ABSTRACT

Title of Dissertation: THE DESIGN AND EMPIRICAL STUDIES OF PERSPECTIVE-BASED USABILITY INSPECTION

Zhijun Zhang, Doctor of Philosophy, 1999

Dissertation directed by: Professor Victor R. Basili
Department of Computer Science

Inspection is a fundamental means of achieving software usability. Past research showed that the current usability inspection techniques were rather ineffective. This dissertation developed perspective-based usability inspection, which divides the large variety of usability issues along different perspectives and focuses each inspection session on one perspective. It is hypothesized that 1) with focused attention and an appropriate inspection procedure for each perspective, each inspection session would detect significantly more usability anomalies covered by that perspective, and that 2) the combined results of different perspectives would uncover more anomalies than the combined results of the same number of inspectors using the same general inspection technique. Three experiments with human subjects were conducted to study the feasibility and

effectiveness of perspective-based usability inspection by comparing it against a widely used technique, heuristic evaluation. The user interfaces being inspected were a Web-based data collection form (experiment I) and a commercial Web site (experiments II and III). The subjects included both professionals (experiments I and III) and students (experiment II), who worked either individually or in 2-person teams. The experimental results showed that on average each inspector using a perspective-based technique detected not only more anomalies related to the assigned perspective, but also more anomalies overall. For the combined effectiveness of multiple inspectors, perspective-based inspection showed a significant improvement over heuristic evaluation (30% for 3 inspectors in experiment I, 90% for 4 inspectors in experiment II, and 45% for 4 inspectors in experiment III). Besides the quantitative results, qualitative information was collected and discussed, along with results from a survey of usability practitioners. This work outlines a research agenda with some initial results, and provides implications for usability practice.

THE DESIGN AND EMPIRICAL STUDIES OF PERSPECTIVE-BASED USABILITY INSPECTION

by

Zhijun Zhang

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
1999

Advisory Committee:

Professor Victor R. Basili, Chairman/Advisor
Professor Ben Shneiderman, Co-Chair/Co-Advisor
Professor Marvin Zelkowitz
Assistant Professor Douglas W. Oard
Assistant Professor Ben Bederson

Dedication

To my parents, and to my wife.

Acknowledgements

I would like to thank all of the 110 participants of the experiments for their time and effort.

I am most grateful to my advisor, Vic Basili, and co-advisor, Ben Shneiderman, for supporting and guiding my dissertation research, and arranging for the experiments to happen. I thank all the members of the Empirical Software Engineering Group at Maryland for their help and feedback throughout the process.

I thank the members of my committee for their careful reading of my work, especially Dr. Doug Oard for his valuable and extensive comments on an earlier version. I also thank Dr. Fred Davis for serving on my proposal committee, and commenting on an earlier version of the dissertation. I would also like to thank Jeff Carver for editing the dissertation.

Many people have helped in different ways in the three experiments. Thanks go to Elizabeth Nicoles, Barbara Sedivi, Nancy VanDerveer,

Ellen Soper, and Al Paez of the US Census Bureau; Dr. Dagobert Soergel, Chris North, Chip Denman, Forrest Shull, Roseanne Tesoriero, Daniil Iakimovitch, Eser Kandogan, and Egemen Tanin at University of Maryland; and Pawan Vora of US WEST. I also thank the survey participants: Bryan Anderson, Margaret Brown, Yee-Yin Choong, and Meredith McQuilkin of GE Information Services, Eileen Schwab of Ameritech, Sherrill Packebush and Barbee Teasley of SBC Communications, and Arnie Lund of US WEST.

Finally, I thank my wife Wei Ding for her support, love, and patience.

Table of Contents

Section	Page
List of Tables	**ix**
List of Figures	**xi**
1 Introduction	**1**
1.1 Problem Statement	3
1.2 Research Questions	4
1.3 Motivation	6
1.3.1 Usability and Usability Inspection	6
1.3.2 Perspective-based Technique	9
1.3.3 Empirical Study	12
1.4 Requirements for the New Technique	12
1.5 Definitions	13
2 Review of the Literature	**16**
2.1 Models for Human-Computer Interaction	16
2.2 Techniques for Usability Inspection and Software Inspection	19
2.2.1 A Framework of Inspection Techniques	20

2.2.2 Existing Inspection Techniques 23
2.3 Empirical Studies of Inspection Techniques 27
2.3.1 Internal Attributes 28
2.3.2 External Attributes 36
2.3.3 Organizational Structure 41
2.4 The Goal/Question/Metric Method 41
2.5 Summary 42

3 Perspective-based Usability Inspection 46

3.1 Overview 46
3.2 Models to be Used in the GQM Analysis 49
3.2.1 The Three-Perspective Usability Model 49
3.2.2 A Process Model of Human-Computer Interaction 52
3.3 Using GQM to Generate the Generic Inspection Questions 55
3.3.1 Characterizing the User Interface 55
3.3.2 Characterizing the Users 57
3.3.3 Characterizing the Tasks 58
3.3.4 Characterizing Users' Working Environment 59
3.3.5 Characterizing the Usability Perspectives 60
3.4 Inspection Scenarios 72
3.4.1 Inspection Procedure for Novice Use 73
3.4.2 Inspection Procedure for Expert Use 74
3.4.3 Inspection Procedure for Error Handling 74
3.5 Generating Inspection Questions for a Specific Project 75

4 Empirical Studies 84

4.1 Goal 84
4.2 Research Hypotheses 85
4.3 Overview of the Empirical Studies 86
4.4 Experiment I 88
4.4.1 Method 88
4.4.2 Results and Discussion 101
4.4.3 Threats to Validity 113
4.4.4 Summary and Guidance for Further Studies 116
4.5 Experiment II 118
4.5.1 Method 118
4.5.2 Results and Discussion 121
4.5.3 Threats to Validity 125
4.5.4 Summary and Guidance for Further Studies 126
4.6 Experiment III 127
4.6.1 Method 127
4.6.2 Results and Discussion 130
4.6.3 Threats to Validity 134
4.6.4 Summary and Guidance for Further Studies 134
4.7 Comparison of the Experiment Results 135
4.7.1 Feasibility of the Technique 135
4.7.2 Effectiveness of the Techniques 138
4.7.3 Team Size and Inspector Expertise 140
4.8 A Survey of Usability Practitioners 142
4.8.1 Method 142
4.8.2 Results and Discussion 142

5 Summary of Dissertation Work **146**

5.1 Contributions 146

5.1.1 Research Agenda 147

5.1.2 Experiment Results 147

5.1.3 Implications for Practitioners 149

5.2 Limitations and Open Issues 150

5.2.1 Study Design 150

5.2.2 Perspectives 151

5.2.3 Application Domain 151

5.2.4 Artifact Format 152

5.3 Future Research 152

A Generic Usability Questions for Each Perspective **154**

A.1 Novice Use 154

A.2 Expert Use 157

A.3 Error Handling 159

B Inspection Techniques Used in Experiment I **161**

B.1 Usability Heuristics 161

B.2 Inspection Procedure for Novice Use 163

B.3 Inspection Procedure for Expert Use 165

B.4 Inspection Procedure for Error Handling 166

C Inspection Techniques Used in Experiments II and III **169**

C.1 Heuristics Evaluation 169

C.2 Novice Use 172

C.3 Expert Use 176

List of Tables

Number		Page
1.1	The differences between the three usability evaluation methods	9
2.1	Results from the (Desurvire 1992) study: number of anomalies detected	31
2.2	Comparison of anomaly detection by perspectives (Desurvire 1994) and other techniques (Desurvire 1992)	33
2.3	Characteristics of usability inspection techniques	44
2.4	Characteristics of software inspection techniques	45
4.1	Layout of experiment I: number of subjects in each treatment	90
4.2	Percentage of anomalies detected by the 7 pilot subjects	91
4.3	Detected usability anomalies by severity (experiment I)	99
4.4	Detected usability anomalies by category (experiment I)	100
4.5	Effect of independent variables on overall detection effectiveness in experiment I (p-values from ANOVA)	101
4.6	Percentage of anomalies found for each interface-order situation in experiment I	103

4.7 The effect of technique on the detection of anomalies by category in experiment I (p-values from ANOVA) 105
4.8 Comparison of different types of anomalies found in experiment I 106
4.9 Subjective ratings of the techniques (experiment I) 107
4.10 Correlation between experience and inspection performance for the 24 subjects in experiment I 107
4.11 Aggregated anomalies found by 3 inspectors (all possible aggregations, standard deviations are in parentheses) 108
4.12 Permutation tests for all possible 12-person teams 109
4.13 Major anomalies uniquely detected by only one technique 111
4.14 The number of data points collected in experiment II 119
4.15 Means and standard deviations for individual inspectors in experiment II 122
4.16 Means and standard deviations for 2-person teams in experiment II 122
4.17 ANOVA and t-tests for the 2-person teams in experiment II . . . 123
4.18 Anomaly detection effectiveness (mean and standard deviation) in experiment III 130
4.19 ANOVA and t-tests for anomaly detection effectiveness in experiment III 131
4.20 The subjective ratings in experiment III 133
4.21 The reported process conformance in experiment III 133
4.22 Different settings of the experiments 136
4.23 Comparison of the experiment results 137
4.24 The number of anomalies detected by inspectors using the same technique 141

List of Figures

Number		Page
2.1	Simplified Model Human Processor	17
2.2	The characteristics and effectiveness of usability inspections	23
2.3	The GQM model	42
3.1	The process for deriving the inspection procedures	48
3.2	A state diagram of the three perspectives	51
3.3	The three perspectives and user needs	52
3.4	A process model of human-computer interaction	83
4.1	Overlapping of anomalies detected by different perspectives	110
4.2	A Spotfire view showing the amount of time spent inspecting interface B and the job category of the subjects (1=programmers, 2=domain experts, 3=technical researchers, 4=cognitive researchers)	112

Chapter 1

Introduction

Usability is an important attribute of interactive computer systems. ISO 9126 [44] defined software quality as consisting of *functionality, usability, reliability, efficiency, maintainability, and portability.*

Software development organizations strive to make their products usable through a set of activities that collectively are called *usability engineering* [37]. *Usability inspection* is one such activity that is well suited as part of an iterative design process where it can be combined with other usability evaluation methods such as *user testing.* Research results have shown that user testing and usability inspection detect largely distinctive usability anomalies (often called usability problems) [18]. A typical strategy is to apply a usability inspection first to detect some of the problems, revise the interface, then conduct user testing [39].

During usability inspections or expert reviews [57], human inspectors detect usability anomalies in a user interface design so that they can be corrected to improve usability. Inspectors detect usability anomalies by examining the interface to see whether it supports the user's tasks and whether it complies with usability principles, using their knowledge and expertise in the task domain and

the user interface domain. The inspection of an interface is often conducted by multiple inspectors working individually or as a team.

However, studies showed that the current inspection techniques are not effective in detecting usability anomalies, especially when the inspectors are non-experts [11, 12, 22, 36]. To deal with this problem, some researchers suggested the use of multiple inspectors and inspectors with high expertise in both usability and the application domain [36].

Therefore, on the one hand, a successful usability inspection using the current techniques requires the participation of multiple expert usability inspectors. On the other hand, most project teams do not have multiple usability experts available. We need a usability inspection technique that is more effective than the existing ones when applied by non-expert inspectors.

This dissertation suggests that during usability inspection, each inspector is assigned one of several specific viewpoints or perspectives. Each perspective covers a subset of usability issues. The combination of all perspectives provides full coverage of the usability issues. It is hypothesized that by taking specific perspectives, non-expert inspectors will detect a significantly larger number of usability anomalies covered by their assigned perspective than they would detect if they all conduct the inspections in a general way, and that the combined inspection effectiveness of multiple inspectors would improve significantly over a general technique.

Three empirical studies have been conducted to investigate the effectiveness of the perspective-based usability inspection technique. The results from these empirical studies support the above hypotheses. The qualitative information gathered from these studies and from interviewing some usability practitioners

provides some insight in understanding the usability inspection process.

The following sections provide more contextual information, the problem statement, research questions, the motivations, and requirements for the new technique. Some definitions of terms are given at the end of this introduction chapter. In subsequent chapters, a literature review, a description of the proposed technique, and a report of the empirical studies are provided. A summary of the contributions, limitations, open issues, and future work is in the final chapter.

1.1 Problem Statement

Currently the most widely used usability inspection technique is heuristic evaluation, developed by Nielsen et al. [40]. Based on 19 studies, Nielsen [36] reported that on average each inspector could detect around 20%, 40%, or 60% of the usability anomalies depending on whether they were novices (with no expertise in either usability or the application domain), single-experts (with expertise in usability principles but without expertise in the specific application domain), or double-experts (with expertise in both usability and the application domain). Desurvire [12] reported that on average each non-expert inspector (inspector who had less than 3 years of usability work experience) detected about 8% of the usability anomalies when using heuristic evaluation. These results suggest that multiple experts in both the usability domain and the application domain be used for heuristic evaluation. Since most projects do not have multiple usability experts available, inspections using heuristic evaluation are not likely to be effective.

Thus, the user interface design practice needs a more effective usability inspection technique for non-expert inspectors. Furthermore, empirical studies need to be carried out to understand the effectiveness of such techniques and the factors that may influence their effectiveness.

1.2 Research Questions

This dissertation first addressed the following research question:

> How can a new usability inspection technique be built that is more effective than the existing techniques when used by non-expert inspectors?

This was done by:

- Analyzing the reasons for the ineffectiveness of current usability inspection techniques.
- Applying the perspective-based reading technique to usability inspection.
- Using the Goal/Question/Metric framework to derive the usability questions for each perspective.

Then empirical studies were conducted to address the following questions about the new technique:

- **Feasibility:** Is the proposed perspective-based usability inspection technique feasible?
- **Effectiveness:** How effective is the proposed technique?

1. Does assigning specific responsibilities have a significant impact on the effectiveness of anomaly detection?
2. Do inspectors using different perspectives detect different subsets of usability anomalies?
3. Given the same number of inspectors, other things being equal, which way is more effective in doing usability inspection: assigning each of them the same general role, or assigning each of them one specific role?

- **Scope:** Under what conditions is the proposed technique effective?
- **Applicability:** Can practitioners apply the proposed technique in practice?

To answer these question, three experiments have been conducted to compare inspection results of the perspective-based techniques against heuristic evaluation, which gives each inspector the same general responsibility. The results showed that the perspective-based technique was more effective than heuristic evaluation at both the individual level and for the combined results of multiple inspectors.

These three experiments have been conducted with different types of subjects and different team sizes, and with different applications. The combination of these studies provides some information for answering the "scope" question. The "applicability" question has been studied by surveying usability practitioners.

1.3 Motivation

This section motivates the issues being studied and the research methods. It describes the importance of usability and usability inspection, gives the reasons for using the perspective-based approach, and explains why empirical study is so important in developing new techniques.

1.3.1 Usability and Usability Inspection

Usability of interactive computer systems has become more important than before because of changes in the computer world including:

Application domain Computers were originally used for mathematical calculations. They worked in a batch-processing manner, with primitive user interfaces. But now computer applications cover a wide range of areas, including read, write, communicate, organize, administer, look for information, entertain [29]. Computer systems are often designed and built to act as assistants, aids, and "power tools" to users to help them with various tasks. The interaction between users and computers becomes more complex than before.

User population With the rapid expansion of computer applications, the population of computer users has been growing dramatically. Computers are being used *directly* by more and more people with different levels of computer literacy and background, including novice users, users with physical disabilities, users with different languages and culture, etc. More and more systems are being used without formal training.

From the point of view of development, a study showed that on average 48% of the software code is for user interface [34]. Efforts need to be allocated to ensure that these user interfaces are usable. The collection of usability-related activities carried out during software development is called usability engineering. It includes user analysis, competitive analysis, setting usability goals, parallel design, participatory design, guidelines, iterative design with prototyping and usability evaluation, etc.

In current practice, people tend to simply use guidelines or standards in developing user interfaces. However, studies have shown that it is not effective to simply ask developers to follow a set of provided usability guidelines and standards. One study [59] found that many developers treat these guidelines and standards as extraneous materials beyond their task and are quite impatient with reading them. They depend heavily on the pictorial examples, often neglecting the accompanying text. They have significant difficulty in interpreting the guidelines, even after they have followed them. In another study [60] where a company required all user interface designs to follow the corporate usability standard, the three products studied broke between 7 and 12 of the 22 mandatory rules in the standard.

Given the fact that user interface developers often focus more on technical issues than on usability issues, and the limitations of our ability to predict human performance and preferences, the practical approach is to take the requirements, design a prototype, evaluate it, modify the design, and repeat this cycle until the design is acceptable [50].

Usability evaluations are a key activity in this development process of user interfaces. Based on the way they are conducted, usability evaluations can be

classified into the following three methods (Table 1.1) [21]:

- *Testing*: representative users work on typical tasks using the system (or the prototype) and the evaluators observe the users to see how the user interface supports the users in their tasks. Testing can include quantitative measures such as task completion time and error rate, as well as qualitative information such as the kind of problems users have. Different usability testing techniques have been developed, with different protocols between the evaluator and the test user, or different protocols among test users when there are multiple users working together.

- *Inspection*: usability specialists – and sometimes software developers, users and other professionals – examine usability-related aspects of a user interface. They look for anomalies from the point of view of the defined users, think of different situations, and check for consistency. Various inspection techniques will be discussed later on in this dissertation.

- *Inquiry*: usability evaluators obtain information about users' subjective opinions, practical needs, as well as their understanding of the system by interviews, observations (not for the purpose of usability testing), or questionnaires. This is for collecting qualitative or subjective information. Different techniques are used for collecting different information under different situations.

One difference between testing and inspection is that in testing, users only care about finishing the assigned tasks. In inspection, inspectors care about whether the defined users will be able to finish the task, whether the different situations and different users inputs are all handled well, whether the user

Table 1.1: The differences between the three usability evaluation methods

Method	User Involved?	Role of Usability Specialists
Testing	Yes	Observe users using the system, collect and analyze data
Inspection	No	Review the user interface and try it out to find anomalies
Inquiry	Yes	Communicate with users to gain insights into usability issues

interface is consistent from screen to screen, etc.

Among these three methods, usability inspection is often the most suitable one to be used during the iterative design of user interfaces. Different techniques have been developed as to how the inspection is going to be conducted, who will participate, whether a meeting is involved, whether task scenarios are used during the inspection, etc. The details of these techniques will be described in Section 2.2.2. As mentioned earlier, studies showed that the current usability inspection techniques are not effective. Better techniques are in demand.

1.3.2 Perspective-based Technique

I started working on software usability by participating in the definition of an anomaly taxonomy for user interfaces [2]. A usability inspection technique was developed based on the taxonomy by providing check items under each anomaly category. This checklist was used to conduct a usability inspection of the file

manager and the help facilities of OS/2 Warp[1]. In this usability inspection, the inspector:

1. defined the task cases;

2. inspected the semantic aspects of the system without going through the task cases, checking the items in the checklist that related to semantics;

3. inspected the syntactic aspects of the system by going through the task cases and checking the checklist items that related to syntax.

The inspector's experience in doing this inspection showed that:

- It is hard to go through a long checklist (the list had more than 50 items) for each task or each screen during the inspection. A better procedure was needed.

- The inspection material should fit the system. It was distracting and ineffective to have a significant number of items that did not apply to the current system.

The major problem to solve is the difficulty for an inspector to handle all different usability issues at the same time. Inspirations for a possible solution came from several studies on reading techniques (for the purpose of software inspection) carried out by NASA SEL [2] and ISERN [3] [3] [48]. These studies showed

[1] OS/2 Warp is a trademark of IBM.

[2] NASA Software Engineering Laboratory is a research consortium among NASA Goddard Space Flight Center, Computer Science Corporation, and University of Maryland.

[3] ISERN is the International Software Engineering Research Network whose goal is to support experimental research and the replication of experiments.

promising results when inspectors were assigned specific defect types or reading perspectives. The perspective-based technique was adopted in developing a new usability inspection technique.

I split the original checklist into usability issues under three perspectives: novice use, expert use, and error handling (see Chapter 3 for details). Two case studies were conducted at US WEST[4], one on an on-line expense report system and another on a Web-based information system. For the latter, usability testing data (with 16 subjects plus 2 pilot subjects) were available for comparison. The inspection uncovered 30 usability anomalies while the usability testing reported 18. Of these 18 anomalies, 12 were identified by the inspection. The reasons why some usability anomalies were not detected by the inspection include:

- The inspector did not choose the same task steps as the test subjects did.
- The inspector failed to predict the users' problem in understanding certain terminology.
- Usability questions in the inspection material may have been too general to help the inspector detect certain anomalies.

The case study produced both promising results and some lessons that have helped in improving the inspection procedures, so that the chances for inspectors to find the usability anomalies are maximized. The improved technique is subject to more formal empirical studies.

[4]The studies were conducted when I was doing summer internship at the Usability Engineering Group of US WEST Communications.

1.3.3 Empirical Study

Software engineering is a laboratory science [1]. Fenton et al. [17] suggest that rigorous experimentation is needed to evaluate new technologies and their effects on organizations. Specifically we need to experiment with techniques to see how and when they really work, to understand their limits, and to understand how to improve them. This is partly due to the "people" factors in software development, i.e., technologies are used by people, either individually or cooperatively. People factors are very difficult to study analytically because we have no universally applicable laws or theories concerning human behavior [54].

Concepts such as inspection effectiveness need to be studied where they occur, in real software development projects. Empirical studies enable researchers to test whether the proposed technology actually works for people, under a real development environment.

Experimentation is also important for technology transfer. It is risky, time-consuming, and expensive for companies to introduce new technologies. Experimentation can test the new technology with much less risk and cost than if it is tested on a real project. Experimentation can identify some problems of the technology and have them fixed before being applied to real projects. Ideally experimentation can provide data to justify the application of the new technology to facilitate the transfer of new technology to practice.

1.4 Requirements for the New Technique

Perspective-based usability inspection is the application of perspective-based reading technique to usability inspection. In designing such a technique, several

issues need to be addressed [3]:

1. The technique should be tailorable, based on the project and environment characteristics.

2. The technique should be detailed, in that it provides each inspector with a well-defined process.

3. The technique should be specific in that each inspector has a particular purpose or goal for the inspection and the procedures support that goal.

4. The technique should be focused in that a particular technique provides a particular coverage, and a combination of techniques provides the whole coverage.

5. The technique should be studied empirically to determine if and when it is most effective.

The fulfillment of these requirements is reflected through the way the new technique is designed and the fact that it is going to be empirically studied as required.

1.5 Definitions

Usability is defined as [45]:

> the effectiveness, efficiency, and satisfaction with which specified users achieve specified goals in particular environment.

Usability can be further decomposed into the following attributes [37] [57]:

- *Learnability*: The system should be easy to learn so that the user can rapidly start getting some work done with the system.

- *Efficiency*: The system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible.

- *Memorability*: The system should be easy to remember, so that the casual user is able to return to the system after some period of not having used it, without having to learn everything all over again.

- *Errors*: The system should have a low error rate, so that users make few errors during the use of the system, and so that if they do make errors they can easily recover from them. Further, catastrophic errors must not occur.

- *Satisfaction*: The system should be pleasant to use, so that users are subjectively satisfied when using it; they like it.

Usability studies focus on the user interface part of computer systems. But usability can be affected by other aspects of a computer system including functionality and reliability. If a system does not have the function the users need, or often fails, it is definitely not usable.

User interface refers to the parts of a system that users are able to perceive or operate on. It includes input/output hardware and the software that organizes and drives these hardware facilities to support users to use the system. This study focuses on the software part, i.e., how the software can be designed in a way to best support the users with their tasks.

A *method* is a management procedure for applying techniques.

A *technique* is a series of steps, producing some desired effect, and requiring skilled application.

A *usability anomaly* (or *usability problem*) is an aspect of the user interface that would hinder the effective, efficient, and satisfactory use by the specified users for the specified tasks.

A *usability issue* is something that has been reported by an inspector as an anomaly but yet to be determined whether or not it is a real anomaly.

Usability inspection is the process of reviewing a user interface design for the purpose of identifying usability anomalies. Although some usability inspection techniques include team work, others (including heuristic evaluation and perspective-based inspection) only have inspectors conducting the review individually and have the inspection coordinators putting the detected usability anomalies together.

An *inspection scenario* is a collection of procedures that operationalize strategies for detecting anomalies from a specific perspective, within a specific context.

A *prototype user interface* is a representation of the user interface that a user can interact with, one that is built to be changed and improved [33]. It might be:

- a short user guide
- a paper simulation
- a software simulation using a prototyping tool or interface generator
- an early version of the software
- the system to be replaced

Chapter 2

Review of the Literature

This dissertation has been built on literature in several areas. Research on modeling human-computer interaction (Section 2.1) is the foundation for a sound usability inspection technique, in that only after understanding the process itself are we able to examine the process for improvements. The existing usability inspection techniques represent the state-of-the-practice. The existing software inspection techniques provide the possibility of being borrowed and applied to usability inspection. By analyzing these inspection techniques, a framework was built in which a set of attributes were identified for characterizing different inspection techniques. The framework was used to survey the empirical studies on inspection techniques, define the proposed new technique, and design the empirical studies in this work.

2.1 Models for Human-Computer Interaction

Theories have been developed to describe the multiple aspects of human-computer interaction. Some of these theories provided the basis for certain usability evaluation techniques. All these models contributed to the design of the new usability

inspection technique, as described in Chapter 3.

Card, et al. [9] proposed the GOMS (goals, operators, methods, and selection rules) model. They postulated that users formulate goals that they achieve using methods (procedures for achieving goals). A method is carried out through a set of operators, which are elementary perceptual, motor, or cognitive acts. The selection rules are the control structure for choosing among the several methods available for accomplishing a goal. They also described a "model human processor" by a set of memories and processors (Figure 2.1), together with a set of "principles of operation". GOMS model and the user model have been used to derive the GOMS method for user interface evaluation, to be described in Section 2.2.2. The "model human processor" model provides a basis for defining the user characteristics in building the new technique.

