifying the additional cause as shown in Figure 5.9.

**DIAGNOSIS COMPONENT:**

<table>
<thead>
<tr>
<th>Cause(s):</th>
<th>Classification Paths:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User Action Framework</td>
</tr>
<tr>
<td></td>
<td>- Planning (Design helping user know what to do)</td>
</tr>
<tr>
<td></td>
<td>- User and work context</td>
</tr>
<tr>
<td></td>
<td>- User's ability to keep track of how much is done</td>
</tr>
</tbody>
</table>

Figure 5-8 The cause(s) are aligned in tree structure to improve readability.

**Additional Classification**

**Selected Causes**

- [ ] User and work context

**Click on where you want to go to find additional clauses:**

- User Action Framework
  - Planning (Design helping user know what to do)
  - User and work context

Figure 5-9 The user can click on any level to start classifying the additional cause.
Chapter 6. Tool usage

6.1. Scenario of tool usage

Company ABC’s software development team is working on the development of an e-Commerce system dealing with car sales and rental. The software will be eventually uploaded to the Internet and will be viewed and used by several users of varying computer skills. This software needs to be as usable as possible; thus the company has a separate usability group which tests the usability of the software submitted to them and sends back a detailed report of the usability problems associated with the software. The usability team decides on doing lab-based formative usability evaluation of this software. The other alternatives could have been to use the Usability Problem Inspector tool to inspect the software or to use the Remote Usability Evaluation tool or to use Ideal tool to collect usability problems. The evaluation is conducted using participants who are knowledgeable in the problem domain since this is a formative evaluation. During the evaluation, the usability evaluator notices several usability problems and makes rough notes of the details on paper. After the evaluation is complete, the usability evaluator runs the Usability Problem Diagnosis tool and enters reports for the usability problems one by one. While entering each problem, the evaluator diagnoses the problem type, sub-type and cause(s) using the User Action Framework incorporated in the Usability Problem Diagnosis tool and specifies the effects on user observed during the evaluation. Thus the evaluator creates a usability problem record for each usability problem, which is then stored in the database. This database is shared with the software development team. The development team is notified of completion of evaluation, and they then start working on fixing the problems observed. The software developer runs the Usability Problem
Database tool, opens the current project, views the complete problem reports and fixes them one by one. After the changes have been made to the software and some more features added this cycle is repeated again until the completion of the software. After this, summative evaluation is performed to test the effectiveness of software where again the Usability Problem Diagnosis tool is used to record usability problems observed.

6.2. Flow Diagrams

The following flow diagrams depict the usage of the Usability Problem Diagnosis tool and the Usability Problem Database tool with other usability tools (section 6.2.1); and the internal working of the Usability Problem Diagnosis tool (section 6.2.2).

6.2.1. Relationship with other usability tools

As shown in Figure 6-1, the usability problems noted using lab-based usability evaluation, Usability Problem Inspector tool and Remote Usability Evaluation tool are fed into the common database using the Usability Problem Diagnosis tool. This tool uses User Action Framework to support diagnosis of usability problems. These usability problems can be retrieved and manipulated using the Usability Problem Database tool.
6.2.2. Work flow of the Usability Problem Diagnosis tool

The internal working of the Usability Problem Diagnosis tool is shown in Error! Reference source not found.. The user runs the Usability Problem Diagnosis tool and reads the description on diagnosing usability problem records using User Action Framework. The user then selects an option depending on whether the user wants to store the usability problem record in the database or practice. The user then selects the problem source and creates the first three components- Raw data, Effects on user and Diagnosis. The other three components have not been implemented. After the completion of each component the user can view the summary and edit the earlier components.
Figure 6-2 Internal working of the Usability Problem Diagnosis tool
Chapter 7. Formative Usability Evaluation Study

Formative Evaluation is the “evaluation of the interaction design, as it is being developed, for the purpose of improving usability” [Hix and Hartson, 1993]. The usability engineering process is centered on formative evaluation. It is early and continual and considers carefully selected participants [Hix and Hartson, 1993].

There a number of steps we followed while working towards doing formative evaluation. Firstly, we designed “benchmark tasks”, decided on the participants, analyzed the requirements for our laboratory setup, designed a feedback form and conducted evaluation.

