

Mixed-Initiative Support for Customizing Graphical User Interfaces

by

Andrea Bunt

BSCH., Queen's University, 1999
M.Sc., The University of British Columbia, 2001

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

The Faculty of Graduate Studies

(Computer Science)

The University Of British Columbia

2007

© Andrea Bunt 2007

Abstract

Graphical user interfaces (GUIs) are becoming increasingly complex, motivating research into ways of providing users with interfaces that are customized or personalized to their individual needs. Two opposing approaches to interface customization that have received the most attention to date are *adaptable* and *adaptive* approaches. An adaptable approach places the user in charge of customizing the interface, whereas with an adaptive approach, the system performs the customization automatically. Since both the adaptive and adaptable approaches have unique advantages and disadvantages, this thesis investigates a *mixed-initiative* solution to interface customization that aims to maximize each of their advantages, while minimizing their disadvantages.

As our first step, we conducted an exploratory experiment with simulated users. Using GOMS analysis, we evaluated the benefits of an appropriately customized interface. We also identified ways in which adaptive support could help users customize more efficiently, and identified user and task factors that impact effective customization.

Based on the results of our simulation experiment, we designed and implemented the MICA (Mixed-Initiative Customization Assistance) system. MICA provides users with a facility to customize their interfaces according to their needs, but also provides them with system-controlled adaptive support to help them customize effectively. MICA's adaptive support relies on a novel application of GOMS analysis to reason about the potential performance implications of different customization decisions. Using this formal reasoning, MICA generates customization recommendations aimed at maximizing the user's performance with the interface. MICA also communicates predicted time savings to the user in its rationale component, which describes *why* and *how* MICA makes recommendations.

We evaluated our mixed-initiative approach through two user studies. Study One examined the general benefits of MICA's approach in comparison to a purely adaptable alternative. The results indicate that users prefer MICA's

support to customizing independently, that MICA's support positively impacts performance with the interface (in terms of time on task), and that MICA reduces customization time. Study Two explored the utility of MICA's rationale. With a focus on qualitative data, the study revealed that the majority of users wish to have access to the rationale for reasons such as increased understanding and predictability of MICA's recommendations and increased trust in the system. The study also indicated that not all users want access to the rationale, suggesting that fine-grained transparency and predictability may not be important to all users in all contexts. Since previous work has advocated the importance of interaction transparency and predictability, the results of Study Two suggest that it would be beneficial to gain a more general understanding of when and why rationale is useful.

Table of Contents

- Abstract** iii
- Table of Contents** v
- List of Tables** xi
- List of Figures** xiii
- Acknowledgements** xv
- Dedication** xvii

- 1 Introduction** 1
 - 1.1 Motivation 1
 - 1.2 Thesis Approach 2
 - 1.3 Thesis Goals and Overview 3
 - 1.3.1 Overview 4
 - 1.4 Summary of Thesis Contributions 5
 - 1.5 Thesis Outline 6

- 2 Background and Related Work** 7
 - 2.1 Overview 7
 - 2.2 Categories of Customized Interactions 7
 - 2.2.1 Types of GUI Customization 9
 - 2.2.2 Types of Content Customization 10
 - 2.3 Approaches to GUI Customization 12
 - 2.3.1 Adaptable Approaches 12
 - 2.3.2 Adaptive Approaches 14
 - 2.3.3 Mixed-Initiative Approaches 16
 - 2.4 Direct Empirical Comparisons of the Different Approaches 19
 - 2.4.1 Adaptive vs. Adaptable 19

Table of Contents

2.4.2	Adaptive vs. Static	22
2.4.3	Mixed-Initiative vs. Adaptable	24
2.4.4	Other Empirical Comparisons of Interest	25
2.5	Other Examples of Mixed-Initiative Interactions	26
2.5.1	Common Domains	26
2.5.2	Styles of Mixed-Initiative Interactions	26
2.5.3	Relevant Evaluations	29
2.6	Rationale: Exposing the Adaptive System to the User	30
2.6.1	Open User Modelling	30
2.6.2	Explaining Adaptive Decisions	31
2.7	Summary	32
3	Proof-of-Concept Experiment with Simulated Users	35
3.1	Objectives and Approach	35
3.2	Customization Mechanism	36
3.3	Using GOMS Analysis to Evaluate Customization	40
3.4	Details of Our GLEAN Simulation	41
3.4.1	Tasks	41
3.4.2	Factors Influencing Performance: Extending GLEAN's Visual Search	41
3.5	Simulation Experiments	45
3.5.1	Experiment 1: Is Customization Worth the Effort and When?	45
3.5.2	Experiment 2: What to Customize	48
3.5.3	Cognitive Overhead	54
3.6	Implications	55
3.6.1	Is Customization Worth the Effort?	56
3.6.2	Do Users Need Help to Customize Efficiently? If Yes, How Can We Provide Adaptive Support?	56
3.7	Beyond Performance Data	57
3.8	Summary	58
4	MICA: Mixed-Initiative Customization Assistance	59
4.1	Overview of the MICA System	59
4.2	Framework: Overview	60
4.3	Framework: Individual Components	61
4.3.1	Customization Support Module (CSM)	61

4.3.2	User Model	68
4.3.3	Knowledge Base	70
4.4	Framework Implementation	74
4.4.1	Choice of Target Application	74
4.4.2	User Model Status	76
4.5	Delivering the Adaptive Support	79
4.5.1	Mixed-Initiative Customization Interface	80
4.5.2	MICA's Rationale Component	82
4.6	Summary	86
5	Study One: Comparing MICA to the Adaptable Alternative	87
5.1	Study Objectives and Approach	87
5.2	Method	88
5.2.1	Participants	88
5.2.2	Design	90
5.2.3	Apparatus	90
5.2.4	Tasks	90
5.2.5	Procedure	93
5.3	Pilots	94
5.4	Measures	95
5.5	Results	97
5.5.1	Performance	97
5.5.2	Customization Behaviour	99
5.5.3	Impact of Recommendations on Customization Decisions	101
5.5.4	Methods Used to Follow Recommendations	102
5.5.5	Interface Preference	103
5.5.6	Reasons for Customizing	104
5.5.7	Feelings Towards Recommendations	106
5.5.8	Impressions of the Rationale	106
5.5.9	Feature Keen/Shy Differences	108
5.6	Discussion	109
5.6.1	Effectiveness of MICA's Mixed-Initiative Support	109
5.6.2	Study Methodology	111
5.7	Summary	117

Table of Contents

6	Study Two: Understanding the Utility of the Rationale	119
6.1	Study Objectives and Approach	119
6.2	Improving the Interface	120
6.2.1	Informal Evaluation	120
6.2.2	Resulting Rationale and Customization Interfaces	121
6.3	Method	126
6.3.1	Participants	126
6.3.2	Design	128
6.3.3	Apparatus	129
6.3.4	Tasks	129
6.3.5	Procedure	130
6.4	Pilots	131
6.5	Measures	131
6.5.1	Qualitative Measures	131
6.5.2	Quantitative Measures	132
6.5.3	Measures Pertaining to Framework Assumptions	133
6.6	Results	133
6.6.1	Impact of Rationale on Qualitative Measures	133
6.6.2	Impact of Rationale on Quantitative Measures	141
6.6.3	Further Exploration of Framework Assumptions	146
6.7	Discussion	148
6.7.1	Rationale Impact and Effectiveness	148
6.7.2	Study Methodology	149
6.7.3	Validity of MICA's Assumptions	150
6.8	Summary	151
7	Conclusions	153
7.1	Thesis Contributions	153
7.1.1	Analysis of Factors Influencing the Effectiveness of User- Controlled Customization	153
7.1.2	Framework for Providing Principled Customization Sup- port	154
7.1.3	Design of a Mixed-Initiative Customization Interface	154
7.1.4	Identification of the Benefits of a Mixed-Initiative Ap- proach to Customization	155
7.1.5	Identification of the Benefits of Rationale Within a Sys- tem for GUI Customization	155

7.1.6	Increased Understanding of How to Evaluate Customiza- tion in a Laboratory Environment	156
7.2	Directions for Future Research	156
7.2.1	Extensions to MICA	156
7.2.2	Further Evaluation	158
7.3	Concluding Comments	160
References	161

Appendices

A	Preliminary Questionnaire	175
B	Expertise Questionnaires	181
B.1	Menu Items	181
B.1.1	Instructions	181
B.1.2	Sample Question	182
B.2	Toolbar Items	182
B.2.1	Instructions	182
B.2.2	Sample Question	183
C	Study One: Call for Participation	185
D	Study One: Instructions	187
D.1	Instructions Provided at the Beginning of the Session	187
D.2	Instructions Provided Prior to the First Task	187
D.3	Instructions Provided After All Repetitions of the First Task are Complete	188
E	Study One: Tasks	191
E.1	Task A	191
E.1.1	Instructions	191
E.1.2	Files Provided to Start the Task	193
E.1.3	Final Product as a Result of Following the Instructions	194
E.2	Task B	196
E.2.1	Instructions	196
E.2.2	Files Provided to Start the Task	197
E.2.3	Final Product as a Result of Following the Instructions	198

Table of Contents

F Study One: Post Questionnaire	199
G Study One: Sample Interview Questions	205
H Analysis of Study One’s Preliminary Questionnaires	209
I Study Two: Call for Participation	213
J Study Two: Instructions	215
J.1 Instructions Provided at the Beginning of the Session	215
J.1.1 NAART Instructions	215
J.2 Instructions Provided Prior to the First Task	215
J.3 Instructions Provided After All Repetitions of the First Task are Complete	217
K Study Two: Tasks	219
K.1 Task A	219
K.1.1 Instructions	219
K.1.2 Files Provided to Start the Task	222
K.1.3 Final Product as a Result of Following the Instructions	223
K.2 Task B	225
K.2.1 Instructions	225
K.2.2 Files Provided to Start the Task	228
K.2.3 Final Product as a Result of Following the Instructions	229
L Study Two: Post Questionnaire	231
M Study Two: Sample Interview Questions	237
N Study Two: Further Performance Results	241
N.1 Effects of Task on Performance Measures	241
N.2 Effects of Version on Performance	241
O UBC Research Ethics Board Certificates	245

List of Tables

- 3.1 Task simulated by our GOMS models in the simulation experiment. 42
- 3.2 Customization times in the simulation experiment (Experiment 1). 48
- 3.3 Independent variables in the simulation experiment (Experiment 2). 50
- 3.4 Comparison of customization strategies in the simulation experiment (Experiment 2), not including customization time. 51
- 3.5 Impact of ratio and expertise in the simulation experiment (Experiment 2), not including customization time. 52
- 3.6 Comparison of customization strategies in the simulation experiment (Experiment 2), including customization time 54
- 3.7 Impact of ratio and expertise in the simulation experiment (Experiment 2), including customization time. 54

- 5.1 Description of the participants in Study One. 89
- 5.2 Comparison of the tasks in Study One. 93
- 5.3 Main quantitative results in Study One. 98
- 5.4 Overall Performance results in Study One. 98
- 5.5 Task Performance results in Study One 98
- 5.6 Customization Time results in Study One. 99
- 5.7 Customization Sessions results in Study One. 100
- 5.8 Features Added results in Study One. 100
- 5.9 Methods used to follow recommendations in Study One. 103
- 5.10 Reasons for customizing in Study One. 105
- 5.11 Feelings towards recommendations in Study One 106

- 6.1 Description of the participants in Study Two. 127
- 6.2 Comparison of the tasks in Study Two. 130
- 6.3 Reasons for viewing the rationale in Study Two. 136

List of Tables

6.4	Reasons for finding the “How” component useful or not useful in Study Two.	138
6.5	Percentage of recommendations followed in Study Two.	142
6.6	Methods used to follow recommendations in Study Two	144
6.7	Performance and customization behaviour results in Study Two.	145
H.1	Feature Profile Scale reliability in McGrenere and Moore’s study and in Study One.	212
N.1	Performance results according to Task in Study Two.	241
N.2	Overall Performance results according to Task in Study Two.	242
N.3	Task Performance results according to Task in Study Two.	242
N.4	Overall Performance results according to Version in Study Two.	243
N.5	Task Performance results according to Version in Study Two.	243

List of Figures

- 2.1 Overview of related work. 8

- 3.1 Example of the two-interface model. 37
- 3.2 Main adaptable dialogue box 38
- 3.3 Adaptable customization screen for adding features 39
- 3.4 Adaptable confirmation dialogue 39
- 3.5 Simulation experiment setup 46
- 3.6 Payoff results in the simulation experiment (Experiment 1). 47
- 3.7 Two dimensions to customization strategy. 49
- 3.8 Magnitude of differences between the customization strategies in the simulation experiment (Experiment 2). 55

- 4.1 MICA’s architecture 60
- 4.2 Factors influencing the User Model’s performance assessment. 69
- 4.3 Knowledge Base components 71
- 4.4 Main mixed-initiative dialogue box. 79
- 4.5 Mixed-initiative customization screen for adding features (used in Study One). 80
- 4.6 “System Recommendations” screen (used in Study One). 81
- 4.7 Customization screen with the rationale expanded (used in Study One). 83
- 4.8 “How” component of the rationale interface (used in Study One). 84

- 5.1 How well additions matched MICA’s recommendations in Study One. 101
- 5.2 Preference results in Study One. 104

- 6.1 Customization interface for adding features in Study Two. 121
- 6.2 “Show Add Recommendations” interface in Study Two. 122
- 6.3 “System Recommendations” screen in Study Two. 122

List of Figures

6.4	Customization interface with the rationale expanded in Study Two.	123
6.5	“How” component of the rationale in Study Two.	124
6.6	No-Rationale interface in Study Two.	128
6.7	Preference results in Study Two.	134
6.8	Whether the “Why” component was motivating in Study Two. .	137
6.9	The usefulness of the individual “How” factors in Study Two . .	139
H.1	Feature Profile distribution in McGrenere and Moore’s study and in Study One.	211
N.1	Interaction between Overall Performance and Task Order in Study Two.	242

Acknowledgements

First I would like to thank my supervisors, Dr. Cristina Conati and Dr. Joanna McGrenere, for their support, insight, and guidance throughout my degree. I have learned an enormous amount from working with them in terms of how to be both a researcher and a supervisor. Cristina's careful attention to detail has taught me to always make sure that I can justify my decisions and choices. Working with her has also taught me how to write scientifically and has improved the quality of my writing. Joanna has been an excellent resource for study design and analysis, but more importantly is a great example of how to be a mentor. Joanna always does everything in her power to help her students meet their deadlines, even when they are self-imposed. I hope to follow her example in the future.

The members of my supervisory committee, Dr. Alan Mackworth and Dr. David Poole, provided valuable feedback at many steps along the way. I always left meetings with my committee feeling confident that I was on the right path. Finally, comments from the additional members of my examining committee, Dr. Gail Murphy, Dr. Izak Benbasat, and Dr. Frank Linton, greatly improved the quality of this thesis.

A number of sources funded my research. In particular, I would like to thank NSERC, the University of British Columbia, IBM, and Precarn for the financial support that they provided either directly or through grants awarded to Cristina and Joanna.

UBC has a great research culture and I have been fortunate to be a member of two research groups here: the Laboratory for Computational Intelligence (LCI) and the Interaction Design Reading Group (IDRG). Individuals in these groups provided feedback on numerous occasions on technical aspects of my work, practice presentations, and paper drafts. There are too many people to list here, but Karyn Moffatt, Leah Findlater, Jen Gluck, Heather Maclaren, Saleema Amershi, Giuseppe Carenini, and Tony Tang all deserve a special thank you. Jen Gluck also implemented a logger that I used in Studies One and Two.

Acknowledgements

My UBC experience would not have been the same without Kasia Muldner. It is really impossible to thank Kasia for everything she has done for me throughout my time in Vancouver. Kasia has been a sounding board to pretty much all of my ideas, supported me when things were not going well, read more drafts and attended more practice talks than I can count, and was even a bridesmaid at my wedding.

My parents Rick and Gail Bunt have supported me throughout this process, helping me on the financial side, listening to my panic attacks and even providing excellent proof-reading services. My in-laws Yves and Christiane Durocher have also been incredibly supportive and encouraging.

Finally, there is a good chance that I would not have made it to the end of this degree (and kept my sanity) if it weren't for my husband Steph Durocher. Steph supported me at all times in a loving, empathetic, optimistic, calm, and patient manner. It was a difficult last year while we lived across the country from one another. But it also gave me that extra bit of motivation to work those long hours at the end so that we could move on with our lives together.

Dedication

To Steph, for all of your love, encouragement and patience.

Dedication

Chapter 1

Introduction

1.1 Motivation

As users interact with software applications in today's computer-rich world, they are regularly made to cope with large and complex interfaces, and these interfaces are growing continually. One reason for the growth is that as companies release new versions of their software applications, there is a tendency to pack these applications with an increasing number of features. These feature-rich software applications have the potential to suit a wider range of individuals and thus are attractive from a marketing standpoint. This increase in functionality, however, has also been accompanied by an increase in the size and complexity of the graphical user interfaces (GUIs).

A wide variety of applications suffer from interface complexity, ranging from spreadsheet packages, to statistical analysis software, to image-editing software, all of which can have hundreds of features distributed throughout their menus and toolbars. These applications are often used by a diverse set of individuals, who differ not only in the types of tasks they wish to perform, but also in their application-specific knowledge and their general computer expertise. For the average user, this high degree of interface complexity translates into a visually cluttered interface, where the presence of excess features can lead to both frustration [93] and decreased performance [21]. Even seemingly straightforward productivity software, such as word processors, can suffer from these problems, as is evidenced by the number of researchers working on ways to assist users with this class of application (e.g., [36], [49], [62], [83], [92], [100]).

Providing the user with an interface customized to suit her needs could mitigate the problem of interface complexity, but how to best achieve an appropriately customized interface is a contentious issue (e.g., [108]). Two opposing ways to solve this problem are *adaptable* and *adaptive* approaches, which differ in terms of who is responsible for performing the customization (the user or the system) and consequently in the amount of control provided to the user.

Adaptable interfaces permit full user control by providing interface mechanisms that allow users to customize their own interfaces. System-controlled adaptive interfaces, on the other hand, perform the customization automatically. To perform this automatic customization, an adaptive system often relies on a *user model*, which stores relevant user-specific information such as work patterns or preferences.

Both adaptive and adaptable interfaces have advantages and disadvantages. With adaptable interfaces, users are in full control, but not all users want this control, and some are simply unwilling to invest the effort necessary to customize [86, 87]. In addition, the customization facility itself is an extra source of complexity within the interface [72]. Finally, even for those users who are willing to customize, it is unclear whether or not they are able to do so in a manner that increases their productivity. Adaptable approaches require the user to take on some of the role of an interface designer, and even though customization doesn't have to involve extensive interface restructuring, it may still be unfair to expect the average user to take on this responsibility. Adaptive interfaces, on the other hand, do not require any extra effort from the user. Their disadvantages, however, include a lack of user control, transparency and predictability (e.g., [58], [65]). Transparency refers to allowing the user to understand why the adaptive system is behaving the way it is, whereas predictability is the ability to use observations of past system behaviour to predict how the system will behave in the future. Poor transparency and predictability can lead to decreased trust in the system and lowered acceptance of the adaptive behaviour, and can negatively impact the user's performance with the interface [65].

1.2 Thesis Approach

Given the unique advantages and disadvantages of adaptive and adaptable approaches, the optimal solution likely involves a combination of the two. Solutions that combine system-controlled automation with user control through direct manipulation are commonly referred to as *mixed-initiative* [60]. Despite the potential for mixed-initiative approaches to strike the right balance between the adaptive and adaptable extremes, they have received little attention from Human-Computer Interaction and Artificial Intelligence research communities with respect to interface customization.

This thesis investigates a mixed-initiative solution to interface complexity

where users are provided with a customization facility that gives them control over the interface, but are also given access to system-controlled adaptive support designed to help them take full advantage of this facility. The system-controlled adaptive support is based on information about user-specific properties stored in a user model, as well as characteristics of the interface being customized. Both sources of information are combined in a formal reasoning process whose output is a set of tailored customization suggestions. By providing this type of adaptive support, we aim to increase the benefits of user-controlled customization in two ways. First, the system tries to help the user create the best possible customized interface for her needs. Second, the system decreases the amount of effort necessary to customize, particularly for those users who are willing to allow the system to do the bulk of the work.

By letting users make the final decision on when and how to customize, our mixed-initiative approach addresses one of the main drawbacks in purely adaptive approaches – lack of user control. With a mixed-initiative approach, however, if users do not understand why and how the customization suggestions are generated, two potential disadvantages of adaptive interfaces remain: lack of transparency and lack of predictability. Our mixed-initiative system addresses these issues by providing the user with access to the rationale underlying the customization suggestions.

1.3 Thesis Goals and Overview

The goal of this thesis is to show that through careful design of both the interface and the underlying adaptive architecture, a mixed-initiative approach has the potential to draw on the strengths of each of the adaptable and adaptive approaches, while avoiding or mitigating their disadvantages. Satisfying this high-level goal involves answering the following questions:

1. What are the performance benefits of customization and in what ways could mixed-initiative support help users customize more effectively?
2. What characteristics of the users and their tasks impact effective customization?
3. How should a mixed-initiative system provide its support?
 - (a) How should the system reason about relevant user and task characteristics to generate adaptive customization support?

- (b) How should the system communicate its customization suggestions to the user to strike the right balance between the adaptive and adaptable approaches?
- 4. What are the quantitative and qualitative benefits of the resulting mixed-initiative approach?

To answer these questions, we first conducted a proof-of-concept experiment with simulated users to motivate the general approach and to guide the development of the adaptive support. We then designed and implemented a mixed-initiative system known as the MICA (**M**ixed-**I**nitiative **C**ustomization **A**ssistance) system. Finally, we conducted two user evaluations (Study One and Study Two) aimed at assessing the qualitative and quantitative advantages and disadvantages of MICA's approach.

1.3.1 Overview

Our proof-of-concept simulation experiment (chapter 3) addresses the first two research questions. To motivate the idea of supporting user customization, we provide a formal analysis of the performance benefits provided by a suitably customized interface. We first identify an appropriate performance metric and use this metric to analyze the impact of various user-specific properties on effective customization. We use this exploratory analysis to inform the design of our mixed-initiative approach.

Our third question is addressed in chapter 4, where we present the details of the MICA system, including how it generates tailored customization recommendations and how it delivers the recommendations to the user. MICA bases its recommendations on a formal quantification of performance by relying on on-line GOMS analysis [20]. In addition to forming the basis of MICA's recommendations, the performance savings predicted by this formal analysis is communicated to users within MICA's rationale component, which also describes how the recommendations are generated.

We address the fourth thesis question through two evaluations. Study One is our first evaluation of the MICA system (chapter 5), where we compare MICA's mixed-initiative support to the purely adaptable alternative. Through a combination of quantitative and qualitative analysis, we provide initial evidence that users prefer the mixed-initiative approach to customizing on their own, and that MICA's support has a positive impact on performance and aspects of cus-

tomization behaviour. Because the adaptable interface that we use in the study performed well in a previous field evaluation [91, 92] and was preferred to an adaptive alternative, our results demonstrate the potential for mixed-initiative interfaces to strike the right balance between the adaptive and adaptable approaches.

Our second evaluation, Study Two (chapter 6), explores the impact of providing the user with MICA’s rationale on measures such as trust, system understandability and predictability. While our focus is on the qualitative impact of the rationale, we also measure its impact on customization-related quantitative measures. Results show that not all users want access to the rationale, but for many users, the rationale increases feelings of trust, transparency and predictability.

1.4 Summary of Thesis Contributions

The core contributions of this thesis research are summarized below. We provide a high-level description of the contributions here and elaborate in our Conclusions (chapter 7).

1. We provide the first in-depth analysis of the performance benefits of customization. Our work identifies factors that influence effective customization, such as task composition and user expertise.
2. Our MICA system is the only mixed-initiative system for GUI customization to reason about the performance implications of customization by relying on formal techniques. While other work on supporting customization also aims to save the user time, MICA is the only system to predict the potential savings formally and use this prediction as a basis for its recommendations.
3. There have been very few detailed evaluations of mixed-initiative approaches. Our evaluation comparing MICA to a purely adaptable alternative is one of the first comparisons between a mixed-initiative and an adaptable approach to interface customization. Our evaluation is more in-depth than previous work and is the first to obtain direct information on whether users prefer to have the support over having to customize on their own.

4. Finally, MICA is the first system in the context of GUI customization to provide users with access to its rationale. Through a formal evaluation, we show how the provision of rationale impacts users' impressions of the system and the manner in which they respond to system recommendations.

We also claim three secondary contributions. The first is the design of MICA's mixed-initiative interface, which presents recommendations non-intrusively and provides the user with varying levels of control. The second is a better understanding of how to conduct laboratory evaluations involving customization. We discuss a number of different study methodology issues that we encountered when designing our two formal evaluations and the lessons that we learned. Finally, while performing our exploratory analysis of the performance benefits of customization, we extended GOMS analysis, which is designed to model expert behaviour, to account for an aspect of user expertise.

1.5 Thesis Outline

We begin by presenting background and previous work related to GUI customization. Chapter 3 describes our proof-of-concept simulation experiment. In chapter 4, we describe the MICA system, including its framework and mixed-initiative customization interface. Our first evaluation of MICA (Study One) is discussed in chapter 5. Study Two, which targets the rationale, is presented in chapter 6. Chapter 7 summarizes the thesis work, highlights its contributions in greater detail, and outlines areas for future research. Finally, there are a number of appendices included at the end of the thesis. These appendices consist of materials used in Study One and Study Two as well as supplementary analysis from the two evaluations.

Chapter 2

Background and Related Work

2.1 Overview

Chapter 2 describes research related to supporting GUI customization. We begin by distinguishing between two different forms of customization: GUI customization and content customization. Next, we present research related to four topics, which are illustrated in figure 2.1. Our first topic is previous research on the three main approaches to GUI customization: adaptive, adaptable, and mixed-initiative. We then describe direct empirical comparisons of the different approaches, which provide indications of their relative merits and illustrate the potential for mixed-initiative solutions. Next, we describe research on mixed-initiative interactions outside of this context, to give the reader a sense of the range of types of system-user collaboration. Finally, we overview work on rationale provision within adaptive systems.