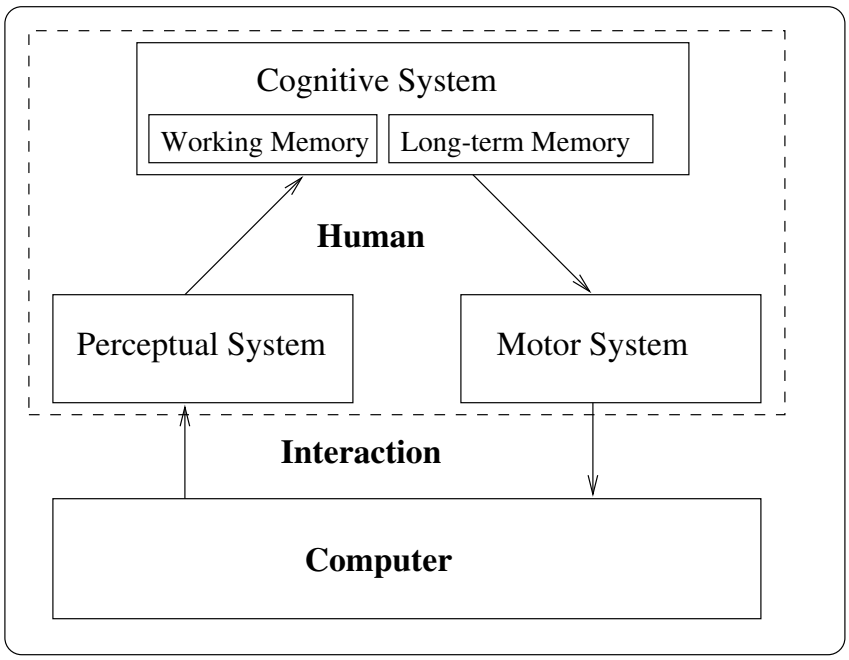

Figure 2.1: Simplified Model Human Processor

Norman [43] used the following seven stages as a model of human-computer

interaction:

1. Forming the goal
2. Forming the intention
3. Specifying an action
4. Executing the action
5. Perceiving the system state
6. Interpreting the system state
7. Evaluating the outcome

Norman placed these stages in the context of cycles of *execution* and *evaluation.* This model leads naturally to the identification of *the gulf of execution* (mismatch between the user's intentions and the allowable actions) and *the gulf of evaluation* (mismatch between the system's representation and the user's expectation) as two general categories of usability anomalies. Norman's model is extended to form a process model upon which usability questions are asked in building the new technique.

Shneiderman [57] proposed the syntactic-semantic model of objects and actions (SSOA model), which is a representation of the user's knowledge in long-term memory. A user's knowledge is divided into syntactic and semantic parts. The syntactic knowledge is varied, device dependent, acquired by rote memorization, and easily forgotten. The semantic knowledge is separated into the computer and task domains. Within these domains, semantic knowledge is structured, device independent, acquired by meaningful learning and stable in

memory. Syntax, semantics, objects, and actions provide a structure for asking usability questions in building the new inspection technique.

Foley, et al. [15] developed the "conceptual, semantic, syntactic, and lexical model" of human-computer interaction:

- The conceptual level is the user's mental model of the interactive system.
- The semantic level describes the meanings conveyed by the user's command input and by the computer's output display.
- The syntactical level defines how the units that convey semantics are assembled together to instruct the computer to perform a certain task.
- The lexical level defines the precise mechanisms by which a user specifies the syntactic units.

This approach is helpful for user interface design because of its top-down nature. Designers are expected to move from the conceptual level to the lexical level, and to record carefully the mappings between levels. In this dissertation, the model has been used in defining a taxonomy of user errors, which is an important part of the "error handling" perspective in the new inspection technique.

2.2 Techniques for Usability Inspection and Software Inspection

This section first describes a framework for characterizing techniques for usability inspection and software inspection. Then the framework is used to review the existing techniques.

2.2.1 A Framework of Inspection Techniques

The framework consists of a list of internal and external characteristics. Internal characteristics are those that are defined in the technique and are not supposed to be changed when the technique is used. External characteristics are factors that are not defined by the technique but will be part of each instantiation of the technique and will have an influence on the effectiveness of anomaly detection.

The list of internal characteristics are:

Prescriptiveness This refers to the extent to which the technique guides the inspectors to do the inspection. This ranges from intuitive, non-systematic procedures, such as heuristics or guidelines techniques, to explicit and highly systematic procedures, such as those in "step-wise abstraction for code inspection."

Individual responsibility Each inspector may be told to conduct the inspection in a general way, i.e., to identify as many anomalies as possible. Or each inspector may be assigned specific roles, i.e., to focus on a limited set of issues for a single inspection session.

Coordination of multiple inspector-sessions Most inspection techniques require multiple inspectors or multiple inspection sessions or both. The coordination of these multiple inspectors can be done in one of three ways:

- *Individual inspections without meeting*
- *Individual inspections followed by a meeting*
- *Inspection meeting only*

The coordination of multiple inspection sessions can be done in one of three ways:

- *Parallel*, with each session inspecting the same version of the artifact. There are no time constraints on different sessions.
- *Sequential with rework*, with anomalies detected in one session fixed before the next inspection session.
- *Sequential without rework*, with each session inspecting the same version of the artifact, but with different focuses. Some sessions have to be conducted before others.

Artifact coverage For software inspection, the artifact is covered according to its linear nature, with consistency checking of interrelated parts. For usability inspection, with the usually large number of screens, objects, and possible actions, it is more difficult to ensure coverage during the inspection. There are two common approaches: (1) have multiple inspectors explore the interface individually in a free way, (2) define a set of representative user tasks and let the inspectors at least check the parts of the user interface executed when going through these tasks.

Usability coverage This refers to the usability issues that the inspection technique addresses. Such issues can be part or all of *ease of learning, efficiency of use, retention over time, error handling, and user satisfaction*, with respect to different users and different working environment.

The list of external characteristics are:

Computer support There are generally two kinds of computer support to inspections. One is to develop tools that can automatically identify cer-

tain types of defects. Another is to use a special computer program to present the inspection instructions, inspection criteria, and the data collection forms.

Artifact format or size This refers to the physical form of the artifact during the inspection. It is often independent of specific inspection techniques. The different formats include a specification of the user interface, possibly with one or more screen shots; prototypes, which include the layout of each screen and a description of the transitions between them; and an executable user interface. Prototypes can be further divided into paper prototype and machine prototype, depending on whether the screen layouts are drawn on the paper, or on the machine.

Inspector expertise This refers to the extent of knowledge and experience the inspector has on user interfaces, usability, and the application domain.

Personal variability This refers to the variation of human performance over time or under different situations.

Organizational structure The refers to the familiarity between the inspectors and the user interface designers, and the familiarity among the inspectors. Familiarity can be measured in terms of whether they have worked together, how closely related they are in the reporting structure, and how close their workspaces are.

The internal and external characteristics and their relationship to the effectiveness of usability inspections are illustrated in Figure 2.2.

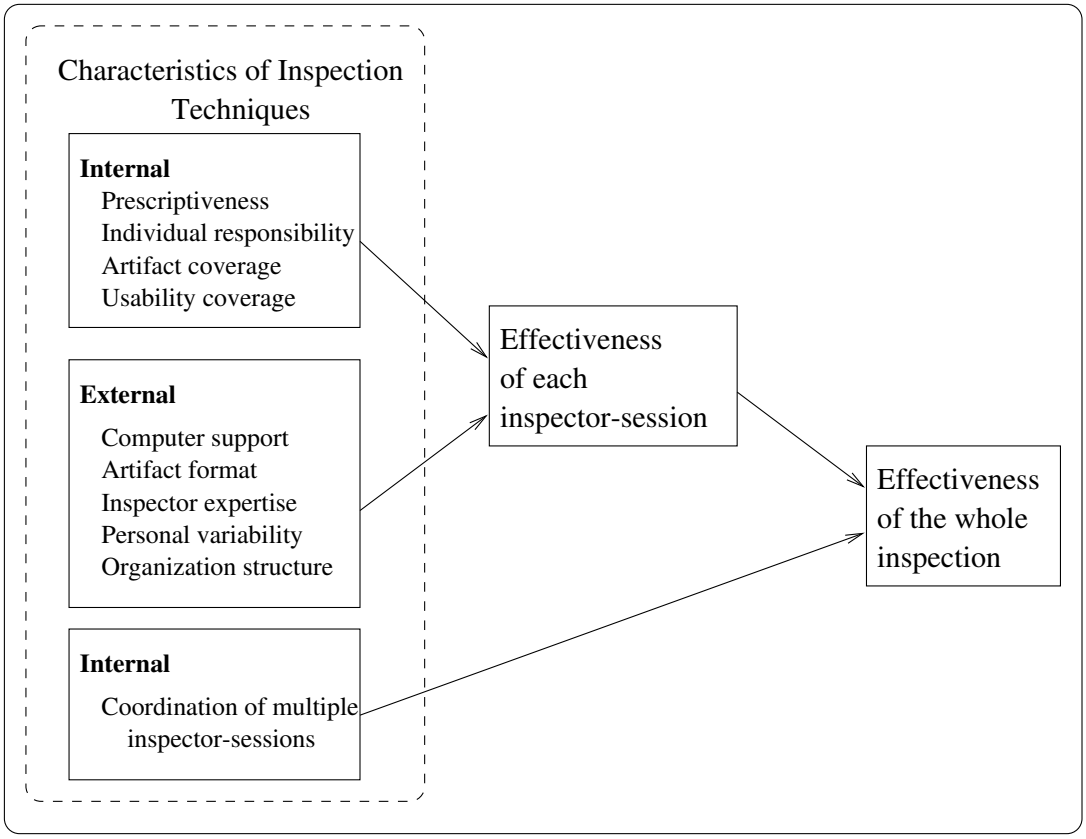

Figure 2.2: The characteristics and effectiveness of usability inspections

2.2.2 Existing Inspection Techniques

The current usability inspection techniques include:

- *GOMS Evaluation* [9, 24, 26] uses a formal model to decompose user tasks into goals, operators, methods, and selection rules to predict users' task completion time, based on a set of standard processing time for human cognitive, perceptual, and motor processors. It focuses on the issue of efficiency for error-free expert use.

- *Cognitive Walkthrough* [61] examines whether the intended users can understand the user interface and successfully complete the defined tasks.

The inspectors all meet together, led by a moderator, to examine each step in the correct action sequence by asking a set of predefined questions. It focuses on understanding and learning in novice use.

- *Guidelines or Standards Inspection* [38] checks to see if the user interface complies with relevant guidelines or standards.

- *Heuristic Evaluation* involves having a set of evaluators examine the user interface and judge its compliance with recognized usability principles (the "heuristics"). Each individual evaluator inspects the system alone, without using task scenarios. Its effectiveness depends on the expertise of the inspectors and the variety of their inspections.

- *Pluralistic Usability Walkthrough* [6] involves a meeting of usability experts, software developers and users, where they work on task scenarios and discuss usability issues.

- *Formal Usability Inspection* [20, 25] is derived directly from Fagan inspections (to be described shortly). Participants represent knowledge areas including software, hardware, usability, and possibly documentation and support. But no users are included. The inspection process consists of individual inspections followed by a meeting. The inspectors are given a set of usability heuristics and a task performance model, which decomposes a user's task performance into four phases: perceiving, planning, selecting, and acting. In the meeting inspectors aggregate detected anomalies and find more anomalies by going through the user task scenarios together.

Software inspections include the inspection of requirements, code, design, test plans, and test cases. Techniques for software inspection can often be bor-

rowed to derive similar usability inspection techniques. The software inspection techniques that have been applied and studied include:

- *Fagan Inspection* [16]. It consists of five steps: overview, preparation, inspection, rework, and follow-up. For overview, the designer describes the work product to be inspected. At the preparation stage, individual inspectors try to understand the work product and study the ranked distributions of error types found by recent inspections. Then at the inspection stage, a "reader" goes over the design, covering every piece of logic at least once and every branch at least once to find defects.

- *Active Design Reviews* [46]. In this approach, inspectors are given questions that can only be answered through careful study of the work product being inspected. Some questions force the inspectors to take a more active role, e.g. to construct a program segment, rather than just reading passively. Furthermore, each inspection is broken up into several inspections. Each of them focuses on one of several aspects, e.g. assumption validity, assumption sufficiency, consistency between assumptions and functions, and access function adequacy.

- *Inspection for Program Correctness* [7]. In this approach, program correctness is decomposed into four aspects: topology, algebra, invariance, and robustness. A list of correctness questions is provided for each of the four aspects. The inspectors are to focus on one aspect at a time, answering the defined questions and may also develop new questions and answer them to examine the correctness of the program.

- *Phased Inspection* [27]. The inspection is divided into several phases. Each phase is aimed at detecting one class of defects. Rework is done to repair the defects found in a previous phase before the next inspection phase begins.

- *N-fold Inspection* [51]. This is based on the assumption that a single inspection team can find only a fraction of the defects in the work product and that multiple inspection teams will not significantly duplicate each other's efforts. So multiple teams each carry out independent inspections in a parallel way. The results of each inspection are collated to form the overall defect list.

- *Defect-based Reading* [48]. This assigns individual inspectors separate and distinct detection responsibilities and provides specialized techniques for meeting them. A possible set of distinct responsibilities is based on different defect classes: data type consistency, safety properties, and ambiguity/missing information.

- *Perspective-based Reading* [3]. This assigns different perspectives to inspectors while they read the work product. For example, the perspectives of the designers, the testers, or the users can be used for reading requirement documents. Each inspection is defined in a way that the inspector will take on an active role to construct either a design, or test cases, or a user's manual. During this process, the inspector is to answer a set of predefined questions that may prompt them to find defects in the document.

These techniques for usability inspection and software inspection are summarized using the framework defined in Section 2.2.1. The results are shown in

Table 2.3 and Table 2.4, which are at the end of this chapter. The issue of usability coverage does not show up in the summary of software inspection techniques (Table 2.4) since it is not relevant to software inspection. For prescriptiveness, a rank of "high" means that the technique has described all of the steps that the inspectors need to follow to do the inspection, while "medium" means that the technique provides moderate guidance and that the inspectors have much freedom in how they are going to do the inspection, and "low" means that the technique gives almost no instruction as to how the inspection should be done.

The next section describes how effective these techniques appeared to be in various studies.

2.3 Empirical Studies of Inspection Techniques

This section reviews the empirical studies that compared different inspection techniques, or applied one technique under different conditions. Based on the framework defined in Section 2.2.1, different techniques differ along the internal attributes: prescriptiveness, individual responsibility, artifact coverage, usability coverage, and the coordination of multiple inspectors or inspection sessions. Different conditions are characterized by the external attributes: inspector expertise, personal variability, organizational structure, artifact format, and computer support. Because of the fundamental similarity between usability inspection and software inspection, results from relevant research on software inspection are also included.

2.3.1 Internal Attributes

Prescriptiveness

The following studies compared inspection techniques that were primarily different in prescriptiveness. They involved techniques with prescriptiveness being "high" (GOMS evaluation and cognitive walkthrough) or "low" (guidelines inspection and heuristic evaluation).

"High" prescriptiveness means that the technique provides a well defined procedure for inspectors to follow. "Low" prescriptiveness means that the technique only provides some principles or criteria such as a short list of (around ten) usability heuristics, but does not tell the inspectors how to do the inspection.

Jeffries, et al. [22] conducted a study in which guidelines inspection and cognitive walkthrough were compared. The subjects were 3 software engineers, who used both techniques. The results showed that both technique detected about 1/6 of the anomalies. Among the anomalies found using guidelines inspection (33 in total number), only about 1/3 were found via the technique itself, with others found as side effect (e.g., while applying a guideline about screen layout, an anomaly with menu organization might be noted) or through prior experience. Most (30 out of 35) anomalies detected using cognitive walkthrough were via the technique. The reason could be that cognitive walkthrough was more prescriptive so that the inspectors were following the technique most of the time. On the other hand, subjects reported that cognitive walkthrough was tedious and sometimes required too much detail. As a result, inspectors were only able to finish seven of the ten tasks during the evaluation session, which was longer than the session for guidelines inspection.

Nielsen and Phillips [41] used GOMS and three forms of heuristic evaluation

to estimate user performance with two alternative designs for database query tasks. The estimated performance (time to complete tasks) was compared to results from user testing. The three forms of heuristic evaluation are cold estimates, warm estimates, and hot estimates. For the cold estimates, 12 inspectors were given a written specification of the two designs. For warm estimates, 10 inspectors were given the running prototype of one design and the written description of another design. For hot estimates, 15 inspectors were given running versions of both interfaces. For GOMS, 19 evaluators were given the same specification of the two interfaces as used by the cold estimates. Therefore, GOMS was comparable the cold estimates situation. The results showed that all methods gave a good estimation of the relative advantage of one design over the other. For absolute user performance (i.e. time to complete tasks), GOMS was always better than the cold estimates, but was not as good as the other two forms of heuristic evaluation. GOMS had much less variance among its 19 evaluators than any form of the heuristic evaluation methods. The variance for GOMS was 19% of the mean, while for the cold, warm, and hot estimates (using heuristic evaluation) the variance was 108%, 75%, and 52% of the mean, respectively.

Desurvire [12] conducted a study in which a phone-based interface was evaluated by groups of three evaluators of different experience levels by using either heuristic evaluation or cognitive walkthrough. The three different experience levels were: experts who had at least three years of human factors work experience; software developers; and non-experts who were not experts in usability or user interface design. The total number of evaluator groups was six, one for each technique and experience level. All groups used task-based evaluation, with 6 basic tasks representative of the system's usage. User testing data was collected

from observing and videotaping 18 potential end users of the system, who performed the same 6 basic tasks. The identified usability anomalies were classified into the following three categories:

- minor annoyance or confusion
- anomaly caused error
- anomaly caused task failure

The results from this study are summarized in Table 2.1. It shows that the groups of non-experts and software developers had almost the same performance for the two different techniques. But the expert group using heuristic evaluation did better than the expert group using cognitive walkthroughs. Again it was reported that cognitive walkthrough was very time-consuming. Perhaps this has caused the experts to spend too much time dealing with the inspection procedure, which made their inspection performance poorer.

In summary, these studies provide evidence that the more prescriptive cognitive walkthrough technique seems to be more helpful than guidelines inspection for software developers to detect usability anomalies (the Jeffries, et al. study); that that the more prescriptive GOMS evaluation generates more consistent results among the evaluators than heuristic evaluation (the Nielsen and Phillips study); and that neither heuristic evaluation nor cognitive walkthrough is effective when used by software developers or non-experts for usability inspection (the Desurvire study).

Table 2.1: Results from the (Desurvire 1992) study: number of anomalies detected

Technique	User testing	Heuristic evaluation			Cognitive walkthrough		
Evaluators	Users	Experts	Developers	Non-experts	Experts	Developers	Non-experts
Anomalies that did occur	25	11	4	2	7	4	2
Potential anomalies	29	9	7	1	9	6	2
Minor annoyance or confusion	5	4	2	1	2	2	1
Anomalies caused error	3	2	0	0	2	0	0
Anomalies caused task failure	17	5	2	1	3	2	1

Individual Responsibility

In the studies reviewed so far, inspectors using the same technique all had the same responsibility and were not asked to take different perspectives at different stages of the inspection. In the following studies, inspectors were asked to review the interface several times, each time with a different perspective, or focusing on a different set of usability issues.

In a study by Desurvire [11], each of the three levels of inspectors – human factors experts, non-experts, and developers – were asked to study flowcharts of a voice interface (for the same interface as in the Desurvire study described earlier) several times, once from each of several quite different perspectives. The perspectives used were of: the inspector's own, a human factors expert, a cognitive psychologist, a behaviorist, a Freudian, an anthropologist, a sociologist, a health advocate, a worried mother, and a spoiled child. All evaluators received the same order of perspectives. For each perspective, after reading a short orientation toward that perspective, the evaluators looked at the flowchart and each of the three tasks and recorded possible user problems.

Compared to the other Desurvire study using the same interface under a similar setup (as described earlier in this section), the author (see Table 2.2 for comparison) suggested that the perspectives approach may offer substantial promise as a technique to enhance interface evaluations by non-experts and developers in several dimensions, such as avoiding false positives, finding real anomalies, and offering suggestions for improvements.

Kurosu, et al. [28] compared heuristic evaluation and a variant, "structured heuristic evaluation." Each usability session was divided into sub-sessions, with each sub-session focusing on one of the following: operability, cognitivity,

Table 2.2: Comparison of anomaly detection by perspectives (Desurvire 1994) and other techniques (Desurvire 1992)

Method	Evaluators	% of anomalies that occurred	% of potential anomalies
Perspectives	Experts	37	27
	Developers	29	33
	Non-experts	34	40
Heuristic evaluation	Experts	44	31
	Developers	16	24
	Non-experts	8	3
Cognitive walkthrough	Experts	28	31
	Developers	16	21
	Non-experts	8	7

pleasantness, novice/expert, and disabled users. Two variations of structured heuristic evaluation were used in the experiment: one with 32 guideline items and the other with 41. There were 5 subjects for each technique, all non-expert inspectors. Each subject spent a total of 3 hours inspecting the usability of a walkman. The results were that structured heuristic evaluation with 41 guideline items revealed more than twice the number of anomalies revealed by heuristic evaluation, and the other condition (with 32 guideline items) revealed about 1.5 times as many anomalies as revealed by heuristic evaluation.

In software inspection, several techniques have different inspectors take different responsibilities, or the same inspectors focusing on different issues at different times. Such techniques include active design reviews, inspection for program correctness, and phased inspection, as described in Section 2.2.2. Recent studies introduced a family of scenario-based reading techniques, which have been applied to the inspection of software requirement documents [3, 48]. The reading techniques use a set of operational scenarios, which are focused, detailed, and specific to a particular emphasis and viewpoint, and the combination of which provide coverage of the work product.

In Porter, et al. [48], one such technique, defect-based inspection, was used. The scenarios were based on different defect classes (data type consistency, safety properties, and ambiguity/missing information). It was compared against Ad Hoc and Checklist techniques. The results were:

- Scenario inspectors (those who used defect-based inspection) detected more defects than Ad Hoc and Checklist inspectors, with an improvement of about 35%,
- Scenario inspectors were more effective at detecting the defects their sce-

narios were designed to cover, and were no less effective at detecting other defects,

- Checklist inspectors were no more effective than Ad Hoc inspectors.

In Basili, et al. [3], another scenario-based reading technique, perspective-based reading, was empirically studied in the context of inspecting software requirement documents. The scenarios were based on the perspectives of the developers, testers, or users, respectively. It was compared against the existing technique the subjects would normally use in their work. The results were:

- for documents of the generic domains (a parking garage document and an ATM document), perspective-based reading performed better than the technique that the subjects usually use,

- for documents of the subjects' working domain (flight dynamics documents), there was no statistically significant difference, possibly because of the tendency of the subjects to ignore the new techniques when faced with familiar documents, or the ineffectiveness of the scenarios with this particular domain.

In summary, results from these studies suggest that it is promising to have each inspector focus on a subset of issues or inspect from a specific perspective, especially for non-expert inspectors in usability inspection. The Desurvire study showed that using perspectives greatly helped the software developers and non-expert inspectors in detecting usability anomalies. Unfortunately there was no discussion of how the list of perspectives was chosen and whether it was efficient to use so many perspectives. It leaves much space for further exploration.

2.3.2 External Attributes

Inspector Experience

In a study conducted by Nielsen [36], one "voice response" user interface was subject to heuristic evaluation by three groups of evaluators: usability "novices" with knowledge about computers in general but no special usability expertise; "single experts" who were usability specialists but not specialized in the domain of the interface; and "double experts" with experience in both usability in general and the kind of interface being evaluated. In the study, the novice inspectors were 31 computer science students who had completed their first programming course but had no formal knowledge of user interface design principles. The "single experts" were 19 usability specialists. The "double experts" were 14 specialists in usability for the specific application domain. On average, the novice evaluators each found 22% of the usability anomalies in the interface; the single experts found 41% of the anomalies each; the double experts found 60% each. Accordingly, it was estimated that to achieve 80% coverage, about 2 "double-experts", 4 "single-experts," or 16 novices are needed.

In the two Desurvire studies mentioned earlier (see Table 2.1 and Table 2.2), it is clear that inspector expertise on usability had a substantial impact on the inspection results. Using the same technique, the same number of expert inspectors found 5 times more anomalies than the non-expert inspectors.

Therefore, one important thing to do in empirically comparing inspection techniques is to have subjects with similar background, to randomly assign subjects into different groups, and to use post hoc analysis to see if the different groups indeed were homogeneous in terms of experience.

Individual Variability

Inspectors with the same qualification can have quite different performance in conducting usability inspection. In eight case studies conducted by Nielsen [39], the ratio between the number of usability anomalies found by the top and bottom quartile (best 25 percent versus worst 25 percent) of evaluators ranged from 1.4 to 2.2, with a mean of 1.7. These numbers represent cases in which evaluators with essentially the same background and qualifications were compared.

The same inspector's performance can vary significantly from one inspection to another. Nielsen [40] conducted a study where 34 inspectors with the same background inspected two different user interfaces using heuristic evaluation. The correlation between the number of usability anomalies found by individual inspectors in the two systems was r=.57. This definitely indicates better than random consistency, but at the same time also indicates substantial unexplained variability in performance from one inspection to the next.

Artifact Format

The artifact format in a usability inspection can be either a specification, a paper prototype, a prototype on the computer, or a running (or partly running) system. A user interface can be inspected in different formats during different development stages.

A study conducted by Nielsen [35] showed that the medium in which a design is presented had a major impact on what kind of usability anomalies can be discovered using heuristic evaluation. The computer mockups seemed to focus the evaluation on the major usability anomalies and offer the evaluator an experience closer to that of a real user. Paper mockups are better with respect

to showing certain inconsistencies in a design. Also, inspectors tended to detect more usability anomalies when working on a paper prototype than looking at a user interface on the computer. The possible reasons include that people are more eager to critique something that has not been implemented yet, and that people have a higher tendency to miss elements on the screen than elements on the paper.

Analytically, an advantage of using a specification seems to be that it makes it easier to cover the user interface in a logical way. The disadvantage of using a specification is that for a complex user interface, it is quite difficult for an inspector to understand the details about how it works from the specification.

The above results and analysis provide some guidance in developing a new inspection technique or applying existing techniques.

Computer Support

Computer support has been provided for various software inspection and usability inspection techniques. It can perform automatic evaluation of certain issues, or facility the inspection process.

A family of support tools built at the University of Minnesota include CSI (Collaborative Software Inspection) [31], CAIS (Collaborative Asynchronous Inspection of Software) [32], and AISA (Asynchronous Inspection of Software Artifacts) [58]. CSI assists by recording and collating the inspectors' comments, and allowing the inspection meeting to be geographically distributed, with the artifact being displayed on each inspector's screen and a voice connection that allows people to talk to each other. CAIS extended CSI to support distributed, asynchronous inspection of textual artifacts. AISA extends the above tools to

provide a World-Wide Web based tool for distributed, asynchronous software inspection of textual and graphical artifacts. Stein, et al. [58] found that the "depth" of the comments made in asynchronous inspection was greater than that normally seen in inspection meetings. It might be because the participants had a chance to compose their thoughts before responding to comments, instead of having to respond immediately as in a meeting.