7.1. Benchmark tasks

During the Usability Evaluation, the evaluator calls in several representative user participants to perform a set of tasks using the product under evaluation in a confined/field environment. These tasks are called as “Benchmark Tasks”. Benchmark tasks are defined after a lot of deliberation and thinking as these tasks have several characteristics, which should be kept in mind while writing.

Some of the characteristics of the “Benchmark Tasks” [Hix and Hartson, 1993] are as follows:

• The tasks should represent the most common tasks performed by the users of this tool.

• These tasks should be worded in such a way that the user knows clearly where he should start and where he should end.
• The user should not be guided during the evaluations and the tasks should not have words, which the user can see on the screen and predict the next step. Doing this will obscure the usability errors.

• All the tasks together should have exercised all the important functions of this tool.

We prepared a set of three tasks (see in Appendix A), which covered the three main functionalities of our tool. The tasks involved creating usability problem records and editing them. This required the participant to use the User Action Framework to diagnose the causes of the usability problem given. Since our main idea was not to test the user action framework but to test the usability of the tool we decided to give the classification for the problem causes to the user.

One of our benchmark tasks is as follows (the other two are listed in Appendix A):

**Usability Problem Scenario:**

A user of a personal document retrieval system has been deleting numbered documents. The user now wants to reuse the old document numbers, but the system does not allow this.

**Your Task:**

1. Consider that you are a usability engineer presented with the above problem.

2. Your first step is to create a Usability Problem Record as a part of Problem Diagnosis. There are three parts to the Problem record for which you must provide values. The Problem record components are as follows: Raw Data, Effects on user and Diagnosis.

   a. Create the Raw Data component.
b. Create the Effects on user component.

c. Create the Diagnosis component. The possible cause of the problem mentioned above is as follows:

Translation (design helping user know how to do something) > Existence (of a way or of a cognitive affordance to show the way) > Existence of a way > Existence of a feature – functionality and physical affordance.

7.2. Selection of Participants

Formative evaluation requires careful selection of participants. The participants should be knowledgeable in the target system domain. Since the tool will be used by usability evaluators, we decided on selecting students who have had experience in usability or who have taken the Usability Engineering course (CS 5714) under Dr. Rex Hartson. These students have a brief idea about the User Action Framework and usability in general. Following these guidelines we selected five participants for evaluation. These participants matched the intended user class i.e. people in development projects with usability responsibility and who have some usability engineering and usability inspection experience.
7.3. Usability Lab-Setup

7.3.1. Requirements

We had the following requirements for the usability lab-setup:

1. The participant and the evaluator should be seated in two different rooms.

2. During the evaluation, the participants may get nervous if they are aware of the evaluator watching them. Thus the set-up should be such that the participant cannot see the evaluator.

3. An intercom system is needed between the two rooms to facilitate communication between the participant and the evaluator.

4. Since we are not interested in capturing the participant’s behavior during the evaluation, we require a tool, which will capture the participant’s screen and mouse movements.

5. Finally, we require the evaluator to make comments during the evaluation, which are not audible to the participant. These comments will be help in reporting critical incidents, which occurred during the evaluation.
7.3.2. Setup

After knowing the requirements we created a laboratory set-up, which satisfied the above requirements.

The setup consisted of the following modules:

1. VNC (Virtual Network Computing) Setup

VNC allows the Evaluator sitting in room McBryde 102 B to view and manipulate the ‘Desktop’ or ‘Screen’ view of the Participant sitting in room McBryde 102 A. This way, the Evaluator can remotely and non-intrusively observe the Participant’s screen. The Evaluator can also take control of the Participant’s computing environment by manipulating the Participant’s mouse and cursor.

2. Microphone Setup

The Microphone Setup in 102 A allows a dual sound stream to be recorded. One sound input comes from the participant microphone in 102 A and the other from the evaluator microphone in 102 B.

3. Recording using Camtasia

We used Camtasia recorder to record the participant’s actions during the evaluation.