2.2 Categories of Customized Interactions

There are numerous ways to provide users with interactions that are customized (these are also known as *personalized* interactions). For the purpose of this thesis, we divide interaction customizations into two categories: (1) graphical user interface (GUI) customization and (2) content customization. We define GUI customization as involving adaptations to the control structures that provide users with access to an application's functionality. More specifically, this type of customization is typically performed on visual, clickable features such as menu items, toolbar items, and buttons. We refer to content customization, on the other hand, as the adaptation of domain- or application-related information. Customization of this sort can involve tailoring subject matter, movie or prod-

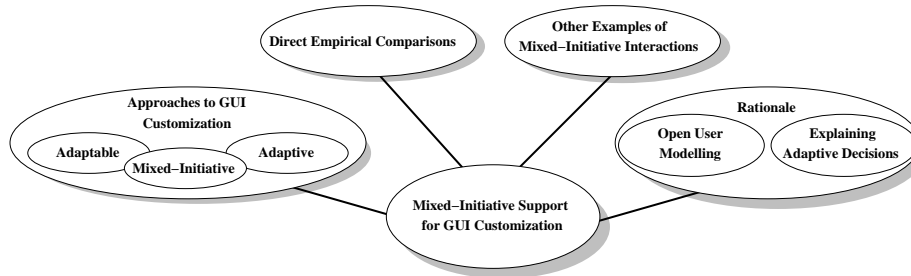


Figure 2.1: Work related to mixed-initiative support for GUI customization.

uct recommendations, or application-specific assistance (e.g., how to use the application’s functionality).

We believe that these two classes of personalized interactions (GUI vs. content) have their own unique issues and challenges. The need for content personalization and the usefulness of adaptive or mixed-initiative approaches to content personalization tends to be more accepted in the research community. These approaches have been especially successful in the area of Intelligent Tutoring Systems, which provide tailored delivery of educational material to support human learning (e.g., [3], [30], and [109]), and in the area of recommender systems, which provide tailored recommendations of products, items, or news stories of interest (e.g., [6], and [22]). Dominant issues influencing the different levels of acceptance of content versus GUI customization seem to be user expectations and the importance of consistency. Content is always changing in any sort of dynamic presentation, and as a result, users might expect that adaptations will occur in such systems. On the other hand, dynamic adaptation of features in a GUI is comparatively rarer and so there might be a general expectation that a GUI will remain static. Regardless of expectations, consistency – an established principle of good interface design – is more critical in a GUI than in a presentation of content. Consistent positioning of features in a GUI enables users to become more proficient with the interface over time. With practice, users are able to remember where features are located within the interface and can eventually begin to automate their motor responses (see Norman’s *The Psychology of Menu Design* for a discussion [98]). Maintaining consistency in an adaptive interface, where the GUI is being customized dynamically, can be a challenge.

Because of the above differences, we distinguish between content and GUI customization in terms of our coverage and categorization of related work and

how we frame the thesis contributions. Our related work focuses on GUI customization while occasionally highlighting work on content customization. There are, however, cases where the line between content and the GUI is blurred. Hyperlinks could be considered one of those cases, since they can both act as control structures (a way to navigate through the information space) and contain content. For simplicity, we group this work in the content category, but realize that it is somewhat unique within that category. Similarly, there are examples of customization that pertain to neither GUIs nor content, such as the customization of input devices (e.g., [117]). We cover the customization of hyperlinks briefly in section 2.2.2, but do not cover input-device customization.

2.2.1 Types of GUI Customization

GUIs can be customized in a number of different ways. For the purpose of this work, we distinguish between: macro definition, feature management, and cosmetic customization. While the distinction is not always clear-cut, a general understanding of the different categories helps position our work. In particular, these classes have different implications for: (i) the amount of user effort required to customize (given an adaptable interface), (ii) the size of the customized interface, and (iii) the type and amount of impact customization can have on user performance.

The first category, macro definition, involves adding new interface features, such as menu items, toolbars items, or buttons, that represent frequently executed sequences of commands (e.g., [89], [100], and [101]). Often these sequences of commands are parameterized (e.g., Font with New Times Roman, Size with 12 pt, and Bold). Macros can also be bound to the keyboard, in which case there is no impact on the size of the GUI.

Feature management involves the customization of existing interface features. One approach is to duplicate the functionality of existing interface elements, allowing access to this functionality from more convenient interface locations, such as toolbars as opposed to menus (e.g., [36] and [101]). Related to this category of customization are history mechanisms or reuse facilities (e.g., [55]). These mechanisms leverage the fact that in some applications, users tend to repeat previously invoked commands. Recently used features are duplicated and grouped in a separate interface area to enable users to re-access this functionality quickly. Rather than duplicating features, some approaches aim to help users manage existing interface elements by customizing the set of already

available GUI features. Examples include customizing the set of features present at any point in time (e.g., [92]), customizing the order in which features appear within menus (e.g., [106]), or customizing the type and layout of the widgets that contain the individual features (e.g., [48]).

Finally, cosmetic customization involves making fairly simple customizations to the appearance of an interface, such as minor changes to the placement of buttons, feature parameters (e.g., the zoom setting), and screen colours. Little research has focused solely on this type of customization, likely because it has more limited impact on user performance. Examples can be found in work that discusses this type of customization in conjunction with the more complex forms of customization discussed above (e.g., [50], [69], and [101]).

Of the different forms of customization, feature management tends to be more relevant to the problem of GUI complexity and is, therefore, the form of customization supported in this thesis. Depending on the particular customization strategy used, this type of customization has the potential to allow users to work from smaller versions or sections of a more complex interface. Even if the customization doesn't involve reducing the size of the interface, effective rearrangement of features can lessen the impact of interface complexity. In other words, the interface can be customized to the particular user such that unused or rarely used features have less of an impact on the user's ability to access the more frequently used features. The impact of macros on interface complexity is less straightforward. Creating macros can lead to an increase in the total number of different features available in the interface, causing its overall complexity to increase. On the other hand, when a macro represents a group of feature selections, it can reduce the number of times the user must search through the complex interface.

2.2.2 Types of Content Customization

There are numerous ways to customize content to an individual user. Since this is not the focus of the thesis, we only briefly outline some of the key dimensions (more detail on this topic can be found in [11]). The first relevant dimension is *which* content is delivered to the user, including customized product recommendations (e.g., [22]), topic selection in an adaptive web site (e.g., [102]), and material selection (e.g., [96]) or hint provision in an Intelligent Tutoring System (e.g., [13]). A second dimension of content customization is *how* content is presented, including the structure, layout, and modality used (e.g., [128]). It is

also possible to customize *when* content is delivered to the user (e.g., [62] and [66]).

As mentioned earlier, hyperlink customization could be considered to entail both GUI and content customization since hyperlinks contain domain-specific information and they act as control structures. Hyperlink customization can involve customizing which links are presented to the user, how they are displayed, and how they are structured (see [9] for a review). Another form of hyperlink customization, which is more towards the GUI end of the customization spectrum, is to create secondary interface structures such as bookmarking or history mechanisms that permit easy access to previously visited sites (e.g., [37], [70], and [73]).

Alternative Approaches to Interface Complexity

In this thesis we address the problem of interface complexity by helping users create customized interfaces. A complementary body of work, which resides in the category of content customization, focuses on helping users understand the available functionality in feature-rich interfaces, as opposed to reducing complexity through GUI customization (e.g., [62], [83], and [97]). The Lumière system [62] provides interface assistance when it believes that the user is having difficulty accomplishing a task or is using a less efficient method of task completion. The OWL system [83] monitors the feature usage of both an individual user and the user's peers to recommend features that will increase the individual user's knowledge of the available feature set. Finally, the Crystal system [97] allows users to ask "why" (or "why not") the application is responding in a particular manner to user input based on the interaction history. The answers to the "why" questions can help the user better understand the functionality that he is currently using. In response to "why not" questions, the system often highlights relevant interface features that could have led to the desired behaviour, potentially increasing the user's awareness of features not yet being used. While Lumière and OWL rely on Artificial Intelligence techniques to provide their support, Crystal's approach requires wide-scale instrumentation of the application's code base. For each command (i.e., any action in the application), the developer must specify a number of properties including the commands' dependencies on other commands. When a command is the result of a feature selection, developers must also ensure that it is notified of which interface feature issued the command.

2.3 Approaches to GUI Customization

In chapter 1 we introduced three common approaches to GUI customization: (i) *adaptable* approaches, which place the user in charge of customizing; (ii) *adaptive* approaches, where the system performs the customization on behalf of the user; and (iii) *mixed-initiative* approaches, which combine aspects of user control and system control. We now describe work related to each type of approach.

2.3.1 Adaptable Approaches

There are a number of examples of purely adaptable solutions to GUI customization within existing applications and environments. Field studies of these solutions have provided an indication of the amount and type of customization that users have been willing and able to engage in. Page *et al.* examined usage of customization mechanisms within WordPerfect 6.0 over a 28-day period [101]. They found that 92% of the 101 participants engaged in some form of customization (including cosmetic customization), 55% created short-cuts, and 16% created macros. Mackay investigated customization within the Athena software environment over the course of two studies (one with 51 participants, the other with 18 participants), and documented reasons why people do and do not customize their environments [86, 87]. While she found that 78% of users performed some sort of customization (including cosmetic customization), only 11% engaged in thorough and systematic customization of their environments. Common reasons for not customizing included being too busy or lacking the customization-related knowledge. Jorgensen and Sauer found that approximately 50% of users engaged in some form of cosmetic customization in studies of three different contexts (with 10, 720, and 27 participants): (i) the “IBM Assistant” (word-processing, spreadsheet, graphics, and database applications); (ii) a business application package; and (iii) an unidentified operating system [69].

To make comprehensive, feature-management customization more accessible and attractive to the average user, McGrenere *et al.* proposed a two-interface adaptable model for MSWord [92]. With the two-interface model a user can create a feature-reduced version of the full default MSWord interface using a lightweight customization mechanism; however, users can continue to access the full interface (via a toggle button). The two-interface model and customization mechanism were evaluated in a six-week field study with 20 participants.

The study revealed that participants liked the two-interface model and made extensive use of the personalized interfaces that they created. For example, the majority of participants (70%) spent the majority of their time working in their personal interfaces. The mixed-initiative approach we designed and evaluated in this thesis builds on this two-interface model.

McGrenere *et al.*'s study also revealed that participants employed a number of different strategies to customize their interfaces. We elaborate on these strategies here, since abstracted versions are discussed in chapter 3. All but one of the participants customized to some extent.¹ Results showed that 32% of these participants performed the vast majority of their customization at the beginning of the study – a strategy referred to as *Up Front*. The remaining participants (68%) customized in a more incremental manner. These participants preferred to wait until they needed to use a feature before they added it to their personal interfaces – a strategy referred to as *As You Go*. In terms of which features users added to their personal interfaces, 63% of those who customized chose to add all features they used, while 37% added only their most frequently used features. Once users added features to their personal interfaces they tended not to remove them, even if at some point the features were no longer needed. Since the evaluation also compared the adaptable two-interface model to an adaptive interface, we refer to it again in section 2.4.1.

While the above studies provide indications of how and why participants customize, there is little information on how user-controlled customization impacts performance with the interface. For example, McGrenere *et al.*'s work showed that users have different strategies for customizing [91, 92], but the relative effectiveness of these strategies was not tested. A field setting generally does not permit for this type of information to be gathered.

Additional examples of adaptable interfaces for which there is no reported evaluation include Maclean *et al.*'s purely adaptable macro creation facility [89] and Stuerzlinger *et al.*'s user interface façades [110]. Façades provide users with essentially unlimited ability to reconstruct their own version of an application's interface in a separate window. Users can change widget types, layout, and which features are present in the interface. While the proposed customization mechanism is powerful, it is yet to be evaluated, and so it is unclear how usable the mechanism is or how much effort users would be willing to expend to perform these types of extensive customization.

¹The remaining participant did customize a minimal amount at the beginning of the experiment, but subsequently used the full default interface almost exclusively.

Finally, layered interfaces (e.g., [24] and [107]) can also be considered a type of adaptable interface. Typically with layered interfaces, the application designer creates a set of pre-determined interfaces or layers (e.g., according to expertise or task), and users are allowed to switch between the layers as they see fit. Since layered interfaces (for the most part) place users in charge of transitioning between the layers, this type of solution does fit within the category of user-controlled GUI customization. However, unless the layers are augmented with a customization facility (as is the case with the two-interface model proposed by McGrenere *et al.* [92]), it is primarily an off-the-shelf solution. For a layered approach to be successful, the developer must design a set of layers that meets user needs. Designing an appropriate and manageable set of layers might become increasingly difficult to do as both the number of features in an application and the diversity of its users increases. As a result, the layered-interface approach may lack scalability. Furthermore, to date, the layered-interface approach has not been extensively evaluated. One exception is a recent study by Findlater and McGrenere, which compared two versions of a two-layered interface and showed that layered interfaces can have positive impacts on performance as compared to a full-featured interface [42].

2.3.2 Adaptive Approaches

Contrary to an adaptable approach, where the user is responsible for the customization, in an adaptive approach the system automatically customizes the GUI according to its understanding of the individual user's needs. A commercial example of a purely adaptive solution to GUI customization is SMART MENUS, introduced in MSWord 2000. When opened, a SMART MENU provides the user with a small version of the full menu, containing only the items that the system believes are the most frequently and/or recently used.² To access an item not present in the smaller menu, users can expand the menu to view its full contents. Unfortunately, when the menu is expanded to the full version, positional consistency is not maintained – the newly visible items are interspersed with those present in the small version. As a result, users have to re-scan the entire menu from top to bottom to find the desired item (as opposed to just the set of newly visible items). Further details of an evaluation comparing the SMART MENUS to McGrenere *et al.*'s adaptable two-interface model are discussed in

²We were unable to obtain a description of the algorithm. Observations of the adaptations suggest that they are based on recency and frequency.

section 2.4.1. The Start menu in Windows XP is another commercial example of an adaptive interface – it has an area devoted to system-created short-cuts to frequently used programs.

An example of an adaptive solution that maintains a higher degree of positional consistency is Sears and Shneiderman’s concept of split menus [106]. With a split menu, the entire menu is always visible, but the more frequently used features are moved to the top. Specifically, a split menu contains two partitions: a top portion containing the most frequently used items and a lower static portion containing the remaining items. A laboratory evaluation with 38 participants showed the general benefits of having the most frequently used items located at the top of the menu. The evaluation did not, however, test a variant where the items in the top half of the split were changing dynamically throughout the session according to usage. Instead, the top half of the split was pre-determined at the start of the session.

The SUPPLE system uses an optimization approach to perform comprehensive GUI adaptations based on the characteristics of the target device and the user’s usage patterns [47, 48]. The goal of the optimization is to provide the user with an interface that both meets the capabilities of the device and minimizes the user’s effort (measured as the number of clicks). The first evaluation of the system was an informal laboratory study, which verified that the system produces reasonable designs compared to those generated by four participants with experience in interface design (as judged by the authors) [48]. Based on feedback from a second laboratory evaluation with 16 participants [47], SUPPLE was extended to incorporate the idea of *partitioned dynamicity* [124]. The idea behind partitioned dynamicity is that consistency can be maintained by keeping part of the interface static, while another part adapts to the user. In SUPPLE, this style of interface is referred to as a split interface.

Gong and Salvendy proposed an adaptive interface where users are gradually “pushed” from a menu-based interface to the command line as they become more familiar with the command names [52]. After a certain number of uses of an individual menu item, the system highlights the item and requires the user to use the command line instead. The system then moves the menu item from the main menu to a hidden portion of the interface. An evaluation, which is described in section 2.4.2, showed that this strategy had potential to improve user performance.

Early research on adaptive interfaces, which tended to focus primarily on designing architectures rather than on evaluations, can be found in the books

Adaptive User Interfaces [8] and *Adaptive User Interfaces: Principles and Practice* [105]. In particular, the latter collection includes an overview of the state adaptive interfaces at the time of publication by Dieterich *et al.* [38]. In addition to surveying existing systems, Dieterich *et al.* outline a number of important issues that must be addressed when designing an adaptive interface, one of which is the manner in which control is divided between the system and the user. Of these early systems, one example is Cote-Munoz's AIDA system, which automatically creates macros on behalf of the user in a CAD application [31]. A second example is Malinowski's adaptive approach to form-based interfaces and dialogue boxes, where the system highlights parameters that the user manipulates frequently (in green) and fades parameters that the user tends not to change [90].

An additional but unevaluated example of adaptive GUI customization can be found in Miah *et al.*'s architecture for adaptive toolbars. With their architecture, toolbar items and entire toolbars are automatically added and removed based on frequency, recency of use, and the time the toolbar item/toolbar was created [94]. Finally, history mechanisms tend to be purely adaptive in that their content is usually determined solely through the system's collection of previously executed commands. It would be possible to augment a history mechanism with a customization facility to allow users to adjust the list of commands to better suit their needs (see [55] for a discussion); however, we are not aware of any implementations of this strategy.

2.3.3 Mixed-Initiative Approaches

In 1993, Fisher [44] discussed the need for systems that contain a mix of adaptivity and adaptability to be capable of adjusting to changing environments and users. Systems with a mix of adaptability and adaptivity are known as *mixed-initiative* systems, which Horvitz defined in 1999 as systems that combine automation based on sensing a user's actions with user control through direct-manipulation mechanisms [60]. As it is now widely accepted that no adaptive mechanism is perfect, nearly all adaptive systems are mixed-initiative to some degree. In the vast majority of cases, however, such systems involve content customization, examples of which are provided in section 2.5.

The majority of the work on mixed-initiative GUI customization has involved helping users create macros, a different form of customization than the one supported in this thesis work (i.e., feature management – see sec. 2.2.1). The first

example is the FlexExcel project, an extension to the Excel software package [77, 100, 115]. FlexExcel integrates adaptability and adaptivity by providing adaptive suggestions for defining new menu entries or short-cuts for functions that are repeatedly used with the same parameters (e.g., Font with Palatino, Size with 14 pt, and Bold). The system also has a Critique feature that provides user-tailored information on how to better use the customization features. Both the suggestions and the Critique topics are determined using a rule-based approach according to usage frequencies. When a new customization suggestion is available, users are notified by a “Tip” icon that blinks three times and makes a sound. The system was evaluated in a laboratory setting with 13 participants, but the authors report only high-level observations, making it difficult to obtain a clear understanding of the merits of their solution. In terms of usage, the authors state that some users had difficulty using the system-tailored suggestions to initiate their own customizations. This difficulty could have been caused by either the manner in which the suggestions are presented, or problems with the underlying adaptive algorithm.

A second example of a system that supports complex and general macro creation is EAGER [33], which is also an example of the more general class of systems that supports “Programming by Demonstration” [34]. Eager monitors the user’s feature usage and other actions within a word-processing application for repetitive behaviour. When a pattern is detected, EAGER pops up an icon notifying the user that EAGER is ready to recommend a macro. To give the user confidence that the pattern has been correctly identified, EAGER begins to highlight the user’s next anticipated feature selection (in green). If the user then clicks on the icon indicating the availability of the new macro, EAGER will automatically complete the remainder of the steps. Based on the paper’s description, it is unclear whether or not a more permanent interface feature is also created so that the user can invoke the same macro again at a later time. An informal laboratory evaluation with seven participants showed that users generally understood and made use of EAGER’s support, but that they also expressed discomfort with giving up control.

As is the case with this thesis work, Debevc *et al.*’s Adaptive Bar helps users manage the complexity of existing features, in particular, the features on the toolbars in MSWord [36]. The system helps users add and remove features from the toolbars through the use of adaptive suggestions based on a combination of recency and frequency of use. When the system has suggestions, it notifies the user by changing the background colour of the toolbar and by generating a

sound. Within the toolbar, the system also displays the relative frequency with which the user is invoking each feature by altering its size. An evaluation of the system compared the Adaptive Bar to the default, customizable MSWord toolbar. Since this evaluation is a direct comparison of an adaptable and a mixed-initiative system, it is discussed in section 2.4.3.

Two additional examples include Benyon's system [5] and Clark and Matthews' approach to layered interfaces [25]. Along the lines of Gong and Salvendy's adaptive approach [52], Benyon's system provides the user with a suggestion to use either a menu-based interface or a command-line interface based on the number of user errors and the user's general computer experience [5]. The layered interface approach proposed by Clark and Matthews allows the user to decide which layers should be present, but the system also makes certain layers visible based on which types of documents the user has edited in the past [25]. Little detail is provided on these systems and evaluations are not reported.

MICA's mixed-initiative support, which is described in detail in chapter 4, differs from the above examples in terms of: (i) the manner in which the underlying adaptive mechanism determines which recommendations to make, and (ii) how MICA delivers these recommendations to the user. While the above systems make customization suggestions to help the user save time, they do not actually rely on a formal computation of the savings. Instead, these systems tend to rely on rules and thresholds to determine which customization recommendations to make. In contrast, MICA formally and comprehensively quantifies user performance, and uses this quantification to inform its decision making. This comprehensive prediction considers not only usage frequency, but also the user's expertise and characteristics of the interface. In terms of delivering the support, MICA is the only system to provide the user with access to its rationale. An additional difference is the timing of the suggestions. MICA presents its recommendations only when the user chooses to customize to avoid interrupting the user. With many of the above approaches, the interruptions are quite subtle, however, even a subtle interruption has a cost.

A system that does perform more comprehensive reasoning is the SUPPLE system described in the previous section. While the SUPPLE work has focused primarily on a purely adaptive approach to interface customization, the system was extended to include a customization facility that allows users to override the system-initiated adaptations [47]. This portion of the work, however, appears to be more preliminary than SUPPLE's adaptive component, since little detail is provided on the underlying mechanisms and it has yet to be evaluated. In

addition to placing more emphasis on user control, MICA also differs from SUPPLE in the way that it evaluates the benefits of customization. SUPPLE includes a measure of user effort in its decision making, but uses only the number of mouse clicks as an approximation, whereas MICA relies on detailed GOMS analysis to predict the user’s actual performance with a given interface. Like all of the above systems, SUPPLE does not provide the user with access to its rationale.

2.4 Direct Empirical Comparisons of the Different Approaches

To better understand the relative advantages of mixed-initiative, adaptive and adaptable approaches, we survey a number of direct empirical comparisons. The majority of these evaluations fall into two categories: (i) comparisons of adaptive and adaptable alternatives and (ii) comparisons of adaptive and static interfaces. We also describe the evaluation of the Adaptive Bar (mixed-initiative vs. adaptable) and two additional evaluations of interest.

2.4.1 Adaptive vs. Adaptable

Despite the disagreement in the research community on the merits of the different approaches (e.g., [108]), there has been very little work directly comparing adaptive and adaptable approaches through empirical studies. Three exceptions are evaluations conducted by McGrenere *et al.* [91, 92], Findlater and McGrenere [41], and Jameson and Shwarzkopf [66]. The studies have had mixed results, motivating a solution that combines aspects of the two approaches.

McGrenere *et al.* conducted a six-week 20-participant field study to evaluate their two-interface adaptable model for MSWord (described in sec. 2.3.1) [91, 92]. Participants used the adaptable interface for the first four weeks of the study, after which they used the SMART MENU adaptive interface (described in sec. 2.3.2) for the remaining two weeks. The majority of the participants in the field study preferred the adaptable interface (65%), but some did favour the adaptive interface (15%). The remaining 20% indicated that they preferred the default MSWord interface. There were, however, two potential confounding variables. First, the adaptive and adaptable interfaces have very different designs, and as a result, might differ in their usability. While there is no published

evaluation, the SMART MENUS adaptive interface is considered to have fairly substantial design flaws, mostly due to the unpredictability of its behaviour and the lack of positional consistency between the small and full menus [65]. Second, the conditions were not counterbalanced – all participants completed the adaptive condition after the adaptable condition.

As a follow-up to the above study, which did not include a performance measure, Findlater and McGrenere ran a laboratory experiment with 27 participants to test the effects of adaptive, static, and adaptable menus on performance [41]. The experiment tested three variants of Sears and Shneiderman’s split-menus [106]: (1) a static menu, (2) a menu where the top half of the split-menu was adaptable by the user, and (3) an adaptive design where the system would dynamically adjust the top half of the split based on frequency and recency. Participants were given streams of menu selections (i.e., given a sequence of menu items to select), with each stream presented twice for each condition. In the adaptable condition, participants were given the opportunity to customize their interfaces after seeing the stream once.

The results comparing efficiencies of the three variants are complicated, primarily because the interfaces’ impact on performance depended on the order in which participants were exposed to the three conditions. Participants were significantly faster with the adaptable interface (not including customization time) than with the adaptive or static interface only when they were not presented with the adaptable interface first. These results suggest that an adaptable approach can improve performance, but that users might need some experience with the application and/or tasks, or support from the system to customize efficiently. To further explore the effects of practice on customization, the authors ran a second experiment, but the results were generally inconclusive [40].

In terms of preference, Findlater and McGrenere found that the majority favoured the adaptable variant (55%), 30% favoured the adaptive version, and 15% favoured the static interface. The authors point out that the increase in support for the adaptive interface from McGrenere *et al.*’s field study (from 15% to 30%) could be attributed to the better design of the split-menu adaptive interface relative to the SMART MENUS. Some who did not like the adaptive version were frustrated with the inconsistencies, again suggesting that dynamically adjusting menus is likely not a good adaptive strategy. MICA’s mixed-initiative strategy does not adjust the menu items dynamically, but rather provides adaptive suggestions to assist users as they customize. In addition, in Findlater and McGrenere’s study, participants were given the opportunity to customize dur-

ing their break between the two menu-selection streams. In a more ecologically valid setting, users would have to take time away from their work to customize, which might make fully user-controlled customization at least somewhat less attractive because of the time required to customize.

Findlater and McGrenere’s study is one of the few endeavors to examine the performance implications of user-controlled GUI customization. The simulation experiment described in chapter 3, which was conducted in parallel to their study, also examined the performance benefits of an adaptable approach and addressed two issues not considered in the above experiment. First, we included the time necessary to customize the adaptable interface in our analysis, since it might be the case that the customization time outweighs any savings. Second, by asking users to select streams of named menu items, Findlater and McGrenere’s experiment reduced variability that could arise from different levels of expertise with an application. Our work in chapter 3 included a measure of user expertise and analyzed how this measure of expertise affected performance.

Jameson and Shwarzkopf performed a direct comparison of system- versus user-controlled customization of content rather than of GUIs [66]. In a laboratory experiment with 18 participants, they examined the effects of having the system versus the user control the timing of updates to a personalized hotlist for events at a conference. Updates consisted of additional events of interest, as recommended by the system. In the user-controlled version (i.e., adaptable), the system would update the list only when the users pressed an “Update” button. In the system-controlled version (i.e., adaptive) the updating would occur automatically, whenever the user modified an event in the hotlist. In a third “no recommendation” condition participants did not received any system recommendations (i.e., static). Since the adaptable condition does involve some system-controlled customization in the form of which recommendations are made, this evaluation could also be considered a comparison of a mixed-initiative system and an adaptive system. We discuss it in this section because the independent variable was the timing of the updates, which was purely system controlled versus user controlled.