A Web-based inspection tool called `hyperCode` has been used at Lucent Technologies [47]. At the initial preparation phase, the inspection package is made available on-line with e-mail notification of availability. The inspection preparation and collection are done concurrently, with `hyperCode` providing the automatic collection of the inspection comments. The resolution of the comments is done by the moderator and author as part of the repair phase, with the collection meeting eliminated. The tool was quickly accepted by practitioners. Inspections using the tools were less expensive in cost and at least as effective in defect detection, as compared to inspections without the tool.

Hartson, et al. [14] provided a taxonomy of remote usability evaluation, including remote inspection, by using computer networks. Remote usability inspection is often used when no usability inspector is available locally. One example remote usability inspection service is provided at Vertical Research, Inc. [42], where the network is used to transfer user interface artifacts to the inspectors and to transfer the inspection results to the user interfaces designers.

In order to deal with the tediousness of the cognitive walkthrough process, Rieman, et al. [49] developed an automated prompting system which aimed to reduce the technique's overhead and focus the inspector's attention on critical issues. The system helped the inspector by organizing the inspection questions

into groups, and prompting the inspectors with a small set of questions under each group at each time. For each action, the tool kept a record of which groups of questions have been answered.

A family of user interface consistency checking tools was developed at the Human-Computer Interaction Laboratory of the University of Maryland [30]. It aids the interface evaluation process by providing a compact overview of possible inconsistencies and anomalies on certain textual and spatial characteristics of the interface. Currently it is limited to Visual Basic applications, but is extensible to other application by writing a translator to convert interface form files created by other development tools to the canonical format used by the evaluation tools.

Sears [55] developed a tool to use a set of metrics for layout appropriateness to evaluate the usability of the user interface layout based on user tasks. It outputs some usability metric values and provides interpretations together with those values.

Scholtz [52] described WebMetrics, which consists of a collection of tools for developing and evaluating Web sites and Web applications. One of the tools, WebSAT, among other similar tools, examines the HTML files of a Web page against usability guidelines and advises the developer about potential usability anomalies.

In summary, tools are potentially good for reducing the human inspectors' workload by detecting certain anomalies automatically, and making the inspection process easier by taking care of some of the logistics.

2.3.3 Organizational Structure

Organizational structure can also affect inspection results. An empirical study conducted by Seaman [53] reported that:

- The more the inspection participants interact with each other on a regular basis, the fewer defects will be reported.
- The more the inspection participants have worked together in the past, the fewer defects will be reported.
- The more closely related the inspection participants are in the reporting structure, the fewer defects will be reported.
- The closer the workspaces of the inspection participants are physically, the fewer defects will be reported.

These results suggest using inspectors who are not familiar with either the author of the artifact, or the other inspectors.

2.4 The Goal/Question/Metric Method

This sections reviews the literature on GQM, which is used in building the new inspection technique and in defining the goals of the empirical studies.

Basili, et al. [4, 5] described the Goal/Question/Metric (GQM) method, which is a mechanism for supporting the setting of operational goals for software projects with regard to the appropriate perspective and relevant environment. Based on models, questions are generated that define those goals as completely as possible. Answering the questions often requires collecting, verifying, and

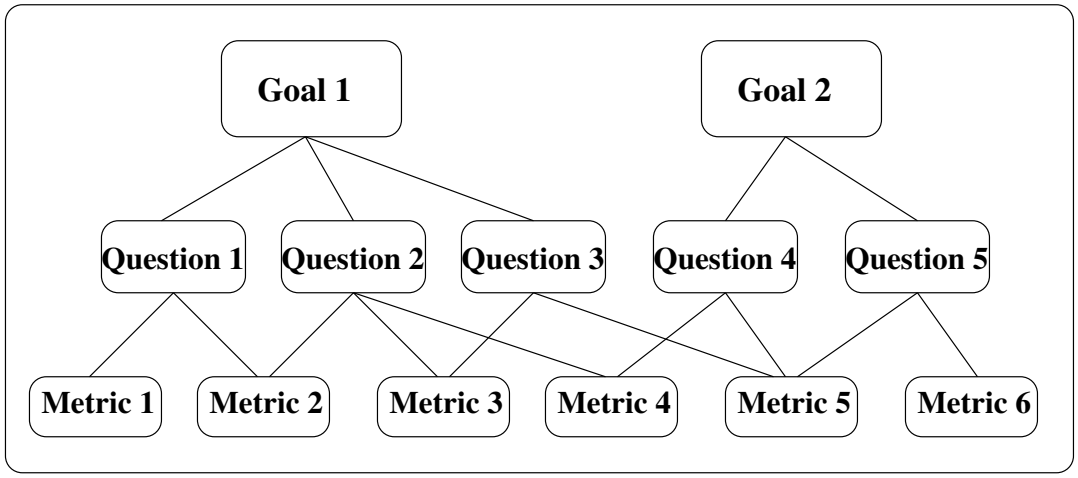

Figure 2.3: The GQM model

analyzing certain data. Therefore the goals are defined in a top-down fashion, and are achieved in a bottom-up fashion.

In this dissertation, the goal of usability inspection is decomposed into questions about different aspects of usability. Answering these questions needs data about the user interface, the users, the tasks, the users' working environment, and criteria for drawing conclusions about usability based on those data. Chapter 3 shows the details of this process.

2.5 Summary

The literature on usability inspection and software inspection techniques presents some promising results, but at the same time shows deficient areas that call for further exploration.

Inspection with a specific focus Promising results came from studies in both usability inspection (Desurvire [11] and Kurosu, et al. [28]) and software inspection (Porter, et al. [48] and Basili, et al. [3]). What are missing

from the usability inspection studies were a well-defined smaller set of inspection scenarios, and research on the possibility of having each inspector use only one scenario. The two existing usability studies did not report how the perspectives or the anomaly categories were defined and how they covered the different usability issues.

Empirical validation of new techniques The empirical studies of usability inspection techniques in the literature often had too few subjects (usually 3 to 5 subjects for each condition) along with other problems. More vigorous studies and replicated experiments are needed to provide stronger evidence about the effectiveness of a new technique.

To address these deficiencies, this dissertation developed perspective-based usability inspection with three usability perspective, novice use, expert use, and error handling, and conducted 3 empirical studies to investigate the effectiveness of the new technique.

Table 2.3: Characteristics of usability inspection techniques

	Prescriptiveness	Individual Responsibility	Coordination of Inspector-sessions	Artifact Coverage	Usability Coverage
GOMS	High	Specific but same	Undefined	Task scenarios	Error-free expert use
Cognitive Walkthrough	High	Specific but same	Group inspection when multiple inspectors are available	Task scenarios	Novice use
Guidelines or Standards	Low	General	Undefined	Undefined	General
Heuristic Evaluation	Low	General	Combination of replications	Free exploration	General
Pluralistic Usability Walkthrough	Medium	Inspector's own role	Inspection meeting with multiple roles	Task scenarios	General
Formal Usability Inspection	Medium	General	Collection meeting after individual inspections	Task scenarios	General

Table 2.4: Characteristics of software inspection techniques

	Prescriptiveness	Individual Responsibility	Coordination of Multiple Inspector-sessions	Artifact Coverage
Fagan Inspection	Medium	General	Inspection meeting	Undefined
Active Design Review	High	Specific	Combination of inspections with different focuses	Focus-dependent
Inspection for Program Correctness	High	Specific	Combination of inspections with different focuses	Focus-dependent
Phased Inspection	Medium	Specific but same within each session	Different sessions focus on different issues, with rework between sessions	Undefined
N-fold Inspection	Medium	General	Combination of replications	Undefined
Defect-based Inspection	High	Specific	Combination of inspections that focus on different defects	Scenario dependent
Perspective-based Reading	High	Specific	Combination of inspections from different perspectives	Perspective dependent

Chapter 3

Perspective-based Usability Inspection

3.1 Overview

There are two steps in the proposed technique for usability inspection. The first step is to derive the inspection plan through a GQM analysis. The second step is to carry out the inspection plan. The first step, i.e. the planning, is performed by one or more usability specialists who are well trained to do it. The second step, i.e. the inspection, is carried out by people who have certain task domain and computer domain knowledge.

The technique provides practitioners with a GQM structure which consists of a set of comprehensive usability goals and a set of generic usability questions associated with the goals. Before conducting a specific inspection, practitioners can first tailor the GQM structure based on the usability goals of the project. Then they can tailor and instantiate the usability questions based on the user interface, the tasks to be supported, the user characteristics, and the users' working environment.

For each inspection session, an inspector will be assigned a specific perspective and check a subset of usability issues by following a provided procedure.

This two-step approach makes the proposed technique different from the existing usability techniques (as described in Section 2.2.2). The existing techniques all let inspectors do inspections from scratch by using the same inspection material for any user interface. The benefit of the two-step approach is that inspection materials become very relevant to the interface being inspected.

Using the framework in Section 2.2.1, the internal characteristics of the proposed techniques are as follows:

Prescriptiveness For inspection with each perspective, the technique provides a description of the perspective, a user profile, a set of task cases, and a list of usability issues associated with the perspective. An inspection procedure will be provided for each perspective. The details are described below.

Individual responsibility Each inspector in each session works on one perspective.

Coordination of multiple inspector-sessions Each complete inspection includes at least one session from each of the three perspectives. These different inspection sessions can be carried out by different inspectors, or by the same inspector at different times.

Artifact coverage This is achieved by the use of task cases. For some systems a small set of task cases can provide full coverage of the functionality. For other systems a well chosen set of task cases can provide coverage of the typical uses of the system.

Usability coverage The different perspectives collectively cover the following usability issues: understandability, ease of learning, ease of use, efficiency of use, and error handling.

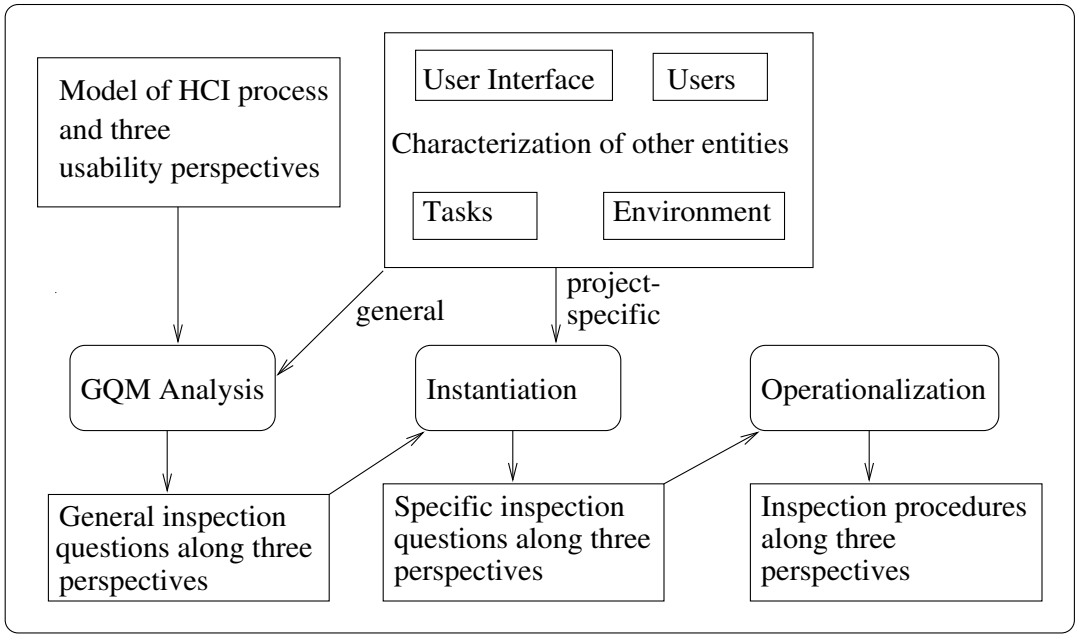

Figure 3.1: The process for deriving the inspection procedures

Figure 3.1 shows how the technique starts with models and finally derives the inspection material. In the following sections, questions will be defined to characterize the user interface, the users, their tasks, and their working environment (collectively those factors are called the usability context). A model for the human-computer interaction process will be built by extending the "seven stages of action" model. Then usability questions are defined along three perspectives: novice use, expert use, and error handling. For each perspective, usability goals are defined, a tailored process model is examined with respect to the usability goals and the characteristics of the usability context (as defined by their respective questions). The result is a set of usability questions for each of the

three perspectives. For each specific evaluation, answers are available for at least some of the questions about the usability context. By using answers to these questions and following the provided guidelines, specific inspection questions can be derived. By operationalizing these questions, the inspection procedures are defined.

3.2 Models to be Used in the GQM Analysis

The models that are adopted from the literature are:

- The seven stages of action model by Norman [43];
- The syntactic-semantic object-action model by Shneiderman [57];
- The user model by Card, Moran, and Newell [9].

Two new models built as part of the dissertation are:

1. A three-perspective usability model proposed in this work;
2. A process model for human-computer interaction proposed in this work, which extends Norman's model.

3.2.1 The Three-Perspective Usability Model

The three-perspective usability model states that when using a computer to accomplish tasks, a user will experience one or more of the following situations:

Novice use The user's knowledge and experience do not tell the user how to use the system to achieve the goal.

Expert use The user knows how to use the system and prefers to achieve the goal efficiently and easily, or wants to achieve higher goals.

Error handling The user has a problem with the effect achieved by the previous action and needs to resolve the problem.

These three perspectives were defined based on the following two questions:

1. Did the user make a mistake in the last step and need to recover from that mistake?

2. Does the user know how to complete the current step?

If the answer to question 1 is "yes", then the situation is covered by "error handling". Otherwise, answering "no" to question 2 leads to "novice use", and answering "yes" leads to "expert use". Therefore, both "novice use" and "expert use" only consider user actions along the correct path. "Error handling" is not further broken down to "error handling for novices" and "error handling for experts" because the latter is not very frequent and does not involves a significant number of usability issues that are not in "error handling for novices."

These three situations form the three perspectives in the proposed usability inspection technique. Other perspectives may be used, especially for specific application or user interface domains. These perspectives are meant to be easily separable, give inspectors a focused attention, and at the same time tie naturally to an inspection procedure. Other possible perspectives that were considered but not chosen include "static and dynamic aspects of the interface", and "user's perception, cognition, and motor activities."

Figure 3.2 gives a state diagram of a user's possible experience while using a computer to complete tasks. A user can experience one or more of the three

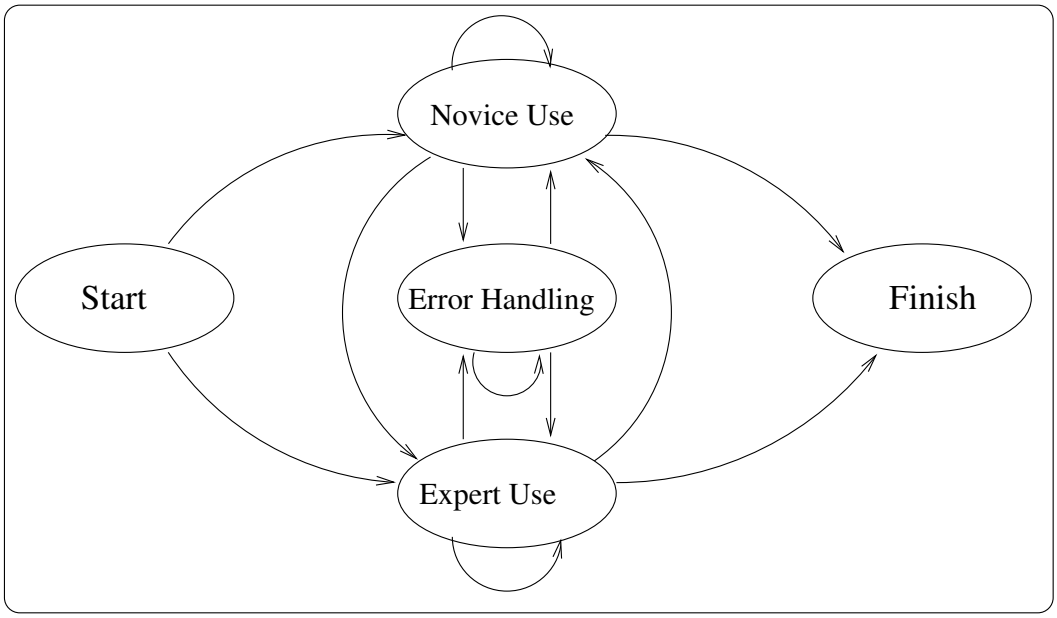

Figure 3.2: A state diagram of the three perspectives

situations from the start to the completion of the task. A user can be at "novice use" at one step and enter "expert use" at the next step, or vice versa, since the user may only be familiar with part of the system. A user can enter the "error handling" state from either the "novice use" or the "expert use" state. Since these three states provide a full coverage of the situations a user may experience, they are used to form the three perspectives for partitioning usability issues.

Figure 3.3 shows the different user interface facilities users will need under each of the three different situations. In the "novice use" state the user is looking for guidance from the user interface. In the "expert use" state the user is looking for facilities that make task completion even faster and/or easier. In the "error handling" state, on the one hand the user needs help understanding what has gone wrong and how to recover from it, while on the other hand the system should avoid putting the user into such a situation. These are some of the

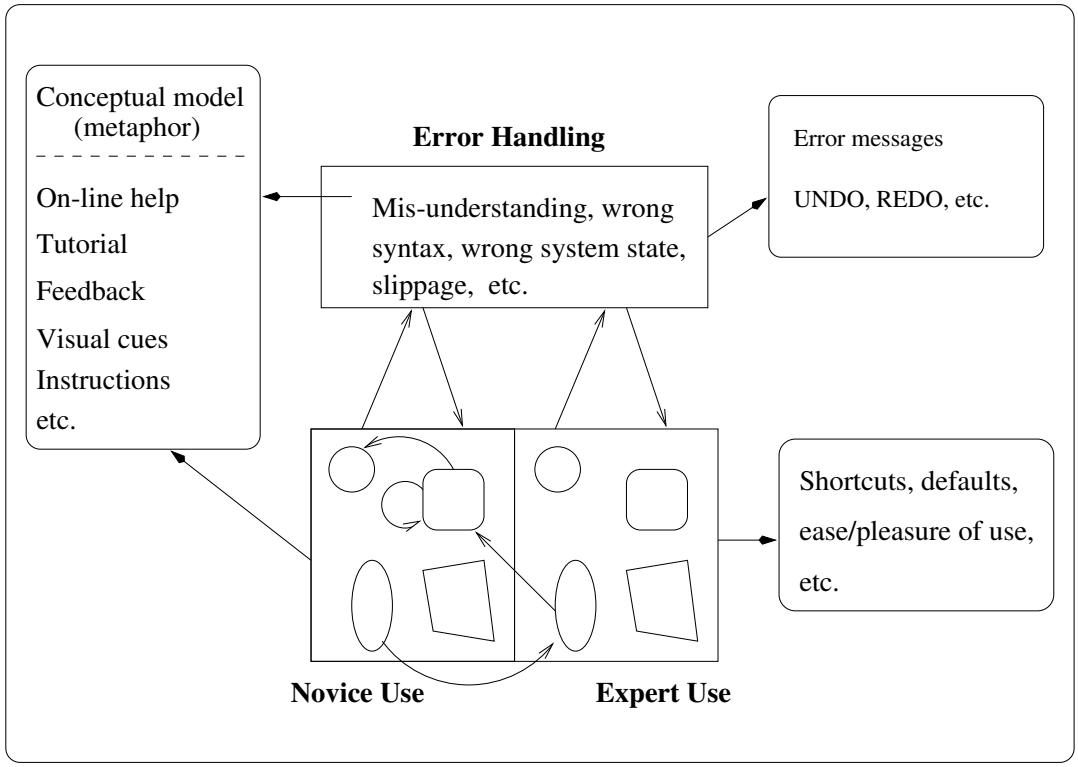

Figure 3.3: The three perspectives and user needs

usability issues related to each perspective, which are to be examined during the inspection using that perspective.

The concept of a user being a "novice" or an "expert" is not used here in reference to the whole system. Rather it is in reference to an "interaction style" in the user interface. For example, a user may have never used a particular Web-based data collection form. But if the user has used "text fields" and "radio buttons" in other Web-based forms, he or she is an "expert" on the same "text fields" and "radio buttons" in the new Web-based form.

3.2.2 A Process Model of Human-Computer Interaction

By extending the "seven stages of action" model with consideration for error handling as well as the start and termination of the system, the following model is defined for the human-computer interaction process. Here each task is accomplished by a series of steps, each of which consists of an action and some objects. Each step can have one or more mini-steps. Each mini-step represents a user effort towards specifying an object or an action. It can be one of the following:

- A keystroke (including combination keys)
- A mouse click (either single click or double click)
- A mouse movement up to when some other object's appearance changes on the display (if any)
- A screen touch that the user interface reacts to (in case of touch screen interfaces)
- The speaking of a word (in case of voice input interfaces)

A step is a series of consecutive mini-steps that as a whole are meant to achieve a goal such as entering a data item, selecting an object, or initiating an operation, etc. In the following procedure, the user may perceive the effects after each mini-step, or the user may perceive the effects only at the end of a step.

The following process model is illustrated in Figure 3.4, which is at the end of this chapter.

1. Start the system if necessary.
2. Repeat the following (a through c) until there are no more tasks to do.
   (a) Map the next task to the effects to be achieved in the computer system.

(b) Identify the actions and associated objects for achieving the effects, as well as the sequence to execute these actions.

(c) Repeat the following (i through v) until the task is completed.

i. Execute the next step or mini-step.

ii. Perceive the effect of the action.

iii. Understand the progress made towards achieving the goal.

iv. Is progress made? If yes, continue to next loop; otherwise go to step 11 for error handling.

v. Identify the kind of error that has occurred and act accordingly:

- For slippage: reverse the effect caused by the slippage; let the original step be the next step;
- For syntax error: reverse the effect caused by the wrong syntax (if there is any); find out the correct syntax; let the original step be the next step;
- For missing task step(s): back up several steps as necessary; let the first step that was missed to be the next step;
- For wrong action(s): reverse the effect caused by the wrong actions (if necessary); re-map the task to actions of the user interface;
- For wrong system mode: reverse the effect caused by the action under the wrong mode (if there is any); switch to the "right" system mode; let the original step be the next step.

3. Terminate the system if necessary.

3.3 Using GQM to Generate the Generic Inspection Questions

Using the GQM method, the goal of perspective-based usability inspection can be defined as:

Analyze the user interface of a computer system, its intended users, tasks and working environment for the purpose of anomaly detection with respect to usability from the point of view of novice use, expert use, and error handling in the context of a specific project (usability goals, etc.).

To achieve this goal, we need to characterize the *user interface*, the *users*, the *tasks*, and the *working environment* by defining a set of questions for each of them. Then we need to define a set of questions for *usability* from each of the three perspectives.

3.3.1 Characterizing the User Interface

The user interface is the medium between the user and the system. Analyzing the interface components will help understand what needs to be inspected.

Questions concerning the different aspects of a user interface are listed below. These questions are defined along fundamentals (entry and exit points, metaphor, system states, etc.), objects (menus, layout, colors, fonts, etc.), actions (task steps, feedback, etc.), and requirements on software and hardware. The objects part of the questions does not cover all possible user interface components. More questions can be asked about concerning other types components in the interface. The answers, i.e. facts about the user interface, can be used to answer related usability questions, defined later on in Section 3.3.5.

Fundamentals:

I1. How can the user start to run the system?

I2. How can the user exit the system?

I3. What is the metaphor used in the user interface?

I4. Is on-line help always available?

I5. What is the number of working states?

I6. If there are multiple working states, how is the current state indicated?

I7. To what extent can the user change the user interface?

For each window or dialog box:

I8. What user interface components are used in each screen/dialog box?

I9. What is the number of items in the menu tree?

I10. What is the number of levels of the menu tree?

I11. What colors are used?

I12. How are different colors used to present different information?

I13. What is the number of different colors used in each screen/dialog box?

I14. What font types and font sizes are used?

I15. How are different fonts used to present different information?

I16. What is the number of different fonts used in each screen/dialog box?

I17. How are the contents laid out?

(Other questions can be asked about other components in a specific inspection such as icons, menu bar, buttons, navigational facility.)

Relating to tasks:

I18. For each task, what kind of objects, actions, and guidance does the user interface provide?

I19. What is the name or visual representation of each action?

I20. What is the syntax for executing each action?

I21. What visual cues are used for the objects, actions, and other?

I22. For each possible user input, what is the reaction by the user interface?

Relating to environment:

I23. What software and hardware support are needed to run the system, including hardware architecture, operating system versions, supporting software versions, free disk space, memory, screen resolution and colors, other input and output devices?

3.3.2 Characterizing the Users

A usability inspection seeks to detect potential problems that the intended users may have when they use the system through the user interface. We need to understand users before we can predict what problems they may have with the user interface.

Combining the "model human processor" [9] and the SSOA model [57], a user can be characterized by the following dimensions:

Motor system: e.g. keyboard skills, possible disabilities;

Perceptual system: consider possible disabilities (including color-blindness);

Cognitive system: working memory and long-term memory (task domain knowledge, computer domain knowledge, syntactic knowledge).

The questions concerning users are listed below.

U1. How are the computer skills of the expected users? (can the users touch-type, can they use mouse, etc.)

U2. May some of the users have motor or perceptual disabilities (problems with their hands, vision, hearing, and speech in case of speech-input systems)?

U3. How is the task domain knowledge of the expected users?

U4. How is the computer domain knowledge of the expected users?

U5. What is the matrix of familiarity of users towards the different types of components in the user interface?

U6. How is the syntactic knowledge of the expected users towards this user interface?

U7. What are the language and cultural backgrounds of the expected users?

3.3.3 Characterizing the Tasks

Tasks are domain specific. A usability inspection is conducted with regard to the defined users and defined tasks. Therefore it should include going through a set of representative user tasks.

The questions concerning tasks are listed below.

T1. What tasks does the user interface support the users to accomplish?

T2. What is the association matrix of tasks and user interface components?

T3. How frequently is each task expected to be executed?

T4. Which subset of tasks should novice users be able to complete?

T5. What are the advanced features and functions that only expert users are expected to use?

During an inspection, each task will be mapped to steps in the user interface, where each step consists of an action and some object(s).

3.3.4 Characterizing Users' Working Environment

The users' working environment is another factor that may affect the human-computer interaction process.