4. Playback

The created file was played with “Real Player” which allowed the Evaluator to scan back and forth through the file. Alternatively, it can be played on Windows Media Player or the Camtasia Player.
5. Communicating with the Participant in 102 A from 102 B

We used the Intercom to communicate one-way with the participant in 102 B. The evaluator can hear the participant in 102 A through the speakers in 102 B. Hence if the evaluator wishes to speak to the participant he can use the Intercom. This can be turned ON by pressing the ‘TALK’ button. If the evaluator wishes to keep the communication on continuously, he can do so by pressing the TALK and LOCK buttons simultaneously.

6. Recording the VNC Listener view using Camtasia.

After starting the VNC viewer, the evaluator will have a view of the Participant’s screen. The evaluator can then start Camtasia and by capturing the VNC Listener Window a screen capture video can be made. With both Microphones on, the video will record both the evaluator and participant’s voices simultaneously.
7.4. Feedback form

We collected feedback regarding the tool and the evaluation procedures. The form was used to understand the usability experience of the participant and the extent of knowledge about the UAF tools. The feedback form is given in Appendix 11.2.

All the participants selected had a good usability background and had worked/used the UAF tools previously. In their feedback, the participants commented on both the positive and negative aspects of the tool. They found it a good tool for noting usability problems and though the task is fairly complex, they could manage it without any prior training. They found some aspects of the tool confusing and at times irritating especially the auto-scrolling feature, which scrolls down to the next effect and also the font color used for headings. The usability problems discovered are discussed in detail in Chapter 7.

7.5. Procedure

Before the evaluation started, we gave a brief explanation of the evaluation setup and the tasks to be performed, to the participant. The participant was then left alone to read the benchmark tasks and begin evaluation. We sat in the opposite room from where we can see the participant but they cannot see us, and started the VNC viewer (more details in section 7.6.2) enabling us to view the participant’s desktop and cursor movements. It also allows us to manipulate their desktop when needed.

During the evaluation, we prodded the participant to think aloud. In case, the participant was stuck for too long at some point, we provided a few pointers in the right direction. We used the Critical Incident method to record usability problems. In this method, the critical incidents of the users is recorded where a critical incident of a task is the most important information associated with task performance [Hartson and Castillo,
1998; Sridharan, 2001]. Thus whenever a critical incident occurred we made notes either on paper or for more complicated incidents we gave a brief description through audio input.

After evaluation, we sorted the notes, reviewed the Camtasia videos made during the evaluation, and entered the usability problems and descriptions in an Excel spreadsheet. After all the problems were entered, we performed the cost-importance analysis explained in section 8.2.
Chapter 8. Results

8.1. Usability Problems Found

After the evaluation, we collected a long list of usability problems. We decided to sort these problems initially by type i.e. feedback, wording etc. and then rank them according to the severity rating [Hartson, Shivakumar, Perez, 2003]. Most of the usability problems were associated with wordings and page layout. The list of problems ranked by severity rating is provided in the next section whereas their descriptions are given in section 8.2.

8.2. Cost-Importance Analysis

The purpose of cost/importance analysis is to help manage design change in accordance with the limited budget and schedule, as explained in Chapter 10 of Hix and Hartson [Hix and Hartson, 1993]. Formative evaluation usually results in identifying a number of usability problems larger than what resources allow to be fixed. We need a way to prioritize them, so that we can first fix the ones that are the most important and the least costly. The importance in this context is importance to fix and can be based on problem severity and also other variables such as marketing inputs, management judgment, etc. Cost-importance analysis is a way to give each problem a priority ratio, a numeric estimation of importance to fix divided by estimated cost to fix. The problems are then sorted and ranked by this metric, favoring the highest importance and lowest cost. [Hix and Hartson, 1993]

During the cost-importance analysis for each problem we:

• Estimated the importance of the problem

• Estimated the cost to fix in person hours.
The results of the cost importance analysis are shown below:

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>UP Description</th>
<th>Importance to fix</th>
<th>Cost to fix (in person hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User does not know where to start with the problem record creation</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Automatic scrolling down of effects page irritated the user</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Images on the usability problem record creation page do not appear as links.</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Imprecise wording for &quot;edit current problem&quot; link</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Improper feedback given to user during diagnosis component modification</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Missing functionality-delete a cause</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>The section headings in UP record page are not noticed because of the green color</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Too much of text on UPD splash page</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>Different fonts on page are confusing</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>Imprecise wording of remove effects button</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>Description of effect types should be given</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>Layout of effects on user page is confusing to the user.</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 1 Cost-Importance ratings for usability problems
8.3. Usability problems and proposed design solutions

We sorted the usability problems, discovered during the lab-based usability evaluation, by type for e.g. feedback, wording etc. We did not use our own tool to diagnose and store usability problems in the database since the backend database functionality is not complete and the interface for viewing the stored usability problems is not developed.

8.3.1. Problems with feedback

When the users try to edit the problem causes in the UPD and the UDB tools, they are not warned about the deletion of the existing causes. A warning message should be displayed to the user warning them that “The existing cause(s) will be deleted and will have to be added again if required”.

8.3.2. Problems with wording

When the users were given the task of editing the recently created problem record, they found the wording of the link “Edit Current Problem” (shown in Figure 8.1), in the Usability Database tool, confusing. They preferred it to be worded as “Edit recent usability problem record” indicating the most recently created problem record.

![Working with Problem record](image)

Figure 8-1 The wordings of the circled link are unclear.
After the users edited the effects on user, they hesitated to click “Enter these values in the current UP record” button, as shown in Figure 8.2. They were unsure whether these effects would be added to the earlier effects and duplicate entries would be created in the database. A better design decision would be to have a button worded as “Save the above changes in effects as part of the UP record”.

Figure 8-2. Incorrect wording of “Enter these values in the current UP record” button confused the user.

The wordings of the “Remove” button on the effects page, as shown in Figure 8.3, should be changed to “Remove selected effects”, reminding the users to select the effects before attempting to remove them from the list.
Figure 8-3. The label of the button "Remove" should be changed to “Remove selected effects”.

8.3.3. Problems with layout/graphic design

The user got irritated while creating the “effects on user” component, shown in Figure 7.4. After adding an effect in a category, the screen scrolls down to the next category of effects. The user felt loss of control during this action and found it difficult to add more than one effect in the same category.
The links to “record the effects on user” and to “identify the problem typing causes” were not immediately obvious to the user, as shown in Figure 8.5. The user did not realize that they were links until they looked for them with the cursor.
The main headings of the components of the usability problem record are written in “green” font, as shown in Figure 8.6. The intention was to attract the attention of the users and make them aware of the different components of the problem record they were creating. During the evaluation, however, we realized that the users ignored the headings in “green” font, failing to understand which component they were working on.
Figure 8-6 The green font is ignored by the user
There is too much of text on the Usability Problem Diagnosis tool splash page, as shown in Figure 8.7, which irritated the users as they were forced to read through the entire text while searching for a link to create problem record.

![Usability Problem Diagnosis Tool (UPD)](image)

Figure 8-7. Too much of text on this page which irritated the user

While trying to create the Problem Diagnosis component and identifying problem causes, the users found too many fonts on the page, which they found confusing and unpleasant to read.

We intended to have a horizontal table, on the top of the problem record creation page, with a cell for each component highlighted using “gray” color on completion of that component as shown in Figure 8.8. However, it confused the users, as they tried to
click on it thinking that it would take them to the respective component. When the users were questioned in this respect, they held the “green font” responsible for giving an impression of a link.

8.3.4. Problems with user planning

The user when provided with the task of creating a usability problem record did not know where to start. Firstly, they were confused about which tool to use - Database or Usability Problem Diagnosis tool. Secondly, after figuring out that they should use the Usability Problem Diagnosis tool, they could not find a button/link for creating the problem record. The solution to this would be to change the wording of the “Continue” button, shown in Figure 8.9, on the Usability Problem Diagnosis tool splash page, to “Continue to create the problem record”.

Figure 8-9. The wording of “Continue” button must be changed to “Continue to create problem record”
8.3.5. Problems with system functionality

The users expect the system to allow them to delete a cause. Currently the system does not support the functionality in the way the user desires. When the user tried to edit problem causes, the previous causes are deleted from the database.