The experimental task was to create a hotlist for a fictitious colleague with a specified set of interests. Performance was measured as the number of correct and incorrect entries in list. An entry was considered “correct” if it related to the fictitious colleague’s interests (presumably as judged by the authors) and “incorrect” otherwise. The results showed that the three conditions had very similar performance scores. While participants were not asked to choose between

the three versions, analysis of qualitative measures indicated that preference was more evenly divided between the adaptive and adaptable interfaces than in the other two evaluations discussed in this section. The short duration of the experiment (seven minutes per condition), however, makes it difficult to draw any firm conclusions.

2.4.2 Adaptive vs. Static

Direct comparisons of adaptive and static interfaces have also yielded mixed results indicating that adaptivity can be beneficial, but that care should be taken in deciding when and how to provide adaptive support. All evaluations discussed in this section are laboratory experiments.

Greenberg and Witten [54] compared an adaptive versus a static strategy for hierarchical menu organization in a telephone directory system (containing 2611 items) in a 26-participant experiment. In the adaptive interface, the hierarchy was adjusted dynamically throughout the interaction according to frequency of use, with the goal of minimizing the depth of traversal to the desired item. In the static strategy, the hierarchy remained constant. They found that participants performed significantly faster with the adaptive interface, completed significantly fewer errors, and the majority preferred the adaptive interface (69%). Given the extremely large number of items present in the interface, and the small number of items that would typically be used by an individual user, this might be the ideal candidate for an adaptive interface to succeed, since an effective adaptive strategy could dramatically reduce the effort required to locate the desired telephone number.

Trevelyan and Browne observed that with the above experiment, users might not have enough exposure to the static interface for its benefits (e.g., positional consistency) to begin to dominate [116]. Therefore, they replicated Greenberg and Witten's experiment, but with a larger number of trials for each condition. The authors found that with practice some users did begin to perform better with the static interface. Unfortunately, the small number of subjects (four in total) and the manner in which the statistics were run preclude drawing any firm conclusions.

An evaluation where the adaptive interface did not perform well is Mitchell and Shneiderman's comparison adaptive and static menus [95]. In the adaptive menus, the positions of the items changed dynamically according to frequency of use. Participants (63 in total) were significantly slower in the adaptive condition

and 81% of participants preferred the static alternative. The specific details of the target application and the menus are not described, however, the interface appears to be much less complex than the telephone directory system described above (e.g., the sample menu in the paper contains four items). As a result, the cost associated with the lack of positional consistency might have outweighed the relatively small benefits associated with having the most frequently used items located at the top of the menu.

Tradeoffs associated with different adaptive approaches were investigated by Gajos *et al.* who compared three adaptive interfaces for toolbar customization to a static alternative [49]. The three adaptive conditions were as follows: (1) a Split interface, where an extra adaptive toolbar contained duplications of “important” features present on the default toolbars; (2) a Moving interface, where the important features on a given toolbar were moved to more easily accessible locations on that same toolbar; and (3) a Visual Popout interface, where the important features were made visually distinct, but positional consistency was maintained. Over the course of two experiments (one with 26 participants and the other with 8 participants), the authors investigated the impact of the different interfaces, the underlying adaptive algorithm (frequency based vs. recency based), adaptation accuracy (70% vs. 30%), and (indirectly) task complexity.

Given the number of factors in the experiment, we highlight only the key findings. In general, both the quantitative and qualitative effectiveness of the different adaptive interfaces depended on task complexity. In terms of performance, the authors found little difference between the interfaces for the more cognitively demanding task. In the less cognitively demanding task (selection streams), two of the adaptive interfaces (Split and Moving) were faster than the Static interface. On measures such as preference, perceived benefit, and perceived cost, the Split and Moving interfaces both fared well in the more cognitively demanding task, whereas users found the Visual Popout interface distracting. There was not as much support for the adaptive interfaces in the less complex task. Complaints included having to look in two locations for features in the Split Interface and the lack of positional consistency in the Moving interface. There was no effect of the type of underlying adaptive algorithm, but user performance did decrease as the adaptive algorithm’s accuracy decreased (as one would expect with a decrease in adaptation accuracy from 70% to 30%).

When discussing the results from their experiments in comparison to results obtained by other relevant studies (e.g., [41], [54], [106], and [118]), the authors propose a number of factors that could impact the success of an adaptive inter-

face. These factors include: spatial stability, adaptation accuracy, adaptation frequency, the frequency with which the user interacts with the interface, the complexity of the user's tasks, and the complexity of the interface itself.

Finally, Gong and Salvendy evaluated their adaptive approach to promoting command-line usage in a four-condition between-subjects experiment with 40 participants [52]. The four conditions were: (1) a command-line interface; (2) a menu-based interface; (3) a hybrid interface, where participants had access to both the menus and the command line; and (4) an adaptive interface, where the system moved users from the menus to the command line (see sec. 2.3.2). The authors found that towards the end of the session participants were significantly faster with the adaptive interface as compared to the non-adaptive, hybrid approach. Unlike the evaluations described above, the study used a between-subjects design, which did not allow the authors to obtain preference information.

2.4.3 Mixed-Initiative vs. Adaptable

We are aware of only one direct comparison involving a mixed-initiative system for GUI customization to either an adaptive or adaptable alternative. Debevc *et al.* compared their Adaptive Bar [36] (the mixed-initiative system described in sec. 2.3.3) to the built-in toolbar customization facility present in MSWord at the time of their study (i.e., the adaptable system). Their experiment was run in the laboratory with 16 participants. Because the study used a between-subjects design, the authors could not gather direct preference data and the study had lower power to detect performance differences. The authors did, however, find that the mixed-initiative system significantly improved performance in one of two experimental tasks. The performance differences were attributed primarily to differences in customization time in the two conditions. The authors also discuss some interesting effects of expertise. Trends suggested that the more novice users customized less in the adaptable condition than in the mixed-initiative condition, while the opposite was true for the more expert users. A key difference between Debevc *et al.*'s experiment and the one in chapter 5 is that by using a within-subjects design we were able to obtain information on which interface users preferred in addition to how the mixed-initiative support affected their performance and customization behaviour. Our evaluation in chapter 5 is also more detailed than Debevc *et al.*'s study, providing further insight into why and how mixed-initiative approaches can be beneficial.

2.4.4 Other Empirical Comparisons of Interest

Tsandilas *et al.* performed a controlled laboratory evaluation with 12 participants to compare the effects of adaptation accuracy on user performance in a large list-based selection task [118]. Users were asked to select a target from a list of 50 items, with the system recommending a given set (four or eight items) dispersed throughout the list. Accuracy was varied by adjusting the percentage of the time that the target item would actually appear in this group (100%, 80%, or 60%). In addition to accuracy, the authors experimented with two ways of drawing the user’s attention to the recommendations. In both cases recommended items were highlighted, but in one of the two conditions, the size of the remaining (non-recommended) items was also decreased until the user moused over the items with a fish-eye lens [4]. Similar to Gajos *et al.*’s study [49], the authors found that adaptation accuracy did impact performance, but with a more dramatic effect in the fish-eye condition. The fact that adaptation accuracy had more of an impact in this condition could be because it was more difficult for the users to compensate for bad adaptation, since non-recommended items would remain too small to read until magnified with the mouse.

Using a more theoretical method of evaluation, Warren employed cognitive modelling to examine the potential costs and benefits of varying the length of an adaptive hotlist in a decision-support system for medical diagnosis [123]. The results of the simulation indicated that a system that considered the following three factors generally performed better than one that used a fixed-length list: (1) the “cost” of processing each item in the list; (2) the “benefit” of using the list to retrieve the desired items as compared to some alternative interface structure (i.e., time saved); and (3) the probability that the items in the hotlist were, in fact, the desired items. Cognitive modelling was used to inform the value for the “cost” parameter, which remained constant throughout the experiment. “Benefit” was an independent variable manipulated in the simulation (i.e., Warren experimented with a number of different values for this parameter). Unlike our experiment in chapter 3, Warren’s experiment did not explicitly simulate the interface operations required to select the desired items in either the hotlist or the alternative interface structure.

2.5 Other Examples of Mixed-Initiative Interactions

In section 2.3.3, we discussed existing mixed-initiative approaches to GUI customization. Because there is little work in the area, our discussion did not illustrate the range of mixed-initiative interactions that are possible. Thus, we cover this range here. We first overview domains outside of GUI customization that commonly rely on mixed-initiative techniques. We then provide examples grouped according to their style of mixed-initiative interaction. We follow this with a short discussion of evaluations targeted at understanding the value of a system’s mixed-initiative approach as compared to either fully system-controlled or fully user-controlled alternatives.

2.5.1 Common Domains

One of the most common uses of the term “mixed-initiative” is in discourse understanding within the area of Natural Language Processing (NLP), where it refers to permitting both the system and the user to choose the direction of a human-computer dialogue (see [29] for a discussion). Mixed-initiative interactions are also common in systems that aim to recognize the user’s plan or goal to assist him in completing his tasks more efficiently (e.g., [2], [62], and [82]). These mixed-initiative interactions range from dialogues to clarify the user’s goals or plans (e.g., [60] and [82]), or inferring the user’s plan and asking permission before executing it [2]. Other popular domains for mixed-initiative techniques are recommender systems and educational settings. In recommender systems, these techniques are used to refine the system’s recommendations, (e.g., [6] and [22]), while in Intelligent Tutoring Systems, they are used to determine which material to cover and in what manner (e.g., [12, 13], [64], and [102]).

2.5.2 Styles of Mixed-Initiative Interactions

Out of the numerous different styles of mixed-initiative interactions that have been explored, we structure our coverage according to the following four categories: (1) conversational interactions, (2) systems that allow the user to control their adaptive behaviour, (3) systems that support user-provided relevance feedback, and (4) systems that allow users to override the adaptive support. These are not mutually exclusive categories (i.e., a mixed-initiative system can support

a number of different interaction styles), nor are they exhaustive.

Conversational Interactions

We consider conversational interactions to be ones where the initiative changes frequently throughout the interaction. Work in discourse understanding often falls into this category, where taking the initiative means directing the topic or flow of the dialogue (see [29] for an overview). Examples include the L2Tutor, a dialogue system for improving fluency in a second language [103], and Atlas, which permits mixed-initiative tutorial dialogues in the physics domain [45]. This conversational interaction style also often occurs in systems that recognize a user’s plan or goal in order to automate tasks. One example is the SMARTedit system, which employs a mixed-initiative approach to training a system to automate certain text-editing tasks [126] and, like the EAGER system discussed in section 2.3.3, is a “Programming by Demonstration” system. The user can propose his own training examples, however, the system can also directly ask the user for examples of certain tasks. Additional examples include the Collagen planner [82] and the LookOut system [60], which engage users in dialogues to determine their goals and plans prior to automating tasks.

User-Controlled Adaptive Behaviour

An interesting way to mix adaptivity and adaptability is to allow the user to directly manipulate parameters in the adaptive algorithm. This can be accomplished indirectly using open user models [71], which display the system’s assessment of relevant user traits, and allow the user to change any assessments that she does not perceive to be accurate (e.g., [23], and [102]). These user-controlled manipulations in turn influence the manner in which the system adapts to the user. Another example within this category involves systems that permit users to influence their behaviour by allowing them to specify system-action thresholds (e.g., [23] and [62]). These thresholds specify how “sure” the system should be of the appropriate course of action and/or how valuable a system intervention might be. Examples of such actions or interventions include providing the user with assistance on how to use the application or automating part (or all) of the user’s task. Finally, some systems allow users to directly program the application’s behaviour through what is known as “end-user modifiability.” In these systems (e.g., [114]), special interface mechanisms allow users to specify rules that govern how the adaptive system behaves.

User-Provided Relevance Feedback

An alternative to giving users direct control of the adaptive mechanism is to exhibit adaptive behaviour, elicit feedback from the user as to the appropriateness of this behaviour, and use the feedback to refine future decisions. In this type of mixed-initiative interaction, the system often presents relevant topics or items and the user provides feedback on how well the topics or items match her needs. The AVANTI system, an adaptive hypermedia system for users with special needs, presents users with tailored information based on the assessment of the user model [43]. Once this information is presented, users can adapt the content by indicating that they do not wish to have certain pieces of information displayed. The RU-INQUERY system allows users to provide relevance feedback and to modify the system-suggested terms in an adaptive search engine [76]. This type of relevance feedback is also applicable in more complex and extended information searches, such as those supported by the HARVEST system [125]. The HARVEST system displays its current perception of the user's information goals (i.e., the search terms and their relationships) and allows the user to provide feedback on the system's interpretation [125].

Recommender systems also often gather feedback from users on the relevance of recommended items to improve future recommendations (e.g., [6], and [22]). For example, in critiquing-based recommender systems, the system iteratively refines its recommendations by generating an example recommendation and asking the user for a critique. These critiques involve discussing one or more attributes of the recommendation whose values do not match the user's preferences (e.g., [22]).

Overriding Adaptive Support

Finally, a mixed-initiative system can provide adaptive support, but also allow users the freedom to ignore or override this support – the primary style of most of the mixed-initiative GUI customization systems discussed in section 2.3.3. This form of mixed-initiative interaction is especially common in adaptive hypermedia systems and open learning environments, both of which place a large emphasis on free exploration and user control. For example, the INSPIRE system provides adaptive sequencing and presentation of instructional material, yet users have the option to ignore the recommended sequencing [102]. The ACE open learning environment follows a similar philosophy [12, 13]. ACE provides adaptive feedback on the user's exploratory behaviour, but the user

can choose to ignore the system’s exploration advice and continue to explore as she sees fits. A related example is the work by Jackson *et al.* where the learner is able to turn off system-generated reminders to use the available scaffolding tools [64]. The system, however, continues to monitor the learner’s use of the tools and will resume the reminders when the system deems necessary.

This style of mixed-initiative interaction can also be found in systems that use goal/plan recognition to automate tasks. Once a system recognizes the user’s plan or goal, it will often verify the appropriate course of action before taking over. For example, the AIDE system, which provides intelligent assistance for statistical analysis, uses a plan library to assist the user in performing exploratory data analysis, but will not execute certain plans without asking for permission [2]. Similarly, the Lumière system [62] will offer the user assistance, but it is up to the user to decide whether to accept the assistance or continue to work autonomously.

2.5.3 Relevant Evaluations

While many of the above systems were evaluated with human-participants, very few of these evaluations controlled for the impact of the mixed-initiative nature of the interaction. Two notable exceptions are the laboratory evaluations of the AIDE system [2] and the RU-INQUERY system [76], both of which showed the benefits of the mixed-initiative approach. In an eight-participant evaluation, AIDE’s mixed-initiative approach was compared to a control condition where users explored the data set unassisted. The results showed that users were able to make significantly more correct observations about the data using AIDE than through purely user-controlled data exploration. The 64-participant evaluation of the RU-INQUERY system compared three different variants of the mixed-initiative search engine to a baseline query mechanism without user-supplied feedback. These variants were as follows: (1) an opaque version, where users marked documents as relevant or not relevant; (2) a transparent version that showed users which search terms were added after their relevance feedback; and (3) a penetrable version, where users could manipulate these additional terms. The results showed that users completed a searching task more efficiently using the penetrable version than using the baseline version. The remaining pairwise comparisons were not significant. Neither evaluation gathered preference information.

2.6 Rationale: Exposing the Adaptive System to the User

One unique characteristic of MICA's mixed-initiative support is that it provides the user with access to its rationale. We are not aware of any other work that provides rationale within a system for GUI customization, however, aspects of rationale provision have been explored in other contexts and domains. First, there is a body of work on open user modelling, which allows the user to see the system's assessments of her relevant traits. There is also work on explaining the outcome of an adaptive system's behaviour or decisions.

2.6.1 Open User Modelling

One method of exposing the adaptive system is to allow a user to see and sometimes edit the contents of her user model. Open user models have been particularly popular in the field of Intelligent Tutoring Systems (e.g., [10], [35], [71], [85], [102], and [127]). An open user model is often displayed as a graph of domain concepts and their relationships, annotated or coloured according to the system's current beliefs in the user's associated knowledge levels or interests (e.g., [71], [85], and [127]). Alternatively, the system can display a list-based description of relevant concepts and their associated system assessments (e.g., [10] and [102]). In addition to domain concepts, open user models can contain information on learning styles, preferences, or learning goals (e.g., [35] and [102]).

Evaluations of the above systems, which have tended to be quite exploratory in nature, have indicated that a large percentage of users are interested in viewing their user models [71][102], and have provided initial evidence that users can generally understand the contents of even seemingly complicated models [127]. The evaluations have also indicated that open user models can promote reflection and can increase learner motivation [10][127].

Open user models have also been explored outside of a learning context. Cheverst *et al.* examined open user modelling in a ubiquitous computing system for controlling an office environment [23]. In their system, the user model is represented as a set of rules that specify when to perform actions, such as adjusting the temperature, and the system's confidence in the rule. Users can view their profile, create or adjust their profile, or view the rule that contributed to a particular adaptive decision (which also places this work in the category

described in the next section). In a fairly informal qualitative evaluation, the authors found that the majority of users generally wanted access to the information, and found the textual representation of the models to be sufficient. The authors also state that there were large individual differences in users' general desire to view and update their user models. An additional early example can be found in Kuhme *et al.*'s work on adaptive prompting. The adaptive prompter proposes a small set of tools and applications for the user depending on the user's current working context [78]. Users are also provided with a facility to view and edit their user models, and to modify the prompting rules.

2.6.2 Explaining Adaptive Decisions

Allowing users to access a representation of their user models provides them with some insight into how the system operates. Unless accompanied with explanations of how the system uses the information, open user models might not always help the user understand why the adaptive system is behaving in a particular fashion. In this section we survey approaches to rationale provision where the system explains or justifies its behaviour.

A first example of this type of rationale can be found in many recommender systems and adaptive search engines. When recommending a product or link, some systems justify the decision by describing attributes of the recommended item(s) that influenced the system's decision (e.g., [104]). Instead of focusing on the recommended item in isolation, another approach is to describe how the recommended item and other alternatives compare (e.g., [7] and [104]). Viewing properties of the alternatives and comparing them to the recommendation can give the user confidence that the system has, in fact, made the best recommendation. In addition to explaining the final recommendation, a recommender system can also explain prior steps, such as the need to ask the user for certain preferences and how these preferences will factor into its decision making (e.g., [122]).

Not all forms of rationale or explanation relate to specific attributes of the recommended item(s). In a collaborative-filtering recommender system, recommendations are based on how others in the community have rated a particular item and the similarity between their ratings as a whole and the ratings of the user in question. Herlocker *et al.* investigated several ways to visualize the effects of other users' ratings in a collaborative filter system, such as displaying ratings of the other most similar users for the recommended item [57]. Similarly,

Coyle and Smyth proposed explaining the output of an adaptive search engine by displaying how others in the community had interacted with the retrieved items [32]. Finally, Czarkowski and Kay's Tutor3 system explains web-page adaptation by indicating which parts of the user model informed the system's decision [35].

Other systems aim to increase transparency and predictability by describing aspects of their underlying mechanisms. For example, the RU-INQUERY system (described in sec. 2.5.2) aims to improve transparency by indicating that additional search terms have been added to the user's original query [76]. Explaining a system's underlying decision-making, such as explaining the relevant rules and reasoning techniques, has also been explored in decision-support and expert systems (e.g., [67], and see [61] for a review) and in e-mail classifiers [111].

Many have argued that including rationale in an adaptive system is crucial to its acceptance and effectiveness [56][81][112][113] – opinions that are generally supported by evaluations. While it is certainly not the case that all users will make extensive use of provided rationale [122], the positive impacts include increased trust in the system [122], more positive perceptions of the system's competency [104], and higher agreement with the recommendations [57]. If not properly designed, however, rationale can be difficult to use [35] and can even lead to less favourable responses towards the system [57].

2.7 Summary

Our survey of related work shows that there are many unresolved issues with respect to how to best help users cope with GUI complexity. First, while there is existing work on why or how users customize, little is known on how customization impacts performance, or on how adaptive support could be used to improve user customization. Second, there have been few attempts at addressing the problem of GUI complexity through mixed-initiative support; most solutions to date are either purely adaptive or purely adaptable. Mixed-initiative approaches that do exist tend to focus on macro creation, and rely primarily on frequency counts and rule-based reasoning. Third, there is no work on providing rationale within this context, despite the fact that rationale provision has been shown to be beneficial in other domains. Finally, mixed-initiative approaches to GUI customization have rarely been the focus of in-depth evaluations. There is only

one direct comparisons of a mixed-initiative approach for GUI customization to either the adaptive or adaptable alternatives [36]. This previous evaluation provides evidence that mixed-initiative support can improve performance, but it remains to be seen which of the approaches users prefer and why.

Chapter 3

Proof-of-Concept Experiment with Simulated Users

In this thesis we investigate a mixed-initiative approach to customization. In particular, we explore an approach that provides users with a customization mechanism that gives them control of the customization process, but that also provides adaptive support to help them customize effectively. This chapter describes a proof-of-concept experiment with simulated users that was the first step in our investigation [14].³ Our simulation experiment was designed to examine the performance benefits of customization and identify how to best support users as they customize. We used the results of our simulation to motivate our general approach and to inform the design of our system.

3.1 Objectives and Approach

Before building our mixed-initiative system for GUI customization, we needed to gain a better understanding of the value of customization and of adaptive support for it. In particular, we needed to understand: (1) if customization is worth the effort; (2) if users can customize effectively; and (3) if not, what specific properties the adaptive system should take into account to provide support for effective customization. The experiment in this chapter focused on issues 1 and 3. In addition, we drew on previous results to examine issue 2 – the adaptable interface designed and evaluated by McGrenere *et al.* [91, 92] (see sec. 2.3.1). Their evaluation provided useful insights into how users chose to

³An earlier version of this chapter appeared as: A. Bunt, C. Conati and J. McGrenere. What role can adaptive support play in an adaptable system? In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2004)*, pages 117-124, 2004.

customize their interfaces when given an easy-to-use mechanism, but did not measure the efficiencies of these strategies. To provide adaptive support to help users take full advantage of an adaptable interface, we needed to be able to identify when and why users are not able to customize effectively on their own. By comparing the efficiency of different customization strategies and relating these differences to factors such as user expertise and task composition, we were one step closer to creating adaptive support for customization. The goal was to lay the foundations for adaptive support that can provide principled, tangible benefits to users with varying needs.

One way to determine how user customization affects performance would be through a user study, but a detailed exploration of the space of relevant user and task properties would require a study that is large, complex, and longitudinal. We chose first to gain insight into what makes customization effective in a lower-cost manner by using a form of cognitive modelling to simulate and compare different customization scenarios and strategies. Our simulation experiment relied on a well-established technique for interface evaluation known as Goal, Operators, Methods, and Selection Rules (GOMS) [20]. We used GOMS analysis to look at two main aspects of customization strategy: *when* users should customize and *what* they should customize. While we used the insight gained from this exploratory analysis to guide the development of our mixed-initiative system, the results of our analysis could also be used to inform the design of user studies that can test specific hypotheses.

An overview of the remainder of the chapter is as follows. Section 3.2 describes the customization context and mechanism. Section 3.3 discusses why we used GOMS analysis to simulate and assess customization behaviour, while section 3.4 describes the details of our simulation. Section 3.5 describes two exploratory experiments that we performed using these simulations. Finally, section 3.6 discusses the implications of the results from these experiments for the future design of adaptive support for user-controlled customization.

3.2 Customization Mechanism

The work described in this chapter, and in the remainder of the thesis, relied on the two-interface model for MSWord proposed by McGrenere *et al.* [91, 92] (introduced in sec. 2.3.1). The two-interface model, displayed in figure 3.1, provides the user with access to two versions of the MSWord interface:

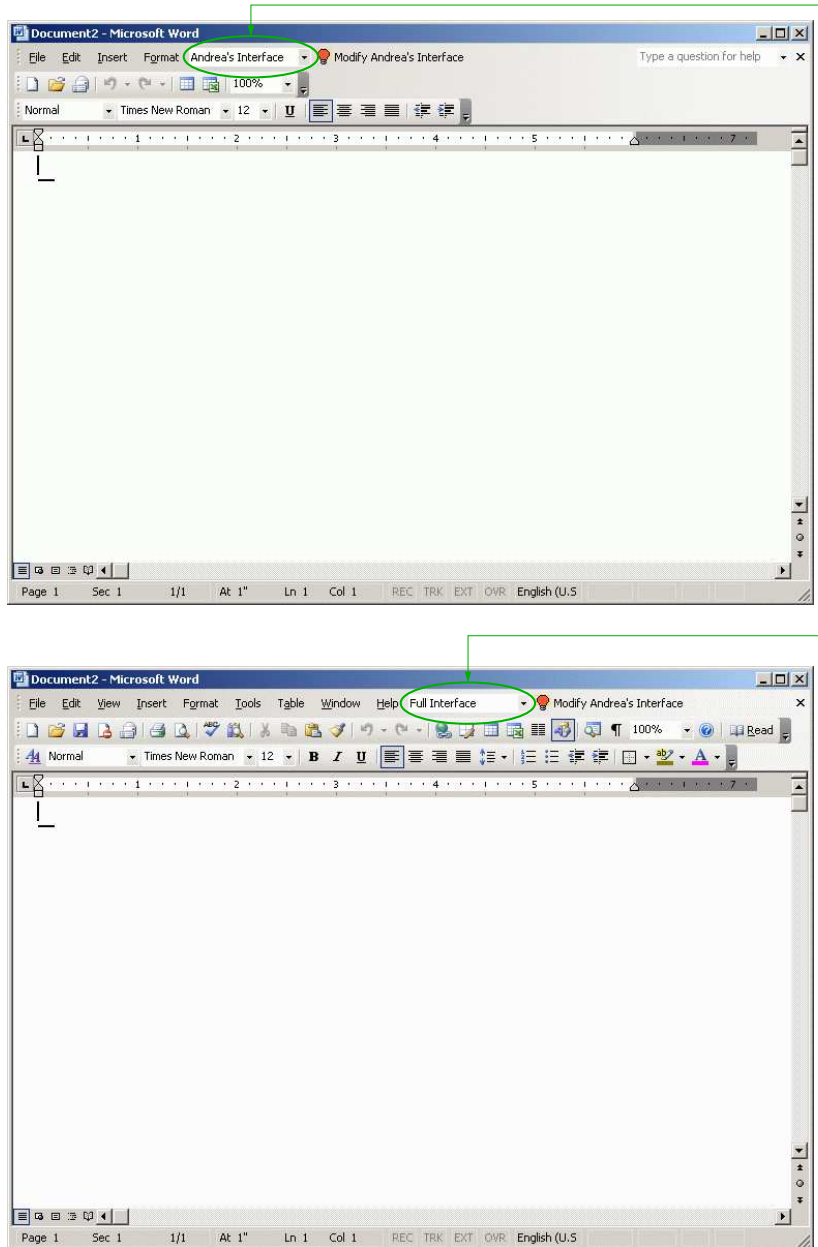


Figure 3.1: An example of the two-interface model. The top half of the figure shows the Personal Interface (PI), which contains only a subset of all available features (“Andrea’s Interface” here). The bottom half displays the Full Interface (FI), which the user can switch to at any point using the toggle button (circled in the figure).