The users' working environment can be characterized by the following factors [10]:

- Security. The importance of keeping the system secure.
- Organization. Including organization policy, culture, goal, and procedure.
- Physical environment. Including lighting, noise, working space, etc.
- Social. Including users' role, the extent of group working, the communication structure, etc.
- Technological set-up. Quality and capacities of hardware and software.

The questions concerning environment are:

E1. What restrictions may the environment have on the use of certain interaction channels: the use of typing or mouse operation for input, the use of speech for input or output, etc.?

E2. What unusual requirements may the environment have on visual display: the contrast, the font size, the colors, etc.?

E3. What kind of hardware and software settings will the specified users have, including hardware architecture, operating system versions, supporting software versions, free disk space, memory, screen resolution and colors, other input and output devices?

E4. Will users tend to use the system alone or together with someone else?

E5. Will users tend to have co-workers around who may help them to use the system?

E6. For security reasons, what kind of results (system delays, delays in finishing the task, errors, etc.) are unacceptable when using the system?

E7. Do users need to enter confidential information?

3.3.5 Characterizing the Usability Perspectives

Perspective-based usability inspection divides the usability issues along different perspectives and focuses on one perspective for each inspection session.

In this section, usability will be characterized along the three suggested perspectives: novice use, expert use, and error handling. Questions will be derived from each usability perspective based on the process model of human-computer

interaction in Section 3.2.2, as well as the characterizing question of user interface, user, task, and environment as defined in the previous sections. Each perspective will have certain assumptions which may answer some of the questions of the characteristics of the users and the tasks. Furthermore, each perspective may relate to only a subset of these characteristics.

The usability questions for each perspective can be derived by going through the process model in Section 3.2.2 from that particular perspective. The process is examined with respect to the usability goals of that perspective. Questions are asked about whether the characteristics of the user interface, the users, the tasks, and the environment will fit together to assure the achievement of those usability goals. It is a tedious process to generate all the possible questions. Domain knowledge is needed in determining which questions are relevant.

Since "error handling" is treated separately, the other two perspectives will focus on the correct paths only. For example, "novice use" will consider whether the user interface provides sufficient guidance to help a novice user follow the correct path, but will not consider the possible incorrect paths users may choose; "expert use" will consider whether the expected short-cuts are available, but will not consider what happens when the user tries to use the unavailable short-cuts.

From the GQM point of view, the usability goals are organized into three groups of subgoals along the three usability perspectives. Through analysis based on the process model, each of these subgoals is defined by a set of questions.

Conceptually, some questions are related to more than one usability perspective. For example, understandability of user interface objects and actions relates to both novice use and error handling; some consistency questions relate to all three perspectives. Operationally, each question only appears under one

perspective.

In characterizing these usability perspectives, visual user interfaces are considered. For special situations such as voice interfaces, interfaces for handicapped users, and CSCW (Computer Supported Cooperative Work) systems, the characteristics need to be changed accordingly.

Perspective 1: novice use

Assumptions:

- The user does not have enough prior knowledge to accomplish a the task through the user interface. The issue becomes whether the knowledge the user obtains directly from the user interface (metaphor, cues, guidances, etc.) is sufficient for accomplishing the task.

Usability Goal:

Goal 1: The fundamental tasks can be accomplished by the defined users with the minimum knowledge.

Tasks:

Only the fundamental tasks that novice users should be able to complete will be considered.

Users:

Users are assumed to have the minimum level of computer domain knowledge and task domain knowledge as described in the user profile, which is an characterization of the users and is provided to inspectors.

Questions:

Now only certain types of users are considered. The tasks to be considered are also constrained to the ones that a novice user is expected to accomplish. For

the user interface, only the parts that a novice user is expected to see need to be inspected. The usability questions can be derived based on the models of human-computer interaction, user, task, user interface, and environment, with the above-mentioned special conditions.

The model of human-computer interaction for "novice use", which is a sub-model of the model in Section 3.2.2, is as follows:

1. Start the system if necessary.
2. Repeat the following (a through c) until there are no more tasks to do.
   (a) Map the next task to the effects to be achieved in the computer system.
   (b) Identify the actions and associated objects for achieving the effects, as well as the sequence to execute these actions.
   (c) Repeat the following (i through iii) until the task is completed.
      i. Execute the next step or mini-step.
      ii. Perceive the effect of the action.
      iii. Understand the progress made towards achieving the goal.
3. Terminate the system if necessary.

The characterizing questions for the user interface that relate to "novice use" are (this is a subset of the questions in Section 3.3.1):
Fundamentals:

I1. How can the user start to run the system?

I2. How can the user exit the system?

I3. What is the metaphor used in the user interface?

I4. Is on-line help always available?

I5. What is the number of working states?

I6. If there are multiple working states, how is the current state indicated?

Relating to tasks:

I18. For each task, what kind of objects, actions, and guidance does the user interface provide?

I19. What is the name or visual representation of each action?

I20. What is the syntax for executing each action?

I21. What visual cues are used for the objects, actions, and other?

I22. For each correct user operation, what is the reaction by the user interface?

Question I22 is tailored to only consider the correct user actions.

Relating to environment:

I23. What software and hardware support are needed to run the system, including hardware architecture, operating system versions, supporting software versions, free disk space, memory, screen resolution and colors, other input and output devices?

The following questions about the environment are chosen because they may have negative impact on novice use.

E1. What restrictions may the environment have on the use of certain interaction channels: the use of typing or mouse operation for input, the use of speech for input or output, etc.?

E2. What unusual requirements may the environment have on visual display: the contrast, the font size, the colors, etc.?

E3. What kind of hardware and software settings will the specified users have, including hardware architecture, operating system versions, supporting software versions, free disk space, memory, screen resolution and colors, other input and output devices?

The usability questions that follow are obtained by going through the sub-model of human-computer interaction and checking to see if the defined users can successfully accomplish the defined tasks. The criteria is whether there is sufficient support from one or more of the following:

- The instructions and online help (Explicit Guidance or EG);
- The visual or auditory cues (Implicit Guidance or IG);
- The user's previous knowledge of task domain and computer domain (External Consistency or EC);
- The user's knowledge obtained from using other parts of the system (Internal Consistency or IC).

The generic inspection questions for novice use are listed in Appendix A.1. Each numbered question comes from a step in the sub-model, and each inspection question under it comes from consideration of the related questions (I1, ..., U1,

..., T1, ..., E1 ...) and criteria (EG, IG, EC, or IC). To keep the questions focused on the main issues, the environment issues (E1, E2, and E3) are not checked in the list wherever they should be. If these issues are particularly important for a specific application, the checking of the environment issues should be added to every place where the user needs to perceive the interface or to execute an action.

Perspective 2: expert use

Assumptions:

- The user has sufficient knowledge and skill to carry out the task. The issues are 1) whether the human-computer interaction process is efficient, easy, and pleasant; and 2) whether productivity and flexibility are supported through advanced features and functions.

Usability Goals:

Goal 2.1 Users can complete each task in an efficient and easy way.

Goal 2.2 Users can customize the system to behave the way they desire.

Goal 2.3 There are advanced functions or features for advanced tasks.

Questions:

Now the users are constrained to those who know how to use the system. In addition to the fundamental tasks considered in novice use, we also need to consider the advanced tasks that only expert users are expected to accomplish.

The human-computer interaction model for "expert use", which is a sub-model of the model in Section 3.2.2, is as follows:

1. Start the system if necessary.

2. Repeat the following (a through c) until there are no more tasks to do.

   (a) Map the next task to the effects to be achieved in the computer system.

   (b) Identify the actions and associated objects for achieving the effects, as well as the sequence to execute these actions.

   (c) Repeat the following (i through iii) until the task is completed.

      i. Execute the next step or mini-step.

      ii. Perceive the effect of the action.

      iii. Understand the progress made towards achieving the goal.

3. Terminate the system if necessary.

This sub-model is the same as the sub-model for "novice use", except for the cycle of "map-identify-execute-perceive-understand", only the "execute," "identify," and "perceive" steps need to be inspected. In the "expert use" perspective, it is assumed that users do no have problem in the "map" and "understand" steps.

The characterizing questions for the user interface that relate to "expert use" are (this is a subset of the questions in Section 3.3.1):

Fundamentals:

I1. How can the user start to run the system?

I2. How can the user exit the system?

I7. To what extent can the user change the user interface?

For each window or dialog box:

I8. What user interface components are used in each screen/dialog box?

I9. What is the number of items in the menu tree?

I10. What is the number of levels of the menu tree?

I11. What colors are used?

I12. How are different colors used to present different information?

I13. What is the number of different colors used in each screen/dialog box?

I14. What font types and font sizes are used?

I15. How are different fonts used to present different information?

I16. What is the number of different fonts used in each screen/dialog box?

I17. How are the contents laid out?

Relating to tasks:

I18. For each task, what are the possible ways to accomplish it?

I22. For each user action, is there always immediate indication of either the completion of the operation or the progress that is being made ?

Question 18 is tailored to disregard the visual cue part and focus on the different possible ways to accomplish a task. Question I22 is tailored to only consider the correct user actions.

Related questions about the environment are:

E1. What restrictions may the environment have on the use of certain interaction channels: the use of typing or mouse operation for input, the use of speech for input or output, etc.?

E2. What unusual requirements may the environment have on visual display: the contrast, the font size, the colors, etc.?

E7. Do users need to enter confidential information?

The usability questions, listed in Appendix A.2, are obtained by going through the sub-model of human-computer interaction and checking to see if the goals defined under this perspective are achieved, considering the relevant user interface components and the set of tasks. Efficiency is achieved through consistency, shortcuts, default values, etc. "Ease of use" is at the motor, perceptual, and cognitive levels. The environment issues are dealt with in the same fashion as in "novice use."

Perspective 3: error handling

Assumptions:

- The user has done something wrong, or there has been a system failure.

Usability Goals:

Goal 3.1 The opportunities for user errors are minimized.

Goal 3.2 The user interface helps users understand the problem when user errors happen.

Goal 3.3 The user interface helps users recover after they have made errors.

Goal 3.4 System failures are dealt with appropriately.

Classes of user errors:

Using the "conceptual, semantic, syntactic, and lexical" model (Foley et al.[15]), the following classes of user errors are derived:

- Conceptual. The user makes mistakes because of the lack of fundamental concepts in the computer domain. For example, the user did not know what a mouse is used for; the user did not know that it was necessary to save a document for the recent changes to take effect.

- Semantic. The user makes mistakes because of not knowing the meaning of some object, action, or system feedback. For example, the user wanted to close the current document window but instead closed the whole application because of not knowing the meaning of the "Exit" command.

- Syntactic. This can be an omission or commission. The user uses the wrong syntax or omits something. For example, the user followed the "action-object" order when an "object-action" order was required.

- Lexical. This can be an omission or commission. The user makes mistakes in specifying an object or action. For example, the user spelled a command name wrong; the user selected the wrong object to delete.

The operational error classes used during the inspection are: wrong action (this can be the result of mistakes at conceptual level, semantic level, or lexical level), execution in wrong system mode (conceptual level), syntax error (syntactic level), missing task steps (a special case of omission at syntactic level), and slippage (lexical level). Besides user errors, a user interface also needs to deal with some system failures.

Questions:

For "error handling," the focus is on the mechanisms in the user interface to prevent errors, the message the user interface gives when an error occurs, and the ways in which error recovery is supported by the user interface. We need

to consider the different errors that different users may make when working on different tasks. We need to consider the environment factors that may increase the possibility of certain user errors.

The model of human-computer interaction for "error handling" is the same as the model described in Section 3.2.2, with focus on the error handling step (step 11):

- Identify the kind of error that has occurred and act accordingly:
  - For slippage: reverse the effect caused by the slippage; execute again;
  - For syntax error: reverse the effect caused by the wrong syntax (if there is any); find out the correct syntax; execute again;
  - For missing task step(s): back up several steps as necessary; start over by executing the first step that was missed last time;
  - For wrong action(s): reverse the effect caused by the wrong actions (if necessary); re-map the task to actions of the user interface;
  - For wrong system mode: reverse the effect caused by the action under the wrong mode (if there is any); switch to the "right" system mode; execute again;

The characterizing questions for the user interface that relates to "error handling" are (this is a subset of the questions in Section 3.3.1):
Fundamentals:

I5. What is the number of working states?

I6. If there are multiple working states, how is the current state indicated?

Relating to tasks

I18. For each task, what kind of objects, actions, and guidance does the user interface provide?

I19. What is the name or visual representation of each action?

I20. What is the syntax for executing each action?

I22. For each possible user error or system failure, what is the reaction from the user interface?

Question I22 is tailored to fit the error handling situation.
Related questions about the environment are:

E1. What kind of errors are more likely to occur because of the environment?

E6. What kind of errors are unacceptable for this application?

Both questions are tailored to the "error handling" situation. They affect the type of errors to check and the importance of different errors.

The usability questions, listed in Appendix A.3, are obtained by going through the model for human-computer interaction and checking to see if the goals defined under this perspective are achieved, considering the relevant user interface components and the set of tasks.

3.4 Inspection Scenarios

An inspection scenario consists of a perspective to be taken for the inspection, an operational procedure for carrying out the inspection from that particular perspective, and other contextual information.

For each inspection, an inspector will be provided with the following:

- A general description of the application domain and the functionality of the system.
- A description of the perspective for the inspection session.
- A description of the users and their working environment.
- One or more specific tasks to be carried out.
- A list of usability questions that are tailored specifically to the current project and are integrated into the inspection procedure. The questions may be associated with historical usability anomalies in similar user interfaces as examples.

The inspectors are asked to use the described perspective and follow the provided inspection procedure. The procedure guides the inspectors through one or more user tasks, and ask the inspectors to use the provided usability questions to detect usability anomalies.

3.4.1 Inspection Procedure for Novice Use

For usability inspection from the "novice use" perspective, inspectors are asked to think about the "novice users," who have the minimum amount of knowledge as indicated by the user profile and no prior experience of doing the task using the system. Based on this assumption, inspectors are going to ask questions as to whether a novice user can form the right intention, identify the right actions and objects, form the right task plan, execute each task step correctly, perceive the system feedback, and understand the progress towards achieving the goal.

The inspection procedure will first guide the inspectors to check the global questions (questions 1 and 2 of the usability questions for "novice use" in Section 3.3.5). Then the procedure will ask the inspectors to go through the tasks and for each task, checking questions 3 through 7.

3.4.2 Inspection Procedure for Expert Use

For usability inspection with respect to "expert use", the inspectors are asked to think about the expert users do not have trouble completing the task, but who are intolerant of inefficiency, inconvenience, or waste of time or energy while working with a computer.

Operationally, the inspectors are asked to work at two levels during the inspection. At a higher level, the goal is to determine whether the task steps the user interface requires are efficient, i.e., whether it is possible to have the task accomplished in fewer steps. At a lower level, the goal is to check each required task step, with regard to the usability questions provided. Inspectors are also asked to check the different parts of the user interface for consistency.

The inspection procedure will ask the inspectors to first check "expert use" question 1 (in Section 3.3.5), if it is relevant. Then the inspectors are to go through all the screens related to the task and check questions 3 to 9 and 12. Finally the inspectors are asked to work on the tasks and for each task check questions 2, 10, and 11.

3.4.3 Inspection Procedure for Error Handling

To inspect the usability with regard to errors, two aspects need to be analyzed: the types of errors that may occur, and how they are dealt with by the system.

The inspectors will be given the class of errors and asked to derive error cases for each task and task step. Then for each error case, they are to check the relevant usability questions.

The types of user errors include slippage, syntax error, omission, wrong action, and wrong system state. System failures include software or hardware failures, e.g. "help application" not available, floppy drive not ready, etc. For some applications these situations need to be handled by the user interface.

These error situations need to be dealt with in three respects: prevention, information, and correction. Errors should be prevented whenever possible. Then errors occur, the interface should communicate to the user in an understandable and constructive way. Facilities should be provided for users to recover from errors.

During the inspection, the inspectors first derive the possible user errors and system failures for each task. Then they check question 1, which relates to starting the system. Then they work on the tasks and check question 2 and 3 (if applicable). Finally they check question 4, which relates to exiting the system.

3.5 Generating Inspection Questions for a Specific Project

The usability questions generated in Section 3.3.5 can be instantiated to derive the questions for a specific usability inspection. That is done by:

- collecting the contextual information, including the characteristics of the user interface, the users, the tasks, and the users' working environment;

- prioritizing and tailoring the three perspectives based on the usability goals of the project;
- assigning tasks to each perspective;
- for each question, deciding whether it should be removed, made more specific, or kept unchanged, using the concrete information about the user interface, the users, the tasks, and the environment.

The rest of this section goes through these steps, using a specific interface as an example. The interface is a Web-based data collection form of the US Bureau of the Census.

Step 1. Collecting contextual information

In Section 3.3 a list of characterizing questions are defined for the user interfaces, users, tasks, and environments. For a specific project, information needs to be collected so that these questions are answered with as much detail as possible.

For the example user interface, the following information is collected. Some assumptions are made about the users (have access to the Internet and a graphical Web browser, know how to use a computer mouse, etc.) because the Web-based form is an alternative to the traditional paper form. It is assumed that people who choose this option have those characteristics.

Application Domain:

The purpose of the form is to collect demographic information. This is done by asking people from each household to fill out a form through a Web browser. Each form will collect information about all the people in the household such as name, gender, date of birth, race, etc.

User Interface:

For a user interface whose prototype is already implemented, some of the questions about the user interface can be easily answered by looking at the prototype. The answers to other questions may not be so obvious.

Fundamentals:

- Users start to run the system by typing the URL provided to them.
- Users finish using the data collection form by submitting it, which is done by clicking on the "Submit" button.
- A form metaphor is used which resembles a paper form.
- On-line help is available to explain each data item. The name of each data item has a link to the explanation.
- There is only one system mode between the user opening the form and exiting the form.
- Users can use the facilities provided by the Web browser to change the size of the window, the fonts, the appearance of the links, etc.

For each window or dialog box:

- The data collection form is a long page which consists of four parts: the introduction, household information, individual information, and afterword.
- The details of the colors, the fonts, and the layout can be seen from the user interface itself since the interface is already implemented and accessible through a Web browser.

Relating to tasks:

- A user needs to go through the four parts: the introduction, the household information, the individual information, and the afterword.
- For navigation, there is a scroll bar on the right that users can use to scroll up or down continuously; or users can use the PageUp or PageDown key to move up or down one page at a time; or they can use the up arrow or down arrow key to move up or down approximately one line at a time.
- To answer a question, users may either select one of the radio buttons in a group, or activate a text area and then type in the information, or both.
- More details can be obtained directly from the user interface.

Relating to environment:

- The user needs a computer with a graphical Web browser to access the data collection form.
- The user needs to use both a keyboard and a mouse to fill out the form.
- Occasionally the user may be required to make a phone call to report a special situation.
- Some data to be entered are confidential.

Users:

- Users know how to use the keyboard and mouse of a computer.
- Users do not have a handicap that prevents them from typing or using the mouse.

- Users may be color-blind, which would prevent them from discerning visual cues that are only color-coded.
- Users have experience in filling out paper forms of some kind.
- Users know how to access a Web site from the given URL.
- Users may not have all the needed information ready.
- Users all read English.

Tasks:

- Get to the Web site by specifying the URL to the Web browser.
- Read the instructions that explain what needs to be done.
- Answer the questions about the household and questions about each individual. Depending on the number of people in the household, the part of the form that the user needs to fill out varies.
- Submit the completed form.

Environment:

- For the point of view of the Bureau of the Census, all information must be collected as accurately as possible.
- Users who choose to use the on-line form have access to computers with an Internet connection and a graphical Web browser.
- Users may fill out the Web-based form either in the office or at home.

Step 2. Prioritizing and tailoring usability perspectives

Examine the usability goals defined in Section 3.3 against the project. Decide whether each of the goals applies to the project. Only inspection questions that are are related to an applicable goal are going to be used for the inspection.

For a specific project, some usability goals may be more important that others. In that case, more effort should be spent on the perspective that covers the more important usability goals.

The usability goals for the example interface are:

- A user with basic Web knowledge should be able to fill out the form successfully.
- Possible users errors that may affect data quality should be handled properly.
- Efficiency and ease of use should be achieved as much as possible.

The last goal has lower priority than the first two goals.

Step3. Assigning tasks to perspectives

A set of well-selected user tasks may provide a good coverage of the user interface. For some systems, the user tasks the system supports are a few and obvious. For others it takes some effort to derive a set of tasks for inspection. Some practical advice for task analysis can be found in [23].

Not every task needs to be included in inspections for all three perspectives. If a task will only be executed by expert users, it does not need to be included in the inspection for novice use. Tasks that can be completed by one step do not need to be included in the inspection for expert use.

Tasks are going to be provided to the inspectors together with the ways they can be accomplished in the user interface. For novice use, the most intuitive solution will be given. For expert use, the short-cuts, if available, will also be provided. For error handling, if the format of the artifact is not a running system, then it should be described how the user interface would respond to possible error inputs of the user.

For the example interface, there are no tasks that will only be executed by expert users. Tasks vary according to the size of the household. Inspectors of all three perspectives are given the same tasks and asked to inspect the user interface with different household sizes considered.

Step 4. Instantiating the questions

In the last two steps it has been decided which usability issues are important, which are less important but still relevant, and which are irrelevant for the project. It has also been decided which subset of tasks are going to be inspected for each perspective. The next step is to instantiate the general usability questions into more specific ones, based on the context (user interface, users, tasks, and environment) of the inspection. The instantiation is done by going through the usability questions under each perspective and possibly doing the following for each question:

- Eliminate the questions that are not applicable;
- Specialize some questions according to the context;
- Provide examples specific to the application domain or to the type of user interface.

Historical data about usability anomalies in similar user interfaces can greatly help this process.

The resulting inspection questions for the example interface are listed in Appendix B.

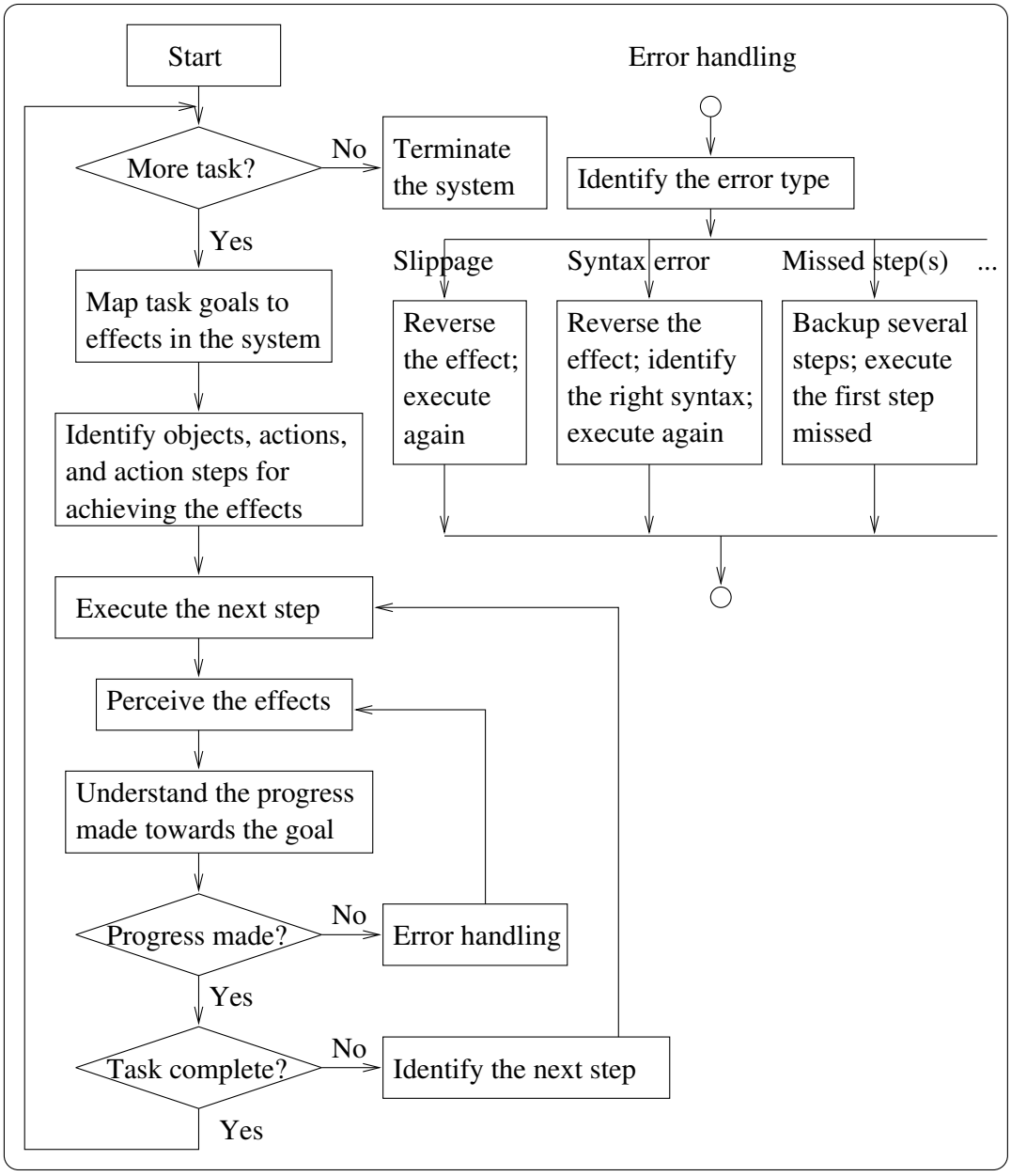

Figure 3.4: A process model of human-computer interaction

Chapter 4

Empirical Studies

One of the research requirements in developing the perspective-based inspection technique (Section 1.4) is to empirically study the technique. The following sections describe the research questions, the research hypotheses, three experiments that have been conducted, a survey of 8 usability practitioners, and the quantitative and qualitative results from these studies. Directions for further studies are discussed in the last chapter of the dissertation.

4.1 Goal

Using the GQM template, the goal of these studies is defined as:

Analyze perspective-based usability inspection for the purpose of evaluation with respect to detection of usability anomalies from the point of view of researchers.