8.4. Iterative Redesign

After evaluating the results of the formative usability evaluation we selected a few important problems to fix.

8.4.1. Layout of the “effects on user” page

The user was distracted by the layout of the “effects on user” page. There was no clear grouping of related functions. The earlier design of the page is shown Figure 8.4

After the evaluation, we changed the design of the page to reflect the grouping of related functions and to place the buttons in a more organized way, to avoid distracting the user. The new design of the page is shown in Figure 8-10 below.
8.4.2. Warning message to the user on editing causes

When the user tries to edit the problem causes, the earlier causes are deleted. Previously, there was no warning message given to the user and they were not sure what proceeds when you click on the “Edit causes and types of problem” button. After the evaluation, a warning message is displayed to the user when they try to edit the problem causes, as shown in Figure 8.11.
Figure 8-11 A warning message is displayed to the user when they click on the “Edit effects on the user” button.

8.4.3. Allowing the user to add multiple effects at the same time

While entering the “effects on user”, the user was irritated that they could not add or remove multiple effects at the same time. As shown in Figure 8-12, the user can now add or remove multiple effects.
These were some of the usability problems encountered during the formative evaluation process that have been fixed in the tool, as part of the “Iterative Redesign” process. There were other serious problems too, as mentioned in Table 1, for which possible design solutions have been suggested in section 8.3.
Chapter 9. Conclusions

During formative evaluation, a number of usability problems are exposed which are reported to the developer to analyze and fix. The developer not being a part of the evaluation, as this might deter the participants from the reporting all the errors, could result in loss of understanding the usability problems revealed. Thus the developer is highly dependent on the evaluator to make clear and precise notes. Our tool addresses this problem by allowing the evaluator to diagnose and save the usability problems in a database.

Also, the way we designed the laboratory setup for formative evaluation has helped us in extracting maximum usability problems from the participants. The cost/importance analysis helped in ordering the usability problems by priority, thus enabling more resources to be diverted to fixing the most important problems.

Our research objectives comprised of three main goals – First, to develop a tool to diagnose and save usability problems in a database and provide a complete, unambiguous report of the problem. Second, to perform formative evaluation of the UPD tool and uncover usability problems associated with the tool itself. Third, to fix a few selective usability problems.

The first part focused on doing prototyping, design walkthroughs and iterative redesign. We succeeded in developing Usability Problem Diagnosis tool, which allows the practitioner to furnish values for the first three components – Raw Data component, Effects on user component, and the Diagnosis component of the usability problem record. The three other components – Cost/Importance component, Implementer component, and the Management component have not been implemented yet.
The second part focused on performing lab-based usability evaluation of the UPD tool. We sorted the list of usability problems discovered, by type and performed cost-importance analysis, analyzing the importance and cost of fixing the problems.

The third part focused on selecting a few important problems to fix. These problems were selected such that they were important and not very costly to fix. Thus, performing a complete iteration of the usability engineering life cycle, which resulted in a revised and enhanced fault-tolerant version of the Usability Problem Diagnosis tool.
Chapter 10. Future Work

The Usability Problem Diagnosis tool and Usability Database tool form a basis upon which several other useful features can be extended.

10.1. Major future work

There is one major, “thesis-worthy” project that can follow on to the work reported in this thesis: A complete summative study to test the tool effectiveness in diagnosing and documenting usability problems is needed. Such a study would be performed in a usability-testing laboratory, using benchmark tasks to performance usability problem analysis – extraction and diagnoses. Results, and claims about tool performance and effectiveness would be based on a comparison of statistically significant differences in tool user performance using this Usability Problem Diagnosis tool and some other tool, or manual method, with similar functionality.

10.2. Minor future work

In addition to the “thesis-worthy” major future work described above, there are numerous minor ways in which my work on the tool can be extended. First, currently the Usability Problem Diagnosis tool does not implement all the components of the usability problem record. The three other components - Cost/Importance, Implementer and Management component are yet to be implemented, on the completion of which, the tool will provide a complete framework for handling usability problems.