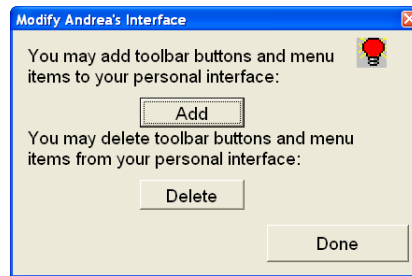


Figure 3.2: The dialogue box that appears when the user clicks the “Modify” button (see fig. 3.1).

1. **Full Interface (FI)**: the default full MSWord interface (fig. 3.1 (bottom)).
2. **Personal Interface (PI)**: a feature-reduced version of the MSWord interface, containing only features that the user has chosen to add (fig. 3.1 (top)).

The motivation for this two-interface model is to allow the user to create a PI that contains only the menu and toolbar items that best suit her needs, but also to allow the user to switch to the FI (using the toggle button circled in fig. 3.1) for rarely used features. The PI is built by the user using a lightweight customization mechanism. The user enters the customization mechanism by clicking on the “Modify” button shown in figure 3.1, at which point she is given the option of either adding features or deleting features from the PI (see fig. 3.2). If the user chooses to add features, she enters a mode in which the FI is displayed and any feature she selects (as in normal usage) will be added to the PI when she exits the customization mode (see fig. 3.3). A similar mechanism allows the user to delete features from the PI – the PI is displayed and anything she selects will be removed from the PI.

We chose this particular customization mechanism as a base for our research because it was fully evaluated by McGrenere *et al.* in a six-week field study [91, 92], ensuring that we were augmenting a customization mechanism that was already highly usable. While responses were generally very positive, their field study also generated suggestions for further improving the customization mechanism, which we partially implemented prior to performing the proof-of-concept simulation experiment described here. Specifically, we modified the design so that users are required to engage in only one confirmation dialogue

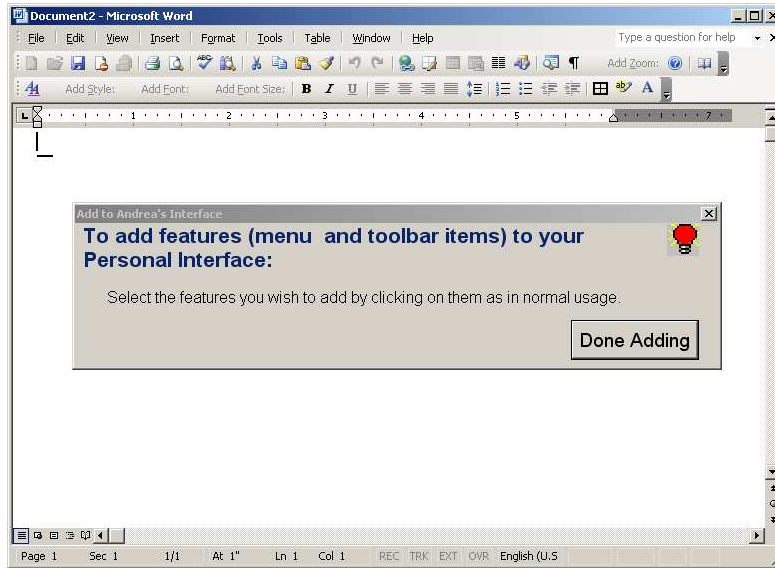


Figure 3.3: The lightweight customization facility for adding features. Anything the user clicks on will be added to the PI. Greyed-out features are already present in the user's PI.

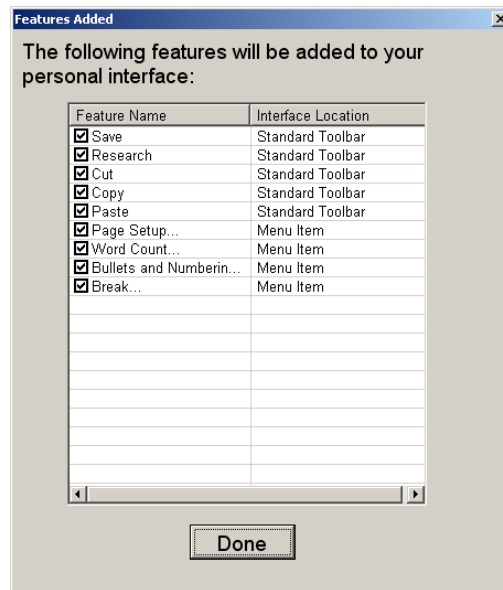


Figure 3.4: The confirmation dialogue that appears once the user indicates that she has finished selecting features.

upon exiting the adding or deleting customization modes rather than having to confirm each individual selection (see fig. 3.4).

There are, of course, other forms of customization that we could have chosen to study and extend. These mechanisms, however, would first have to be thoroughly tested to avoid having lack of usability as a confounding variable when comparing a purely adaptable customization approach to the mixed-initiative approach proposed in this thesis.

3.3 Using GOMS Analysis to Evaluate Customization

GOMS analysis [20] is a low-cost cognitive modelling technique used to evaluate a given interface design by predicting performance of simulated sequences of actions. In our simulation experiment, we used GOMS analysis to compare how different types of customization affect user performance in terms of *time on task*. Creating a GOMS model requires identifying the user's goals (i.e., tasks) in a given interface and specifying the methods available to achieve them. Methods are decomposed hierarchically until they consist of primitive interface operators (e.g., keystrokes, moving the mouse from one target to another and clicking mouse buttons). These primitive operators have associated times, which have typically been established empirically.

Running a GOMS model allows one to simulate and quantify the set of interface actions that a user would perform to complete a particular task. For example, to select a menu item, the user would perform the following sequence of actions: (1) find the menu heading, (2) point the mouse at the menu heading, (3) click the mouse button (at which point the menu items are displayed), (4) find the desired menu item, (5) point the mouse at the menu item, and (6) click the mouse button to select the item.

GOMS has been shown to be particularly effective at comparing two or more interface designs [53][68]. This was exactly our goal in this portion of the work, where the different interface configurations that we compared corresponded to the outcome of different customization behaviours.

While GOMS performance estimates are usually calculated by hand, we ran our models automatically using the GLEAN tool [74, 75].⁴ GLEAN takes as input a GOMS model written in a procedural-like language and a simulated

⁴We used GLEAN version 3 (i.e., GLEAN3), which is described in [74].

interface written in C++. Using these two inputs, GLEAN executes the model and generates a predicted execution time, broken down by individual methods, based on established values for the primitive operators.

In the next section, we describe the details of our GLEAN simulation, including the extensions we made to GLEAN to perform this work.

3.4 Details of Our GLEAN Simulation

The GLEAN tool generates performance predictions by executing a GOMS model of user actions on a simulated interface. As we mentioned in section 3.2, for the purpose of this experiment we simulated the adaptable two-interface model of MSWord. Since MICA also relies on this simulated interface to generate tailored recommendations, we provide further detail in chapter 4.

3.4.1 Tasks

Different tasks and different combinations of tasks are likely to benefit from different types of customization. Our goal was to determine the efficiency of different customization strategies and how they depend on factors such as user expertise, the user's combination of tasks, and individual task complexity. Thus, we built five GOMS models, all of which are based on tasks that users would realistically perform using MSWord. The five tasks are described briefly in table 3.1. The tasks were designed to vary in complexity, where *task complexity* is a function of the number of task features involved (the fourth column in table 3.1) and the total number of feature invocations (referred to in this chapter as *task size* – the fifth column in table 3.1). The GOMS models of the five tasks contained only the menu and toolbar actions the user would have to perform to complete the task, and did not simulate any of the typing or cognitive work. This is based on the assumption that those components would be the same regardless of customization strategy. We were interested only in comparing differences owing to customization.

3.4.2 Factors Influencing Performance: Extending GLEAN's Visual Search

At the level of GOMS primitive operators, the number of features in one's Personal Interface is likely to affect performance in at least two ways. One

Task	Description	Summary of Relevant Actions	# of Different Task Features	Total # of Feature Invocations (Task Size)	Distribution
Letter	a basic editing task	<ul style="list-style-type: none"> • formatting the layout and font of parts of the text 	8 items	14	across 3 menus
Report	a conference-style multi-sectioned document	<ul style="list-style-type: none"> • formatting section headings • formatting parts of the text • setting the document layout • inserting page numbers • formatting references • language checking 	16 items	66	across 5 menus
Table	a split-cell table	<ul style="list-style-type: none"> • inserting a table • inserting additional rows and columns • formatting the table 	8 items	10	across 2 menus + 1 toolbar
Revisions	MSWord's revision-tracking functionality	<ul style="list-style-type: none"> • inserting comments • accepting or rejecting comments 	3 items	9	across 2 menus
Figure	adding a figure	<ul style="list-style-type: none"> • inserting a figure from a file • adding a caption 	3 items	3	across 2 menus

Table 3.1: The tasks simulated by our GOMS models.

factor is mouse pointing time. It takes longer for a user to point to a menu item that is further down the list. GLEAN’s primitive operator for mouse pointing addresses this issue by using Fitts’ Law to calculate the time to point to a target [88]. Fitts’ Law states that movement time is a function of the distance to the target (or amplitude) and the target’s size.⁵

The second factor likely to impact overall performance is visual search time. A menu with more items will take most users more time to search through. In addition, it will take most users longer to find the correct menu when there are more menu headings. The search time is also likely to depend on the level of user expertise, i.e. the user’s familiarity with the interface [98]. The basic types of visual search documented in the literature include:

1. Exhaustive Linear Search: the user examines every menu item in a linear fashion [80].
2. Self-Terminating Search: the user examines every menu item but terminates the search when he finds the target [80].
3. Hick’s Law: the time to decide among the menu items is a logarithmic function of the number of alternatives [79].
4. Default: any visual search operation is assigned a constant value [98].

The original GLEAN models only the Default search.⁶ It assigns a default value of 1.2 seconds to any visual search operation regardless of menu length. As we will discuss shortly, the different types of search strategies can be related to user expertise. Our goal was to study how user expertise affects customization, and so we modified GLEAN so that it could simulate all four visual search strategies listed above. We then defined four expertise categories with respect to the visual search strategies. In the next section we describe how we defined these categories.

Relating Experience to the Visual Search Parameters

Norman states that, with experience, users are no longer affected by the number of items in a menu [98]; thus, the search behaviour of a highly experienced user can be reasonably approximated by the Default strategy. Landauer and

⁵GLEAN uses a modified Welford formulation of Fitts’ Law.

⁶Whenever we refer to the “original” GLEAN tool, we are referring to the version of the tool prior to extensions made as part of this thesis work (i.e., GLEAN3 [74]).

Nachbar state that Hick’s Law is applicable when users find the menu items easily distinguishable, when they know the menu item is present in the menu, and when they have had substantial practice with it [79]. This search strategy seems to indicate a user who is very familiar with the interface but not sufficiently experienced to be unaffected by the number of menu items. The work by Cockburn *et al.* [27] and Sears and Shneiderman [106] provides evidence that users transition from linear search strategies to logarithmic ones as expertise increases. There is no work, however, that explicitly distinguishes between Exhaustive Linear Search and Self-Terminating Search according to user expertise. It is reasonable to assume, nevertheless, that Self-Terminating would be a search strategy easier to employ by users who are more familiar with a target menu, since they would be able to tell when they had reached the item they are looking for without having to evaluate the remaining items.

We used the four different search strategies to identify four levels of expertise, ranging from Default to Exhaustive Linear Search in order of decreasing expertise. Exhaustive Linear Search, Self-Terminating Linear Search, and Hick’s Law include a parameter that can be related to the amount of time it takes the user to process an individual item (we call this parameter Time per Item). Time Per Item is likely a function of both a user’s reading speed and his familiarity with the menu items. Thus, we defined our categories of expertise as follows:

- **Extreme Expert:** Default Search.
- **Expert:** Hick’s Law with a Time Per Item of 0.15 seconds.
- **Intermediate:** Self-Terminating Search with a Time Per Item of 0.5 seconds.
- **Novice:** Exhaustive Linear Search with a Time Per Item of 1.0 seconds.

The values for Time per Item in each category are compatible with values given in the literature, but our experiments do not exactly replicate the conditions of previous experiments. As a result, we chose values that are conservative estimates of the Time Per Item for each category.

Our claim is not that these four categories perfectly model the behaviour of users with those levels of expertise, but rather that they provide four reasonable models that span user experience. In fact, studies that have used either cognitive modelling (e.g., [59]) or eye-tracking (e.g., [1], [17], [19], and [121]) to understand visual search strategy agree that the systematic strategies presented above (i.e.,

Self-Terminating and Linear) are rarely 100% accurate. These studies do not, however, agree on alternative models of visual search, partially because of small differences in their experimental designs.

3.5 Simulation Experiments

This section discusses two experiments we performed using our GLEAN simulation and the results that they generated. The first experiment was designed to address the issue of whether or not the overhead of customization pays off within a reasonable amount of time (i.e., is customization worth the effort?). It also compared customization strategies that vary in terms of *when* features are added to the Personal Interface. The second experiment was aimed at comparing strategies that differ in terms of *what* features users choose to add to their Personal Interface. We now discuss the details of these experiments.

3.5.1 Experiment 1: Is Customization Worth the Effort and When?

Does customization have the potential to improve user performance? A reduced interface is bound to be more efficient, but customization takes time. The goal of this experiment was three-fold. First, we wanted to see whether or not the overhead to perform customization pays off within a reasonable amount of time, assuming that the user has perfect foresight (i.e., she adds exactly the features that she needs to the Personal Interface). Second, we wanted to compare the performance of two different customization strategies that differ in *when* customization is performed during the interaction. Third, we wanted to see how task complexity and user expertise influence the payoff.

To achieve these goals, we implemented GLEAN models for three customization strategies. These strategies are abstracted versions of those adopted by users in McGrenere *et al.*'s study [91, 92] where, given the nature of the field study, they were free to choose any strategy.

1. *No Customization*: the user does not customize and always uses the Full Interface.
2. *Up Front*: the user adds all interface features that she plans to use before she starts the given task.

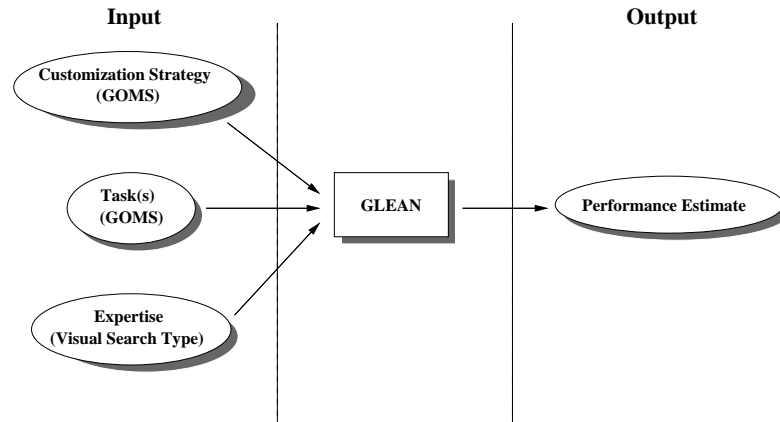


Figure 3.5: An overview of the setup for the simulation experiments.

3. *As You Go*: the user adds the interface features incrementally (one feature at a time) when they are first required to complete a given task.

The procedure for running the simulation experiment, which is also illustrated in figure 3.5, was as follows. We ran the GLEAN simulation with models for the three customization strategies, two tasks (Letter and Report), and each of the four expertise categories. The assumption here is that the user repeats the same task over time and does not perform any other task. Letter and Report were chosen because they represent different levels of *task complexity* while being plausible tasks that one would repeat every day. To answer the question “can customization improve performance,” we present the results in terms of the number of times the user would have to complete the target task (*task completions*) with the Personal Interface before it would outperform the Full interface, in terms of time to complete the task (*payoff*).

Can Customization Improve Performance?

Our results indicate that customization can be worth the necessary effort. Figure 3.6 displays the number of task completions required for each expertise category before the given customization strategy paid off compared to *No Customization*. For example, it shows that Intermediates using the *As You Go* strategy needed only to complete the Letter task approximately two times in the PI to outperform the FI. In the great majority of cases, the simulated users saw a payoff within twenty task completions. In all cases, as expertise increased, the number of task completions necessary for customization to become beneficial increased.

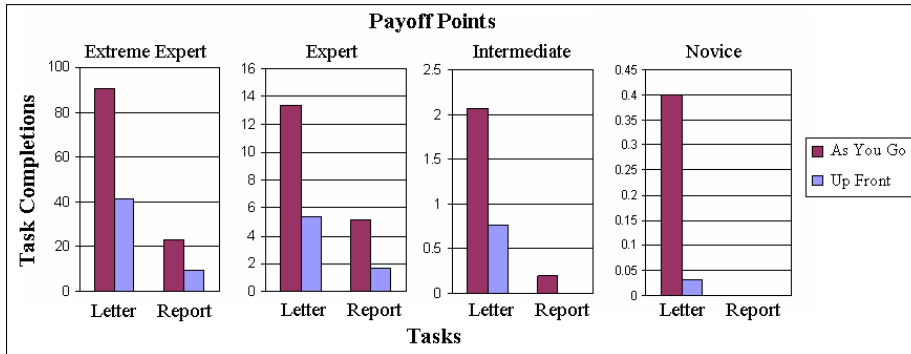


Figure 3.6: Experiment 1 results, showing the number of Task Completions it took for the Personal Interface to outperform the Full Interface for each of the *Up Front* and *As You Go* strategies. For clarity of display, the graphs have different scales.

This indicates that, as expected, simulated users with more experience were less impacted by unused features in their interfaces and, therefore, had to wait longer to see a payoff in customization. With larger task size (greater number of feature invocations), customization began to pay off sooner because of the larger number of feature selections per task that were performed faster. The effect was particularly dramatic for Intermediates and Novices, who often saw the payoff immediately. We would also expect an effect based on the number of different task features (the other component of task complexity). In particular, the higher the number, the more delayed the payoff, because the size of the Personal Interface would begin to approach the size of the Full Interface. With the Letter and Report tasks, task size was the dominating factor.

The overall trends are not surprising, but the magnitude of the effect for Intermediates and Novices was unexpected. For example, regardless of customization strategy or task, Novices did not even need to complete the task once in the PI before outdoing their performance in the FI.

If Customization Can Improve Performance, When Should Users Customize?

Table 3.2 compares the two customization strategies in terms of the amount of time users spent on customization-related actions. Customization using the *Up Front* strategy always took less time than the *As You Go* strategy – in some

Task	Expertise	Up Front (seconds)	As You Go (seconds)
Letter	Extreme Expert	72	123
Letter	Expert	61	112
Letter	Intermediate	104	156
Letter	Novice	144	296
Report	Extreme Expert	137	250
Report	Expert	116	282
Report	Intermediate	205	318
Report	Novice	493	605

Table 3.2: The customization time for Letter and Report based on expertise and customization strategy.

cases the *Up Front* was almost twice as fast. The benefit of the *As You Go* strategy is that users would spend a longer percentage of their tasks with smaller Personal Interfaces. Figure 3.6 shows, however, that this extra saving did not outweigh the extra customization time incurred by this strategy, since the *Up Front* strategy always led to an earlier payoff than the *As You Go* strategy.

In summary, the results of this experiment show that customizing can be worthwhile in terms of time savings, particularly for Novices and Intermediates. The most efficient time to perform the customization is *Up Front* (i.e., needed features are added at the beginning of the task), if the user can customize perfectly (i.e., she has perfect foresight of the needed features). It might seem quite obvious that customization should save time by allowing one to work in a smaller interface, but we believe that a quantitative measure of this savings is important for determining whether or not it is worthwhile helping users to customize. We are aware that performance is not the only factor that triggers user customization, but we argue that knowledge of performance gains could make it easier to motivate a user to customize in a particular way.

3.5.2 Experiment 2: What to Customize

Unlike in Experiment 1, users are not likely always to be performing exactly the same tasks. Rather, they are likely to perform a combination of different tasks, with certain tasks being performed more frequently than others. Our second experiment was designed to explore which features should be added to the Personal Interface in the presence of multiple tasks, and how this depends on task combination and user expertise. Figure 3.7 illustrates the differences between the two experiments. Experiment 1 addressed the *When* dimension of

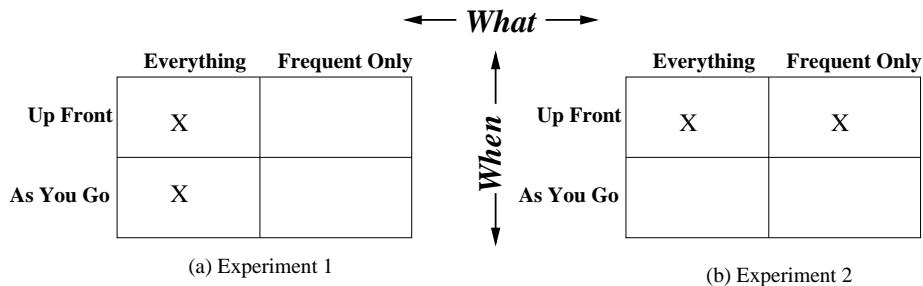


Figure 3.7: The two dimensions to customization strategy. The X’s represent the combinations of strategies investigated in each experiment.

customization strategy, while always adding all features necessary to complete the tasks (*Everything* in fig. 3.7). In Experiment 2, we focused on the *What* dimension by analyzing the performance of two customization strategies that differ in the subset of features that are added to the Personal Interface. As in the previous experiment, these strategies are based on customization behaviours observed in McGrenere *et al.*’s field study [91, 92].

1. *Everything*: the user adds all features to the Personal Interface necessary for both the frequent and infrequent tasks.
2. *Frequent Only*: the user adds only those features necessary for the more frequently performed task, and then must switch to the Full Interface to perform the infrequent task.

To isolate the effect of the *Everything* vs. *Frequent Only* strategies, we assumed *Up Front* customization. We restricted our investigation to combinations of two tasks, with the non-frequent task being performed once in the task sequence. The pairs of tasks (see the “Combination of Tasks” row in table 3.3) were selected based on *task composition*, which we define as a task’s complexity (task size and number of task features) and the distribution of task features across the menus. We considered the distribution of features across menus (depth vs. breadth of the menus) because it affects interface complexity. Longer menus take longer to search through, but additional menu headings factor in all menu item searches as the user must first locate the appropriate menu heading and then the appropriate menu item.

Which customization strategy is more effective involves tradeoffs between three main factors:

Independent Variable	Description	Levels
Strategy	Which features are added to the Personal Interface	<i>Everything, Frequent Only</i>
Combination of Tasks	Frequent/Infrequent	Letter/Table, Report/Table, Report/Figure, Letter/Table, Report/Revisions, Letter/Revisions
Ratio	The number of times the user will perform the same frequent task for every one infrequent task	1:1, 2:1, 3:1, 4:1, 10:1, 20:1, 100:1
Expertise	Categories of user expertise	Extreme Expert, Expert, Intermediate, Novice

Table 3.3: A description of the independent variables in Experiment 2.

1. *Time to complete each instance of the frequent task.* With *Everything*, the more frequent task is executed in a larger Personal Interface than with *Frequent Only*, because the Personal Interface contains the features for both tasks as opposed to just the features for the frequent task. As a result, we expect the frequent task to be slower in the *Everything* condition. How much slower it is should depend on the complexity of the Personal Interface, user expertise, and the composition of the infrequent task.
2. *Time to complete each instance of the infrequent task.* With *Everything*, the user can perform the infrequent task in the Personal Interface rather than in the Full Interface. Thus, the execution time of the infrequent task should be faster in the *Everything* condition. How much faster it is should depend on the complexity of the Personal Interface, user expertise, and size of the infrequent task.
3. *Customization time.* The *Everything* condition requires additional features for the infrequent task to be added to the Personal Interface. The difference in customization time for the two strategies will depend on the number of features in the infrequent task and user expertise.

Table 3.3 summarizes the variables that we manipulated in this experiment. The dependent variable was the total time necessary to complete the given ratio of tasks. As in Experiment 1, we used GLEAN to simulate the performance of the two customization strategies, with various combinations of tasks, task ratio, and user expertise (see fig. 3.5). Tables 3.4 and 3.6 provide a high-level summary of our results. Table 3.4 indicates which strategy was more efficient for

Task Combination (Frequent/Infrequent)	Everything	Frequent Only	Depends on Expertise or Ratio
Report/Figure	√		
Letter/Table			√
Report/Table			√
Letter/Figure			√
Letter/Revisions			√
Report/Revisions			√

Table 3.4: The most efficient customization strategy for each task combination without including customization time.

each combination without including customization time, while table 3.6 includes the additional customization time required for the *Everything* strategy. For each task combination in tables 3.4 and 3.6, where the most efficient strategy depended on user expertise or ratio (the last column in each table), tables 3.5 and 3.7 indicate the task ratio at which the *Frequent Only* strategy began to outperform the *Everything* strategy for each of the expertise categories. A range of ratios (e.g., (10-20):1) indicates that the exact ratio at which the *Frequent Only* began to outperform *Everything* lay somewhere within that range. We now discuss the results in these tables by first ignoring customization time to isolate the effects of different interface configurations on performance. Later we examine how the additional customization time for the *Everything* strategy affects the results.