The quality aspects of interest (i.e. "detection of usability anomalies") can be characterized by the following questions:

1. What percentage of usability anomalies are detected by inspectors using

the perspective-based usability inspection? (overall effectiveness by individual)

2. What percentage of different types of usability anomalies are detected by inspection using each perspective? (specific effectiveness by individual)

3. What percentage of usability anomalies are detected by an aggregation of inspectors using different perspectives? (overall effectiveness by team)

4. How much time is needed for the inspectors to detect defects using the technique? (objective feasibility)

5. What are the inspectors' subjective opinions about the technique? (subjective feasibility)

6. Does the technique fit well with the current usability inspection practice? (compatibility with practice)

4.2 Research Hypotheses

Relating to the above quality aspects, the research hypotheses are listed below. Here a "general technique" refers to an existing technique that gives each inspector the same general responsibility.

1. Compared with a general technique, assigning each inspector a specific individual responsibility will significantly improve the detection of anomalies covered by the responsibility.

2. Inspections with different perspectives will detect different subsets of usability anomalies.

3. Given the same number of inspectors with similar expertise, using the perspective-based technique will significantly improve the detection of anomalies over a general technique.

4.3 Overview of the Empirical Studies

To test the research hypotheses listed in the last section, and to empirically investigate the other quality aspects of perspective-based inspection, a series of experiments were conducted. These experiments had subjects ranging from students taking a human-computer interaction class, to professionals from one organization with different job categories, to professionals from different organizations and different background taking a usability engineering class. They either worked individually or in two-person teams. The experiment arrangements had subjects either work in one 100-minute inspection session with two different designs of the same interface and one task, or work in one 60-minute session with one interface and four tasks, or work in three 15-minute sessions with one interface and each session devoted to one task. One experiment was conducted in lab rooms where each room only had one subject at a time. The other two experiments were conducted in classrooms, where all subjects conducted the inspection at the same time.

A major source of limitation of this work was the availability of environments for running the experiments. Experiments usually require a larger number of subjects than an organization can provide as participants. Running experiments on live projects often faces constraints that the experimenter does not have power to control, as in the case of the first experiment in this work. Running experi-

ments without direct impact on a live project is less interesting to organizations, but may provide training benefits for both the organization and the participants. It is usually more feasible to have students taking a relevant class participate as subjects, as long as they have enough knowledge to understand the task domain and the user interface domain. One problem with this situation is that the subjects are participating as volunteers, therefore there is always the risk of potential subjects not showing up at the experiment, as happened to a certain extent in experiment II. Another problem of using students in a class as subjects is fairness in terms of the benefits students will receive from the experiment. In this work, each participant got to learn a usability inspection technique they had not been taught before. After the experiment, they were given the chance to understand the other techniques that were used in the experiment and to understand the experimental design and results.

The environment for running these experiments brought about various threats to validity, which will be discussed for each study. Although each experiment may have certain threats to validity, some conclusions can be drawn based on the consistent results from different runs and replications which compensate each other's weaknesses or have varied settings.

The technique was proposed mainly to improve the effectiveness of usability inspections by non-expert inspectors. The subjects were mostly not usability engineers, but had similar characteristics to people in organizations who may well be chosen to conduct usability inspections. To understand how this technique may fit and improve usability inspection practices by usability engineers, 8 usability practitioners from four companies were surveyed to understand their usability inspection practice and the get their opinion about the possibility of

conducting perspective-based inspections.

The results from these studies are described and discussed in this chapter. The limitations, implications, and open issues are discussed in the next chapter.

4.4 Experiment I

The first experiment was conducted at the US Bureau of Census in September 1997, with 24 professionals as subjects.

4.4.1 Method

This was an exploratory study in that it was a live project and the number of subjects we could get for running the experiment was limited. It was not practical to set a significance level beforehand and calculate the statistical power to determine the number of subjects to have. Also, the project required the evaluation of two interfaces under equal conditions. The experimental design had to balance the order in which the two interfaces were inspected, and thus introduced the interface order as an independent variable. Due to these constraints, the study did not have enough statistical power to test the hypotheses at the traditional 0.05 level. Also, this was the first time an experiment was run to study perspective-based usability inspection. Therefore I viewed this study as exploratory and used the 0.10 significance level, so that potentially interesting results would not be rejected by forcing a more rigorous significance level.

Experimental Design

I used a post-test only control group experimental design [8]. The control group used heuristic evaluation. The experimental group was further divided into three sub-groups, each assigned one of the three perspectives.

Each subject inspected two versions of a Web-based data collection form. One version was developed in HTML (referred to as interface A). The other version was developed in Java (referred to as interface B). For interface A, there were two variations: one with a menu bar and one without a menu bar. The different versions and variations were presented to inspectors in four possible orders (A/M is interface A with a menu bar; A/N is interface A with no menu bar; B is interface B):

1.	A/M	A/N	B
2.	A/N	A/M	B
3.	B	A/M	A/N
4.	B	A/N	A/M

As show in Table 4.1, in each of the three perspective groups one participant received each of the orders. In the heuristic group, three participants received each of the orders.

The 24 subjects in the experiment were familiar with both the interface domain and the task domain. They were either programmers, domain experts, technical researchers, or cognitive researchers. Efforts were made to evenly distribute participants of different backgrounds among the different groups. But due to some schedule changes, there were 5 programmers in the control group and 3 in the experimental group. The experimental group had 3 cognitive re-

Table 4.1: Layout of experiment I: number of subjects in each treatment

	Heuristic (12)	Novice (4)	Expert (4)	Error (4)
A/N, A/M, B	3	1	1	1
A/M, A/N, B	3	1	1	1
B, A/N, A/M	3	1	1	1
B, A/M, A/N	3	1	1	1

searchers while the control group had only 1. This imbalance will be discussed in the threats to validity.

Pilot Study

To test the instruments to be used in the experiment, a pilot study was conducted at the University of Maryland. Seven subjects participated in the pilot study, including 6 Ph.D. students and 1 assistant professor.

The pilot subjects only worked on interface A, with both variations (with or without the menu bar). Each subject was first given information about the data collection form and the technique to be used. Then they did the inspection without being observed. They reported usability anomalies on the anomaly report forms. After finishing the inspection, they were given a debriefing form to fill out. A short interview was conducted with each pilot subject after the inspection. The anomaly report forms and debriefing forms were the same as the ones used in the real experiment.

The anomaly detection performance of the pilot study subjects is summarized in Table 4.2. It shows the percentage of anomalies detected out of all anomalies, "novice use" anomalies, "expert use" anomalies, and "error handling" anomalies

Table 4.2: Percentage of anomalies detected by the 7 pilot subjects

Role	% of overall	% of novice	% of expert	% of error
Novice Use	24	35	9	7
(2 subjects)	13	22	18	21
Expert Use	18	13	36	14
(3 subjects)	15	13	27	14
	12	9	27	14
Error Handling	7	0	0	43
(2 subjects)	6	0	9	29

(Section 4.4.1 will describe how these lists were obtained). On average, each pilot subject for "novice use" detected about 28% of the "novice use" anomalies; each pilot subject for expert use detected about 30% of the "expert use" anomalies; and each pilot subject for "error handling" detected about 36% of the "error handling" anomalies.

Based on the comments made by the pilot subjects, some small changes were made in the presentation of the inspection materials, including highlighting keywords in the inspection questions.

External Expert Reviews

To help us better understand the existing usability anomalies of the interfaces, two external usability experts were asked to review the forms for usability. They were Chris North of the Human-Computer Interaction Lab of the University of Maryland, and Pawan Vora of the Intranet Technology Services Group of US WEST. They worked separately to detect usability anomalies based on their own

expertise. One expert reviewed both interfaces, while the other only reviewed interface A.

Overall they detected 30 usability anomalies for interface A, of which 7 were not detected by any of the 24 experimental subjects and the 7 pilot subjects. For interface B, the expert detected 15 usability anomalies, and 4 of them were not detected by any of the 24 subjects.

The unique anomalies found by one expert mainly came from his expertise and experience in designing for the Web. The unique issues he raised included creating the sense of security, providing "ALT" components for the graphics, using graphics carefully, and processing graphics in a professional way. The unique anomalies found by the other expert were mainly on consistency issues, where the inspector needs to be sensitive to the inconsistent look-and-feel of similar components in different places of the interface. On the other hand, the subjects raised more issues about the details of form design, which is the task domain that is more familiar to the subjects than to the experts. This implies that a good inspection needs to incorporate expertise in both the task domain and the interface domain, as well as good skills in doing the inspection, although every single inspector does not have to have all these capabilities.

Procedure for the Main Study

The experiment was conducted at the cognitive lab of the US Bureau of Census during the week of September 15-18, 1997. The cognitive lab can accommodate two participants at a time, each in a separate room. There is an observation room where observers can watch the subjects through one-way mirrors. Also there are two cameras for each lab room. One camera captures the screen of

the computer in the lab room. The other camera captures the subject's facial expressions and body movements. Both views can be seen on a display in the observations room and be recorded. There were 1 or 2 observers to manage each experiment session.

Step 1. Introduction

At first, some background information was given to the participants. They were given information about the data collection forms to be evaluated and were told that the goal of the evaluation was to find as many anomalies as possible so that the best possible interface can be built.

Then information was provided about the usability inspection technique to be used. Participants using the heuristic evaluation were given information about the usability heuristics, while participants using the perspective-based technique were given an overview of the usability perspectives and some instructions on how to do the inspection.

The subjects were told not to discuss the inspection with the other subjects before all participants had finished their inspection sessions.

These instructions were videotaped and played at the beginning of each experiment session. This way all participants in the same group received exactly the same instructions. After the video, the participants could ask questions. Then they were asked to sign a consent form.

Step 2. Inspection

Each subject was given the inspection material according to the inspection technique and perspective to be used. The material included the usability heuristics (in case of heuristic evaluation) or the inspection scenario (in case of perspective-based inspection), and the forms for subjects to report detected usability anoma-

lies.

The observer helped the subject start the first version of the user interface to be inspected. The observer was always available in the observation room to answer questions. The subjects were told to signal the observer when they were done with one version and ready to switch to the next.

The subjects were told that they had a total of 100 minutes for the two versions. They were told to plan on spending about 50 minutes on each version. Observers would give the subject a 5-minute warning when necessary. Rarely did the observers have to give such warnings.

The inspection sessions of all participants were videotaped. There were two views in the video. One view was on participant's facial and upper body activities, and the other was on the computer screen.

Step 3. Questionnaire

After each inspection session, the observer gave the subject a debriefing form with a list of questions. The questions asked included the subjective feeling and comments about the technique used, the subjective "guess" of what percentage of the existing usability anomalies had been found. It also included questions about their roles at work and their previous experience in user interface design and evaluation. When the subject was filling out the debriefing form, the observer would go through the anomaly report form to make sure that it was readable and clear. At the end, the observer asked the subject questions about the inspection session or the reported anomalies.

Materials

The usability heuristics and perspective-based inspection procedures used in the experiment are included in Appendix B.

The usability heuristics used were:

1. Speak the users' language
2. Consistency
3. Minimize the users' memory load and fatigue
4. Flexibility and efficiency of use
5. Use visually functional design
6. Design for easy navigation
7. Validation checks
8. Data entry
9. Provide sufficient guidance

Each heuristic included a detailed explanation of the related usability issues.

For the "novice use" perspective, inspectors were asked to think of novice users with a list of characteristics: being able to use a keyboard and a mouse, without visual disabilities, etc., which were defined based on the context of the application. Inspectors were given the description of the application and the user tasks. For each task, they were asked to think about whether a novice user would be able to choose the correct action, execute it successfully, and understand the outcome. They were provided with a list of detailed usability questions. For example, for data entry:

Are formats for data entries indicated?

For "expert use", inspectors were asked to think about expert users and check the interface for efficiency, flexibility, and consistency in supporting the user tasks. They were given a list of usability questions relating to these issues. For example, for data entry:

Are possible short-cuts (e.g. using the Tab key to switch to the next field) available?

Are possible default values used?

For "error handling", inspectors were given a classification of user errors. They were also given the characteristics of the users as the "novice use" inspectors were. For each user task, inspectors were asked to list the possible user errors and check the following questions for each user error:

Does the user interface prevent the error as much as possible?

Does the user interface minimize the side effects the error may cause?

When the error occurs, will the user realize the error immediately and understand the nature of the error from the response of the user interface?

When the error occurs, does the user interface provide guidance for error recovery?

Independent and Dependent Variables

The primary independent variable was the inspection technique. But another independent variable, the interface order, was introduced in the experimental design. Statistical tests failed to reveal a significant interaction effect between the inspection technique and the interface order, as shown in Table 4.5.

The dependent variables can be classified along two dimensions: objective vs. subjective, and quantitative vs. qualitative.

- **Objective and quantitative.** This category includes the number of usability anomalies identified by each individual inspector and by simulated teams, the number of each class of usability anomalies identified by each individual inspector, and the time taken by each inspector to do the inspection.

- **Objective and qualitative.** This category includes the information about how the inspectors carried out the inspection, from the observation record and reviewing the videotapes.

- **Subjective and quantitative.** This category includes the subjective rating of the easiness of the techniques and the subjective "guess" of the percentage of usability anomalies that had been detected.

- **Subjective and qualitative.** This category includes the comments made by the subjects about the techniques.

Among the dependent variables, the number of each class of usability anomalies detected was the most important as it showed the ability of each inspection technique to focus on a particular class of usability anomalies and suggested benefits for a team of inspectors to use different perspectives.

Data Coding

Step 1 A list of usability issues raised by each inspector

After the experiment, we went through the usability report forms and built a master list of detected usability issues for each interface. For each issue raised by an inspector, if it did not exist in the current list of issues, a unique number would be assigned. The issue would be added to the master list under that unique number. The number would then be written down on the inspector's anomaly report form. If the same issue had been raised before, then just the number of that issue would be written on the appropriate place in the anomaly report form.

The same procedure was followed to process the usability issues raised by the 7 subjects in the pilot study and the 2 external expert reviewers.

In this way a list of usability issues raised by each inspector was obtained. A list of usability issues for each interface was obtained in the same way.

Step 2 A list of all detected usability anomalies for each interface, categorized by severity level

Nielsen's rating scale [37] was used to assign severity levels to the usability issues. The rating scale is as follows:

- 0 – This is not a usability anomaly.
- 1 – Cosmetic anomaly only, need not be fixed unless extra time is available.
- 2 – Minor usability anomaly, fixing this should be given low priority.
- 3 – Major usability anomaly, important to fix, so should be given high priority.

Table 4.3: Detected usability anomalies by severity (experiment I)

Interface	Total	Cosmetic	Minor	Major	Catastrophe
A	82	19	36	26	1
B	61	16	13	22	10

- 4 – Usability catastrophe, imperative to fix this before product can be released.

After severity levels were assigned, the issues that had severity rating of 0 were removed from the list. In cases where two issues were recognized to be the same, it would be recorded that the corresponding two numbers referred to the same anomaly and one of them would be removed from the list. The final list of usability anomalies detected by each inspector was obtained by removing the ones that were not usability anomalies, changing the numbers of the ones that were removed from the overall list because they were the same as others in the list, and removing any duplicates. An example usability catastrophe was "There is no confirmation with the Reset button. User can lose all the data entered by accidentally clicking that button." An example issue that was regarded not to be a usability anomaly was "The button Check Entries is redundant" even though the button had a purpose.

The list of usability anomalies for each interface was obtained after removing from the accumulated list the duplicates and the ones that were regarded as not usability anomalies.

The detected usability anomalies are summarized by severity in Table 4.3. The statistical results were similar whether or not the severity ratings of the usability anomalies were considered. Therefore the results reported below ignore

Table 4.4: Detected usability anomalies by category (experiment I)

Interface	Total	Novice	Expert	Error	Heuristics	Other
A	82	23	11	14	47	33
B	61	20	16	11	37	20

the severity ratings.

Step 3 Usability anomalies under each category

For the purpose of comparing the percentage of anomalies each inspector detected within the responsibility, we went through every usability anomaly to see if it is covered by the heuristics, the novice use perspective, the expert use perspective, and the error handling perspective. Each anomaly can be covered by more than one of them. If a anomaly is not covered by any of the above, it is placed in the "other" category. For example, the anomaly "No format is specified for date of birth" was classified as covered by "novice use;" the anomaly "The age should be automatically calculated based on the date of birth" was classified as covered by "expert use;" and the anomaly "No error checking for inconsistent input of date of birth and age" was classified as covered by "error handling."

The summary of detected usability anomalies by category is given in Table 4.4

Table 4.5: Effect of independent variables on overall detection effectiveness in experiment I (p-values from ANOVA)

Source	Order	Technique	Order × Technique
Interface A	0.50	0.19	0.23
Interface B	0.71	0.15	0.76
Both A & B	0.76	0.19	0.48

4.4.2 Results and Discussion

Individual Detection Effectiveness for All Anomalies

As stated before (Section 4.4.1), there were two independent variables: the interface order (interface A first or interface B first) and inspection technique (heuristic evaluation or perspective-based inspection). ANOVA was run to test the effect of each of these two variables as well as their interaction on the individual detection scores on interface A and on interface B. MANOVA was run to test these effects when the detection scores on the two interfaces by each inspector were considered at the same time. The detection score here is the number of usability anomalies detected.

The results of the ANOVA and MANOVA tests, as shown in Table 4.5, failed to reveal a significant effect by the interface order. For the inspection technique, there was also no significant effect shown (p=0.19 for interface A, p=0.15 for interface B, and p=0.19 for both). The interaction between inspection technique and interface order was found to be non-significant, which means that the effect of each of these two variables can be tested separately.

Another way to deal with the order effect is to compare the performance of

the two techniques on each interface when only the subjects who reviewed the interfaces in the same order are considered. Thus four t-tests were performed and the results are given in Table 4.6. It shows the average detection effectiveness in terms of the percentage of anomalies detected, as well as the p-values when the means from the two techniques are compared. In all cases, the perspective-based technique performed better than the heuristic technique, although in only one of the four situations was there a statistically significant difference (at 0.10 level). The sample size in each of these tests is 6 data points in each group, which is half of the subjects.

For interface A, the perspective-base technique performed much better than the heuristics technique when interface A was inspected first, while the two techniques performed almost the same on interface A when interface B was inspected first. Interface A was developed in HTML and had a "standard" look-and-feel that was familiar to all the subjects. Interface B was developed in Java and had an "ad hoc" interface that was much unfamiliar to the subjects. Therefore, this may indicate that late in the inspection process, when the artifact being inspected was familiar to the inspector, the inspector may tend to ignore the inspection technique being used and fall back to his/her own way of doing the inspection. Thus the effect of the techniques tend to diminish in such situations.

As evidence about how the subjects followed the assigned techniques, observation records and video-recordings show that most subjects read the instructions for the technique at the beginning. Some of them referred to it several times in the first 20 minutes. Almost nobody looked at the instructions again for the second interface. It is possible that they understood the general idea of the technique after a while. But it is unlikely that they had remembered the

Table 4.6: Percentage of anomalies found for each interface-order situation in experiment I

Interface	Order	Heuristic	Perspective	p-value
A	A-B	8.0	11.8	0.07
A	B-A	8.8	9.0	0.46
B	A-B	9.5	14.3	0.12
B	B-A	9.3	12.5	0.20

(N=6 for each cell)

specific usability issues and the inspection procedures.

In summary, when data from all 24 subjects (with two independent variables) were considered, the inspection techniques did not have a significant effect on the detection of overall anomalies, as shown in Table 4.5. When only half of the subjects who reviewed the two interfaces in the same order were considered each time, for the subjects who reviewed interface A first, perspective-based inspection performed significantly better than heuristic evaluation (p=0.07). There was not a statistical significance for other situations.

But the perspective-based technique asks each inspector to focus on a subset of issues. Therefore each perspective inspector is not expected to find more overall anomalies. Our hypotheses were that individual perspective inspectors should find more anomalies related to their assigned perspectives, and that the combination of inspectors using different perspectives should be more effective than the combination of the same number of heuristic inspectors.

It is surprising that, as shown in Table 4.6, perspective inspectors outperformed the heuristic inspectors at the individual level for overall anomalies (al-

though the differences were not statistically significant for 3 out of 4 cases). This is consistent with results from two other studies [3, 48] where inspectors with a focused responsibility detected more overall defects when reviewing software requirement documents.

Individual Detection Effectiveness for Different Types of Anomalies

Hypothesis 1 (Section 4.2) about the perspective-based technique was that compared to inspectors using heuristic evaluation,

- Inspectors using the "novice use" perspective would detect a significantly higher percentage of anomalies related to the "novice use" perspective.
- Inspectors using the "expert use" perspective would detect a significantly higher percentage of anomalies related to the "expert use" perspective.
- Inspectors using the "error handling" perspective would detect a significantly higher percentage of anomalies related to the "error handling" perspective.

First, ANOVA and MANOVA tests were run to test the effect of technique (four levels: heuristic and the three perspectives), interface order (two levels: interface A first or B first), and their interaction on the detection of anomalies covered by each perspective. Each test involved data from all 24 subjects. The interface order and the interaction between the technique and order were found to have no significant effect in any case. Table 4.7 shows the effect by technique. It shows a significant effect of inspection technique on the detection of "novice use" and "error handling" anomalies. The use of the "expert use" perspective did not have a significant effect, possibly because the inspectors themselves were

Table 4.7: The effect of technique on the detection of anomalies by category in experiment I (p-values from ANOVA)

Category	Interface A	Interface B	Both
Novice	0.065	0.043	0.044
Expert	0.42	0.29	0.40
Error	0.039	0.044	0.032

all experts in the application domain and user interface domain. Thus they were able to capture a large portion of the "expert use" anomalies even without help from the "expert use" perspective.

Then 3 ANOVA tests were run between the heuristic group and each of the 3 perspectives, with both the technique and order variables considered. Each ANOVA test involved data from 16 subjects (12 from heuristic evaluation and 4 from a perspective sub-group). Table 4.8 shows the results of these tests. For usability anomalies related to each perspective, the average percentage of such anomalies detected by the 4 inspectors using that perspective and the average percentage by the 12 heuristic inspectors are listed. The standard deviations are in parentheses. It shows that the use of the "novice use" and "error handling" perspectives significantly improved the inspector's detection effectiveness for anomalies related to those perspectives.

In summary, the results of this analysis supported the hypotheses for both "novice use" and "error handling" perspectives. The "novice use" inspectors found significantly more anomalies related to novice us than the heuristic inspectors. The "error handling" inspectors found significantly more anomalies related to user errors than the heuristic inspectors. But there was not a statis-

Table 4.8: Comparison of different types of anomalies found in experiment I

	Category	% of anomalies by 12 heuristic subjects	% of anomalies by 4 perspective subjects	p-value
A	Novice	8.0(6.6)	18.5(9.0)	0.025
	Expert	15.9(10.3)	20.5(8.7)	0.477
	Error	14.3(12.2)	33.9(6.8)	0.012
B	Novice	11.7(9.1)	26.3(11.1)	0.019
	Expert	14.9(11.1)	28.0(18.5)	0.134
	Error	9.0(10.9)	29.3(13.5)	0.013

(standard deviations are in parentheses)

tical significance for the "expert use" perspective. A possible reason of this was given in the above discussion.

Subjective Ratings

During the debriefing, the subjects were asked to give two subjective ratings: the rating of what fraction of the anomalies they think they had detected (on a 1–9 scale, with 1 being almost none and 9 being all) and the rating of "ease of use" of the technique that was used (on a 1-9 scale, with 1 being very difficult and 9 being very easy). The results of these subjective ratings are given in Table 4.9. The numbers in parentheses are standard deviations. Part of the reason that the inspectors found perspective-based inspection harder to use might be that in this experiment the materials for each perspective were presented on several pages. During the inspection, inspectors had to flip through several pages in order to follow the technique. This may have created some difficulty.

Table 4.9: Subjective ratings of the techniques (experiment I)

Rating	Heuristic	Perspective	p-value
Coverage	5.33(1.37)	4.50(1.68)	0.098
Ease of Use	6.08(2.23)	4.0(1.95)	0.012

(N=12 for each cell)

Table 4.10: Correlation between experience and inspection performance for the 24 subjects in experiment I

	Use Experience	Develop Experience
Anomalies for A	-0.416	0.010
Anomalies for B	-0.187	0.194
Time for A	-0.175	0.067
Time for B	0.339	0.316

Correlation Between Experience and Performance

Table 4.10 shows the correlation coefficients of the inspectors' experience in using the Web or developing for the Web and the number of anomalies found as well as the time spent doing the inspection. There was no strong correlation between experience and inspection performance.

Aggregation of 3 Inspectors

One hypothesis was that perspective-based inspection would significantly improve the performance of groups of inspectors, or inspection teams. Although the techniques used in this experiment did not involve any team work, simulated teams were constructed to test this hypothesis. This section and the next section

Table 4.11: Aggregated anomalies found by 3 inspectors (all possible aggregations, standard deviations are in parentheses)

Interface	Technique	% of anomalies found	Improvement
A	Heuristic	21.8(5.0)	
	Perspective	27.7(4.4)	26.5%
B	Heuristic	24.1(7.2)	
	Perspective	32.8(7.4)	35.7%

present results for these simulated teams.

First, I compared the number of unique anomalies identified by 3 perspective inspectors (one from each of the three perspectives) and 3 heuristic inspectors. The results are shown in Table 4.11. There were 220 possible aggregations for heuristic evaluation and 64 for perspective-based inspection. Since the data points in each group were not independent from each other, no statistical test was performed.

Permutation Test of All Possible 12-person Aggregations

I did a permutation test [13] of simulated 12-person teams. This involves constructing all possible 12-person teams and see how the un-diluted perspective team ranked among all possible 12-person teams in terms of number of unique anomalies detected. Whether or not we can claim that the perspective-based technique had a beneficial effect on team performance depends on how the un-diluted perspective team (with all 12 perspective inspectors) appears towards the top of the ranking. The p-value is the rank of the un-diluted team divided by the total number of teams. There were 2,704,156 possible 12-person teams

Table 4.12: Permutation tests for all possible 12-person teams

	Number of possible teams	Rank of the perspective team	p-value
A	2,704,156	262,577	0.097
B	2,704,156	122,993	0.045

out of the 24 subjects. The results of this test are given in Table 4.12. It shows that at $p < 0.10$ level, the perspective-based inspection technique significantly improved the effectiveness of an inspection team.

Overlapping Among Anomalies Detected by Perspective Sub-groups

This analysis looked into the overlapping of anomalies detected by each perspective sub-group. As shown in Figure 4.1, the number in a circle slice represents the number of usability anomalies uniquely detected by the combination of 1, 2, or 3 perspective sub-groups, depending on whether the circle slice is occupied by 1, 2, or 3 full circles of the three perspectives. For example, for interface B, there were 6 anomalies that were detected by all three perspectives, 4 detected by novice and error perspectives but not by expert perspective, and 15 detected by novice perspective alone. Although there is no other data to compare against at the moment, it shows that for both interfaces the different perspective sub-groups detected fairly different usability anomalies.