In addition, the tool can be extended to extract problems noted by other tools such as Usability Problem Inspector, Remote Usability Evaluation tool and Ideal. Another future feature that asks the tool users if they want to add any keyword tags to the classification nodes can help to more completely characterize the usability problems. For example, a
tool user may add the “affordance” keyword to specific classification nodes. Later, a user can search all problems related to “affordance”, which will display all those problems that have at least one of the above classification nodes in their diagnosis path. Also a feature for letting the users save incomplete usability problem records to complete them in future can be incorporated providing great deal of flexibility. Finally, The Usability Database tool can be extended to provide a feature for visualizing the usability problems collected, which will help detect the major flaws in the interface design.
Chapter 11. References


Chapter 12. Appendices

Appendix A. Benchmark Tasks

1. Usability Problem Scenario:

A user of a personal document retrieval system has been deleting numbered documents. The user now wants to reuse the old document numbers, but the system does not allow this.

Your Task:

1. Consider that you are a usability engineer presented with the above problem.

2. Your first step is to create a Usability Problem Record as a part of Problem Diagnosis. There are three parts to the Problem record for which you must provide values. The Problem record components are as follows: Raw Data, Effects on user and Diagnosis.
   a. Create the Raw Data component.
   b. Create the Effects on user component.
   c. Create the Diagnosis component. The possible cause of the problem mentioned above is as follows:
      Translation (design helping user know how to do something) >
      Existence (of a way or of a cognitive affordance to show the way) >
      Existence of a way > Existence of a feature – functionality and physical affordance.
2. **Usability Problem Scenario:**

A person using a Windows program for FTP file transfer wants to rename a file. S/He selects the file name and tries to type over it (as s/he does in the Explorer program on her/his PC), but this does not work. Eventually s/he figures out that you have to select the filename and then click the Rename command from a button bar. That leads to a small dialogue box with the filename in a text field where s/he can edit the name and click OK. When s/he clicks OK, the system puts the new filename back into the list. S/He completes the task wondering why the system did not provide a way to do the task directly.

**Your Task:**

1. Consider that you are a usability engineer presented with the above problem.

2. Your next step is to create a Usability Problem Record as a part of Problem Diagnosis. There are three parts to the Problem record for which you must provide values. The Problem record components are as follows: Raw Data, Effects on user and Diagnosis.

   a. Create the Raw Data component.

   b. Create the Effects on user component.

   c. Create the Diagnosis component. **This problem can have two possible causes.** The causes are given below:

   - *Translation > Task Structure & Control > Directness of interaction*

   - *Translation > Task Structure & Control > Consistency and compliance of task structure.*
3. **Scenario:** You have just created a Usability Problem Record for the Benchmark task 2. Later you realize that the second cause you have selected is not appropriate for this problem and you wish to modify the Diagnosis component of the Problem Record (*The problems are stored in the Database.*)

**Your Task:**

1. View the created problem record.
2. Modify the Diagnosis component by reclassifying using only the first path mentioned in benchmark task 2.
3. View the modified record.
4. Change the Effects on user component and modify the effects.
5. View the modified record.
Appendix B. Equipment Manual

B.1. Dual Monitor Setup

<table>
<thead>
<tr>
<th>Enabling Dual-Monitor display</th>
<th>Disabling Dual-Monitor Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start&gt; Settings&gt; Control Panel&gt; Display&gt; Settings&gt; Advanced&gt; Dual-Head</td>
<td>Start&gt; Settings&gt; Control Panel&gt; Display&gt; Settings&gt; Advanced&gt; Dual-Head</td>
</tr>
<tr>
<td>Click on <em>Dual Monitor MultiDisplay</em></td>
<td>Click on <em>Dual Monitor Disabled</em></td>
</tr>
</tbody>
</table>

---

**Plug and Play Monitor and Matrox Millennium G450 Dual Head - E...**

- **General**
- **Adapter**
- **Monitor**
- **Troubleshooting**
- **Color Management**
- **Information**
- **Options**
- **DualHead**
- **Monitor Settings**

- **DualHead disabled**
- **DualHead Clone, VideoMax, or Zoom**
- **DualHead Multi-Display**

**Mode description:**

Use "DualHead Multi-Display" mode to add a secondary display to your Windows desktop. With this feature, Windows uses your Matrox graphics card as if it were two separate graphics cards.