Comparing Strategies Without Customization Time

The results given in tables 3.4 and 3.5 provide preliminary evidence of two factors that influence the efficiency of customization strategies when customization time is ignored:

1. The distribution of the features in the infrequent task
2. The complexity of the frequent task relative to the infrequent task

The distribution of features in the infrequent task is an important factor since certain features, when they were added to the Personal Interface with the *Everything* strategy, had a greater impact on the execution of the frequent task than others. An infrequent task feature can have a large impact because: (1) it gets added to a menu in the Personal Interface that is often used by the frequent task, or (2) it requires an entirely new menu heading to be added to

Task Combination (Frequent/Infrequent)	Extreme Expert	Expert	Intermediate	Novice
Letter/Table	*(>100):1	(10-20):1	(10-20):1	(10-20):1
Report/Table	*(>100):1	4:1	3:1	3:1
Letter/Figure	(20-100):1	(10-20):1	(4-10):1	4:1
Letter/Revisions	*(>100):1	(10-20):1	(4-10):1	(4-10):1
Report/Revisions	*(>100):1	(10-20):1	(10-20):1	(10-20):1

Table 3.5: Task ratio at which *Frequent Only* became more efficient than *Everything* for the task combinations in table 3.4 when the most efficient strategy depended on ratio and expertise, and customization time was not included. The asterisk indicates that, for the ratios tested, *Everything* was always more efficient than *Frequent Only*

the Personal Interface. We call features that have this large impact *intrusive*. Adding a new menu heading is especially costly because it affects every visual search operation (for menu items) in the Personal Interface for all but Extreme Expert users. When infrequent task features were intrusive to the frequent task, *Frequent Only* became the more efficient strategy at certain ratios and levels of expertise. To illustrate this effect, we can examine the Report/Figure and the Report/Revisions combinations in table 3.4. In both combinations, the infrequent task contained only three features, yet *Everything* was always the most efficient strategy for Report/Figure, while *Frequent Only* was more efficient at certain levels of expertise and ratios for Report/Revisions (see table 3.5). The difference can mostly likely be attributed to the fact that the features in the Revision task added an entirely new menu heading, while two of the three Figure task features were added to menus that were rarely used by the Report task.

Let's now consider how task complexity, in conjunction with feature distribution, influences strategy efficiency. First, when an infrequent task feature is intrusive, the impact is greater for a larger frequent task than a smaller frequent task. For example, consider the Report/Table vs Letter/Table combinations in Table 3.5. The Table task adds another menu heading. The impact of this addition was greater for the larger frequent task (Report) than for the smaller one (Letter), since the larger task had more feature invocations to be impacted by the extra menu heading. As a result, *Frequent Only* became more efficient than *Everything* at lower ratios for Report/Table than for Report/Letter. In the absence of especially intrusive features, the number of features in the frequent task relative to the infrequent task is also likely to be a factor. If the Personal

Interface contains a large number of features (for the frequent task), adding a small number of infrequent task features will not have much effect. This is not the case, however, if the Personal Interface is small to begin with. Both the Report/Figure vs. Letter/Figure combinations and the Report/Revisions vs. Letter/Revisions combinations illustrate this trend (see table 3.5).

User Expertise

As in Experiment 1, the less experience a simulated user had, the more she was affected by extra features. Thus, as user expertise decreased, table 3.5 shows that *Frequent Only* usually started to become more efficient than *Everything* at a smaller ratio. This is because of the higher impact of additional features on the speed of the frequent task. For example, we see that for the Report/Table task combination (without including customization time) *Everything* was the most efficient strategy for a Novice until she performed the Report task three times for every one Table task (3:1), as opposed to an Extreme Expert, for which *Everything* was always more efficient.

Including Customization Time

The discussion in the previous two subsections concentrated only on how the presence or absence of features in the two customization strategies affected performance. This ignores, however, the fact that *Everything* requires an extra amount of customization time that varies according to the number of infrequent task features and the user's expertise. Tables 3.6 and 3.7 include this additional customization time. For many of the combinations, the extra customization time did determine which customization strategy was more effective. *Everything* remained the most effective strategy for the Report/Figure combination for all ratios and expertise categories because the extra cost was minimal given the few features in the Figure task. On the other hand, for three of the combinations that depended on expertise and/or ratio when ignoring customization time (Letter/Table, Report/Table and Letter/Figure), *Frequent Only* was always the most efficient strategy when customization time was included. Therefore, the performance saving from having the additional features present in the reduced interface was often not as great as the time needed to customize them.

Task Combination (Frequent/Infrequent)	Everything	Frequent Only	Depends on Expertise or Ratio
Report/Figure	√		
Letter/Table		√	
Report/Table		√	
Letter/Figure		√	
Letter/Revisions			√
Report/Revisions			√

Table 3.6: The most efficient customization strategy for each task combination including customization time.

Task Combination (Frequent/Infrequent)	Extreme Expert	Expert	Intermediate	Novice
Letter/Revisions	1:1	1:1	2:1	3:1
Report/Revisions	1:1	1:1	1:1	(4-10):1

Table 3.7: Task ratio at which *Frequent Only* became more efficient than *Everything* for the task combinations in table 3.6 when the most efficient strategy depended on ratio and expertise, and customization time was included.

Magnitude of Differences

In addition to determining which strategy is more efficient, our results indicate the magnitude of the differences between the strategies, which varied according to expertise and task combination. As an example, figure 3.8 shows that after having performed the Report task 20 times and a Table task once, a Novice saved 23 minutes by using *Frequent Only* rather than *Everything*, while an Extreme Expert at the same ratio saved about 1 minute (including customization time). At the same ratio (20:1) for the Letter/Figure tasks, a Novice only saved 7 minutes while an Extreme Expert still saved about 1 minute. This is because the few features in the Figure task did not add much to the complexity of the Personal Interface.

3.5.3 Cognitive Overhead

In the above experiments, we did not consider the impact of two types of cognitive overhead involved with customization: (1) having to decide which features to include in the Personal Interface, and (2) having to figure out that a needed feature is not in the Personal Interface and that switching to the Full Interface is required. The first would have made the results in Experiment 1 somewhat less favourable toward customization. The second would have made the *Frequent*

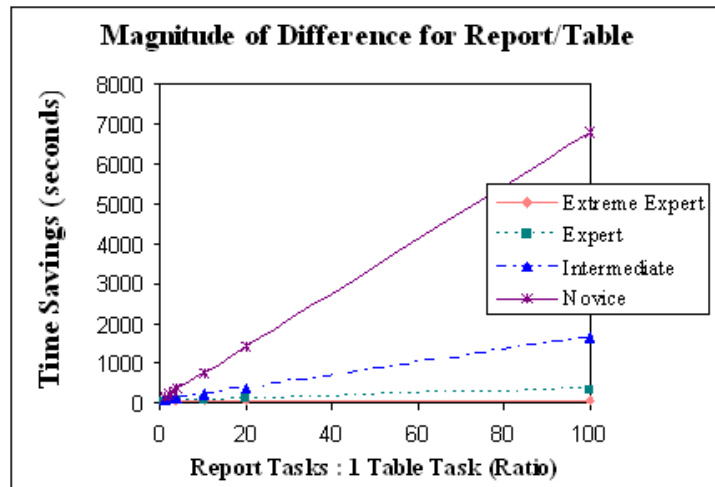


Figure 3.8: The magnitude of the difference between the *Everything* and *Frequent Only* customization strategies for the Report/Table combination for different ratios.

Only strategy in Experiment 2 somewhat less favourable than our results suggest. We are fully aware of the importance of both of these cognitive overheads. Future work includes obtaining accurate estimates of their impacts so that they can be included in our GOMS models.

3.6 Implications

In this section we describe the implications of our experiments. The main focus of this chapter has been to investigate whether supporting user customization is worthwhile and whether there is potential value in providing this support adaptively. This can be decomposed into three questions. (1) Is customization worth the effort? That is, does customization have the potential to improve user performance? (2) Do users need help to customize efficiently? (3) If yes, what user and task properties should the system take into account to provide support for effective customization? In other words, how can we provide adaptive support?

3.6.1 Is Customization Worth the Effort?

The results we obtained indicate that if customization is done well, it has the potential to be very beneficial in terms of reduced time on task, even including the time it takes for users to customize. Most users will see performance benefits within the first 20 times they execute a task, and within the first task completion if they are Novice users.

3.6.2 Do Users Need Help to Customize Efficiently? If Yes, How Can We Provide Adaptive Support?

Our results show a number of ways in which users might be unable to customize efficiently. We can also use these results to see the potential for adaptive support to help users overcome their difficulties. This section discusses general avenues for providing adaptive support. The specific approach investigated in this thesis is discussed in detail in chapter 4.

First, let us consider whether or not users know *when* to customize. We found that *Up Front* is much more efficient than *As You Go*, yet 68% of users described in McGrenere *et al.*'s field study [91, 92] did not engage in this type of strategy. Adaptive support could be used to encourage users to do as much customization as possible *Up Front*. It is possible, however, that in many situations, users will not be able to foresee all the features that will be needed. This is particularly true for Novices, who, as our results show, have the most to lose from inefficient customization. Therefore, adaptive support could be useful to implement efficiently an *As You Go* strategy, for example by recommending modifications at regular intervals based on the assessment of user behaviour and the estimation of what features the user might need the most. Another possibility would be to engage the user in a dialogue while he is performing *Up Front* customization. Since the system knows the factors that influence performance, it could ask the user questions that lead him to think about his future tasks in terms of these factors. For example, the system could ask the user questions about how often he anticipates having to complete certain tasks relative to others.

Second, let us consider whether or not users know *how* to customize (i.e., *what* features to add). Our results show that when users perform one task infrequently compared to another, adding all features is not always as efficient as adding only those from the frequent task. Yet McGrenere *et al.* [91, 92]

showed that 63% of users preferred to add all features. Whether or not the infrequently used features should be added depends on a number of factors, including:

- The number of infrequently used features
- Where these features are located in the menus
- The ratio at which the infrequent features will be used compared to the frequent features
- The user's expertise

There are likely too many factors for a user to take into consideration when deciding what to customize, particularly for a Novice, who would again have the most to lose from inappropriate customization. Adaptive support could be used to help users be selective about which features they add to their Personal Interface.

Finally, our results show that the presence of additional features not used on a regular basis can hurt performance. Yet McGrenere *et al.* [91, 92] showed that users very rarely removed features from their Personal Interfaces and those users who adopted the *As You Go* strategy often had trouble continuing to customize as the study progressed, particularly if their tasks changed. Helping users maintain their Personal Interfaces as their tasks evolve is another example of how adaptive support could help improve user performance.

3.7 Beyond Performance Data

In this chapter, we looked at how customization strategies affect performance. Performance, however, might not be the only factor that should guide an adaptive system. Results of previous experiments (e.g., [41], [66], and [91, 92]) indicate there might be additional subjective factors that should be taken into consideration. These factors include how users feel about full-featured interfaces versus reduced interfaces and how much control users desire over the content of their interfaces. Some users prefer customizing their own interfaces, while others do not mind adaptive approaches. As more evidence becomes available as to how users would chose to trade off these factors against potential performance gains, an adaptive system could balance these factors with performance considerations when providing the user with customization suggestions. In addition,

sometimes the magnitude of the difference between efficient and non-efficient strategies is small. Therefore, depending on type of adaptive support provided, the system might want to weigh potential performance gains against the potential costs associated with providing customization suggestions, both in terms of user satisfaction and the cognitive overhead involved in dealing with any advice from the system.

3.8 Summary

This chapter examined the value of customization in terms of its effects on user performance defined as time on task. First, we identified a performance measure that allowed us to evaluate the potential benefits of a personalized interface and of acquiring this personalized interface through customization. Then we described a study with simulated user interaction data to show that, using this measure, customization can be beneficial. We also discussed how our results, in combination with previous findings on real users' customization behaviours [91, 92], show that some users might have difficulty engaging in the most efficient customization strategies. Furthermore, our results provide indications of characteristics of the users, their tasks, and the interface itself that might impact effective customization. Relevant characteristics include user expertise, the ratio of feature usage and where features are located in the interface. Finally, we identified how adaptive support could help improve user performance in adaptable environments. The avenues that look the most promising are: (1) support for Novices; (2) helping users to perform their customization as early as possible, rather than in an incremental manner; (3) helping users to be selective about what they customize; and (4) helping users maintain their customized interfaces over time. In chapter 4 we discuss the design and implementation of our mixed-initiative system, MICA, which currently focuses primarily on avenue 3.

Chapter 4

MICA: Mixed-Initiative Customization Assistance

In chapter 4 we present the MICA system (Mixed-Initiative Customization Assistance), which provides adaptive, tailored support for user customization [15].⁷ MICA's support is designed to leverage as many of the advantages of the purely adaptable and adaptive approaches as possible, while eliminating or reducing their respective disadvantages. We describe how MICA determines which customization suggestions will help improve user performance and how MICA delivers these suggestions to the user.

4.1 Overview of the MICA System

MICA employs an innovative mixed-initiative approach to GUI customization, where the system and the user cooperate to produce a customized interface. Specifically, users are provided with an interface mechanism that gives them full control over the customization process, as well as adaptive support to help them customize their interfaces effectively. This approach is novel in that it uses on-line GOMS analysis [20] to provide MICA with two unique functionalities. First, MICA makes customization suggestions based on a formal and comprehensive quantitative assessment of how these suggestions are expected to impact the user's time performance with her target tasks. To the best of our knowledge, no other work on interface customization uses a formal quantitative assessment of performance to make informed decisions on how to customize. Second, MICA communicates these underlying expected performance savings to the users by providing them with access to its rationale, potentially improving the lack of transparency and predictability often present in adaptive interfaces. This also

⁷An earlier version of parts of this chapter appeared as: A. Bunt, C. Conati and J. McGrenere. Supporting interface customization using a mixed-initiative approach. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 22-31, 2007.

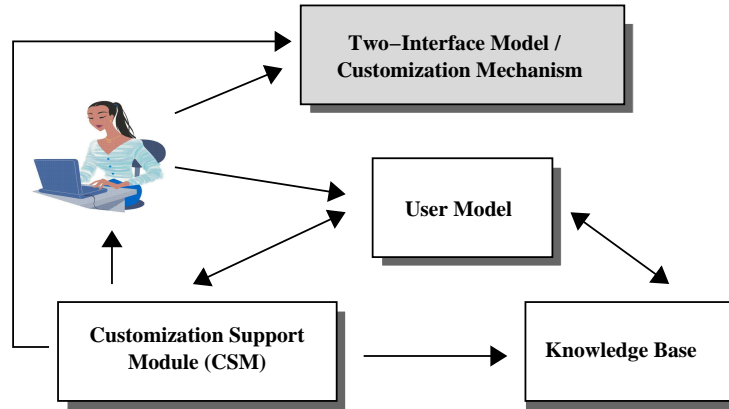


Figure 4.1: MICA's architecture

distinguishes MICA from other related systems.

The remainder of chapter 4 is structured as follows. Section 4.2 provides an overview of MICA's computational framework, while section 4.3 describes each component in the framework. In section 4.4, we discuss how we have implemented the framework, including our choice of target application. Finally, using this target application, section 4.5 illustrates how MICA presents its customization suggestions to the user. The MICA system described in this chapter is the version that we evaluated in Study One (chapter 5).

4.2 Framework: Overview

MICA's mixed-initiative support is designed to help the user customize given the two-interface model and customization mechanism proposed by McGrenere *et al.* [91, 92]. As a reminder, the two-interface model, which is described in more detail in section 3.2, provides the user with two versions of an interface: a Full Interface (FI) and a feature-reduced Personal Interface (PI). The two-interface model was extensively evaluated and shown to be highly usable [91, 92], thus providing us with a sound starting point for our mixed-initiative approach.

MICA's mixed-initiative approach relies on finding the user's optimal PI based on the time it would take him to invoke the features that he needs given the distribution of these features between the PI and FI. Figure 4.1 depicts MICA's architecture. The Customization Support Module (CSM) is responsible for determining the optimal PI and using it to generate customization

suggestions. Determining this optimal interface is done with the help of the User Model, which assesses the user's performance given a particular PI. This performance assessment is based on GOMS analysis [20], which has traditionally been used off-line to evaluate interfaces. In chapter 3, we discussed how GOMS can be used to evaluate the benefits of customization. Building on the approach used in chapter 3, the User Model performs *on-line* GOMS analysis to evaluate specific customization possibilities, corresponding to different potential interfaces. The CSM compares these evaluations and uses the results to make optimal customization suggestions. When assessing performance, the User Model collaborates closely with the Knowledge Base. The Knowledge Base stores and executes the relevant GOMS methods, and contains detailed information on interface layout.

4.3 Framework: Individual Components

We now describe each component in the framework.

4.3.1 Customization Support Module (CSM)

The CSM decides when to provide the user with tailored customization suggestions and which suggestions to make. To minimize disruption, currently the CSM provides the user with suggestions only when she initiates customization. These suggestions consist of features that the user should add to or remove from her PI and are targeted at optimizing her performance with the two-interface model. In general, the more features present in an interface, the greater its complexity, which has the potential to hinder user performance. Therefore, to decide whether to recommend a given set of features for inclusion in the PI, the CSM weighs the extra complexity that these features would introduce into the PI against the time it would take the user to switch to the FI and make the selections from the more complex interface. For an individual feature, this involves a tradeoff between the performance savings that would result from selecting the feature in the PI, versus the negative impact this feature's presence in the PI would have on the remainder of the expected PI feature selections. More formally, a feature f_x will be recommended for inclusion in the PI if and only if the following inequality holds, where $\text{SelectTime}(X, Y)$ is the time required to perform all expected selections of feature X in interface Y, and EA is the set of all features that are expected to be accessed:

$$\begin{aligned}
 & \text{SelectTime}(f_x, FI) - \text{SelectTime}(f_x, PI + f_x) > \\
 & \sum_{i \in EA - f_x} \text{SelectTime}(f_i, PI + f_x) - \sum_{i \in EA - f_x} \text{SelectTime}(f_i, PI - f_x).
 \end{aligned}$$

These selection times depend on: (i) user-specific information stored in the User Model, (ii) the contents of the PI currently under consideration, and (iii) the layout of both the PI and the FI. To decide which suggestions to make, the CSM performs a greedy search on the space of candidate PIs, each of which has a different subset of features present, to find the one that would maximize the current user’s performance. Determining this optimal PI involves asking the User Model (in collaboration with the Knowledge Base) to assess the user’s expected performance given each candidate PI.

In chapter 3, we discussed the idea of incorporating a cost of the suggestions into the system’s recommendations, such as the time to attend to or follow system advice. Because of the timing and format of its recommendations (described in detail in sec. 4.5), the CSM does not currently factor in such a cost. In particular, the CSM’s support does not involve interrupting the user, and its support has a number of potential performance benefits including: (i) informing good decisions (i.e., helping the user create the best possible PI); (ii) speeding up customization decision making; and (iii) increasing the speed of customization-related interface actions. Extending the CSM to incorporate a “cost of suggestions” into its decision making would certainly be possible if, for example, results of future evaluations indicate that it would be appropriate to do so.

Searching for the Optimal PI: General Strategy

As mentioned above, the CSM searches through the space of potential PIs for the one that would maximize the user’s performance with the interface. Unfortunately, an exhaustive search would generally be computationally infeasible since its complexity would be exponential in the number of features that the user is expected to use (as assessed by the User Model). Instead, the CSM employs a search strategy that is both based on heuristics and greedy in nature. In particular, the CSM relies on assumptions and properties of the search space for pruning purposes and to increase the efficiency of requested User Model computations.

The first heuristic that the CSM uses to prune the search space is to restrict its evaluation to only the features that are expected to be used infrequently. All features that are expected to be used frequently are automatically included in the optimal PI, since it is hard to imagine circumstances where it would be more efficient to be continually switching to the FI to invoke these features. That is, only infrequently used features are considered as candidates for exclusion from the PI. A configurable threshold parameter within the CSM (*frequency threshold* from now on) determines what constitutes a feature that is expected to be frequently used versus infrequently used. The frequency threshold represents an accuracy versus efficiency tradeoff in the CSM's computations. Currently, informed by the results of our simulation experiment, the CSM sets the frequency threshold to be three times that of the smallest (non-zero) expected usage frequency at the start of each optimal PI computation.

Next, to simplify computation, the CSM considers whether adding a particular feature to the PI has *local* or *global* effects, which determines the number of computations requested of the User Model and Knowledge Base and the extensiveness of these computations. Including a feature in the PI that requires adding a new menu heading to the PI is considered to have *global* performance effects. Specifically, the presence of the additional menu heading affects all menu-item selection times in the PI, since when the user searches for a menu item, she must first search through the available menu headings. On the other hand, adding a feature within an existing menu is considered to have only *local* effects. In other words, the presence of this new feature affects the selection times for only those features within that particular menu. The CSM assumes that additions to toolbars have local effects – i.e., that additions to a given toolbar affect selection times for other features only on that toolbar. When features with global effects are added, all menus in the interface need to be re-evaluated. When an added feature has a local effect, only the given menu (or toolbar) has to be re-evaluated.

Finally, the CSM considers which features to evaluate in a greedy manner. Within a given menu or toolbar, the infrequent features are added to the candidate PI one at a time in order of decreasing frequency and the CSM stops considering features as soon as it finds one that is deemed to be more efficient to keep solely in the FI (i.e., excluded from the PI).

Algorithm Outline

The outline of the CSM's algorithm is as follows. Whenever we mention feature usage, we are always referring to expected feature usage, as assessed by the User Model. A more detailed and formal algorithm appears in the next subsection.

Step 1: Ask the User Model to provide two lists according to the *frequency threshold*: (1) a list of features that are expected to be frequently used and (2) a list of infrequently used features.

Step 2: Create the *minimalPI*, which consists of all frequently used features and serves as the starting point for the optimal PI.

Step 3: Find the set of infrequent features with global effects (i.e., features that would require new menu headings to be added to the *minimalPI*).

Step 4: For each subset H in the power set of menu headings of these infrequent features:

Step 4.1: Add the headings in H to the *minimalPI* and call it the *currentPI*.

Step 4.2: Determine the optimal configuration for each menu m in the *currentPI* by performing a greedy search (according to usage) on the infrequent items belonging to m . The effects of these additions are local (since the required menu headings have been added).

Step 4.2.1: Add the next infrequent item within m to the *currentPI*.

Step 4.2.2: Ask the User Model to assess the selection times for all used features within m (including those that currently reside solely in the FI).

Step 4.2.3: If it is more efficient to include the feature in the *currentPI* than to have it reside solely in the FI (i.e., if the performance of the current candidate configuration for menu m is better than the previously tested candidate, which had this feature only in the FI), return to Step 4.2.1. Otherwise (according the greedy strategy) remove the feature from the candidate PI and determine the optimal configuration for the next menu in the *currentPI* (see Step 4.2).

-
- Step 4.3:** Ask the User Model to assess the performance for all expected menu selections given the *currentPI*. Since the performance for features in each of the menus in H was computed (and stored) in the previous steps, the User Model is asked to assess the performance for only those features that are not within the menus in the *currentPI*.
- Step 4.4:** If the total performance assessment of the *currentPI* is less than the best configuration found so far, the *currentPI* is marked as the *currentBestPI*.
- Step 5:** For each toolbar, determine the optimal configuration by performing the greedy search outlined in Step 4.2.
- Step 6:** The *optimalPI* consists of the *currentBestPI* determined in Step 4 and the optimal toolbars found in Step 5.

Formal Algorithm

For the interested reader, in this subsection we provide a more formal and detailed description of the algorithm. Using the definitions listed below, Algorithm 1 (FindOptimalPI) determines the overall optimal PI using Algorithm 2 (FindOptimalConfiguration) to find optimal configurations for the individual menus and toolbars.

Definition 1. Let *AllFeatures* be the set of all menu items and toolbar items.

Definition 2. Let *AllHeadings* be the set of all menu headings and toolbars.

Definition 3. Let *AllMenus* be the set of all menu headings.

Definition 4. Let *AllToolbars* be the set of all toolbars.

Definition 5. Let *AllPossiblePIs* be the set of all possible Personal Interfaces.

Definition 6. Let function $eu : AllFeatures \rightarrow \Re$ be defined such that $eu(a)$ returns the expected usage of feature a .

Definition 7. Given a frequency threshold t , let $FrequentFeatures = \{a \mid a \in AllFeatures \wedge eu(a) \geq t\}$.

Definition 8. Given a frequency threshold t , let $InfrequentFeatures = \{a \mid a \in allFeatures \wedge 0 < eu(a) < t\}$.

Definition 9. Let $AllUsed = FrequentFeatures \cup InfrequentFeatures$.

Definition 10. Let function $heading : AllFeatures \rightarrow AllHeadings$ be defined such that $heading(a)$ returns the menu heading or toolbar name of feature a .

Definition 11. Let function $menusInPI : AllPossiblePIs \rightarrow \mathcal{P}(AllMenus)$ be defined such that $menusInPI(potentialPI)$ returns the set of all menu headings within the potentialPI.

Definition 12. Let function $toolbarsInPI : AllPossiblePIs \rightarrow \mathcal{P}(AllToolbars)$ be defined such that $toolbarsInPI(potentialPI)$ returns the set of all toolbars within the potentialPI.

Definition 13. Let function $menusNotInPI : AllPossiblePIs \rightarrow \mathcal{P}(AllMenus)$ be defined such that $menusNotInPI(potentialPI) = \{heading(f) \mid f \in InfrequentFeatures \wedge heading(f) \in AllMenus \wedge heading(f) \notin menusInPI(potentialPI)\}$.

Definition 14. Let function $performance : AllFeatures \times AllPossiblePIs \rightarrow \mathbb{R}$ be defined such that $performance(f, potentialPI)$ returns the time to invoke feature f based on its expected usage and the potentialPI.⁸

Definition 15. Let function $featuresUnder : AllHeadings \rightarrow \mathcal{P}(AllUsed)$ be defined such that $featuresUnder(h) = \{f \mid f \in AllUsed \wedge heading(f) = h\}$.

Definition 16. Let function $infrequentUnder : AllHeadings \rightarrow \mathcal{P}(InfrequentFeatures)$ be defined such that $infrequentUnder(h) = \{f \mid f \in InfrequentFeatures \wedge heading(f) = h\}$.

⁸Note that if f is present in the potentialPI, $performance(f, potentialPI) == SelectTime(f, potentialPI)$. Otherwise, $performance(f, potentialPI) == SelectTime(f, FI)$. See section 4.3.1.