Major Anomalies Detected Only by One Technique Group

In this analysis, we went through all the detected anomalies that were ranked 3 (major usability anomaly) or 4 (usability catastrophe) and counted how many

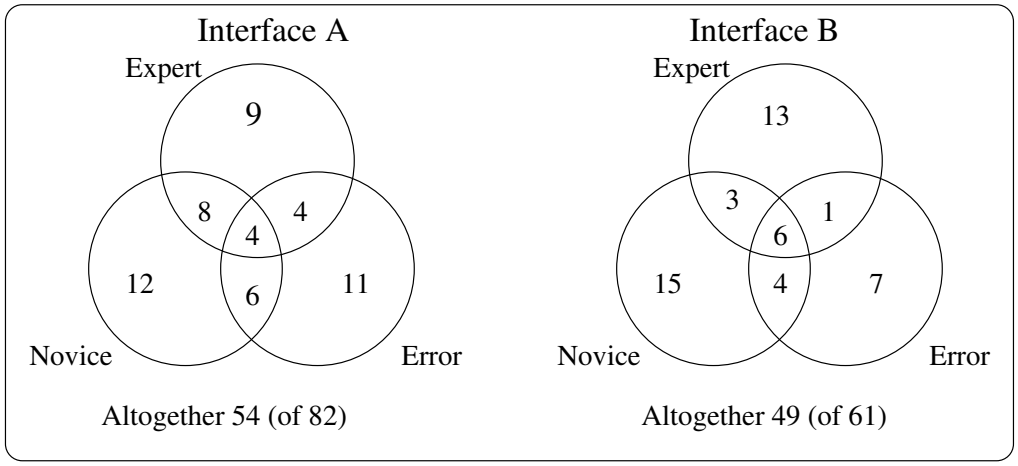

Figure 4.1: Overlapping of anomalies detected by different perspectives

unique anomalies were detected by only one of the two technique groups (the control group and the experiment group). For interface A, heuristic inspectors (control group) did not find any unique anomalies, while perspective inspectors (experiment group) detected 9 unique anomalies a total of 16 times (i.e. some anomalies were detected by more than one perspective inspectors). Of these 9 anomalies, 4 were detected by only one inspector; 3 were detected by two inspectors; and 2 were detected by three inspectors. For interface B, each technique group detected 4 unique anomalies. But each of the 8 unique anomalies were only detected by one inspector. This shows that giving inspectors specific responsibilities did not make them less effective in detecting major usability anomalies.

Table 4.13: Major anomalies uniquely detected by only one technique

Interface	Heuristic	Perspective
A	0 (0)	9 (16)
B	4 (4)	4 (4)
Both	4 (4)	13 (20)

Results from Visualizing the Data

Two information visualization tools, Spotfire[1] and XGobi[2] were used to visualize the data from this experiment. This analysis is to visualize the experiment data in different views (such as the one shown in Figure 4.2) to discover interesting information.

Observations made from the visualization that were not obtained from statistical analyses include:

- For interface A, 6 heuristic inspectors found fewer anomalies than all the other inspectors.
- For interface B, 4 heuristic inspectors found fewer anomalies than all the other inspectors.
- For interface A, 6 perspective inspectors and 2 heuristic inspectors found more than 10 anomalies.
- For interface B, 3 perspective inspectors and 3 heuristic inspectors found more than 10 anomalies.

[1]Spotfire is a product of Spotfire Inc. (http://www.spotfire.com/).

[2]XGobi is developed by Deborah F. Swayne and Andreas Buja of AT&T Labs and Di Cook of Iowa State University.

Figure 4.2: A Spotfire view showing the amount of time spent inspecting interface B and the job category of the subjects (1=programmers, 2=domain experts, 3=technical researchers, 4=cognitive researchers)

- For interface B, 5 heuristic inspectors spent less time than all the other inspectors. No such matter can be found for inspection time on interface A.

- For interface A, which is familiar to the inspectors, among the four job categories, programmers found the least number of anomalies while subject matter experts found the most number of anomalies.

- For interface B, which is unfamiliar to the inspectors, cognitive researchers

found more anomalies than inspectors of other categories.

4.4.3 Threats to Validity

Threats to validity [8] are factors other than the independent variables that can affect the dependent variables. Such factors and their possible influence are discussed in this section.

Threats to Internal Validity

The following threats to internal validity are discussed in order to reveal their potential interference with the results:

- **Maturation:** In this experiment, the whole inspection took up to 1 hour and 40 minutes, with no break. The likely effect would be that towards the end of the inspection session, the inspector would tend to be tired and perform worse. Also since the two interfaces had the same content, it is likely that for the second interface inspected, the inspector would get bored and not do an inspection as thoroughly as before. However, from observation records of the experiment, there were no signs showing that the subjects looked tired or bored. The experimental design let half of the subjects inspect interface A first while the other half inspect interface B first. The two interfaces differed to a large extent in terms of look and feel, which helped to keep the subjects interested. An ANOVA test failed to show a significant effect of the order on individual performance.

- **Testing:** Getting familiar with the material and the technique may have an effect on subsequent results. This is potentially a threat to this experiment since each subject used the same technique for both interfaces,

and that the two interfaces had the same content with different presentations. The experimental design had exactly the same number of subjects within each technique group inspect the two interfaces in two different orders. This should counter-balance some of the effect both between the two groups and within each group.

- **Instrumentation:** In this experiment, the decisions in data coding were made by a group of three people through meetings and discussions. These decisions included whether an issue raised by an inspector is a usability anomaly, what severity level should be assigned to each anomaly, and whether a particular anomaly is covered by the heuristics and any of the three perspectives. It might be better to have each person do it separately, and have meetings to see how consistent they are and to resolve the differences.

- **Selection:** As stated before, we tried to balance the number of subjects of different job background between the control group and the experimental group. The number of domain experts and technical researchers were balanced between the two groups. But due to some unexpected schedule change, the control group had 5 programmers and 1 cognitive researcher. The experimental group had 3 programmers and 3 cognitive researchers. This imbalance may have contributed to the differences between the two groups.

- **Process conformance** Another threat is that people may have followed the techniques poorly. For heuristic evaluation, the introduction video read through all the heuristics and the related usability issue of each heuris-

tic. The inspectors had these heuristics and issues with them during the inspection. For perspective-based inspection, the introduction video described the idea of doing inspection from three different perspectives and mentioned briefly the usability issues under each perspective. Almost all subjects in the perspective group read through the provided instruction thoroughly before the inspection. But some subjects in the perspective group reported that they did not follow the technique well or could not follow the technique since "it would take too long" or "I don't fully understand it". Given the 2-hour limitation we were not able to provide better training and make sure all subjects understood and felt comfortable with applying the technique. Also the inspection procedure for each perspective as given in this experiment appeared to be too detailed and somewhat intimidating. Given the time limitation, it may have not been practical to literally follow the procedure. But we believe all subjects in the perspective group got the general idea about the perspectives and the usability issues. Most of them tried to follow the technique and focus on the assigned perspective. The different techniques asked different inspectors to conduct the inspection in different ways. If the process conformance had been better, the differences between the different technique groups should be larger, and thus achieving better experimental results.

Threats to External Validity

One possible threat to external validity is:

- **Reactive effects of experimental arrangements.** In this experiment, we did not tell the subjects that we were comparing two inspection tech-

niques. The subjects only knew that they were supposed to use the assigned technique to detect as many usability anomalies as they could. We asked the subjects not to discuss with other subjects what they had done during the inspection until all subjects had finished participating in the experiment. Our impression was that the subjects were more interested in finding usability anomalies than using the techniques. The lab environment kept them concentrated on the inspection without distraction or interruption. The awareness that they were observed by others and video recorded may have some impact on their behavior. But since all these apply to both technique groups in the same way, they might not make a significant difference on the relative performance of the two techniques.

4.4.4 Summary and Guidance for Further Studies

The results from experiment I with 24 professionals, as analyses have shown so far, are summarized as below:

- Overall individual performance (i.e. the number of all usability anomalies detected by each inspector) for the perspective technique was better than for the heuristic technique. (See Table 4.6.)

- As to the detection of each class of usability anomalies, "novice use" and "error handling" perspectives significantly improved the detection of the related anomalies. "Expert use" perspective did not have a significant effect. (See Tables 4.7 and 4.8.)

- Consistent with the hypothesis, using the perspective-based inspection significantly (at 0.10 level) improved performance of simulated 12-person

teams. (See Table 4.12.)

- For interface A, the perspective inspectors detected 9 major usability anomalies (a total of 16 times) that were not detected by any heuristic inspector. For interface B, inspectors on each technique group detected 4 unique major anomalies, but each of these 8 anomalies was only detected once, i.e., by one inspector.

In summary, the results from the experiment showed that perspective-based inspection performed better than heuristic evaluation. The results from the pilot study showed even more promising performance of perspective-based inspection. This supports the belief that if we had run the experiment in a better way (provide better training session, match inspector expertise with the perspective to be used, etc.), the results could have been better.

The results from the first experiment and the threats to validity provided the following guidelines for further studies:

Inspection length To reduce the possible effect of maturation, the inspection time should be shorter. We decided to have each inspection session last for about an hour.

Process conformance To achieve better process conformance, we considered having inspectors work in pairs and ask one of them to focus on process conformance. For individual inspectors, we would provide a page of the inspection procedure once for each task and suggest inspectors to check off items and make sure that the inspection procedure is followed.

Usability coverage We would make sure that the usability issues covered in the heuristics are the same as the combined issues covered in the three

perspectives.

4.5 Experiment II

The second experiment was conducted in Spring and Summer 1998 at the University of Maryland.

4.5.1 Method

Based on the number of subjects available and statistical power analysis, the significance level was set to be at 0.05 to test the hypotheses listed in Section 4.2.

Experimental Design

In this experiment, the control group used heuristic evaluation. The experimental groups used the "novice use" or "expert use" perspective. The "error handling" perspective was not used for two reasons. One was that the artifact to be inspected did not have a significant number of usability anomalies related to user errors. Another reason was that with the limited number of available subjects, if we had one more experiment group, the number of data points in each group would be too few to have enough statistical power for testing the hypotheses at the defined level.

Most of subjects (30 out of 44) were gradate or upper-level undergraduate students taking an Human-Computer Interaction (HCI) class in the Spring 1998 semester. The original intention was to have students work in pairs as much as they could. But because some students did not show up (as they were participating on a volunteer basis), 8 subjects ended up working on their own. After that,

Table 4.14: The number of data points collected in experiment II

	Heuristic	Novice	Expert
2-person	6	6	6
1-person	4	0	4

6 graduate computer science students were recruited. They all had knowledge about usability. Also recruited were 8 students who were taking another HCI class in Summer 1998.

The data points collected with all these subjects are shown in Table 4.14.

The subjects from the two HCI classes conducted the inspections in their classrooms. The other subjects conducted the inspections one pair at a time, outside the classroom.

Procedure

The subjects were asked to inspect several portions of a Web site by going through 4 tasks: an Internet tutorial, an on-line help, finding specific information by browsing, and a search facility. They were given a one-hour time limit for the inspection, with suggested time provided for each of the 4 tasks. After the experiment, each subject was asked to fill out a questionnaire.

For 2-person teams in all three groups, one person was asked to monitor the inspection process to make sure that the technique was followed. The same worksheet of the technique was provided once for each task, so that the subjects could check off items on the page to keep track of their progress for each task.

Materials

In preparing the experiment material, it was made sure that the usability issues covered in the heuristics (with detailed issues under each brief heuristic) were exactly the same as the union of the issues covered by the "novice use" and "expert use" perspectives. Another change made to the experimental material was that the background information was presented on a separate page. The inspection procedure and questions were presented on a separate sheet of paper, so that the subjects could hold the one-page work sheet during the inspection, as opposed to flipping between several pages as the earlier subjects had done.

Independent and Dependent Variables

There were two independent variables: the inspection technique (heuristic evaluation, novice use, or expert use) and the team size (1 or 2). Since there were rather few data points for the individual inspection cases (i.e. with team size being 1), only anecdotal information was obtained regarding the team size variable. Quantitative analyses were done only on the data points with the team size being 2. In those analyses the inspection technique was the only independent variable.

The dependent variables were the number of anomalies detected by each individual inspector or pair of inspectors (overall and under each categories), the subjective rating of the inspection technique by each inspector, the subject's self-report as to how well the technique was followed, and the reported extra time needed to complete the inspection.

Data Coding

Data were processed and coded in a similar way as for experiment I. I worked with one person who was in charge of the interface part of the Web site to decide on the severity of each issue raised by the subjects. We used the same severity scheme used in experiment I, where severity 0 means the issue is not a usability anomalies, while severity 4 means the issue is a usability catastrophe, etc.

4.5.2 Results and Discussion

Altogether 64 usability anomalies were detected, of which 18 were covered by the "novice use" perspective, and 46 by the "expert use" perspective.

Results of the Subjects Who Did Individual Inspection

The performance of the 8 individual inspectors is summarized in Table 4.15. It shows the number of overall anomalies detected, the percentage of "novice use" anomalies detected, and the percentage of "expert use" anomalies detected. The standard deviations are in parentheses.

Results of the Individual 2-person Teams

The results of the 18 2-person teams were statistically analyzed. The data are summarized in Table 4.16. These results are based on the number of anomalies detected. The results did not change significantly when the severities of the anomalies were considered.

ANOVA and post hoc analyses were applied to the performance of 2-person teams (6 data points for each technique), with the inspection technique as the single factor. The results are summarized in Table 4.17.

Table 4.15: Means and standard deviations for individual inspectors in experiment II

	# of all anomalies	% of novice anomalies	% of expert anomalies
Heuristic evaluation	5.00 (1.63)	12.50 (3.00)	6.00 (3.92)
Expert use	9.75 (3.77)	8.75 (5.50)	17.75 (7.80)

(N=4 for each row)

Table 4.16: Means and standard deviations for 2-person teams in experiment II

	# of all anomalies	% of novice anomalies	% of expert anomalies
Heuristic evaluation	6.17 (3.66)	9.50 (4.42)	9.83 (6.74)
Novice use	7.67 (2.16)	23.00 (2.45)	7.67 (5.13)
Expert use	14.33 (3.72)	8.50 (7.77)	27.83 (8.66)

(N=6 for each row)

Table 4.17: ANOVA and t-tests for the 2-person teams in experiment II

	ANOVA	Post hoc analyses at p=0.05 level
All anomalies	0.0013	Expert > Heuristic Expert > Novice
Novice anomalies	0.0004	Novice > Heuristic Novice > Expert
Expert anomalies	0.00026	Expert > Heuristic Expert > Novice

(N=6 for each of the 3 treatments)

For the number of all anomalies detected, ANOVA showed a significant difference. Post hoc analyses showed that "expert use" was significantly better than both heuristic evaluation and "novice use." "Novice use" was not found significantly different from heuristic evaluation.

For the number of "novice use" anomalies detected, ANOVA showed a significant difference. Post hoc analyses showed that "novice use" was significantly better than both heuristic evaluation and "expert use." "Expert use" was not found to be significantly different from heuristic evaluation.

For the number of "expert use" anomalies detected, ANOVA showed a significant difference. Post hoc analyses showed that "expert use" was significantly better than both heuristic evaluation and "novice use." "Novice use" was not found to be significantly different from heuristic evaluation.

As to major usability anomalies detected, perspective inspectors detected 10 unique anomalies that were not detected by any heuristic inspectors. These 10 anomalies were detected a total of 22 times combined. All the major anomalies

detected by heuristic inspectors were also detected by one or more perspective inspectors.

The results from the 8 individual inspectors were consistent with all the above results for 2-person teams, although there were too few data points for meaningful statistical results.

Aggregated Results of Two 2-person Teams

For the combination of two teams, there are 36 possible perspective combinations (with one "novice use" team and one "expert use" team). There are 15 possible heuristic combinations out of the 6 heuristic teams.

The perspective combinations (4 inspectors) found a range of 16 to 27 unique anomalies (the mean was 19.83), with 0 to 6 anomalies found by both teams (the mean was 2.17). On average, about 11% of the anomalies in the combined list were found by both teams. The heuristic combinations (4 inspectors) found a range of 3 to 16 anomalies (the mean was 10.27), with 1 to 5 anomalies found by both teams (the mean was 2.07). The improvement of perspectives over heuristic evaluation was about 90% for 4 inspectors.

Subjective Ratings

In the post-experiment questionnaire, subjects were asked to rate the ease of use of the inspection technique they used in the experiment. The rating was on a 1 to 5 scale, with 5 being "very easy" and 1 being "very difficult". For heuristic evaluation, the average rating of the individuals was 2.75, while the average of the 2-person teams was 3.46. For "expert use", the average rating of the individuals was 3.5, while the average of the 2-person teams was 3.96.

The average rating for "novice use" teams was 3.5. The ratings by teams were better than those by individuals. The ratings of perspective techniques were better than those of heuristic evaluation, but with no statistically significant differences (p=0.35 from one-way ANOVA on the pairs).

Also in the questionnaire there was a question "to what extent have you followed the technique". There were 3 options (1 – I ignored it. 2 – I followed it in general but not step-by-step. 3 – I followed it step-by-step.). The average answer of the individual heuristic inspectors was 2 (they all answered 2). The average answer of the heuristic 2-person teams was 2.33. For "expert use", the average answer of the individuals was 3 (they all answered 3), while the average answer of the 2-person teams was 2.83 (one team answered 2). The average answer of the "novice use" teams were 2.5.

For the inspection time, all heuristic 2-person teams reported that they had enough time. Of the 4 individuals who used heuristic evaluation, 3 said they needed more time (20, 40, and 120 more minutes). Of the 6 "expert use" 2-person teams, 3 reported that they needed more time (5, 20, and 60 more minutes). Of the 4 individuals who used "expert use", 1 needed more time (30 minutes). Of the "novice use" 2-person teams, 3 teams needed more time (10 minutes each). Therefore, among the 2-person teams, the perspective teams needed more time, which is an indication that they could have detected more anomalies if more time were given.

4.5.3 Threats to Validity

The following threats to validity are discussed in order to reveal their potential interference with the results:

Selection In this experiment, the subjects were from three different sources (students from two HCI classes and those who were not taking any of these classes). The distribution of subjects from different sources among the different technique groups was not even. For "novice use", the 6 pairs were all from the Spring class. For "expert use", 2 pairs were from each of the three sources. For "heuristic evaluation", 3 pairs were from the class in the Spring, 2 pairs from the class in the Summer, and 1 pair not taking any class. This increased the variety among the subjects. This may have created some bias. The data show that the performance variance among subjects from different sources was rather small, in contrast to the strong differences among different technique groups. But it would be better if all subjects were from the same source.

Experimental arrangements For the subjects who were taking either HCI class, the experiment happened in the classroom. Since as many as 11 pairs were in the same room, although there was no sign of people overhearing each other's discussions, the somewhat crowded environment may have a negative effect on the performance of subjects. But having the experiment in class was the best way for the students to participate.

4.5.4 Summary and Guidance for Further Studies

In summary, the perspective techniques helped the inspectors find significantly more anomalies covered by the perspectives. At the same time the perspective techniques did not cause inspectors to find significantly fewer anomalies that were outside their responsibility (anomalies not covered by the perspective) than the heuristic inspectors.

The heuristic inspectors on average only found a slightly larger number of "novice use" anomalies than the "expert use" inspectors, and a slightly larger number of "expert use" anomalies than the "novice use" inspectors. The difference was not statistically significant in either case.

For the combination of two 2-person teams (4 inspectors altogether), the best case for heuristic evaluation only matches the worst case (16 detected anomalies) for perspective-based inspection. The percentage of anomalies detected repeatedly was higher for heuristic combinations than for perspective combinations.

Further studies can extend on this experiment in the following two directions:

1. Replicate the 2-person team part with subjects from the same source, and with more subjects;

2. Replicate the whole experiment with the same number of data points for individuals and for 2-person teams.

4.6 Experiment III

A third experiment was conducted in Fall 1998. It was a replication of the 2-person team part of experiment II, with all subjects from one source. To test the hypotheses as listed in Section 4.2, the significance level was set at 0.05.

4.6.1 Method

Experimental Design

The subjects were 42 professionals who were taking a "Usability Engineering" class under the master's of software engineering program of the University of

Maryland. The experiment was conducted in the classroom. All subjects worked in pairs, i.e., 2-person teams. Each pair used one of the three techniques: heuristic evaluation, "novice use" perspective, or "expert use" perspective. There were 7 pairs using each technique.

The subjects had varied experience in user interface design and evaluation. A post hoc analysis of their reported experience showed that the subjects in heuristic evaluation had somewhat more usability evaluation experience than the subjects in the other two groups. For the 14 subjects using heuristic evaluation, 6 reported that they had a lot of experience doing usability evaluation (had evaluated at least 5 user interfaces for usability), and only 1 did not have any prior experience in usability evaluation. For the 14 subjects using "novice use" perspective, the numbers were 2 and 3, respectively. For "expert use", the numbers were 3 and 4, respectively.

The interface evaluated was the same as in experiment II. But only 3 of the 4 tasks were used in this experiment. They were the Internet tutorial, the on-line help, and a search facility.

Just as in the second experiment, for each 2-person team one person was asked to monitor the inspection process to make sure that the technique was followed. They were provided the same worksheet of the technique once for each task, so that they could check off items on the page to keep track of their progress for each task.

Procedure

Unlike in experiment II, which gave the subjects an hour and suggested the amount of time for each task but did not enforce it, experiment III had three

inspection sessions, one for each task. Each session was 15 minutes. At the beginning of each session the experimenter told the subjects what they were going to evaluate for the session. Therefore in their handout material there was no description of which parts of the interface were going to be evaluated.

Materials

The usability heuristics and perspectives used in this experiment were essentially the same as in the second experiment, with minor wording changes. The usability issues covered in the heuristics (with detailed issues under each brief heuristic) were exactly the same as the union of the issues covered by the "novice use" and "expert use" perspectives.

Independent and Dependent Variables

There was one independent variable: the inspection technique, with three treatments: heuristic evaluation, novice use, and expert use.

The dependent variables were the number of anomalies detected by each pair of inspectors (overall and under each category), the subjective rating of the inspection technique by each inspector, each subject's self-report as to how well the technique was followed, and the reported extra time needed to complete the inspection.

Data Coding

Since experiment III used the same artifact as experiment II, the data were processed and coded in a similar way.

Table 4.18: Anomaly detection effectiveness (mean and standard deviation) in experiment III

	# of all anomalies	% of novice anomalies	% of expert Anomalies
Heuristic evaluation	9.29 (3.59)	5.18 (4.86)	21.42 9.79
Novice use	11.86 (2.61)	27.27 (9.09)	15.42 (5.13)
Expert use	14.00 (2.71)	7.77 (3.45)	32.34 (6.92)

(N=7 for each of the 3 treatments)

4.6.2 Results and Discussion

There were a total of 60 usability anomalies detected by the subjects, of which 22 were classified as related to "novice use" and 38 as related to "expert use".

The independent variable was the inspection technique, with three treatments: heuristic evaluation, "novice use" perspective, or "expert use" perspective.

The dependent variables were the number of overall anomalies detected, the overall detection score (with severity level considered), the number of novice anomalies detected, and the number of expert anomalies detected.

Results of the Individual 2-person Teams

The number of anomalies detected by each group are summarized in Table 4.18.

Table 4.19: ANOVA and t-tests for anomaly detection effectiveness in experiment III

	ANOVA	Post hoc analyses at p=0.05 level
All anomalies	0.029	Expert > Heuristic
Novice anomalies	0.000	Novice > Heuristic Novice > Expert
Expert anomalies	0.002	Expert > Heuristic Expert > Novice

(N=7 for each of the 3 treatments)

ANOVA and post hoc analyses were applied to the performance of all 21 teams (7 data points for each technique), with the inspection technique as the single factor. The results are summarized in Table 4.19.

For either the number of overall anomalies detected, or the overall detection score, ANOVA testes showed that there was a statistically significant difference among the three groups ($p<0.05$). Post hoc tests showed that for both dependent variables, at 0.05 level, the "expert use" group was significantly better than heuristic evaluation. The "novice use" group was not significantly different from the other two groups.

For the number of "novice use" anomalies detected, ANOVA test showed a statistically significant difference among the 3 groups ($p<0.01$). Post hoc tests showed that the "novice use" group was significantly better than the other two groups. The "expert use" group was better than the heuristic group, but without statistical significance.

For the number of "expert use" anomalies detected, ANOVA test showed a statistically significant difference among the 3 groups ($p<0.01$). Post hoc tests showed that the "expert use" group was significantly better than the other two groups. The heuristic group was better than the "novice use" group, but without statistical significance.

As to major anomalies detected, 14 were found only by perspective inspectors and not by any heuristic inspectors. These 14 anomalies were detected a total of 34 times (8 were detected more than once). Only 1 major anomaly was detected uniquely by heuristic inspectors, a total of 4 times. This anomaly was concerned about the overall structure of the on-line help.

Aggregated results of two 2-person teams

For the combined performance of two teams, there are 49 possible perspective combinations (with one "novice use" team and one "expert use" team). There are 21 possible heuristic combinations out of the 7 heuristic teams.

The perspective combinations (4 inspectors) found a range of 18 to 31 unique anomalies (the mean was 22.90), with 1 to 6 anomalies found by both teams (the mean was 2.96). On average, about 13% of the anomalies in the combined list were found by both teams. The heuristic combinations (4 inspectors) found a range of 9 to 22 anomalies (the mean was 15.81), with 1 to 7 anomalies found by both teams (the mean was 2.76). On average, about 17% of the anomalies in the combined list were found by both teams. The improvement of perspectives over heuristic evaluation was about 45% for 4 inspectors with regard to the number of un-duplicated anomalies detected.

Table 4.20: The subjective ratings in experiment III

	Mean	StdDev	Median
Expert use	3.43	0.84	4.00
Novice use	3.36	1.03	4.00
Heuristic	3.36	0.63	3.50

(ANOVA: $p > 0.50$; N=7 for each row.)

Table 4.21: The reported process conformance in experiment III

	Mean	StdDev	Median
Expert use	2.86	0.24	3.00
Novice use	2.50	0.76	3.00
Heuristic	2.21	0.27	2.00

(ANOVA: $p = 0.07$; N=7 for each row.)

Subjective Ratings

There was no statistically significant difference among the groups in the subjective rating of how easy the techniques were to follow. See Table 4.20.

There was no statistical significance among the groups in their reporting about how well they had followed the techniques. See Table 4.21.