Your secondary display device can be a traditional computer monitor (CRT), a flat panel monitor, a TV or a VCR.

- **Use DualHead Clone**
- **Use DualHead VideoMax**

**Clone Options...**

**VideoMax Options...**

**Zoom Options...**

**TV standard**

- **NTSC (North America)**
- **PAL (Europe)**

**Max secondary resolution**

1600 x 1200 @ 75 Hz
B.2. VNC (Virtual Network Computing) Setup

VNC allows the Evaluator sitting in room McBryde 102 B to view and manipulate the ‘Desktop’ or ‘Screen’ view of the Participant sitting in room McBryde 102 A. This way, the Evaluator can remotely and non-intrusively observe the Participant’s screen. The Evaluator can also take control of the Participant’s computing environment by manipulating the Participant’s mouse and cursor.

1. Run WinVNC(App Mode) in the machine in 102 A. (This is the server)

   Start>Programs>VNC>Run WinVNC(App Mode)

   This will start the WinVNC server and produce an icon in the system tray

![WinVNC Server Icon]

2. Run VNCViewer in Listen mode in 102 B. (This is the viewer)

   Start>Programs>VNC>Run VNCViewer(Listen Mode)

   This will start the WinVNC Viewer and produce an icon in the system tray

![VNCViewer Icon]
The order in which you start the Viewer and Server does not matter.

3. In the machine in 102 A: Start>Run>**Type ”command”**. This will open the Command Prompt. Now Type “ipconfig”. Note down the IP Address. Then type “exit” to close the Command Prompt.

4. Now Double-click on the VNCViewer icon in the system tray of the machine in 102B and enter the IP Address of the machine in 102 A.
5. The VNCViewer now asks you the password of the machine in 102 A so that it can connect to it. The password is currently set to “********”.

Alternatively, you can also directly grant access to the Evaluator machine in 102 B from the machine in 102 A. For this, the VNC Listener should already be running in 102 B. Note down the IP Address of the machine in 102 B in a similar way as above. Then right-click on the VNC icon in the system tray of the machine in 102 A and select “Add a New Client”. Enter the IP address of the Evaluator machine (102 B machine) here.
B.3. Microphone Setup

The Microphone Setup in 102 A allows a dual sound stream to be recorded. One sound input comes from the participant microphone in 102 A and the other from the evaluator microphone in 102 B. In the event that either of the microphones is not working, follow the instructions given below:

1. The Microphone Preamplifier (dbx 760x) in 102 B should be connected to the power supply.

   *The Microphone Preamplifier has two sets of controls. The controls on the left are for the evaluator and the duplicate set of controls on the right are for the participant.*

2. The evaluator’s microphone in 102 B should be connected to the balanced input on the left in the Microphone Preamplifier.

3. The participant’s microphone in 102 A should be connected to the balanced input on the right in the Microphone Preamplifier.

4. The unbalanced outputs from the evaluator and the participant should be connected to a connector which is then connected to the microphone input on the evaluator’s machine in 102 B.

5. Adjust the volume levels on the Microphone Preamplifier.
B.4. Recording using Camtasia

1. Start> Programs> Camtasia> Camtasia Recorder

2. Press the record button.

3. When done, press the square STOP button.

4. Save the file. The file will be saved with a “.avi” extension.
B.5. Playback

The created file can be best played with “Real Player” which allows the Evaluator to scan back and forth through the file. Alternatively, it can be played on Windows Media Player or the Camtasia Player.

B.6. Communicating with the Participant in 102 A from 102 B

Use the Intercom to communicate one-way with the participant in 102 B. The evaluator can hear the participant in 102 A through the speakers in 102 B. Hence if the evaluator wishes to speak to the participant he can use the Intercom. This can be turned on by pressing the ‘TALK’ button. If the evaluator wishes to keep the communication on continuously, he can do so by pressing the TALK and LOCK buttons simultaneously.

B.7. Recording the VNC Listener view using Camtasia.