Algorithm 1 FindOptimalPI

```
1: Create the minimalPI consisting of the FrequentFeatures (see def. 7)
2: currentBestPI  $\leftarrow$  minimalPI
3: currentBestTime  $\leftarrow$   $\sum_{f \in AllUsed} performance(f, minimalPI)$  (see defs. 9
   and 14)
4: for all  $H \in \mathcal{P}(menusNotInPI(minimalPI))$  (see def. 13) do
5:   currentPI  $\leftarrow$  minimalPI
6:   time  $\leftarrow$  0
7:   Add headings in H to currentPI
8:   for all  $m_{in} \in menusInPI(currentPI)$  (see def. 11) do
9:     time  $\leftarrow$  time + findOptimalConfiguration( $m_{in}, currentPI$ ) (see Alg.
       2)
10:  end for
11:  for all  $m_{out} \in menusNotInPI(currentPI)$  (see def. 13) do
12:    time  $\leftarrow$  time +  $\sum_{f \in featuresUnder(m_{out})} performance(f, currentPI)$ 
      (see defs. 14 and 15)
13:  end for
14:  if time < currentBestTime then
15:    currentBestPI  $\leftarrow$  currentPI
16:    currentBestTime  $\leftarrow$  time
17:  end if
18: end for
19: optimalPI  $\leftarrow$  currentBestPI
20: for all  $tb \in toolbarsInPI(optimalPI)$  (see def. 12) do
21:   time = FindOptimalConfiguration( $tb, optimalPI$ )
22:   currentBestTime  $\leftarrow$  currentBestTime + time
23: end for
```

Algorithm 2 FindOptimalConfiguration(m , $currentPI$)

```

1:  $currentBestTime \leftarrow \sum_{f \in featuresUnder(m)} performance(f, currentPI)$ 
   (see defs. 14 and 15)
2:  $featuresToConsider \leftarrow infrequentUnder(m)$  (see def. 16)
3: while  $featuresToConsider \neq \emptyset$  do
4:   Let  $f_{max} \in featuresToConsider$  be such that  $eu(f_{max}) =$ 
    $\max_{f \in featuresToConsider} eu(f)$ 
5:   Add  $f_{max}$  to the  $currentPI$ 
6:    $time \leftarrow \sum_{f \in featuresUnder(m)} performance(f, currentPI)$ 
7:   if  $time < currentBestTime$  then
8:      $currentBestMenuTime = time$ 
9:   else
10:    remove  $f_{max}$  from the  $currentPI$ 
11:    return  $currentBestTime$ 
12:   end if
13:    $featuresToConsider \leftarrow featuresToConsider - f_{max}$ 
14: end while
15: return  $currentBestTime$ 

```

4.3.2 User Model

The User Model's role is to respond to the CSM's requests for performance assessments given a candidate PI. Accomplishing this task involves a tight collaboration with the Knowledge Base. The User Model stores the user-specific information necessary to compute how long it will take the user to perform her tasks with the candidate PI, and drives the computation process. The Knowledge Base, on the other hand, simulates and estimates the time necessary to perform a given feature selection (in response to a request from the User Model). In this section we describe the user-specific information stored in the User Model. We present further detail on how these factors are used in the overall performance assessment in section 4.3.3, where we describe the Knowledge Base.

To generate performance assessments, the User Model stores information on three relevant factors (also depicted in fig. 4.2): expected usages, expertise, and switching overhead. We now describe each of these factors.

Expected Usages

Expected usages represent how often the user is expected to access each feature in the interface. To maintain a probabilistic assessment without having large distributions, the User Model is designed to maintain a distribution over *ranges*

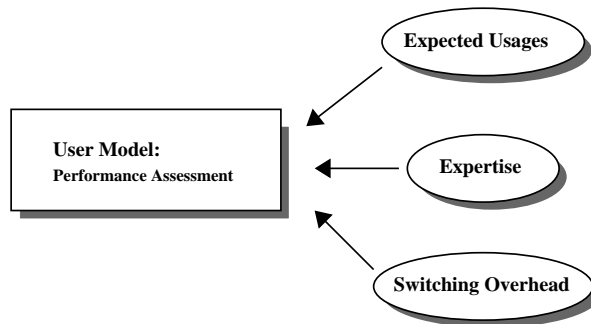


Figure 4.2: Factors influencing the User Model's performance assessment.

of plausible values – a strategy commonly used for probabilistic variables whose sets of values are large, but countable (e.g., [39]). These ranges can be application dependent and informed by typical usage data (e.g., [83] and [91, 92]). The expected usage of a feature is defined as the expected value of this distribution, calculated using the mid-point of each range. Expected usages do not represent how often the user will access a feature in her lifetime, but rather how often the user is expected to use the feature relative to other features in the interface (e.g., the user is expected to use “Save ” 10 times for every “Open”).

Expertise

In chapter 3, we discussed how the performance of users with lower expertise is likely to be more negatively impacted by excess functionality than that of more expert users, because it takes lower-expertise users more time to visually search for individual features. In particular, in chapter 3, we defined four expertise categories (Extreme Expert, Expert, Intermediate, and Novice), which specify the degree to which the user's selection of a given feature is impacted by complexity in the interface. To account for these effects of expertise in its performance assessments, the User Model maintains a probability distribution over these four categories for each feature in the interface. Unlike with expected usages, the full probability distribution is supplied to the Knowledge Base (discussed in further detail in sec. 4.3.3).

As we will see in section 4.3.3, the User Model's expertise assessment is used in the on-line GOMS analysis as part of the Knowledge Base's visual search calculation. Thus the User Model's definition of expertise allows the GOMS analysis, which has traditionally been used to model expert behaviour [20], to

account for varying levels of expertise in one specific way – namely the impact of expertise on the time necessary to visually search for features (given the number of items present in the interface).

In addition to impacting visual search, expertise and interface complexity might interact in a number of other ways. For instance, as interface complexity increases, novice users might make more feature-selection errors than experienced users because they are less familiar with what a feature does. Broadening the User Model’s definition of expertise and extending the Knowledge Base to handle this more general definition are potential areas for future investigation.

Switching Overhead

The switching overhead factor in the User Model represents the amount of time it takes the user to realize that switching to the FI is required for a feature not present in the PI. This factor allows the User Model to account for the performance implications of having a feature reside solely in the FI if that feature is expected to be used. In addition to this cognitive overhead, once the user realizes that the feature is not present in the PI, the user must also perform the relevant interface actions to switch to the FI. The time to perform these interface actions is estimated by the Knowledge Base.

4.3.3 Knowledge Base

The Knowledge Base is responsible for determining the time necessary for a given user to select a given feature in the two-interface model with a given PI (supplied by the CSM). To predict overall performance, the User Model asks the Knowledge Base for the time necessary to select each feature with an expected usage greater than zero, supplying the Knowledge Base with the relevant expertise distribution. The User Model then takes the Knowledge Base’s estimated feature-selection time, multiplies it by the feature’s expected usage, and, if the feature is not in the given PI, adds the switching overhead estimate.

The Knowledge Base estimates the time necessary for a given feature selection using information on interface layout and a GOMS simulation environment, which is based on the GLEAN tool [74, 75] used in our simulation experiments. In chapter 3, we described how we extended GLEAN to account for varying levels of expertise. Here we provide further details on our extension, and also describe how our GOMS simulation environment handles the User Model’s assessment of expertise, which is probabilistic (and on a per-feature basis). In

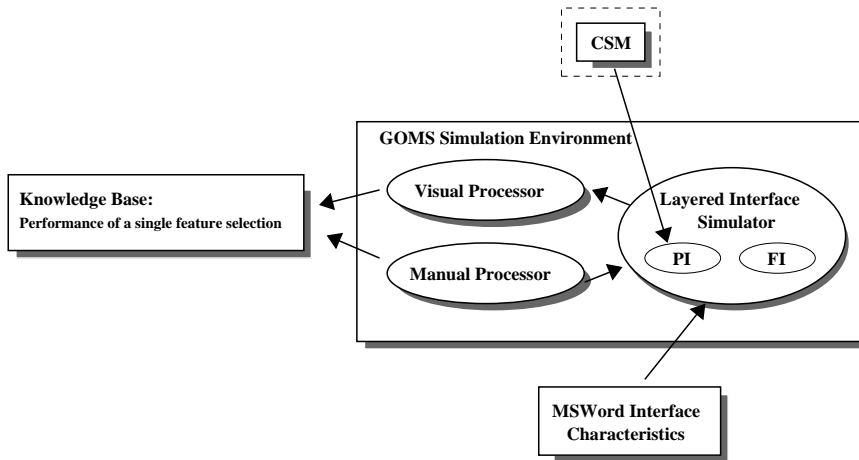


Figure 4.3: Components that allow the Knowledge Base to generate predictions of how long it will take the user to select a single feature in the two-interface model.

in addition, our GOMS simulation environment differs from the original GLEAN tool in that we removed complexities (see [74] for further detail) that are not necessary for MICA’s purposes.⁹ For example, the original GLEAN tool has an Auditory Processor and a complex Cognitive Processor that operates on simulated Working and Long-Term memories. Since MICA does not rely on these components, we removed them (and their interactions with the remainder of the components) for efficiency purposes.

To assess the performance of a single feature selection in the two-interface model, the Knowledge Base contains methods to estimate the time necessary to perform the following basic interface actions: (1) visually search for a feature, (2) point to a feature, and (3) click on a feature. A single feature selection consists of a sequence of these operations. For example, if the User Model requests an estimate for a menu item that resides in the PI, the Knowledge Base simulates the following actions: (1) visually searching for the menu heading, (2) pointing to the menu heading, (3) clicking on the menu heading, (4) visually searching for the menu item, (5) pointing to the menu item, and (6) clicking on the menu item. If the feature resides in the FI, the Knowledge Base simulates a toggle before and after the selection. The toggle button, which is circled in figure 3.1, allows the user to switch back and forth between the PI and the FI. The

⁹As in chapter 3, we use the term “original” to denote the version of the GLEAN tool available prior to our extensions (i.e., GLEAN3).

Knowledge Base assumes that the starting point for a feature selection is always the PI.

The Knowledge Base's GOMS Simulation Environment relies on three main components (depicted in fig. 4.3): a Manual Processor, a Visual Processor, and a simulated version of the two-interface model (Layered Interface Simulator in fig. 4.3). We now describe each component.

Layered Interface Simulator

The Layered Interface Simulator is a complete simulation of the two-interface model [92]. The infrastructure necessary to incorporate a simulated interface into the GOMS predictions comes from the original GLEAN tool. The specifics of this particular simulated interface are unique to our system.

The Layered Interface Simulator supplies the Manual and Visual Processors with information on currently visible interface items (i.e., interface items that the user can see on her screen), including their positions and their dimensions (i.e., their size). In particular, the simulation consists of two simulated versions of a menu- and toolbar-based interface (the FI and the PI), and a toggle button that switches between the two interfaces. While not currently utilized by the CSM component, the Layered Interface Simulator can also simulate the customization mechanism described in section 3.2, should the CSM eventually wish to factor the cost necessary to perform customization-related interface actions into its decision making (i.e., selecting features to add to/delete from the FI/PI).

To allow the Layered Interface Simulator to maintain an up-to-date list of visible items, the Manual Processor notifies the Layered Interface Simulator of pointing or clicking actions. For example, if the user clicks on a menu heading, the Layered Interface Simulator adds the menu's contents to its list of currently visible items. Alternatively, if the user clicks on the toggle button to switch from the PI to the FI, the Layered Interface Simulator indicates that the contents of the FI (i.e., menu headings and toolbar items) are visible rather than those in the PI.

The Layered Interface Simulator operates on a generic menu- and toolbar-based interface. Information on interface layout specific to the MSWord interface is found in the "MSWord Interface Characteristics" component (see fig. 4.3). Relevant interface-layout information includes: (i) which features reside in which menus and toolbars, (ii) the location of the menus and toolbars on

the display, and (iii) the size of menu items and toolbar buttons. The contents of the PI currently under consideration are supplied by the CSM immediately before it asks the User Model to begin assessing performance.

Manual Processor

The Manual Processor, which is a simplified version of its GLEAN counterpart, estimates the time necessary for pointing and clicking operations. The estimated time for pointing is calculated using Fitts' Law ([88]) and requires the dimension and interface location of the target feature, which is obtained from the Layered Interface Simulator. For a click operation, the Manual Processor calculates the time and informs the Layered Interface Simulator of the action. The Manual Processor does not consider expertise when generating estimates for pointing and clicking operations.

Visual Processor

The Visual Processor estimates the time necessary to visually search for a feature. The estimate is based on a probabilistic assessment of user expertise for the feature in question, supplied by the User Model. For our simulation experiments in chapter 3, we extended GLEAN's visual search component to account for the effects of expertise. In this section, we describe the details of our extension. We also describe how our Visual Processor handles the fact that the User Model's assessment is probabilistic, which was not the case in chapter 3.

To assess the amount of time necessary to search for an interface item, the Visual Processor asks the Layered Interface Simulator (i) whether the target item is present and (ii) for the number of "relevant items" present in the interface. The rules for what constitutes a relevant item are as follows:

- Menu heading: If the user is searching for a menu heading, the number of relevant items is the number of menu headings present on the interface.
- Menu item: If the user is searching a menu item, the number of relevant items is the number of items within the open menu.
- Toolbar item: If the user is searching for a toolbar item, the number of relevant items is the number of items on the given toolbar.

The Visual Processor uses the number of relevant items to generate probabilistic performance estimates as follows. First, the Visual Processor calculates

the visual search time for each expertise category using the functions defined in the chapter 3, and repeated here for convenience:

- **Extreme Expert:** Constant (1.2 seconds).
- **Expert:** Logarithmic function of the number of relevant items (Hick’s Law with a coefficient of 0.15 seconds).
- **Intermediate:** Self-terminating linear search of the relevant items (“Time Per Item” of 0.5 seconds).
- **Novice:** Exhaustive linear search of the relevant items (“Time Per Item” of 1.0 seconds).

Next, the Visual Processor calculates the expected value based on the expertise distribution for the feature in question, which is supplied by the User Model. Note that the User Model maintains a unique expertise distribution for each feature in the interface.

4.4 Framework Implementation

MICA’s framework is implemented for MSWord (2003). The framework components are implemented in C++, while the mixed-initiative customization interface is implemented in Visual Basic for Applications (VBA). The framework and interface communicate through a dynamic-link library (DLL).

4.4.1 Choice of Target Application

We chose MSWord as our target application primarily because it allowed us to directly extend McGrenere *et al.*’s purely adaptable prototype, avoiding the extra steps necessary to implement and test an effective customization facility. We soon discovered, however, that providing adaptive support using a commercial application as a testbed for an approach (without access to the source code) has advantages and disadvantages. The first advantage is that MSWord satisfies the key requirement of being feature-rich. It also has a large user base, which is diverse in terms of application-specific expertise, general computer knowledge, and existing attitudes towards its feature complexity. This diversity both justified enabling effective customization and allowed us to recruit a range of participants for our evaluations. Finally, we were able to maintain the option of

utilizing previously collected usage data (e.g., [84] and [91]). A large disadvantage, however, is limited programmatic control over the application’s behaviour. While MSWord’s GUI is generally highly programmable, limitations in the available MSWord APIs affected the manner in which we were able to deliver the adaptive support (discussed in the next section). There were also limitations in the type and quantity of directly accessible information on user interface actions to be used for (i) User Model assessments (e.g., expertise) and (ii) logging purposes in evaluations. For example, the APIs do not expose lower-level mouse events, help-system events, and do not even expose all feature-selection events (e.g., toolbar items such as “Undo/Redo” which have drop-down menus that contain images). These limitations, which were not anticipated, led to both non-insignificant delays and research compromises.

We explored alternatives to MSWord, but found that they had their own unique challenges. For example, it is possible to create one’s own research test-bed application, which would provide unlimited control over the application’s behaviour. However, creating a fully functional feature-rich application requires substantial resources. We also considered using OpenOffice, an open source word processor that is similar (but not identical) to MSWord. While building on an open source project seemed like an attractive alternative initially, we soon learned that manipulating the menus and toolbars was not one of the easily supported extensions (see [99] for further details). Customizing these elements would have required diving into the source code, and like many open source projects, OpenOffice is extremely large and complex (e.g., eleven pages of instructions on how to compile). Furthermore, both alternatives (OpenOffice and a research test-bed application) would have changed the characteristics of our participant pools, and made it more difficult to extend and leverage previous work. OpenOffice does have an existing user base, but it is smaller than that of MSWord and likely consists of users with higher computer-related expertise, since OpenOffice is not nearly as widely marketed or distributed.

Choosing a target application was necessary to (i) illustrate how MICA delivers its support and (ii) evaluate our system. MICA’s framework, however, is designed to be easily applicable to any graphical user interface that consists of toolbars and menus, and is customizable using the two-interface approach. First, the framework is completely separate from the application’s interface since, as described above, the two communicate through a DLL. Second, within MICA, all changes necessary to apply the framework to a different application would be confined to the “MSWord Interface Characteristics” component in the

Knowledge Base (see fig. 4.3). This component would have to be updated to reflect the characteristics of the new application, such as the size and locations of toolbars and menu headings, and the size of toolbar items and menu items. The actual contents of the toolbars and menus (e.g., which features reside in which menus) are specified in a text file.

The type of formal performance-based reasoning that MICA engages in is also applicable to other forms of customization that don't involve a two-interface model. Applying MICA's techniques would require identifying how one could simulate the implications of different customization decisions within the new customization context. For example, with the two-interface model this involves simulating feature selections in the PI with and without the additional complexity, and comparing these results to simulations in the more complex FI. Implementing new simulations would likely require modifications to all three of MICA's components.

4.4.2 User Model Status

Some aspects of MICA have not yet been fully implemented. MICA's general framework, including the on-line GOMS performance assessments, is fully implemented. The User Model, on the other hand, is not yet capable of assessing the user's expertise and expected usages on-line. In this section, we first describe techniques relating to assessing expertise and expected usages on-line, and discuss why we chose to focus on evaluation before fully implementing all components of the framework. We then describe our process for determining an appropriate initial value for the switching overhead factor.

Expected Usages and Expertise

There are existing techniques that could be explored to guide the assessments of both expected usages and expertise. Expected usages could be assessed through a mixture of plan recognition (e.g., [18]), usage history (e.g., [54]) and dialogues with the user (since MICA is a mixed-initiative system). The Lumière work [62] and the recent work by Hurst *et al.* [63] could serve as guides to assessing expertise. We felt, however, that it was first necessary to assess the overall approach before investing time to implement these techniques. While these techniques are available, embedding them within MICA's User Model would not be completely trivial since they would have to be tailored to suit our specific requirements. We believed that evaluating the system to understand the value

of the overall approach and the utility of providing the user with the rationale would provide the most benefit to GUI customization research, despite a user model with some “black box” components. Our decision was also influenced by the lack of easily available on-line, reliable, and detailed information on user interface actions to assist the User Model in performing these assessments (see sec. 4.4.1). In particular, for expertise we would need access to signs that the user is having difficulty locating features in the interface, such as the number of menus opened prior to a feature selection or the number of items moused-over. Obtaining this information would not be problematic if we had access to the source code. In chapter 7 we discuss a potential alternative way to gain access to this information, which was discovered at later stages of this work.

Switching Overhead

For the User Model’s switching overhead component, we used sensitivity analysis in combination with a small study to inform our choice of an appropriate initial value. Sensitivity analysis is the process of quantifying the sensitivity of the outputs of a computational model to variances in parameters in that model. It is often used to determine how much time and how many resources should be devoted to refining these parameters. To understand the impact of switching overhead on the CSM’s decision making, we performed “nominal range” sensitivity analysis, a method that evaluates the effect of varying one parameter across the range of plausible values (switching overhead in our case), while holding the remainder of the system parameters at constant levels (i.e., expertise and expected usages for us) [46]. Our output of interest was the number of features with an expected usage greater than zero that the CSM would recommend *not* be included in the PI (i.e., these features should reside solely in the FI).

To perform the above analysis, we required “default values” for the other model parameters: expected usages and expertise. For expected usages, we used the feature-usage data from seven participants from McGrenere’s *et al.*’s pilot and field studies [91, 92]. For two of these participants (those in the pilot evaluation), the data was collected over a four-month period, while the data for the remaining five participants was collected over a six-week period. We had to determine values for expertise ourselves, since previous studies did not provide this data. For lack of a better alternative, we used the usage counts themselves. Specifically, we used the frequency of usage for each interface feature to define the default expertise value for that feature. In other words, we used the same

source of data to define expected usages and expertise. The usage ranges we used to clamp the expertise distributions to a particular value were as follows: Novice [0,5]; Intermediate [6, 40]; and Expert [41, 100]; Extreme Expert [100, +]. While ideally we would have had actual expertise data for each of these participants, we felt that it was reasonable to use these settings given the scope of this aspect of the work.

The data sets from these seven participants contained an average of 48.9 used features (SD: 18.0). Using the above settings for expertise and expected usages, the sensitivity analysis revealed that for switching overhead values of 60 seconds or more, the CSM always recommended that all used features be included in the PI. Below this threshold, the CSM recommended that anywhere from 0 to 7 used features be excluded from the PI. The average of number of used features that were excluded from the PI was 1.6 and the standard deviation was 2.0. This range and standard deviation indicate that there was some degree of sensitivity below the 60-second threshold, but not an overwhelming amount. It is difficult, however, to assess the practical implications of these differences on either users' performance with the system or their attitudes towards MICA's recommendations without a detailed user evaluation.

To see whether switching overhead could reasonably be larger than this 60 second threshold and to get a sense of the range of plausible values, we conducted a small study with three participants (ranging from novice to expert users). Our objective was to obtain estimates for the "best-case" and "worst-case" scenarios. In both cases, the participant was started off in the PI and asked to select a specific feature. This feature was present only in the FI (i.e., not in the PI). In the "best case," the participant was explicitly told that the feature was not present in the PI and that switching to the FI was required. In the "worst case," the participant had not seen the contents of the PI and was not told whether the feature was present in the PI, or only in the FI. In addition, in this "worst case," the feature that the participant was asked to select was a more advanced and difficult to find feature. In both cases, we observed participant behaviour and timed how long it took the participant to initiate the switch. In the "best case," it took participants 1-5 seconds to initiate the switch, while in the "worst case", it took participants 5-13 seconds. In the "worst case", all participants looked in the target menu before switching, and one participant looked in a second menu briefly before initiating the switch.

These results show that the range of times obtained from this experiment was well under the threshold that would cause the CSM always to recommend

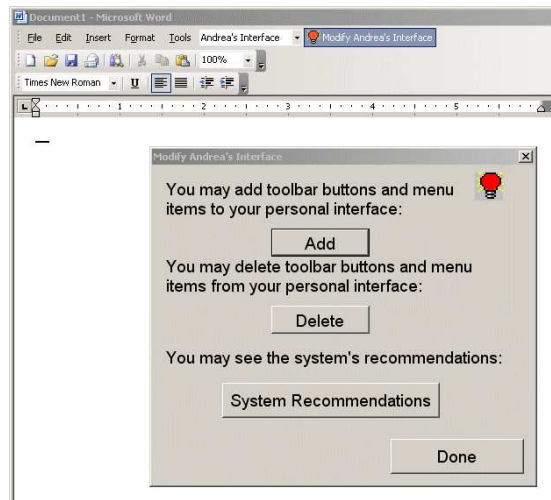


Figure 4.4: The main customization dialogue for the mixed-initiative customization interface.

that the user include all used features, indicating that it is important to get a meaningful value for this parameter. Since the range of times was relatively narrow, however, we are currently using the average value (5.8 seconds) from the small study as the User Model's assessment of switching overhead. This time includes any actions that the user might perform prior to realizing that the feature is missing, such as opening one or more menus. While there was some sensitivity in the CSM recommendations within the range of times obtained in the small study, we did not feel that it was large enough to warrant the effort to further refine this parameter prior to our first evaluation aimed at showing the general validity of our approach.

4.5 Delivering the Adaptive Support

In addition to searching the space of candidate PIs for the optimal PI, the CSM is also responsible for delivering the tailored suggestions to the user. To avoid some of the common disadvantages of purely adaptive interfaces, the delivery of the CSM's customization suggestions is designed to: (1) maintain user control, (2) provide customization support non-intrusively, and (3) maintain interaction predictability and transparency. In this section we describe how we achieved these goals.

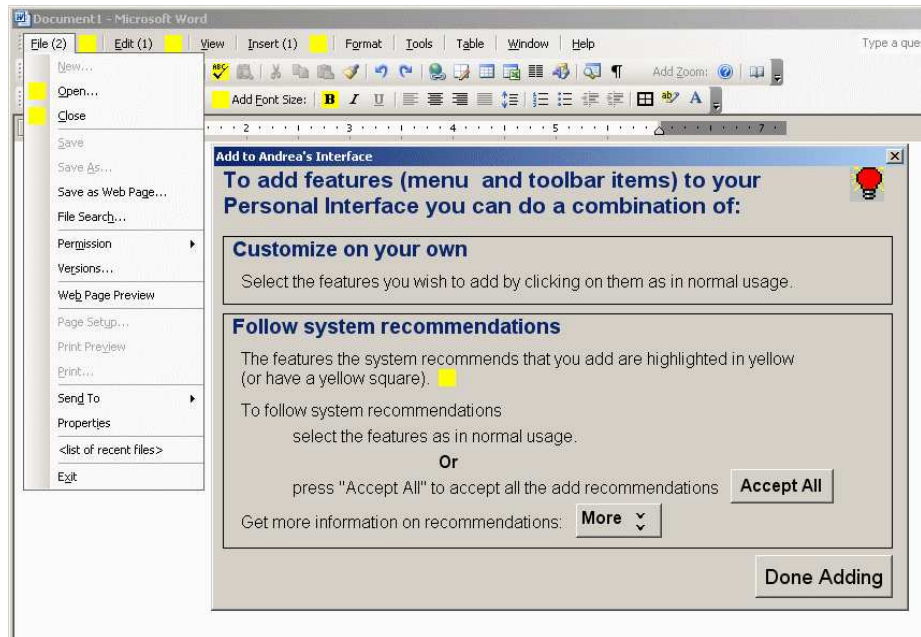


Figure 4.5: The customization screen for adding features in the mixed-initiative interface. To see MICA's recommendations within the menus and toolbars, the figure must be viewed in colour.

4.5.1 Mixed-Initiative Customization Interface

The mixed-initiative customization interface is a direct extension of the original purely adaptable customization mechanism proposed by McGrenere *et al.* [91, 92] (described in sec. 3.2). MICA provides customization recommendations only when the user initiates customization by clicking the "Modify" button (see fig. 3.1). At this point, the dialogue box displayed in figure 4.4 gives the user the option of adding, deleting, or going directly to a list of system recommendations.