There was no statistically significant difference among the groups with regard to the extra time they wished to have for the inspection. More time was needed by 3 "expert use" teams (10, 15, and 45 minutes), 2 "novice use" teams (20 and 60 minutes), and 3 heuristic teams (10, 10, and 45 minutes).

4.6.3 Threats to Validity

Similar to experiment II, experiment III was conducted in the classroom. Since there were more subjects than experiment II in the same classroom, there is some chance that people may overhear each other's discussions. The somewhat crowded environment may have had a negative effect on the performance of subjects. My observation of the experiment was that subjects were concentrating on their own work. But still they were not perfectly isolated from each other. On the other hand, the distribution of the material made it more likely that subjects were using different techniques from they neighbors. If they had copied some anomalies of their neighbors, it would have made the detected anomalies by different techniques more similar to each other, which would have made the experiment results less significant. Same as in experiment II, having the experiment in class was the best way for the students to participate.

4.6.4 Summary and Guidance for Further Studies

In summary, the perspective inspectors found significantly more anomalies covered by the perspectives. At the same time the perspective techniques did not cause inspectors to find significantly fewer anomalies that were outside their responsibility than the heuristic inspectors. The perspective inspectors also found more major anomalies. These results are the same as in experiment II.

The heuristic inspectors on average only found a slightly larger number of "expert use" anomalies than the "novice use" inspectors. The heuristic inspectors found fewer "novice use" anomalies than "expert use" inspectors. The difference was not statistically significant in either case.

For the combination of two 2-person teams (4 inspectors altogether), the per-

spective combinations on average found 45% more anomalies than the heuristic combinations. The percentage of anomalies detected repeatedly was higher for heuristic combinations than for perspective combinations.

Further studies can improve on the following aspects of this experiment:

Environment It would be ideal to have each 2-person team work in a lab environment without any outside interference.

Balanced number of anomalies As in experiment II, this experiment reported more "expert use" anomalies than "novice use" anomalies. It would be better if each perspective covered about the same number of anomalies in the artifact.

4.7 Comparison of the Experiment Results

The three experiments differed in factors such as experimental design and subjects characteristics that may have contributed to the differences in their experimental results. These factors are summarized in Table 4.22.

The results of the three experiments are summarized in Table 4.23. The following describes these differences and and discusses the possible causes.

4.7.1 Feasibility of the Technique

In these experiments, the feasibility of each inspection technique was reflected through the time needed for an inspection, how well inspectors followed the technique, and the subjective ratings of the technique in terms of its ease of use.

In the first experiment, all but one subject reported that they had enough time. Heuristic evaluation was rated significantly easier to use than the perspec-

Table 4.22: Different settings of the experiments

Factor	Experiment I	Experiment II	Experiment III
Team size	all individuals	individuals and pairs	all pairs
Subjects	professionals (domain experts)	students in a HCI class	professionals in a HCI class
Application domain	Web-based form	Web-based tutorial, on-line help, information seeking, and search	Web-based tutorial, on-line help, and search
Process conformance	most followed the techniques in general but not step-by-step	pairs better than individuals; for pairs, expert use better than novice use, which was better than heuristic	expert use better than novice use, which was better than heuristic, with no significant difference
Time	only one subject said that more time was needed	heuristic individuals needed more time than expert use individuals and heuristic pairs; for pairs, expert use and novice use needed more time than heuristic	several teams in each technique needed more time, with no significant difference

Table 4.23: Comparison of the experiment results

Measure	Experiment I	Experiment II	Experiment III
Subjective rating	heuristic evaluation rated significantly easier to use	expert use rated easier better than novice use and heuristic	all three techniques had about the same rating
Overall anomalies	perspective better than heuristic, no significance	expert use significantly better than heuristic or novice use; novice use better than heuristic, with no significance	expert use significantly better than heuristic; no other significant differences
Team performance	perspective 30% better for 3-person teams	perspective 90% better for 2 pairs (4 persons)	perspective 45% better for 2 pairs (4 persons)
Novice use anomalies	novice use significantly better		
Expert use anomalies	expert use *not* significantly better	expert use significantly better	
Error handling anomalies	error handling significantly better	N/A	

tives. The process conformance was not as good as the other two experiments for either the heuristics or the perspectives.

In the other two experiments, some changes were made to the perspective techniques. Also most subjects worked in pairs. These factors may have caused the subjective ratings to be consistently in favor of the perspective techniques, both in subjective rating and in process conformance.

In the second experiment, "expert use" and heuristic evaluation were used by both individuals and pairs. It showed that both techniques were rated easier to use by pairs than by individuals. Pairs using perspective techniques needed more time than pairs using heuristic evaluation. But the process conformance was also better for the perspective pairs than for the heuristic pairs.

In the third experiment, there was no significant difference among the techniques in process conformance, extra time needed, or subjective rating. Combining the results from experiments II and III, it was shown that the perspective-based techniques (as used in these two experiments) were easier to use than heuristic evaluation, that process conformance was better for perspective-based techniques than for heuristic evaluation, which does not have a well-defined procedure.

4.7.2 Effectiveness of the Techniques

Given that each perspective only covers a subset of the usability issues, one may think that a perspective inspector on average would detect less anomalies than a heuristic inspector. But in all cases, each perspective inspectors on average detected more overall anomalies, with or without statistical significance. This result, together with the same results reported in experiments on defect-based

[48] and perspective-based reading [3], may suggest that given a certain amount of time, when a fairly large number of anomalies exist in the artifact, an inspection focusing on a subset (e.g. 30-40%) of the anomalies will tend to detect more anomalies than a general inspection.

The effectiveness of the "novice use" perspective was consistent across the three experiments. Since most inspectors were not novice users, the "novice use" perspective was often different from an inspector's own perspective, and thus helped reveal a significantly higher number of anomalies related to "novice use".

The effectiveness of the "expert use" perspective was not significant in experiment I, but was significant for experiments II and III. One reason for this difference might be that the process conformance in experiment I was not as good as it was in the other two experiments. The subjects in experiment I all did the inspection individually. The perspective-based techniques were rated harder to use than heuristic evaluation. Before experiment II, the perspective-based techniques were modified to make them easier to use. Another possible reason is that the inspectors in experiment I were all experts in the task domain, while the inspectors in the other two experiments were not. Or it could simply be because the number of subjects for each perspective, which was 4, was too small.

The effectiveness of the "error handling" perspective was significant in experiment I. This indicates that "error handling" perspective was quite different from an inspector's own perspective, and thus can make a difference. This perspective was not used in the other two experiments.

As to the effectiveness for multiple inspectors, in experiment I, the improve-

ment of perspective-based inspection over heuristic evaluation was about 30% for 3 inspectors. For experiment II, the improvement was about 90% for 4 inspectors. For experiment III, the improvement was about 45% for 4 inspectors. The subjects for experiment II were from different sources. Therefore the distribution of subjects may have been biased in favor of the perspective groups. Experiment I did not have as high a level of process conformance that the other two experiments did. Therefore, the results from experiment III, might be a more reliable indication of how much the perspective-based inspection can improve effectiveness for the combination of multiple inspectors.

4.7.3 Team Size and Inspector Expertise

The experiments had subjects conduct the inspection either individually or in a two-person team. Since perspective-based inspection was new to all the subjects, and heuristic evaluation was new to most subjects, a learning process of the inspection technique was going on during the inspection. The subjects who worked in pairs rated the techniques as easier to use than the subjects who worked individually. This suggested that the two-person team made the learning process easier.

Using these inspection techniques for the first time, the subjects needed to pay attention to both the technique and the artifact being inspected. This seemed to be a daunting task for an individual. That was probably the reason why process conformance was significantly better for the two-person teams, when one person kept an eye on the inspection technique, while the other person could concentrate on the artifact.

Personal expertise and capabilities are important factors in inspection per-

Table 4.24: The number of anomalies detected by inspectors using the same technique

	Exp. I - A 12 singles	Exp. I - B 12 singles	Exp. II 4 singles	Exp. II 6 pairs	Exp. III 7 pairs
Heuristic	4-14	2-12	3-7	2-11	5-14
Novice	6-11	7-15	N/A	5-11	8-16
Expert	6-10	3-13	5-14	11-20	11-19
Error	6-12	4-9	N/A	N/A	N/A

("Exp. I - A" means results for interface A in experiment I)

formance. For the first experiment, the domain experts reported a lot of issues related to the specific task domain. The two external reviewers in the first experiment each reported some unique anomalies. One reviewer found anomalies in using graphics in the Web-based forms. The other reviewer found anomalies in consistency issues.

As shown in Table 4.24, in the same experiment, using the same technique, some inspector could find 6 times the number of anomalies found by another inspector. The data suggest that the performance of perspective-based techniques is less varied than that of heuristic evaluation.

Comparing the team performance between experiment II and experiment III, subjects in experiment III did better than subjects in experiment II, given that experiment III only involved 3 of the 4 user tasks used in experiment II, and that the subjects in experiment III had less time for those 3 tasks (45 minutes vs. 50 minutes). The observation was that the subjects in experiment III, who were professionals, were more productive than the subjects in experiment II, who

were students.

4.8 A Survey of Usability Practitioners

4.8.1 Method

The last research question defined in Section 4.1 was to investigate how well the proposed perspective-based usability inspection fits with the current practice. Therefore, 8 usability engineers from 4 companies were either interviewed face-to-face or surveyed by a questionnaire through e-mail.

Of the 8 usability engineers in this study, 4 were from GE Information Services. The other 4 were from US WEST Advanced Technologies, Ameritech, and SBC Communications. Two of them had only inspected 2 or 3 interfaces. The others all had inspected at least 10 interfaces.

4.8.2 Results and Discussion

Commonalities Among the Practitioners

There are some commonalities among these practitioners' usability inspection practices. They are:

- use user tasks;
- go through an interface multiple times; and
- remind oneself the usability issues to check.

Differences Among the Practitioners

Different people had different ways to remind themselves what usability issues need to be considered. Most of them use guidelines, standards, checklists, or heuristics, going though them briefly before the inspection or during the inspection. One of them put a list of usability heuristics on the wall of his office so that he could easily raise his head to go through them while doing an inspection. Another person would go through old usability reports before each inspection to be familiar with the typical usability issues.

The practitioners were quite different in their inspection processes. They would all go through an interface multiple times, each time doing one of the following:

- Get familiar with the interface, and see if users can find the functions they need.
- Go through a user task to find usability anomalies. Some would first concentrate on the important issues and have another round to check minor issues.
- Go through each screen, menu, or dialog box etc. to check their conformance to standards and guidelines.
- Go through the interface again to see if there is anything missed or that the usability issues raised early are still valid.

Comments on Perspective-based Usability Inspection

For perspective-based inspection, the practitioners were given a brief description and asked for comments. One practitioner said that it was very similar to the

way she conducted usability inspection, i.e., she would always concentrate on a subset of issues for each round. For example, for a Web interface, one round may focus on navigation, while another round may focus on content and layout, etc. Most practitioners felt the idea of perspective-based inspection was interesting and worth investigating.

One practitioner commented that although it was possible to concentrate on a subset of issues at a time, she would prefer not to because she would like to have different issues coming to mind without restriction. This preference might not be a big problem since perspective-based inspection tries to organize the inspector's way of thinking but does not prohibit the reporting of any other issues. However, to effectively conduct perspective-based inspection, one needs to follow the procedure and concentrate on the issues listed in the technique.

Two practitioners raised another concern that if each inspection session concentrated on one perspective, inspectors may miss issues that are only likely to be found when different perspectives are considered together. To check whether such usability anomalies exist, I went back to the experimental data to see what are the anomalies that were only detected by heuristic evaluation.

For experiment I, there were 3 usability anomalies only reported by heuristic inspectors (all for interface B):

- "How do I quit?"
- "Cannot put a space character in the 'Last Name' field."
- "There is a loss of control and scope when persons tabs aren't generated automatically right after the 'number of people' question was answered."

For experiment II, there was one anomaly that was only reported by heuristic

inspectors:

- "Icon for going back to main help page looks too similar to icon for going back to glossary."

For experiment III, the following anomalies were only reported by heuristic inspectors:

- "The left frame of on-line help should be wider."
- "The structure of on-line help is messy. Should be better organized."
- "Left frame in search is not well organized."

Most of these anomalies were covered by the perspective-based techniques used in the experiments. However, the last anomaly from experiment I and the last two anomalies from experiment III suggest that perspective inspectors may have concentrated more on detailed issues than on global issues such as overall organization, and overall look-and-feel. Currently these issues are covered in the "expert use" perspective. It might be necessary to separate them out and have another perspective to deal with such higher level usability issues.

Chapter 5

Summary of Dissertation Work

5.1 Contributions

In this work, I have:

1. Defined a framework of usability inspection techniques by identifying a list of internal and external characteristics (Section 2.2.1).

2. Developed a perspective-based usability inspection technique along "novice use", "expert use", and "error handling" perspectives, using the GQM (Goal/Question/Metric) method (Chapter 3).

3. Conducted 3 experiments to study the feasibility and effectiveness of the technique, with both professionals and students as subjects (Sections 4.4 through 4.7).

4. Surveyed 8 usability engineers to understand the compatibility of the technique with the current practice (Section 4.8).

This work contributes to the field a research agenda, the experimental results, and some implications for practitioners.

5.1.1 Research Agenda

The framework of usability inspection techniques (part 1) defined the variables that can be manipulated in developing a new technique (part 2), as well as in the experimental design for empirical studies (part 3).

Different techniques can be developed by varying the internal attributes defined in the framework. A series of experiments can be designed, using one or more of those attributes as independent variables, in order to study the effects of the variables in question.

According to the framework, the new technique proposed in this dissertation is high in prescriptiveness, and assigns specific and different responsibilities to different inspectors.

The experiments were designed to compare the new technique against a current technique. The two techniques differed along several characteristics. Therefore the reported differences between the techniques compared in these experiments may have been the combined results from all the different characteristics, along with other factors as discussed in the threats to validity.

Other techniques with different attribute values might be developed. Further empirical studies can investigate the effects of each single attribute as well as the meaningful combinations of multiple attributes.

5.1.2 Experiment Results

The results from the experiments are:

Detection of overall anomalies In all three experiments, on average each perspective inspector detected more overall usability anomalies than each

heuristic inspector, although each perspective only covered a subset of the overall usability issues.

Detection of different types of anomalies Anomalies were classified based on whether they were covered by each of the perspectives used in the experiment. In all three experiments, inspectors using each perspective always detected more anomaly related to that perspective than heuristic inspectors. The effect was statistically significant in all cases except for "expert use" in experiment I.

Detection of major anomalies For all cases but 1 (interface B in experiment I), perspective inspectors detected 6 to 14 unique major anomalies, while heuristic inspectors detected 0 or 1 unique major anomaly. For interface B in experiment I, perspective inspectors and heuristic inspectors each detected 4 unique major problems, each of which was detected only by one inspector.

Feasibility The survey of usability engineers showed that they often inspect an interface several times for different issues. Thus perspective-based inspection may not require more overhead and be that distant from what usability practitioners do. In the three experiments, different inspectors were able to use perspective-based inspection to detect more anomalies than those who used heuristic evaluation. The effectiveness was shown under different time constraints from more than enough time in experiment I to fast-paced inspections in experiments II and III. The improved version of perspective-based inspection used in experiments II and III received better subjective ratings from the subjects.

Effect of team size All subjects in experiment I and 8 subjects in experiment II did individual inspection. The other subjects did inspection in 2-person teams. Having two people working together made it easier to learn the inspection technique, follow the inspection procedure as described on the worksheet, and at the same time look at the interface on the computer screen to detect anomalies. A significant number of subjects reported that two people often compensate each other by detecting different anomalies.

Effect of personal expertise The experiments show that people with similar backgrounds vary dramatically in their ability to detect usability anomalies. Different people often focused on different types on usability issues during the inspection, especially when they were not guided by an inspection technique.

5.1.3 Implications for Practitioners

The results from this research provide the following guidance to usability practice:

One-person inspection Sometimes only one usability engineer is available for a usability inspection. The survey showed that usability engineers always inspect an interface several times. The demonstrated effectiveness of perspective-based usability inspection suggests that usability engineers can focus on different usability issues each time an interface is inspected. This may help them detect more anomalies.

Multiple-inspector inspection If there are a number of people available to conduct the inspection, the research results suggest that it is more effec-

tive to assign different responsibilities to different people than to have all the people conduct the same general inspection. Responsibilities should be assigned in a way to make sure that all usability issues are covered, including the high-level issues.

Team size The research shows that team inspection (specifically two-person teams) is preferable when inspectors are non-experts who are not familiar with the inspection technique. In such cases, team work makes it easier to learn and apply an inspection technique.

Personal expertise Since people's ability to detect usability anomalies varies dramatically, it is important to find the "good" inspectors in the organization and to understand who is good at finding what types of anomalies. In this way, "good" inspectors with different specialties can be hired to form an effective inspection team.

5.2 Limitations and Open Issues

5.2.1 Study Design

This research is limited by the environment available for running the experiments. Such limitations include number of subjects, time to run the experiment, and the amount of training provided. For all 3 experiments, subjects were given a fixed amount of time to conduct the inspection.

For experiment I, there were 2 independent variables (inspection technique and interface order) and 24 subjects. There were only 4 subjects for each perspective. The inspection time was enough, but the training time was very limited.

For experiments II and III, there was one independent variable (inspection technique) with 3 treatments (heuristic evaluation, "novice use", and "expert use"). There was 6 pairs in each treatment for experiment II, and 7 pairs in experiment III. Since all subjects were in the same classroom, training was provided in the material with a chance for subjects to ask questions. Inspection time was not enough for about 1/3 of the subjects using both techniques.

The two techniques compared in these experiments differ along more than one characteristic, based on the defined framework of inspection techniques (Section 2.2.1). This was because the proposed technique was compared against the most widely used technique in current practice. From a researcher's point of view, the effect of each characteristic should be studied by comparing techniques that differ only along that characteristic.

5.2.2 Perspectives

The three perspectives used in this research are just one way to divide usability issues. Other perspectives can be defined and used to form another set of perspective-based inspection techniques. This is the topic of future studies.

5.2.3 Application Domain

The user interfaces inspected in this research were a Web-based data collection form, and several parts of a commercial Web site (tutorial, on-line help, information seeking by browsing, and search). It would help to have a classification of different user interface features under different platforms (Web, Windows, voice interface, etc.) and systematically study whether the proposed technique is effective in the usability inspection of interfaces with various features under

various platforms.

5.2.4 Artifact Format

The three experiments all inspected fully-functioning interfaces. Further studies should be conducted to test if perspective-based inspection is feasible and effective in inspecting user interfaces of other formats, such as a user interface specification, a prototype that is still not fully functioning.

5.3 Future Research

The studies described in this dissertation provide some initial evidence as to the effectiveness of perspective-based usability inspection. It takes a series of experiments and replications to build up a body of knowledge as to how effective the technique is under different conditions. Research questions for further studies are:

- Are the results in these studies replicable with other non-expert inspectors, with other Web-based applications, or with other types of user interfaces (Windows, voice, etc.)?
- Is perspective-based usability inspection effective for other artifact formats such as user interface specification, early prototypes?
- Which factor has more influence on the effectiveness of usability inspections by non-experts: prescriptiveness (whether to guide the inspection process by a procedure) or specific responsibility (whether to focus on a subset of anomalies or take a specific perspective)?

- What are other effective sets of usability perspectives for generating other perspective-based usability inspection techniques, either for all user interfaces in general or for a specific user interface domain?

- Given the same number of inspectors and the same amount of time, which is more effective: asking inspectors to work individually or in pairs? The question can be asked for both perspective-based inspection and heuristic evaluation, and for both inspectors who are unfamiliar with the inspection technique and inspectors who are familiar with it.

Along future studies that address these questions, more research questions may emerge that need further research. Such continuous experimentation would not be feasible without the existence and cooperation of a community of researcher with similar interests. By working together and sharing experiment materials, results, and experiences, the community can help an area to grow and mature.

Appendix A

Generic Usability Questions for Each Perspective

These are the generic inspection questions for each perspective. They are generated following procedures described in Chapter 3.

A.1 Novice Use

1. Will users be able to start the system? (step 1)

    - Is the identification of the system easily memorable (in case of command name or URL) or recognizable (in case of icon or name as appeared in a list)? (I1, IG)
    - Does the identification of the system relate directly to its function? (I1, IG)
    - Is the hardware and software support required to run the system beyond what users have? (E3)

2. Will users be able to exit the system? (step 3)

- Is there a consistent way for users to exit the system from different system states? (I2, IC)
- Is the way to exit the system clearly marked under all different system states? (I2, EG)
- Can users apply the syntactic knowledge they have with other systems to find out the way to exit the system? (I2, EC)

3. For each task, will users know what effects are to be achieved in the computer system? (step 4)
   - Is on-line help always available? (I4, EG)
   - Does the metaphor used in the user interface create a natural connection between what is to be achieved in the task domain and that in the computer system? (I3, I18, IG)
   - Can users apply the computer domain knowledge they have to correctly map the task to the effects to be achieved in the computer system? (I18, EC)
4. For each task, will users know what needs to be done to achieve the desired effects, in terms of actions and objects and the order of actions? (step 5)
   - Can users find out how to accomplish the task from the instruction or on-line help? (I4, I18, EG)
   - Is there a direct connection between the description of the task and the objects (menu, label, etc.) and actions directly perceivable from the user interface? (I18, IG)

- If the task requires more than one action, does the system provide some natural ordering of the actions? (I18, IG or EG)

5. For each task step, will users be able to execute it successfully? (step 7)

   - Will users be able to execute the action under the current system mode? (I18)
   - Is the way to execute the action obvious from the user interface type: menu selection, form fill-in, direct manipulation, etc.? (IG)
   - Can users apply the syntactic knowledge they have with other systems to correctly execute the action? (EC)
   - Do actions in the system have consistent syntax whenever possible? (IC)
   - Is there guidance with entry point from the current state as to how the action can be executed? (EG)
   - Do users have all the necessary knowledge to carry out the action? (U3, U4, U6)
   - May the user's working environment make it difficult or impossible to execute the step? (E1)
   - After each step, will the user interface guide users to the next step for completing the task, if the task is not complete yet? (IG or EG)

6. After each step, will the user be able to perceive the effect? (step 8)

   - Does the user interface provide perceivable feedback that the user is not likely to miss, considering the obtrusiveness and the distance

between the place of system feedback and place of the user's current attention? (I22)

- Will the user's working environment prevent the user from perceiving the feedback? (E1)

7. After each step, will users understand from the perceived effects how much progress has been made? (step 10)

   - Are the effects in the user interface presented in a way that can be directly related to effects in the task domain? (I3)

A.2 Expert Use

1. Can the system be customized to enter a certain mode when started? (step 1, flexibility)

2. Can users exit the system with minimum effort when so desired? (step 3, efficiency)

3. Are colors used consistently across the user interface, i.e., the same color is used to carry the same meaning? (I11, I12, consistency)

4. Are fonts used consistently across the user interface, i.e., the same font is used to represent the same type of information? (I14, I15, consistency)

5. Is the layout of objects and actions consistent across the user interface? (I17, consistency)

6. Does the menu structure follow the "shallow-and-wide" strategy instead of the "deep-and-narrow" strategy? (efficiency, reference [56])

7. Are colors used conservatively (use no more than 6 colors in each window) and appropriately (use colors that are widely spaced on the color spectrum)? (I13, references [56] [19])

8. Are fonts used appropriately (use no more than 2 styles and no more than 3 sizes)? (I16, reference [19])

9. Does the user interface avoid using background or visual attributes (font, color, blinking, animation, orientation, etc.) that would make text hard to read on the screen? (I11, I14, ease of use, perceptual)

10. For each task, can expert users complete the task with minimum perceptual, cognitive, and motor efforts? (step 7)

    - Is it true that the task cannot be completed in less steps? (efficiency)
    - If the steps for expert use are different from those for novice use, is there any reminder that informs users the more efficient way to do the task when they do it the less-efficient way? (support for transition from novice to expert)
    - Does the system do computation or remember information for the user whenever possible? (efficiency and ease of use)
    - Is the information organized in a way that the most important information can be read first? (efficiency)
    - Does the user interface use an recognizable order whenever a list is presented? (efficiency)
    - Does the system avoid providing redundant information? (efficiency)

- Does the user interface support conventional shortcuts? (efficiency, consistency)
- Are appropriate default values used whenever possible? (efficiency)

11. After each step, is it easy to perceive the effect of the action? (step 8)
    - For long actions, does the system provide information about how much has been done? (ease of use, cognitive)
12. Does the user interface assure the user of security or confidentiality when necessary? (E7)

A.3 Error Handling

1. If the system cannot start (due to problem with the hardware or software environment, syntax error, etc.), will the user be informed properly? (step 1)
2. For each task step, what are the possible errors a user may make, out of the following types: wrong action (because of misconception or misunderstanding), execution at wrong system mode, syntax error (commission and omission), missing task steps, slippage (mis-type, select the wrong object or action)?
    - How frequently might each error occur?
    - What are the consequences if the error is not prevented?
    - What are the consequences if the error is not detected immediately?

- For each possible error, has the user interface been designed to minimized the possibility of this error?
- For each possible error, will the user realize the error immediately from the response of the user interface?
- For each possible error, does the user interface minimize the side effects it may cause and enable the user to reverse the effects?
- For each possible error, when the user realizes the error, does the user interface provide guidance for the user to recover from the error, including guidance about how to reverse the side effects?

3. If an error occurs when the system does validation in a batch mode, does the error message brings the user to the place in error?

4. Does the system prevent users from terminating the system accidently without saving their work first? (step 4)

Appendix B

Inspection Techniques Used in Experiment I

The inspection techniques as used in the first experiment are presented here.

B.1 Usability Heuristics

1. **Speak the user's language**: Use words, phrases, and concepts familiar to the user. Present information in a natural and logical order. Define new concepts the first time they are used.

2. **Consistency**: Indicate similar concepts through identical terminology and graphics. Create consistent interfaces for tasks that are essentially the same. Adhere to uniform conventions for layout, formatting, phrasing, interface controls, task actions, etc.,for tasks that closely resemble one another.

3. **Minimize the users' memory load and fatigue**: Take advantage of recognition rather than recall. Do not force users to remember key infor-

mation across tasks. Minimize physical actions such as hand movements, and mental actions such as visual search or decisions.

4. **Flexibility and efficiency of use**: Accommodate a range of user sophistication. For example, guide novice users through a series of progressive steps leading to the desired goal, but provide proficient users with shortcuts that do not violate data collection procedures.