After starting the VNC viewer, the evaluator will have a view of the Participant’s screen. He can then start Camtasia and by capturing the VNC Listener Window a screen capture video can be made. With both Microphones on, the video will record both the evaluator’s and participant’s voices simultaneously.
Appendix C. Feedback form

How would you rate your usability knowledge, on a scale of 1-5?

Scale:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td>Good</td>
<td>Excellent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How would you rate your knowledge of the User Action Framework (UAF) on a scale of 1-5?

Scale:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td>Good</td>
<td>Excellent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Have you taken the ‘Usability Engineering’ course (CS5714)?

☐ Yes ☐ No

Have you ever participated in a usability evaluation of software?

☐ Yes ☐ No
Did you have any problems with the Usability Problem Diagnosis tool?

Do you have any positive/negative comments about the Usability Problem Diagnosis tool?

Were you comfortable with the experimental setup? Did you have any problems?

Suggestions?
VITA

Reenal Mahajan

EDUCATION

M.S Computer Science (Expected graduation, May 2003) GPA: 3.81/4.0
Virginia Polytechnic Institute and State University, Blacksburg, VA

B.E Computer Engineering (May 2001) GPA: 3.85/4.0
Fr. Conceicao Rodrigues College of Engineering, University of Mumbai, India

WORK EXPERIENCE

Graduate Research (http://hemlock.cs.vt.edu/upd-demo) (Jan 2002 - present)
* Designed and implemented a web-based Usability Problem Diagnosis tool to diagnose usability problems, following the Usability Engineering life cycle-Design, Implementation, Formative Usability Evaluation, and Iterative Redesign.
* Conducted Formative Usability Evaluation, analyzed results, and performed cost/importance analysis.
* Involves ASP, JavaScript, VbScript, SQL, Usability Engineering principles, IIS Server and MS Access.

Graduate Teaching Assistant, Virginia Tech

* CSCW (Computer Supported Co-operative work) (Jan 2003 – present): Assisted students in conducting usability evaluations of collaborative systems, and assessed paper presentations.
Part-time Java Programmer, Virginia Tech (Feb 2002 - Sept 2002)

* Designed and implemented a tool to optimize multidimensional functions in Java.

* Involved Java, Java Swing, and JNI.

Intern, Mahindra Consulting Ltd, Mumbai, India (July 2000 - May 2001)

* Designed and implemented a web-based tool to automate the recruitment process.

* Involved ASP, JavaScript, ODBC, and IIS.

RELATED PROJECTS


* Developed XML schemas, XML parser and converter for converting Digital Library scenarios to a textual representation of collaboration diagrams useful as input to an automated user-interface generation tool.

* Developed a web-based eCommerce shopping system. Involved Java, JSP, Servlets, JavaScript, CSS, and Tomcat.

* Developed an Inventory Management System using Java, JSP, Servlets, CSS, JDBC, and Tomcat.

* Developed a tool to Visualize Multidimensional Functions. Involved complex mathematics, Java and Java Swing.

* Designed and Implemented an efficient and reliable Data Link Protocol in C. Modified and implemented Sliding Window Protocol.
* Designed and developed a wireless Smart-Cart interface for Kroger. Conducted Formative and Summative usability evaluation. Involved Visual Basic and Usability Engineering principles.

* Developed an eCommerce application in Java RMI and Client-Server Networking.

**COMPUTER SKILLS**

**Languages:** C/C++, Java, ASP, UML, J2EE, RMI, JNI, Java Beans, JSP, Servlets, JavaScript, CORBA, EJB, VbScript, HTML, DHTML, CSS, XML, XSL, XHTML, CSS, Visual Basic, Visual C++, Prolog, Scheme, SQL, COBOL, Web Services, Pascal.

**Software tools:** Rational Suite, Macromedia Dreamweaver, Internet Information Server (IIS), Tomcat, Flash, Adobe Photoshop, VISIO, Jbuilder, MS Office.

**Operating Systems:** Windows 9X/ 2000, Unix, Linux

**Databases:** MS Access, ORACLE, SQL Server, ODBC, JDBC

**HONORS**

Received the "Ratan Tata Scholarship" for outstanding academic achievement in the year 1998-1999.