Figure 4.5 shows MICA's mixed-initiative customization interface for adding features. When the user selects the "Add" option and enters this mode, the FI is displayed, along with the dialogue box located in the central part of figure 4.5. MICA's recommended additions are made visually distinct by highlighting (in yellow) recommended toolbar items (see the top of fig. 4.5) or by having squares (also yellow) beside recommended menu items (see the pull-down menu in fig. 4.5) and beside menu headings with recommended features inside them. Our original intention was to highlight the entire menu item/heading in yellow,

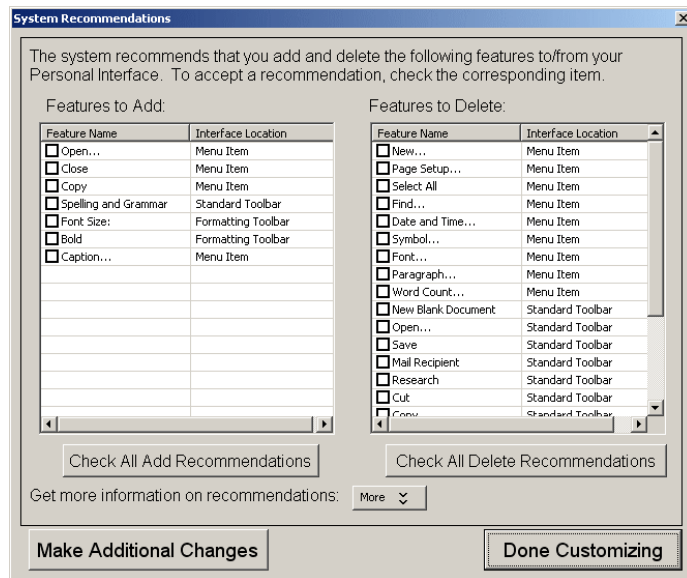


Figure 4.6: The screen containing a list of all system recommendations accessible from the “System Recommendations” option in figure 4.4.

but this was not possible with the available APIs for MSWord.

Users can accept MICA’s recommendations by selecting the highlighted features as in normal usage. Alternatively, the “Accept All” button in the dialogue box allows users who trust the system’s recommendations to accept all of them at once. Users maintain control because it is ultimately up to them to decide when and how to customize and to what degree they wish to follow MICA’s recommendations.

In terms of the other two options in figure 4.4, the “Delete” customization mode is analogous to the “Add” customization mode, except that the PI is displayed with the recommended deletions highlighted as described above. The “System Recommendations” button takes the user directly to a list of all system recommendations. This method of customization, which is shown in figure 4.6, is designed for users who know that they only want to follow system recommendations, since this screen does not provide the user with the opportunity to add or delete non-recommended features. Using this method, however, the user can pick and choose which recommendations to follow without having to go through the menus and toolbars.

The design of the mixed-initiative interface was informed by informal us-

ability testing with pilot participants. In particular, when we discovered that the Word APIs did not allow us to highlight an entire recommended menu item (or menu heading containing recommended items), we pilot-tested a number of alternatives. These alternatives included increasing item size and having borders appear around recommended toolbar items. The solution described above allowed users to distinguish most easily between recommended and non-recommended items.¹⁰

4.5.2 MICA’s Rationale Component

To maintain interaction predictability and transparency, the lack of which is cited as a common disadvantage of purely adaptive interfaces [58], MICA provides the user with access to its rationale. MICA’s rationale component describes why the system is making recommendations and the relevant user- and interface-specific factors impacting its decision-making process. Presenting this rationale has the potential to provide the user with valuable insight into how the system works, but effectively communicating the information to the average user is a challenging design task, particularly since MICA’s algorithm is relatively complex.¹¹ The rationale interface presented in this section represents our first attempt at conveying this information to the user. While explaining adaptive behaviour has been explored in other contexts (e.g., [35]), this is the first attempt to show system rationale in GUI customization research. As a result, we knew that evaluation would be necessary to ascertain what types of information, if any, users find useful, along with how to convey the information. In section 6.2, we describe a revised version of this interface based on feedback from the evaluation described in chapter 5 and a subsequent informal evaluation (also described in sec. 6.2).

Users can access MICA’s rationale by clicking the “More” button next to the line “Get more information on recommendations” in the dialogue box shown in figure 4.5. Having the rationale accessible but initially hidden makes the information available for those who want it, without overloading those who don’t. Clicking the “More” button expands the dialogue box to include the additional pane of information shown in figure 4.7.

¹⁰One limitation of this approach is that the yellow squares cannot be added next to sub-menu headings. Instead, the number of recommendations within the sub menu is displayed next to its name (as is done for the “File” menu heading in fig. 4.5).

¹¹For example, MICA’s algorithm is more complex than one making recommendations based solely on a weighted average of the frequency and recency of feature usage.

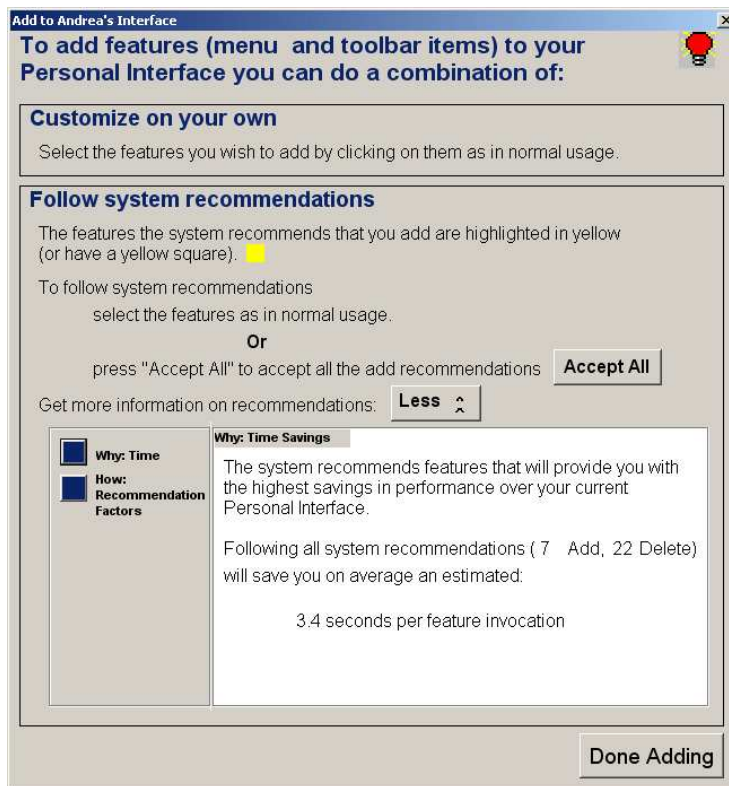
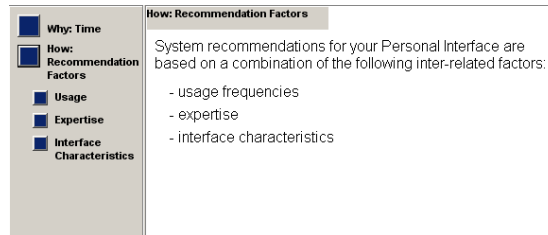
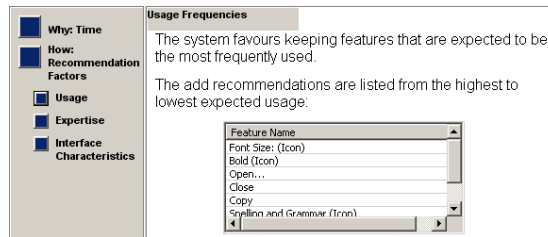


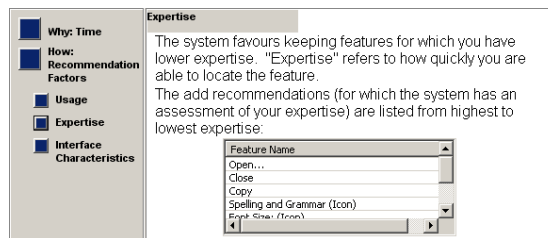
Figure 4.7: The customization dialogue box for adding features with the rationale expanded. The “Why” component is displayed.



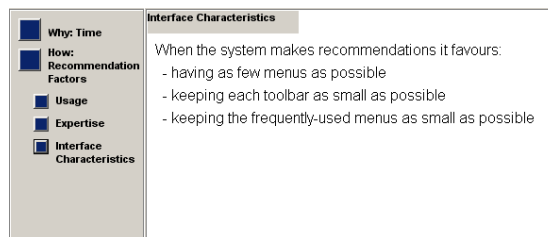
(a) How: Recommendation Factors



(b) Usage Frequencies



(c) Expertise



(d) Interface Characteristics

Figure 4.8: The “How” component of the rationale

MICA’s rationale consists of two main components. The first is the “Why” component – the screen that is first displayed when the user expands the rationale (see fig. 4.7). The “Why” component explains the guiding principle behind MICA’s recommendations, which is that they are designed to provide the user with a PI that will save him time. In addition, because of the formal manner in which performance is quantified during recommendation decision making, MICA is able to report the average estimated time savings per feature invocation should the user choose to accept all recommendations (i.e., the entire optimal PI). This estimated time savings is based on the difference between the User Model’s performance assessments for the user’s current PI and the optimal PI. Since the magnitude of the savings will depend on how much the user uses the interface, the savings is expressed in terms of an average savings per feature invocation.

The second component to MICA’s rationale, the “How” component, provides the user with a more in-depth explanation of the sources of information that the system considers when generating its recommendations. In particular, the “How” component (accessible by clicking the button next to “How: Recommendation Factors” in fig. 4.7) describes three factors that impact the system’s decisions: usage frequencies, expertise, and interface characteristics. As discussed in section 4.3.2, MICA’s User Model considers a fourth factor – switching overhead. We decided not to include a description of switching overhead within the rationale, because we felt that this factor is lower level and would over-complicate the explanation. When the user selects the “How: Recommendations Factors” option, the system starts by presenting an overview of the three “How” factors (see fig. 4.8(a)). The user can obtain more information on each factor by clicking on the corresponding button on the left-hand side of the rationale interface. This action brings up an explanation of the factor and, if relevant, access to a high-level snapshot of the User Model’s assessment (see figs. 4.8(b), 4.8(c), and 4.8(d)). Specifically, the descriptions of the usage frequencies and expertise factors contain ranked lists of the recommendations according to the User Model assessments for the factor (see fig. 4.8(b) and fig. 4.8(c)). These lists are provided to (i) indicate that the system is basing the recommendations on personalized assessments, and (ii) allow users to factor the degree to which they agree with the assessments into their decisions on how to customize.

4.6 Summary

Chapter 4 presented the details of the MICA system. MICA uses on-line GOMS analysis to provide the user with recommendations that are tailored to her expertise, expected feature usage, and characteristics of the interface. MICA's recommendations are presented in a manner that avoids intrusion and keeps the user in full control. MICA suggestions are designed to assist the user in making good customization decisions, but it is ultimately up to the user to decide when and how to follow them. In the next chapter, we describe our first evaluation of MICA (Study One). Study One explores the effectiveness of the system described in this chapter by comparing it to a purely adaptable alternative.

Chapter 5

Study One: Comparing MICA to the Adaptable Alternative

In the previous chapter, we described the design and implementation of the MICA system, which supports user customization using a mixed-initiative approach. In this chapter, we describe our first evaluation of MICA, which we call Study One [15].¹² Study One explored the general effectiveness of MICA's support and how users respond to MICA's approach.

5.1 Study Objectives and Approach

The objectives of Study One were to find out whether users prefer the mixed-initiative support to customizing on their own, and to see how MICA's support impacts both how users customize and their performance with the interface. With these objectives in mind, we designed a laboratory experiment with two conditions: (1) the purely adaptable two-interface model, where users could customize but did not receive system recommendations; and (2) MICA's mixed-initiative interface described in the previous chapter. The conditions were identical except for the mixed-initiative component. We chose this overall design to provide insight into three main questions: (1) Do users prefer the mixed-initiative support to customizing on their own? (2) Does MICA's support have positive effects on task performance? (3) How does the presence of recommendations impact customization behaviour?

In addition to the above main questions, we were also interested in better

¹²An earlier version of parts of this chapter appeared as: A. Bunt, C. Conati and J. McGrenere. Supporting interface customization using a mixed-initiative approach. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 22-31, 2007.

understanding users' reasons for customizing and obtaining feedback on the manner in which MICA delivers its support. Furthermore, we wanted to obtain preliminary feedback on the rationale interface. A study designed to explicitly test the effectiveness of MICA's rationale (Study Two) is described in chapter 6.

5.2 Method

5.2.1 Participants

Twelve participants completed the study (nine females, three males). Participants were recruited by posting signs throughout the university campus because we wanted a relatively broad range of participants (to the degree possible within the university setting). We avoided posting signs within the computer science department, as these students might have opinions with respect to feature complexity and mixed-initiative interactions that are not representative of the general population. All participants were in the 18-29 age range, with the exception of one participant who was in the 50-59 age range. Ten were students (in a variety of fields), one was an administrative manager, and one was a retired teacher. Participants were paid \$10 per hour of participation.

Our study had two pre-screening requirements. First, because of the large number of written and verbal instructions throughout the experiment, our Call for Participation (see appendix C) asked that participants be native English speakers. Second, prior to signing up for the experiment, interested participants first completed a preliminary questionnaire developed by McGrenere and Moore [93] that classifies users as Feature Keen, Feature Shy, or Feature Neutral based on their answers to questions on the following: (i) how they feel about having many functions in the interface, (ii) how much they want to have a complete version of their interface, and (iii) how up-to-date they would like their interface to be. Details of the preliminary questionnaire and the Feature Profile Scale can be found in appendices A and H. We selected only Feature Keen and Feature Shy participants (an equal number of each) because we wanted to avoid having a large number of participants who might have little opinion on their interfaces (i.e., Feature Neutrals). While we did want our participants to care about the state of their interfaces, we did not anticipate that the differences between Feature Keen and Feature Shy users would impact our independent variable (Mixed-Initiative vs. Adaptable) in any significant way. We did not require

#	Feature Keen/Shy	Gender	Age	Occupation	Average Expertise Level (/3)	MSWord usage per week (hrs)
S1	Shy	F	18-29	Student (Arts)	2.54	2-3
S2	Shy	F	18-29	Student (Psychology)	2.68	4-5
S3	Keen	F	18-29	Student (Arts)	2.73	2-3
S4	Shy	F	18-29	Student (Arts)	2.43	2-3
S5	Shy	M	18-29	Student (Psychology)	2.73	1-2
S6	Keen	M	18-29	Admin Manager	2.90	5+
S7	Shy	F	50-59	Retired Teacher	2.10	0-1
S8	Shy	M	18-29	Student (Applied Science)	2.59	2-3
S9	Keen	F	18-29	Student (Microbiology)	2.32	5+
S10	Keen	F	18-29	Student (Science)	2.22	2-3
S11	Keen	F	18-29	Student (Arts)	2.27	3-4
S12	Keen	F	18-29	Student (Visual Art)	2.56	2-3

Table 5.1: Description of the study participants. The “Average Expertise Level” is based on the participants’ self-assessments of each feature used in the experiment on a scale of 1-3 (see appendix B).

participants to have specific amounts of experience using MSWord (other than having used it at least once). Full participant descriptions can be found in table 5.1.

We decided to have 12 participants in the experiment because we felt that this number would provide us with valuable information on the benefits and drawbacks of the overall approach, while keeping the experiment a manageable size. Given that this was MICA’s first evaluation and we did not yet know how users would respond to its mixed-initiative approach, we hoped that the number would provide the right tradeoff between statistical power and research resources. Each session was designed to last approximately three hours and

required the thesis author to be present throughout to observe and conduct the post-session interview. In addition, the length of the session and the pre-screening criteria led to some difficulty recruiting (75 participants completed the preliminary questionnaire). We were aware that 12 participants would not necessarily be enough to achieve statistical significance, but we felt that this number would provide interesting insight into MICA's effectiveness.

5.2.2 Design

The experiment used a within-subjects factorial design with Interface type (Mixed-Initiative or Adaptable) as the primary factor. A within-subjects design (i.e., each participant completes both conditions) was chosen for its ability to gather direct preference data and account for variability owing to individual differences. In addition, this design requires fewer participants, since each participant provides data for two conditions. Screenshots of the interface in Mixed-Initiative condition were presented in the previous chapter (see figs. 4.4 - 4.8). The Adaptable interface was the same version of the interface that was used in the simulation experiments described in chapter 3 (see figs. 3.2 and 3.3).

Participants completed two tasks, one with each version of the interface (described in sec. 5.2.4). Therefore, Task was a within-subjects control variable. Both Interface Order and Task Order were between-subject controls. To account for learning effects, we counterbalanced the order of interface and task, resulting in four configurations.

5.2.3 Apparatus

The experiment was conducted on an IBM Thinkpad running Windows XP with a 2.0 GHZ processor, 1.5 MB RAM, and a 15" screen. The Adaptable and Mixed-Initiative interfaces were coded for MSWord 2003 using Visual Basic for Applications (VBA) macros. The MICA's framework was implemented in C++.

5.2.4 Tasks

One of the biggest challenges in designing a laboratory study involving customization is how to motivate users to customize, since customization is typically meant to be beneficial over a period of time longer than a laboratory study. Thus, the experimental tasks had to be designed such that: (1) customization

would *actually* have the potential to be beneficial, and (2) participants would *feel* that customization could be beneficial. Satisfying both constraints required that: (i) the tasks be designed so that participants would spend most of the task time selecting features from the menus and toolbars as opposed to entering text, and (ii) participants would feel that there was enough regularity in the feature usage to make customizing their interfaces worthwhile.

Participants performed two tasks (A and B), one with each version of the interface. Full details of the tasks can be found in appendix E. Each task consisted of a list of step-by-step instructions and a target final document. Each step described an interface operation to be performed (or in some cases, a small amount of text entry) and indicated whether to use the toolbars or menus to complete the step, but did not explicitly give the name of the command. We refer to this type of task as a *guided task*. The restricted nature of the guided tasks served two purposes: (1) the tasks required a large number of menu selections and could still be completed within a reasonable-length session (3 hours in total), and (2) it allowed us to assign accurate values to the expected usage component in the User Model (described in the “Procedure” section – sec. 5.2.5).

Alternatives to guided tasks include asking participants to select a stream of named menu and toolbar features (an approach used in a previous study comparing an adaptive and adaptable interface [41]), or a more open-ended task, such as “write a short report on topic X.” We wanted to make our tasks somewhat more ecologically valid and engaging than the selection-stream alternative, but an open-ended task would result in too few menu selections in the same study duration and less accurate information for the User Model. Guided tasks appeared to strike the right balance between these two extremes.

To further motivate customization, we used task repetition in combination with a small amount of deception. Each task was actually repeated three times; to make customization appear even more beneficial, however, participants were told that each task would be repeated up to five times. The customization mechanism was enabled only after the first repetition of each task to allow participants to become familiar with the task before customizing. To motivate usage of the PI, for each task, participants were given a starting PI that contained many, but not all of the features required for the task and some features that were not needed.

Task Similarity

Since our primary factor of interest in this within-subjects experiment was Interface (Mixed-Initiative or Adaptable), the tasks were designed to be similar in overall length and complexity. Ideally, in a within-subjects experiment, tasks are isomorphic to isolate the effects of the primary factor. Aspects of our study requirements, however, made it difficult to use the more standard and straightforward approaches to isomorphic task design. These approaches typically involve having the same basic task construction, and making superficial changes so that participants are not required to complete the same task twice in different conditions (because of transfer effects). One example of a superficial change would be to have users select features at the same locations in the interface, but change their names both within the interface and in the task (e.g., [41]). Since we were using the same application in both tasks (MSWord), this approach was not possible. A second option involving superficial changes would be to have the same menu selections, with changes to the underlying documents (e.g., different topics). With this option, we felt that the transfer effects between conditions, especially the learning effects, would be too high.

Our remaining option was to explore different task constructions that would be similar, while maintaining two key requirements that we considered to be key to the success of this particular experiment. First, we wanted little overlap between the features required to complete the two tasks in order to minimize both learning effects between conditions and transfer effects in terms of how participants customized. For the latter we were concerned that if the tasks were too similar (without being identical), participants would have difficulty distinguishing between the two tasks when it came time to decide how to customize their Personal Interface in a given condition. Second, we wanted to maintain a high degree of ecological validity within the guided task structure, which meant that the tasks had to have a logical flow.

In an application like MSWord, there are numerous dimensions to task similarity. In addition to the more easily quantifiable dimensions listed in table 5.2, relevant dimensions include:

- The amount of work necessary to both invoke a feature and complete the related step (some features require searching through drop-down lists; others require interacting with a dialogue box after the selection)
- The expected usages of the features that are not in the starting PIs and

Dimension	Task A	Task B
Needed features not in the starting PI	7	5
Features in the starting PI	34	40
Different features required	20	21
Feature invocations total	59	35
Task steps on instruction sheet	56	45

Table 5.2: A comparison of Task A and Task B along five dimensions.

when they are invoked

- The “difficulty” of the features, which depends on the prior experience of each participant
- The absolute positioning of the features and where they are located relative to other features
- When features are invoked repeatedly, how close the repetitions are

Fully understanding the tasks along all of these dimensions would have required extremely detailed task analysis and extensive piloting. We did not feel that this type of effort was warranted for this particular evaluation. Rather, we focused on our two key requirements, minimal overlap and logical flow, and settled on the roughly similar tasks summarized in table 5.2. The biggest difference between the two tasks is the number of total feature invocations. Task A has more total feature invocations, but on occasion a number of invocations of the same feature are repeated in quick succession. In addition, some of Task B’s feature invocations involve dealing with a dialogue box after a feature selection. Therefore, despite the different number of feature invocations, we expected the two tasks to take roughly the same amount of time to complete.

5.2.5 Procedure

The procedure for the experiment was as follows: (1) Participants completed a detailed questionnaire designed to assess their expertise for each feature used in the experiment (see appendix B). (2) The questionnaire results and the information on each feature’s anticipated usage frequency in each of the guided tasks were used to initialize the User Model. Recall that this step is necessary because the User Model currently cannot assess these measures on-line. (3) The two-interface model and customization mechanism were briefly demonstrated

to the participants using the interface (i.e., Mixed-Initiative or Adaptable) assigned to them for their first condition. Both demonstrations involved showing participants how to add a menu item and a toolbar item. In the Mixed-Initiative interface, participants were also shown an example of a system recommendation and were told that it was up to them to decide to what extent they wished to follow the recommendations. (4) Participants performed the first guided task (repeated three times, with the customization mechanism enabled after the first of three repetitions – see sec. 5.2.4). (5) The interface to be used in the second condition was introduced. For the Mixed-Initiative interface, this new interface was briefly demonstrated (i.e., participants were shown example recommendations as in step 3 above). For the Adaptable interface, participants were simply told that the customization mechanism would no longer contain system recommendations. (6) Participants performed the second guided task (repeated three times, with the customization mechanism enabled after the first of three repetitions). (7) Participants completed a post questionnaire. (8) Participants were interviewed by the thesis author. A session typically lasted 2 hours and 30 minutes, but ranged from 2 to 3 hours.

Participants were told that if they did not know how to complete a task step, they should give it their best attempt, after which, if they were still unable to figure it out, they could ask the experimenter. The experimenter would provide help (either as a result of a request or pro-actively) if a participant was unable to complete a step within approximately 30 seconds. Such occurrences were marked on an observer sheet and occurred on only a small percentage of each participant’s total task steps (average: 2.5%, SD: 1.6%). No customization-related help was given at any point during the experiment.

5.3 Pilots

In addition to informal testing with friends and colleagues, the above protocol was refined and pilot tested with five participants. Some decisions were influenced by these pilots. The first was the decision to add the restriction to the Call for Participation that participants be native English speakers. This requirement was added after it appeared that language difficulties contributed to two pilot participants having difficulty understanding both the task instructions and the customization mechanism. Second, we decided to briefly demonstrate the customization mechanism after one participant had difficulty understanding

how to use it. We had considered not demonstrating the interface to illustrate its simplicity but were concerned that given the short duration of the study relative to the high task load, some participants would be unwilling to spend even a small amount of time exploring and learning how to use the mechanism. Third, we decided to allow participants to ask for help but also to be pro-active in providing help if the participant spent approximately 30 seconds searching for a given feature. This decision was made to keep the experiment a reasonable length for all participants, including those with lower MSWord expertise, and because pilot participants varied in their willingness to ask for help. Finally, we decided to remove one of the four expertise categories defined in chapter 3 from our expertise questionnaire (referred to as “Access Levels” in appendix B). We removed the Extreme Expert category because we noticed that, while in a number of cases participants tended to over-estimate their expertise, most of the over-estimation occurred with the highest expertise category. Since the highest level should occur for only the most expert users on only a subset of their most frequently used features, we decided to omit the category entirely. The downside of this decision is that participants with extremely highly expertise might not have received the best possible recommendations and the time savings listed in the rationale component might have been overestimated.

5.4 Measures

Our evaluation had a number of quantitative and qualitative measures. These measures are described below, grouped according to category.

Performance

- Overall Performance: the amount of time the participants took to complete the task overall (three repetitions), including customization time.
- Task Performance: the amount of time the participants spent on the task (three repetitions), ignoring time spent in the customization mechanism.

Customization Behaviour

- Customization Time: the amount of time spent in the customization mechanism when some customization actually occurred.

- Customization Sessions: the number of times participants entered the customization mechanism and customization actually occurred.
- Features Added/Deleted: the total number of features added to/deleted from the Personal Interface.

Impact of Recommendations on Customization Decisions and Methods Used to Follow Recommendations

In addition to the above measures, which apply to both conditions, we measured how user customizations matched system recommendations in the Mixed-Initiative condition (the percentage of recommendations followed and the percentage of customizations that did not correspond to recommendations). In the Adaptable condition, we measured how user customizations matched what the system would have recommended. We also analyzed which method(s) participants used to follow the recommendations.

Interface Preference

On the post questionnaire, participants were asked to state which of the two interfaces they would install on their machine (Overall Preference). Participants were also asked to state which interface they preferred (or whether they found them equal) for the following five criteria: (1) ease of use (Easy to Use), (2) ease of deciding which features to add to the PI (Easy to Add), (3) ease of deciding which features to delete from the PI (Easy to Delete), (4) whether the PI matched their needs after customization (Match Needs), and (5) how fast they found the customization process (Fast).