5. **Use visually functional design**: Visually structure the user's task. Support frequent repetition of a small set of well specified tasks. Make it hard to confuse different tasks. User's eyes should be drawn to the correct place at the correct time, e.g. to actions to be performed, items to be remembered or referred to.

6. **Design for easy navigation**: Allow the user to move as necessary through the form, either forward or back to an earlier question. Enable an easy return from a temporary excursion to another portion of the survey. Enable user to determine current position easily.

7. **Validation checks**: Make sure error messages are clear. Resolution is easy. Placement of edit validations makes sense. Error validations will be performed.

8. **Facilitate data entry**: Easy to enter data. Data are visible and clearly displayed. Allow the users to change data previously entered. Easy to find data already entered. Necessary entries are clearly defined. Entries are in correct format.

9. **Provide sufficient guidance**: Convey sufficient text or graphical information for the user to understand the task, but do not provide more information than users need. Implicitly convey task instructions where possible through non-verbal cues, such as those provided by the spatial relationships among form elements on the screen. Provide help when necessary, either auditory or on-line.

B.2 Inspection Procedure for Novice Use

The user's goal is to fill out the form and submit it. The goal can be decomposed into a series of sub-goals. For each sub-goal, go through the following stages and check the questions for each stage.

1. **Map** the **sub-goal to** the **effects** to be achieved in the user interface.
   (a) Will the user know when the subgoal is achieved?
2. **Identify** the **actions** for achieving the effects.
   (a) Are there *instructions or online help* that are *understandable* to the user and provide *sufficient guidance* as to what actions to execute?
   (b) Does the user know how to get to the online help, and how to come back from the online help?
   (c) Are *visual or auditory cues* like labels, icons and sound *understandable* to the user, and *consistent* from place to place in the user interface?
   (d) Do buttons and other clickable objects look clickable?
   (e) Are items in a list *unambiguous* in meaning?

3. **Execute** the **actions**. For each action

   (a) Are there *instructions or online help* that are *understandable* to the user and provide *sufficient guidance* as to how to execute the action (selection, data entry, navigation, submission, etc.)?

   (b) Can the user refer to the online help while answering questions?

   (c) Can the user execute the action correctly based on his/her previous knowledge?

   (d) Are same *actions* executed in a *consistent* way *among* different places in *the user interface*?

   (e) Are formats for data entry indicated?

4. **Perceive** the system **feedback**.

   (a) Does each user action (selection, data entry, navigation, submission, etc.) generate *feedback* that the *user is not likely to miss*?

   (b) Can users with disabilities or insufficient computer support (as described in the user profile) perceive the feedback?

5. **Understand** the **progress** made.

   (a) After each user action (selection, data entry, navigation, submission, etc.), will the *feedback* from the user interface *help the user* to *understand* if *progress* has been made?

   (b) Can the user constantly see what has been achieved so far?

B.3 Inspection Procedure for Expert Use

The user's goal is to fill out the form and submit it. The goal can be decomposed into a series of sub-goals. For each sub-goal, go through the following stages and check the questions for each stage.

1. **Scan through the instructions, objects, and actions** in the user interface.

   (a) Is the text *easy to read*?

   (b) Is the information organized in a way that the most *important information can be read first*?

   (c) Is each list presented in a way that the *more frequently selected items appear earlier*?

   (d) Is *redundant information avoided*?

2. **Execute the actions** for achieving the sub-goal, using short-cuts whenever possible. For each action,

   (a) Are possible *short-cuts available*, e.g., allowing users to use keyboard to switch to the next text field?

   (b) Are possible *default values used*?

   (c) Does the system *do computation or remember information for the user* whenever possible?

   (d) Can the user *make a selection by clicking on a larger area* associated with the object to be selected, e.g., by clicking on the text next to the radio button to be selected?

(e) Are *unproductive activities minimized*? These include navigation, mouse movements, hand movements between the mouse and the keyboard, and eye movements, etc.

(f) Are *stressful actions minimized*? These include keeping a mouse button pressed for a long time, clicking a mouse button multiple times consecutively, using the mouse to click on a very small object.

3. **Wait for system response** if necessary.

   (a) Does each user action *immediately* generate *perceivable results* in the user interface?

Besides the above detailed inspection, you should also consider the following higher-level question:

- Can the structure of the Web-based form be re-designed somehow to significantly reduce the user's unproductive activities (navigation, mouse movement, hand movement between the mouse and keyboard, and eye movement, etc.)?

B.4 Inspection Procedure for Error Handling

User errors often occur during human-computer interaction. The possible user error situations include, but not limited to:

- **Omission**: the user forgot to answer one or more questions; forgot to submit the form; etc.

- **Slippage**: the user typed something wrong; selected the wrong item or executed the wrong action (e.g. RESET) by accident; etc.

- **Wrong perception**: the user did not see a full list of possible answers because some items are not visible on the screen, or there is a visual break; etc.

- **Failed trial**: the user's guess turned out to be wrong. A novice user may guess on the basic functions, while an expert user may guess on short-cuts, etc.

- **Wrong system mode**: the user executed an action at the wrong mode (e.g. typing before activating a text field); entered data at the wrong location; navigated to the wrong place; etc.

With this Web-based form, a user's goal is to fill out the form with the complete and correct information and submit the form. This goal can be achieved by a series of steps. For each step of the user, go through the relevant parts of the user interface and consider all possible user errors that may occur. For each such error, ask the following questions:

1. Has the user interface done its best to *prevent the error*? (prevention)

2. When the error occurs, will the *user realize the error* immediately and *understand* the nature of *the error* from the response of the user interface? (information)

3. Does the user interface *minimize the side effects* the error may cause? (correction)

4. When the error occurs, does the user interface *provide guidance for error recovery*, including guidance about how to reverse the side effects? (correction)

Whenever the answer to one of the above questions is "no", a usability problem is detected. You may also detect problems not covered by these questions, but please make sure that you **focus on error handling issues as much as possible**.

Appendix C

Inspection Techniques Used in Experiments II and III

The inspection techniques (including the introductions) as used in experiment III are presented here. Three tasks were provided to the subjects one by one, at the beginning of each of the three sessions. For experiment II, the only difference is that there were four tasks, which were presented one paper (one task on each page) and were all given to the subjects at the beginning.

C.1 Heuristics Evaluation

Introduction

You are going to inspect the GE TradeWeb (http://www.getradeweb.com) to find usability problems. Usability problems are defined as issues that may hinder the effective, efficient, and satisfying use of the system by the defined users for the defined tasks.

The GE TradeWeb is an Electronic Data Interchange (EDI) service for small businesses and organizations. It enables businesses and organizations to use a

Web browser to exchange business documents with their trading partners. You are going to inspect several parts of the GE TradeWeb, each in a 15-minute session. You will be told which part of the interface to inspect at the beginning of each session.

You are to conduct the inspection using heuristic evaluation. A list of usability heuristics is provided. Each heuristic is associated with specific issues about Web usability.

The provided heuristics will help you identify usability problems. But you may also report usability problems not covered by these heuristics. In reporting usability problems, please indicate the number of the session (i.e. 1-3) where the problem occurs.

The inspection requires teamwork between the two of you. One person should take charge of problem reporting. The other person should manage the inspection process. During the inspection, the second person can make check marks or other marks on the heuristics sheet to keep track of what usability issues have been checke d and what are yet to be checked. The two of you share the other responsibilities, including detecting usability problems, operating the computer, etc. If you like, you may switch roles in the middle of the inspection.

The heuristics sheets are the same for all 3 sessions. Before we start the first session, the two partners can read the heuristics together.

Thank you for your participation!

Usability Heuristics

Heuristic evaluation is to examine the user interface and see if any of the following usability heuristics is violated. If so, report the violation by describing what the interface has done wrong.

1. Speak the user's language. Use objects, actions, and system messages that are understandable to the user. Present information in a natural order.

2. Consistency. User consistent look-and-feel within each page and among all pages of the same site, including color, font, layout, and interaction style.

3. Minimize the users' memory load and fatigue. Use URL's that are easy to remember. Use meaningful Web page titles and hyper-link labels. Make the page easy to read (pay attention to color, font, orientation, animation, and blinking). Use graphics conservatively, provide size information and ALT tag description. Provide feedback when necessary. Indicate progress made.

4. Flexibility and efficiency of use. Support conventional shortcuts. Minimize user actions. Use appropriate default values whenever possible. Use screen space efficiently to reduce scrolling. Make it easy for the user to bookmark a page.

5. Use visually functional design. Visually structure the user's task. User's eyes should be drawn to the correct place at the correct time, e.g. to actions to be performed, items to be remembered or referred to, system messages to be read. Support the user's task flow. Put important information toward the top. Allow the user to obtain important information by scanning through the interface.

6. Design for easy navigation. Allow the user to move easily through the Web site (forward, back, to home, to end, to a specific location). Enable an easy return from a temporary excursion to another page. Enable user to

determine current position easily. Make sure the browser's "Back" button works fine.

7. Prevent errors. Make it hard to confuse objects and actions. Graphical or textual representation of actions should relate directly to the goal to be achieved. Items in a selection list should be easy to distinguish.

8. Help user recognize, diagnose, and recover from errors. Make sure error messages are clear and resolution is easy. Validate user input. Make sure that after clicking a wrong link, the user can immediately realize being at the wrong place. Help the user understand when an action is executed in the wrong system mode.

9. Provide sufficient guidance. Convey sufficient text or graphical information for the user to understand the task. Provide task instructions through textual description or visual cues. Provide help when necessary.

10. Aesthetic and minimalist design. Group and align information. Do not provide more information than users need. Avoid having meaningless objects in a page.

C.2 Novice Use

Introduction

You are going to inspect the GE TradeWeb (http://www.getradeweb.com) to find usability problems. Usability problems are issues that may hinder the effective, efficient, and satisfying use of the system by the defined users for the defined tasks.

The GE TradeWeb is an Electronic Data Interchange (EDI) service for small businesses and organizations. It enables businesses and organizations to use a Web browser to exchange business documents with their trading partners. You are going to inspect several parts of the GE TradeWeb, each in a 15-minute session. You will be told which part of the interface to inspect at the beginning of each session.

You are to conduct the inspection from the perspective of novice use. Think of users who have never used the GE TradeWeb or even a Web browser before. But assume that users have the following characteristics: Users can use the keyboard and the mouse. Users do not have disabilities with their hands or vision. Users have enough task domain knowledge. Users understand English and can speak English. You need to examine if novice users can complete their tasks without trouble. The specific usability issues to check are provided in the procedure sheet.

You should focus on the issues included in the inspection procedure. But you may also report other usability problems. In reporting usability problems, please indicate the number of the session (i.e. 1-3) where the problem occurs.

The inspection requires teamwork between the two of you. One person should take charge of problem reporting. The other person should manage the inspection process. During the inspection, the second person can make check marks or other marks on the procedure sheet to keep track of what usability issues have been checked and what are yet to be checked. The two of you share the other responsibilities, including detecting usability problems, operating the computer, etc. If you like, you may switch roles in the middle of the inspection.

The procedure sheets are the same for all 3 sessions. Before we start the first

session, the two partners can read the sheet together.

Thank you for your participation!

Procedure

The "novice use" perspective checks if novice users can understand the interface, identify the correct actions, execute the actions, perceive the feedback, and understand the outcome. If the answer to any of the following questions is "No", a usability problem has been found. Report the problem by describing what the interface has done wrong. (Not all questions apply to every situation.)

The inspection procedure is as follows:

1. Work through the task steps to understand the general procedure for completing the task. Then go back to the beginning of the task.

2. For the current page, check the following:

   - If the page is accessed by a URL, is the URL easily memorable?
   - If the page is accessed by a link, is the link descriptive of the page?
   - Is the title of the web page good for identifying the page/site?
   - Is there a consistent way for users to go back one step or go to the home page?
   - Is the Back button of the browser good for going back one step?

3. Identify the next action for accomplishing the task, check the following:

   - Are graphics and text understandable to novice users?
   - Is it obvious to novice users what action to take? (which object to click; what search string to type; what format to use; etc.)

- Does the graphical or textual object users need to click relate directly to the user's current goal?
- When users need to select from a list of items, are the items easily distinguishable from each other?
- If needed, can users find information about how to do the task from on-line help?
- If the user selects a wrong link (or button, icon, etc.), does the interface help the user understand the error immediately?
- Is the current system mode (e.g. currently activated frame, data field, button, etc.) obvious to users?
- If the action cannot be executed in the current system mode, does the interface indicate so?
- If the user executes an action in a wrong system mode, does the interface help the user understand the error immediately?

4. Execute the action, check the following:
   - Is feedback (messages, change of display, etc.) provided when needed?
   - Is the feedback easily noticeable to users?
   - Is the feedback understandable to users?
   - If more actions are needed to complete the task, does the system provide a natural flow of the steps that follow (e.g. a top-down hierarchy for information seeking) or a reminder of the next step?
5. If the task is not complete, go back to step 2; otherwise, you are done with this task.

C.3 Expert Use

Introduction

You are going to inspect the GE TradeWeb (http://www.getradeweb.com) to find usability problems. Usability problems are issues that may hinder the effective, efficient, and satisfying use of the system by the defined users for the defined tasks.

The GE TradeWeb is an Electronic Data Interchange (EDI) service for small businesses and organizations. It enables businesses and organizations to use a Web browser to exchange business documents with their trading partners. You are going to inspect several parts of the GE TradeWeb, each in a 15-minute session. You will be told which part of the interface to inspect at the beginning of each session.

You are to conduct the inspection from the perspective of expert use. Think of users who have plenty of experience using Web browsers and are familiar with the GE TradeWeb interface. Such users demand efficiency, flexibility, and consistency. The detailed usability issues are provided in the procedure sheet.

You should focus on the issues included in the inspection procedure. But you may also report other usability problems. In reporting usability problems, please indicate the number of the session (i.e. 1-3) where the problem occurs.

The inspection requires teamwork between the two of you. One person should take charge of problem reporting. The other person should manage the inspection process. During the inspection, the second person can make check marks or other marks on the procedure sheet to keep track of what usability issues have been checked and what are yet to be checked. The two of you share the other responsibilities, including detecting usability problems, operating the computer,

etc. If you like, you may switch roles in the middle of the inspection.

The procedure sheets are the same for all 3 sessions. Before we start the first session, the two partners can read the sheet together.

Thank you for your participation!

Procedure

The "expert use" perspective checks to see if the interface is efficient, flexible, and pleasant to use. If the answer to any of the following questions is "No", a usability problem has been found. Report the problem by describing what the interface has done wrong. (Not all questions apply to every situation.)

The inspection procedure is as follows:

1. Work through the task steps to understand the overall task procedure:

    - Does the interface support the task in an efficient way?

2. For the current page, check the following:

    - Can users easily bookmark the current page if they want to?
    - Does the page have a consistent look within itself and with other pages of the same site (color, font, layout, etc.)?
    - Is the page easy to read, considering color, font, orientation, animation, and blinking?
    - Is screen space used efficiently to reduce scrolling?
    - Is important information shown toward the top of the page?
    - Does the interface avoid providing redundant information or having meaningless components?

- Are graphics used conservatively and with size and ALT information provided in the IMG tag?
- Is information well grouped and aligned?
- In presenting a list, does the interface use a natural order?
- Can users get the important information by scanning only the text that stands out, without reading every detail?

3. Identify the next action for accomplishing the task, check the following:

   - For a long page or a series of pages, can users go directly to the beginning, the end, or any other locations the user may want to go?
   - Does the interface support conventional shortcuts (e.g. Tab key for switching, Enter key for submission)?
   - Are appropriate default values used whenever possible?
   - If the user submits a form (e.g. a search request) without filling out a mandatory field, will the interface help the user understand the error and make corrections?
   - If the user types the wrong search string (mis-spell, wrong format, etc.), will the interface help the user understand the error and make corrections?

4. Execute the correct action, check the following:

   - Are users shown the progress made?
   - Is there an indication of where the user is within a larger scope? (e.g. in a list of menu items highlight the one that corresponds to the current page.)

5. If the task is not complete, go back to step 2; otherwise, you are done with this task.

Bibliography

[1] Victor Basili. The role of experimentation in software engineering: Past, current, and future. In *Proceedings of the 18th International Conference on Software Engineering*, pages 442–449. IEEE, 1996.

[2] Victor Basili, Gianluigi Caldiera, Ben Shneiderman, and Zhijun Zhang. Anomaly taxonomy in user interfaces. In *Human-Computer Interaction Laboratory 12th Annual Symposium and Open House*. University of Maryland, June 1995.

[3] Victor Basili, Scott Green, Oliver Laitenberger, Forrest Shull, Sivert Sorumgard, and Marvin Zelkowitz. The empirical investigation of perspective-based reading. *Empirical Software Engineering*, 1(2):133–164, 1996.

[4] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Goal Question Metric paradigm. In *Encyclopedia of Software Engineering*, pages 528–532. John Wiley & Sons, 1994.

[5] Victor R. Basili and H. Dieter Rombach. The TAME project: Towards improvement-oriented software environment. *IEEE Transactions on Software Engineering*, 14(6):758–773, June 1988.

[6] Randolph G. Bias. The pluralistic usability walkthrough: Coordinated empathies. In Jakob Nielsen and Robert L. Mack, editors, *Usability Inspection Methods*, chapter 3, pages 63–76. John Wiley, 1994.

[7] Robert N. Britcher. Using inspections to investigate program correctness. *IEEE Computer*, 21(11):38–44, Nov. 1988.

[8] Donald T. Campbell and Julian C. Stanley. *Experimental and Quasi-Experimental Designs for Research.* Houghton Mifflin Company, 1966.

[9] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction.* Laurence Erlbaum Associates, 1983.

[10] Gilbert Cockton, Steven Clarke, and Philip Gray. Theory of context influence the system abstractions used to design interactive systems. In *People and Computers X, Proceedings of HCI'95.* Cambridge University Press, August 1995.

[11] Heather W. Desurvire. Faster, cheaper!! Are usability inspection methods as effective as empirical testing? In Jakob Nielsen and Robert L. Mack, editors, *Usability Inspection Methods*, chapter 7, pages 173–202. John Wiley & Sons, Inc., 1994.

[12] Heather W. Desurvire, Jim M. Kondziela, and Michael E. Atwood. What is gained and lost when using evaluation methods other than empirical testing. In A. Monk, D. Diaper, and M. D. Harrison, editors, *People and Computers VII*, pages 89–102. Cambridge University Press, 1992.

[13] E.S. Edington. *Randomization Tests.* Marcel Dekker Inc., New York, NY, 1987.

[14] H. Rex Hartson et. al. Remote evaluation: The network as an extension of the usability laboratory. In *Proceedings of CHI'96*, pages 228–235. ACM SIGCHI, April 1996.

[15] James Foley et al. *Computer graphics : principles and practice.* Addison-Wesley, 2nd edition, 1990.

[16] Michael E. Fagan. Design and code inspection to reduce errors in program development. *IBM Systems Journal*, 15(3):182–211, 1976.

[17] Norman Fenton, Shari Lawrence Pfleeger, and Robert Glass. Science and substance: a challenge to software engineers. *IEEE Software*, 27(7):86–95, July 1994.

[18] Limin Fu, Gavriel Salvendy, and Lori Turley. Who finds what in usability evaluation. In *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting*, pages 1341–1345, Oct. 1998.

[19] Wilbert O. Galitz. *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques.* John Wiley & Sons, Inc., 1997.

[20] Cathy Gunn. An example of formal usability inspections in practice at Hewlett-Packard company. CHI'95 Conference Interactive Poster, May 1995.

[21] James Hom. The usability methods toolbox. http://www.best.com/~jthom/usability/usable.htm, 1996.

[22] R. Jeffries, J. R. Miller, C. Wharton, and K. M. Uyeda. User interface evaluation in the real world: A comparison of four methods. In *ACM CHI'91 Conference Proceedings*, pages 261–266. ACM, 1991.

[23] Robin Jeffries. The role of task analysis in the design of software. In *Handbook of Human-Computer Interaction*. Elsevier Prss, 1997.

[24] Bonnie E. John and David E. Kieras. The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction*, 3(4):320–351, December 1996.

[25] Michael J. Kahn and Amanda Prail. Formal usability inspection. In Jakob Nielsen and Robert L. Mack, editors, *Usability Inspection Methods*, chapter 6, pages 141–172. John Wiley & Sons, 1994.

[26] David E. Kieras. Towards a practical GOMS model methodology for user interface design. In J. Psotks, L. D. Massey, and S. Mutter, editors, *The handbook of human-computer interaction*, pages 135–158. North-Holland, Amsterdam, 1988.

[27] John C. Knight and E. Ann Myers. Phased inspection and their implementation. *ACM SIGSOFT Software Engineering Notes*, 16(3):29–35, July 1991.

[28] Masaaki Kurosu, Masamori Sugizaki, and Sachiyo Matsuura. Structured heuristic evaluation. In *Proceedings of the Usability Professionals' Association Conference*, pages 3–5, June 1998.

[29] Thomas K. Landauer. *The Trouble with Computers: Usefulness, Usability, and Productivity*. The MIT Press, 1995.

[30] Rohit Mahajan and Ben Shneiderman. Visual & textual consistency checking tools for graphical user interfaces. *IEEE Transactions on Software Engineering*, 23(11):722–735, Nov. 1997.

[31] Vahid Mashayekhi, Janet M. Drake, Wei-Tek Tsai, and John Riedl. Distributed, collaborative software inspection. *IEEE Software*, 10(5):66–75, Sep. 1993.

[32] Vahid Mashayekhi and John Riedl. CAIS: Collaborative asynchronous inspection of software. In *Second ACM SIGSOFT Symposium on the Foundation of Software Engineering*, Dec. 1994.

[33] Andrew Monk, Peter Wright, Jeanne Haber, and Lora Devenport. *Improving Your Human-Computer Interface: A Practical Technique*. Prentice-Hall, 1993.

[34] B.A. Myers and M.B. Rosson. Survey on user interface programming. In *ACM CHI'92 Conference Proceedings*, pages 195–202, May 1992.

[35] Jakob Nielsen. Paper versus computer implementations as mockup scenarios for heuristic evaluation. In D. Diaper et al., editor, *Human-Computer Interaction – INTERACT'90*, pages 315–320. IFIP, 1990.

[36] Jakob Nielsen. Finding usability problems through heuristic evaluation. In *ACM CHI'92 Conference Proceedings*, pages 373–380. ACM, 1992.

[37] Jakob Nielsen. *Usability Engineering*. Academic Press, Inc., San Diego, California, 1993.

[38] Jakob Nielsen. Heuristic evaluation. In Jakob Nielsen and Robert Mack, editors, *Usability Inspection Methods*, chapter 2, pages 25–62. John Wiley, 1994.

[39] Jakob Nielsen and Robert Mack, editors. *Usability Inspection Methods*. John Wiley & Sons, Inc., 1994.

[40] Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *ACM CHI'90 Conference Proceedings*, pages 249–256. ACM, 1990.

[41] Jakob Nielsen and Victoria L. Phillips. Estimating the relative usability of two interfaces: Heuristic, formal, and empirical methods compared. In *ACM INTERCHI'93 Conference Proceedings*, pages 214–221. ACM, April 1993.

[42] Nolan. Remote inspection at vertical research. http://www.nolan.com/serv/rem_insp.html, 1995.

[43] Donald A. Norman. *The Design of Everyday Things*. Basic Books, New York, 1st doubleday/currency edition, 1988.

[44] International Standards Organization. *ISO/IEC 9126. Information technology – Software product evaluation – Quality characteristics and guidelines for their use*. 1991.

[45] International Standards Organization. *ISO 9241. Ergonomics requirements for office work with visual display terminals*. 1994.

[46] David Parnas and David M. Weiss. Active design reviews: Principles and practices. In *Proceedings of the 8th International Conference on Software Engineering*, pages 132–136, Washington, D.C., August 1985. IEEE.

[47] J. M. Perpich, D. E. Perry, A. A. Porter, L. G. Votta, and M. W. Wade. Anywhere, anytime code inspection: Using the web to remove inspection bottlenecks in large-scale software development. In *19th International Conference on Software Engineering*, pages 14–21. IEEE, May 1997.

[48] Adam A. Porter, Lawrence G. Votta, Jr., and Victor R. Basili. Comparing detection methods for software requirements inspections: A replicated experiment. *IEEE Transactions on Software Engineering*, 21(6):563–575, June 1995.

[49] John Rieman, Susan Divies, D. Charles Hair, and Mary Esemplare. An automated cognitive walkthrough. In *ACM CHI'91 Conference Proceedings*, pages 427–428, 1991.

[50] Richard Rubinstein and Harry Hersh. *The Human Factor: Designing Computer Systems for People.* Digital Press, 1984.

[51] G. Michael Schneider, Johnny Martin, and W. T. Tasi. An experimental study of fault detection in user requirements documents. *ACM Transactions on Software Engineering and Methodology*, 1(2):189–204, Apr. 1992.

[52] Jean Scholtz. WebMetrics: A methodology for producing usable web sites. In *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting*, page 1612, Oct. 1998.

[53] Carolyn B. Seaman. *Organizational Issues in Software Development: An Empirical Study of Communication.* PhD thesis, University of Maryland at College Park, 1996.

[54] Carolyn B. Seaman and Victor R. Basili. An empirical study of communication in code inspections. In *Proceedings of the 19th International Conference on Software Engineering*, pages 96–106. IEEE, May 1997.

[55] Andrew Sears. Using automated metrics to design and evaluate user interface. Technical Report 94-002, DePaul University-Department of Computer Science, 1994.

[56] Ben Shneiderman. *Designing the User Interface : Strategies for Effective Human-Computer Interaction.* Addison-Wesley, 2nd edition, 1992.

[57] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison-Wesley, 3rd edition, 1998.

[58] Michael Stein, John Riedl, Soren J. Harner, and Vahid Mashayekhi. A case study of distributed, asynchronous software inspection. In *19th International Conference on Software Engineering*, pages 107–117. IEEE, May 1997.

[59] Linda Tetzlaff and David R. Schwartz. The use of guidelines in interface design. In *ACM CHI'91 Conference Proceedings*, pages 329–333. ACM, April 1991.

[60] Henirik Thovtrup and Jakob Nielsen. Assessing the usability of a user interface standard. In *ACM CHI'91 Conference Proceedings*, pages 335–341. ACM, April 1991.

[61] Cathleen Wharton, John Rieman, Clayton Lewis, and Peter Polson. The cognitive walkthrough method: A practitioner's guide. In Jakob Nielsen and Robert L. Mack, editors, *Usability Inspection Methods*, chapter 5, pages 105–140. John Wiley & Sons, Inc., 1994.