Reasons for Customizing and Feelings Towards Recommendations

The post questionnaire also asked participants to rate (on a five-point scale) each of three potential reasons for customizing (listed in table 5.10): (i) to avoid using the FI, (ii) to have a small but appropriate PI, and (iii) for improved performance. Participants were also asked to rate their feelings towards the system recommendations on four dimensions (listed in table 5.11): (i) their trust in the system's recommendations, (ii) how easy it was to tell which features were recommended, (iii) the quality of the recommendations, and (iv) the understandability of the recommendations.

Additional feedback on customization, the recommendations, and the rationale were gathered in open-ended parts of the post-questionnaire and during the interview.

5.5 Results

Out of the twelve participants who completed the study, eight customized in both conditions. The remaining four participants customized in only one condition: three in the Adaptable condition (S2, S5, S7) and one in the Mixed-Initiative condition (S1). For three of these participants (S1, S5, S7), the customization occurred in the second condition. Unless otherwise specified, the results presented in this section are based on the data from only the eight participants who customized in both conditions.

The quantitative dependent measures pertaining to both conditions were analyzed using univariate repeated-measures ANOVA with Interface (Mixed-Initiative or Adaptable) as the primary within-subjects factor. Two between-subjects control factors were included in the analysis as a result of the counterbalancing: Interface Order and Task Order. Along with statistical significance, we report partial eta-squared (η^2), a measure of effect size. Effect size measures the practical significance of the differences found. To interpret this value, 0.01 is a small effect size, 0.06 is medium, and 0.14 is large [28]. The results for these quantitative measures are summarized in table 5.3. We also provide summary tables for these measures.

Before performing the ANOVA with Interface as the primary within-subjects factor, we checked for an effect of Task (A vs. B). As intended, we did not find a significant main effect of Task on any of our dependent measures.

5.5.1 Performance

For Overall Performance, the participants were faster in the Mixed-Initiative condition, spending an average of 28 minutes 6 seconds total time compared to 30 minutes 19 seconds in the Adaptable condition. This main effect was marginally significant ($F(1, 4) = 6.522$, $p = 0.063$) with a large effect size (partial $\eta^2 = 0.620$) (see table 5.4). The results were similar when considering Task Performance only (see table 5.5). Participants spent less time completing the tasks in the Mixed-Initiative condition (26 minutes, 40 seconds) than in the Adaptable condition (28 minutes, 44 seconds), a difference that was also

Dependent Variable	Mean		SD		F(1,4)	p	η^2
	MI	AD	MI	AD			
Overall Performance (minutes)	28:06	30:19	6:09	5:29	6.522	0.063	0.620
Task Performance (minutes)	26:40	28:44	5:29	5:05	6.587	0.062	0.622
Customization Time (minutes)	1:06	1:35	0:33	0:38	8.170	0.046	0.671
Customization Sessions	2.88	3.75	1.8	1.9	1.324	0.314	0.249
Features Added	6.1	6.8	0.8	1.5	2.778	0.171	0.410

Table 5.3: A summary of the results for the main quantitative measures. (N = 8) MI=Mixed-Initiative, AD = Adaptable

Source	SS	df	MS	F	Sig.	partial η^2
Interface	71156	1	71156	6.522	0.063	0.620
Interface*TO	2426	1	2426	0.222	0.662	0.053
Interface*IO	16706	1	16706	1.531	0.284	0.277
Interface*IO*TO	37539	1	37539	3.441	0.137	0.462
Error	43651	4	10910			

Table 5.4: The univariate repeated-measures ANOVA for Overall Performance. (N = 8) TO = Task Order and IO = Interface Order

Source	SS	df	MS	F	Sig.	partial η^2
Interface	61256	1	61256	6.587	0.062	0.622
Interface*TO	3660	1	3660	0.394	0.564	0.090
Interface*IO	18496	1	18496	1.989	0.231	0.332
Interface*IO*TO	18496	1	18496	1.989	0.231	0.332
Error	37199	4	37199			

Table 5.5: The univariate repeated-measures ANOVA for Task Performance. (N = 8) TO = Task Order and IO = Interface Order

Source	SS	df	MS	F	Sig.	partial η^2
Interface	3321	1	3321	8.170	0.046	0.671
Interface*TO	638	1	638	1.617	0.272	0.288
Interface*IO	52.6	1	52.6	0.133	0.733	0.032
Interface*IO*TO	410	1	410	1.040	0.365	0.206
Error	1577	4	394			

Table 5.6: The univariate repeated-measures ANOVA for Customization Time. (N = 8) TO = Task Order and IO = Interface Order

marginally significant ($F(1, 4) = 6.587$, $p = 0.062$, partial $\eta^2 = 0.622$). While both results are only marginally significant, the fact that both had large effect sizes suggests that the Mixed-Initiative interface had a practical impact on both performance measures.

5.5.2 Customization Behaviour

We analyzed the impact of Interface on customization behaviour in terms of the time necessary to customize, the number of separate customizations sessions, and the number of features that participants added and deleted.

For Customization Time, participants spent significantly less time customizing in the Mixed-Initiative condition than in the Adaptable condition ($F(1,4) = 8.170$, $p=0.046$, partial $\eta^2 = 0.671$), with averages of 1 minutes 6 seconds and 1 minute 33 seconds respectively. There was also a significant between-subjects main effect of Interface Order ($F(1,4)= 10.062$, $p=0.034$, partial $\eta^2 = 0.716$), showing that participants spent less time customizing across both conditions if they saw the Adaptable condition first (average: 55 seconds, SD: 24.4 seconds) than if they saw the Mixed-Initiative interface first (average: 1 minute 46 seconds, SD: 12.3 seconds). Interpreting this order effect is difficult since there were only four participants per order and there appeared to be large individual differences. This might be an indication, however, that the simpler Adaptable interface provided scaffolding for the Mixed-Initiative interface and not vice versa. This result could also potentially be attributed to the fact that participants in the Adaptable/Mixed-Initiative order received an additional interface demonstration (see sec. 5.2.5).

While Interface did have an effect on Customization Time, it did not significantly impact the number of customization sessions (see table 5.7). Participants customized slightly fewer times in the Mixed-Initiative condition than in the Adaptable condition (2.9 vs 3.75). This difference was not statistically

Source	SS	df	MS	F	Sig.	partial η^2
Interface	3.06	1	3.06	1.324	0.314	0.249
Interface*TO	1.56	1	1.56	0.676	0.457	0.145
Interface*IO	7.56	1	7.56	3.270	0.145	0.450
Interface*IO*TO	5.06	1	5.06	2.189	0.213	0.354
Error	9.25	4	2.31			

Table 5.7: The univariate repeated-measures ANOVA for Customization Sessions. (N = 8) TO = Task Order and IO = Interface Order

Source	SS	df	MS	F	Sig.	partial η^2
Interface	1.56	1	1.56	2.778	0.171	0.410
Interface*TO	3.06	1	3.06	5.444	0.080	0.576
Interface*IO	1.56	1	1.56	2.778	0.171	0.410
Interface*IO*TO	3.06	1	3.06	5.444	0.080	0.576
Error	2.25	4	0.56			

Table 5.8: The univariate repeated-measures ANOVA for Features Added. (N = 8) TO = Task Order and IO = Interface Order

significant ($F(1, 4) = 1.324$, $p = 0.314$), but the effect size was large (partial $\eta^2 = 0.249$). We comment on main effects that were not significant (or marginally significant) but had large effect sizes in our discussion section (sec. 5.6).

There was also no significant effect of Interface on the number of features added ($F(1, 4) = 2.778$, $p=0.171$, partial $\eta^2 = 0.410$), but table 5.8 shows two marginally significant interactions. The first is a marginally-significant two-way interaction between Features Added and Task Order ($F(1,4) = 5.444$, $p = 0.080$). Second, there is a three-way interaction between Features Added, Task Order, and Interface Order ($F(1,4) = 5.444$, $p = 0.080$). Interpreting these interactions is difficult because (a) the number of participants per cell is small (two per group for the three-way interaction and four per group in the two-way interaction) and (b) they might be in part due to differences in the tasks. If the instructions were followed perfectly, Task A had two more used features missing from its starting PI than Task B. Since participants added roughly the same number of features in both conditions, the decreased customization time in the Mixed-Initiative condition was not a result of participants failing to customize.

None of the 12 participants deleted any features. When asked why in the interview, the majority of participants (eight) said that the extra features in the PI weren't bothersome (67%). Four explicitly mentioned that it wasn't worth the time necessary to delete them (33%), two said that the PI was small enough already (17%), and two thought the extra features might be useful at some point

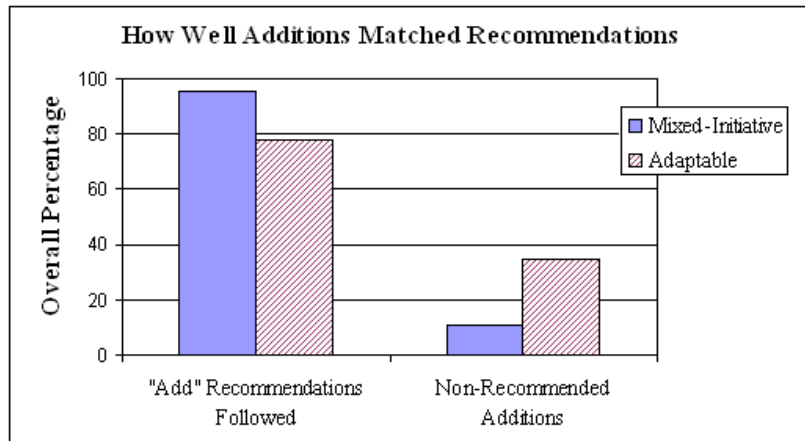


Figure 5.1: How well participants additions to their PI matched MICA’s recommendations in the Mixed-Initiative Condition (N=9). The data for the Adaptable condition indicates how well participants’ additions matched what MICA *would have* recommended for these particular users (N=11).

(17%). The percentages do not sum to 100 because some participants gave more than one reason.

When analyzing the data for Customization Time and Customization Sessions, we discovered that the design of the “Accept All” feature (shown in fig. 4.5) might need to be revisited. This more automatic form of customization was utilized by five participants for some portion of their customization. After customizing using “Accept All,” one participant (S10) had particular difficulty remembering what she had customized, entering the mechanism an additional seven times without customizing. In the interview, this participant revealed that she would enter the customization facility when she could not immediately locate the feature in the PI: “I clicked on there thinking that I didn’t have it and then I went through and realized that I did have it, but that I didn’t know where it was.”

5.5.3 Impact of Recommendations on Customization Decisions

We found that the nine participants who customized in the Mixed-Initiative condition followed the vast majority of the system’s recommended additions. Specifically, out of the total number of Add recommendations, figure 5.1 shows

that 96% were followed. Participants in the Mixed-Initiative condition also added features that were not recommended. In particular, figure 5.1 indicates that 11% of their customizations were not recommended features.

To gain an understanding of whether participants would have made the same customization decisions with or without the system’s help, we also examined the customizations of the 11 participants who customized in the Adaptable condition. We compared their customizations to what would have been recommended by the mixed-initiative system (with the User Model appropriately initialized). These results are also displayed in figure 5.1. We found that in the Adaptable condition participants added only 78% of the features that would have been recommended (compared to the 96% discussed above). Furthermore, 35% of their customizations did not match features that would have been recommended (compared to the 11% discussed above).

The above results when combined with the performance results suggest that the recommendations impacted the participants’ customization decisions in a positive manner. The data provides encouraging evidence that in the Mixed-Initiative condition, participants followed the recommendations, added fewer non-recommended features, and performed better than those in the Adaptable condition.

5.5.4 Methods Used to Follow Recommendations

As discussed in section 4.5.1, participants could follow recommendations using any combination of the following three methods:

1. *Self Selection*: following recommendations by selecting the features highlighted by MICA as in regular usage (see fig. 4.5).
2. *Accept All*: clicking the “Accept All” button (see fig. 4.5).
3. *Get Recommendations*: going directly to the list of recommendations accessible from the main customization dialogue (see fig. 4.4), and selecting from the list (see fig. 4.6).

Table 5.9 summarizes how participants followed the recommendations. The majority of the recommendations were followed using the Self Selection method (65%), with four participants using this method exclusively. The Accept All method was also popular, accounting for 35% of the followed recommendations. None of the participants choose to add recommendations through the Get Recommendations method. The lack of use of this method of customization might

	Self Selection	Accept All	Get Recs
% recommendations added	65%	35%	0%
# participants who used method	7	5	0
# participants who used method exclusively	4	2	0

Table 5.9: Summary of methods used to follow the recommendations. The five participants who used a combination of Self Selection and Accept All followed a portion of the recommendations using Self Selection and then clicked the “Accept All” button to follow the remaining recommendations. (N=9)

be at least partly due to the fact that, unlike the add screen, there was no demonstration of this part of the customization mechanism.

In the interview, the five participants who used the Accept All method indicated that they did so because they trusted the system recommendations (3/5), and/or to save time or make customization easier (5/5). One participant also mentioned that she used the method part way through customizing when she realized that everything she wanted to add was recommended. Four different reasons were given by the four participants who did not make use of the Accept All method: (1) one participant thought it would be easier to use the Self Selection method than it actually was, (2) one participant did not want to use the method without knowing what the recommendations were, (3) one participant did not feel that all of the recommendations were necessary and did not want any extra clutter, and (4) one participant said that he knew exactly what to add on his own. Three of these users said they would consider using this method in the future, but one participant indicated that he would continue to stick with the Self Selection method:

“In this case the recommendations were tailored sort of towards the tasks but then there was still ones I didn’t need. If I were setting it up for my own use, if I had the full program, then again ... there would be things the computer wouldn’t know I needed and things that the computer thought I needed that I wouldn’t” (S8).

5.5.5 Interface Preference

In addition to the above mainly quantitative results, our evaluation also provides qualitative support for the Mixed-Initiative interface. In particular, our within-subjects design allowed us to obtain direct preference information. Figure 5.2 displays these results. For Overall Preference, MICA’s Mixed-Initiative

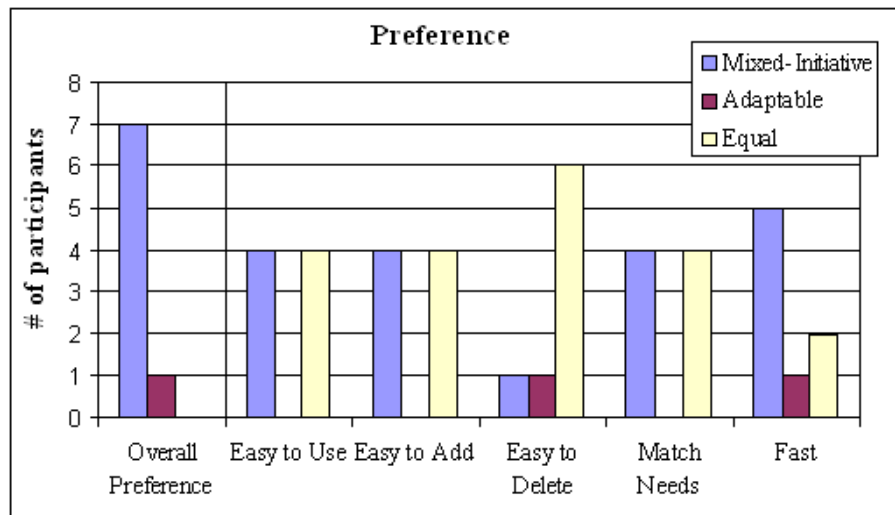


Figure 5.2: The Interface preference data. For Overall Preference, participants were forced to choose. For the individual criteria, they were given the option to rate the two “Equal.” (N=8).

interface was preferred by seven of the eight participants who customized in both conditions. This is consistent with the individual criteria, which showed that participants either preferred the Mixed-Initiative interface or found the two to be equal on all criteria. There were two exceptions to this: one user (S9) rated Adaptable best for Easy to Delete and another user (S6) rated Adaptable best for Fast. For Easy to Delete, all responses were hypothetical since none of the participants entered this mode of customization. The participant who found customization faster with the Adaptable interface (and preferred it overall) was an expert user who said that he knew exactly which features to add and found that there was too much text in the dialogue box describing the customization procedure in the Mixed-Initiative interface (see fig. 4.5).

5.5.6 Reasons for Customizing

Table 5.10 summarizes the participants’ responses concerning three potential reasons for customizing. The data is based on all participants except S2, who missed this section of the questionnaire. The fact that the highest rated reason is task performance is encouraging, since it forms the basis of MICA’s recom-

Reasons for Customizing	Mean	SD
To reduce the number of features that had to be accessed using the Full Interface.	3.09	1.70
To make the Personal Interface as small as possible while still being appropriate for the tasks.	3.45	1.39
To help complete the tasks more quickly.	4.64	0.50

Table 5.10: Reasons for customizing ranked on a 5-point scale. 1 = Strongly Disagree and 5 = Strongly Agree (N=11 – missing responses from S2)

mentations. A free-form section provided participants with an opportunity to list additional reasons for customizing. Three participants entered comments in this section indicating that the nature of the experimental setup was also a factor. In addition to the reasons listed in table 5.10, these participants said that they customized because the tasks were restricted and repetitive or they were curious to try out the customization mechanism. During the follow-up interviews, participants were also asked why they customized and why they added features. Again, performance had the strongest support (50%, 8/12) but some participants also liked the simplicity of the PI (33%, 4/12) and some seemed to want to use the PI exclusively (33%, 4/12). The percentages do not sum to 100 because some users provided more than one reason.

For three of the four participants who customized in one condition only, the interview revealed that experimental load might have been an issue. The responses of these three participants, all of whom did not customize until the second condition, suggest that they might have customized in both conditions given the right experimental setup. Two of the three participants said that they forgot that customization was an option. The third participant said she wasn't sure why she did not customize in the first condition and wished that she had. For those who forgot to customize in the first condition, the second demonstration in between conditions likely served as an additional reminder.¹³ The fourth participant customized in the first condition only. This participant preferred the FI and decided to use it exclusively in the second condition.¹⁴

¹³Those who interacted with the Adaptable interface in the second condition did not receive a second demonstration, but customization was discussed a second time (see step 5 of the procedure in sec. 5.2.5).

¹⁴The participant said that she wasn't sure that using the FI exclusively was an option until the second condition. It is unclear why she came to this realization in the second condition, since she did spend a portion of her time in the FI in the first condition.

Statement About Recommendations	Mean	SD
I trusted the system to make good recommendations	4.11	0.60
It was easy to tell which features were recommended.	3.78	1.09
Recommendations were appropriate for the tasks.	4.44	0.53
I understood why the system made the recommendations it did.	4.00	1.00

Table 5.11: Feelings towards recommendations ranked on a 5-point scale. 1 = Strongly Disagree and 5 = Strongly Agree (N = 9)

5.5.7 Feelings Towards Recommendations

On the post questionnaire, the nine participants who customized in the Mixed-Initiative condition were asked to rank aspects of the system’s recommendation. Table 5.11 summarizes these results. Overall responses were positive, as were responses on a free-form section of the questionnaire, where participants were asked to indicate what they liked about the recommendations. The most common responses were liking how the recommendations were presented (33%, 3/9), or that the recommendations were appropriate for their tasks (56%, 5/9). Finally, one participant (S11) liked that MICA reminded her to make multiple customizations in one session.

In terms of what participants disliked about the recommendations, some wanted all features needed for the task to be recommended (22%, 2/9). Three participants (33%) pointed to three different usability issues: (1) having to look through the menus to see what was recommended, (2) the amount of text in the dialogue box, and (3) the fact that the entire menu heading wasn’t highlighted (see fig. 4.5). Finally, as we noted when presenting the preference results, one participant (S6) did not like the amount of text in the dialogue box (see 4.5).

5.5.8 Impressions of the Rationale

Despite not being an explicit focus of the study, we had hoped that at least some participants would view MICA’s rationale in order to provide preliminary information on its usefulness. Unfortunately, none of the participants in the study chose to look at the rationale. In the post-session interview, the majority of the nine participants who customized in the Mixed-Initiative condition indicated that either they were too focused on completing the tasks (44%, 4/9) or they did not need the information (44%, 4/9). When asked if they would look at the rationale in a more realistic setting, only 33% (3/9) of the participants felt that they would. It is important to note, however, that participants were

not aware of what information was in the rationale, since this feature of the Mixed-Initiative interface was not demonstrated during the interface training.

To obtain some feedback on rationale usage, participants were then asked to take a minute to look through the information that would have been provided. Participants were asked to comment on both the “Why” and “How” components of the rationale. We asked more focused questions about the “Why” component, since the ability to express predicted time savings as a result of accepting recommendations is one of MICA’s distinguishing characteristics. The discussion below is based on comments from only the nine participants who customized in the Mixed-Initiative condition, since they had interacted with the recommendations.

The majority of participants responded positively to the information in the “Why” component. In particular, 56% (5/9) of the participants felt that the time savings suggested by the system would have motivated them to accept the system recommendations, 33% (3/9) felt that it probably would have (but they were not as certain), and 11% (1/9) indicated that it probably would not have acted as a motivator. When asked specifically about the magnitude of the time savings (which ranged from 2.1 to 5.4 seconds per feature selection), 44% (4/9) thought the number was quite significant and 22% (2/9) felt that the number was small but indicated that the small amount of savings would add up over time. Of the remaining 33% (3/9), two participants indicated that while they found the number small, it would make them consider following the recommendations. The remaining participant, who was not motivated by the time savings in general, did not find the time savings to be significant. The following quotes give an idea of the types of response and, in conjunction with the above percentages, indicate that how best to communicate the estimated time savings requires further exploration:

- “Generally, I use the keyboard and a lot of the right-click, which I find to be faster than using either the toolbar or the menu. I mean, I guess that’s why I do it, ’cause it’s faster, so I mean if this is going to make it faster when I do do things using the toolbar and menu, then I would do it, for sure” (S4).
- “Maybe if I was a secretary and I used Word a lot more that would mean more to me but, 3.8 seconds, it is interesting so it makes me want to try it, but it doesn’t mean a lot” (S12).

- “2.1 seconds doesn’t mean too too much to me, probably because I have a strong knowledge of where things are to begin with” (S6).

We also asked participants if they had any comments on the usefulness or helpfulness of the “How” component of the rationale. Two participants found the information interesting (but not necessarily useful), with one of these participants mentioning that it would make her feel motivated to accept recommendations:

- “Kind of interesting that the computer would kind of keep track of that sort of thing and make it easier for the user and that would definitely motivate to me to accept [the recommendations]” (S11).
- “[helpful] for understanding the program, I guess ... I wouldn’t necessarily say helpful, I’d say it is kind of interesting because it tells you how the computer does that” (S10).

The remaining users were less positive. At least two users felt the language of the rationale was too technical. The remaining users expressed a general lack of interest in the information, e.g.,

- “It doesn’t tell you much. It doesn’t like convince you to use your own Personal Interface” “Everything [the information] is there, I just don’t find it convincing myself” (S3).

5.5.9 Feature Keen/Shy Differences

While we did not anticipate the Feature Keen/Shy classification would impact our main quantitative or qualitative measures, we did look for any clear trends. One noticeable difference was that out of the eight participants who customized in both conditions, six were Feature Keen and two were Feature Shy. In addition, Feature-Shy participants rated all the customization reasons listed in table 5.10 higher (on average) than the Feature-Keen participants. These findings might suggest a difference between these two groups with respect to customization, although not necessarily in terms of mixed-initiative interfaces. The only notable difference with respect to the mixed-initiative support was that out of the five participants who used the Accept All method for all or part of their customization, only one participant was Feature Shy. This might suggest that delegating the responsibility completely to the system might be less attractive to Feature-Shy participants (at least initially); however, we also had more Feature-Keen data points.

Further research would be required to substantiate all of the above findings, before which additional validation work on the Feature Profile Scale might be needed. After the completion of the study, we analyzed all responses to the preliminary questionnaire (75) and found that a much larger percentage of our users were classified as Feature Neutral as compared to the distribution obtained by McGrenere and Moore [91, 93], and that the reliability of the individual scales and the overall profiling scale (measured using Cronbach’s alpha (α)) had decreased. For the most part the decrease in Cronbach’s α was fairly small, but one of the sub scales saw a substantial decrease in reliability. Further detail on the analysis and comparison can be found in appendix H.

5.6 Discussion

5.6.1 Effectiveness of MICA’s Mixed-Initiative Support

Our results provide encouraging evidence that when MICA has a fairly accurate User Model, users prefer the mixed-initiative system to the purely adaptable alternative. In addition to users preferring MICA’s support, participants followed the vast majority of the system’s recommended additions (96%) and the data suggest that these recommendations helped improve performance in terms of time on task. Although the performance differences were small, this is to be expected given the relatively short duration of the study. The time savings should add up given longer periods of use in real settings, especially if the user’s tasks maintain a certain amount of consistency. Exactly how much consistency must be present in the user’s feature usage to make customization beneficial, in terms of both objective and subjective benefit, remains an area of future investigation. Feature usage likely doesn’t have to be as stable as it was in our study. If the set of regularly used features is changing substantially and frequently, however, then the time to customize might begin to outweigh any benefit.

The data also show that MICA’s support has the potential to decrease customization time. Since the effort necessary to customize is one of the disadvantages of purely adaptable interfaces, decreasing customization time might make users more willing to customize. The results, however, also point out a potential downside of allowing the system to do more of the customization on behalf of the user. For one user in particular, the more automatic form of customization led to her having difficulty remembering what she had already added to her PI when it came to features with which she was not as familiar. Not all partici-

pants who used this feature experienced this problem, indicating that this more automatic form of customization might be problematic for some and not others. Furthermore, it might simply be a matter of improving the current confirmation dialogue, which displays the names of the newly added and deleted features.

Apart from a significant decrease in customization time, the effects of the mixed-initiative support on other measures pertaining to customization behaviour did not reach significance, but the effect sizes were large. Given that we had twelve participants, only eight of whom could be used in our within-subjects analysis, our experiment was likely underpowered. More participants would be required to validate other trends observed. One such trend is that participants customized slightly fewer times with the mixed-initiative interface than with the adaptable interface. As we described in chapter 3, the Up Front strategy is generally more efficient than the As You Go strategy, yet As You Go was the primary strategy used by participants in McGrenere *et al.*'s field study of the purely adaptable interface [92]. While our study was not able to show that MICA's support helps users batch their customizations so that they do not have to initiate customization as frequently (i.e., helps them implement a strategy that is more Up Front than As You Go), the trend was in the right direction. If MICA's support is not helping users engage in customization that is more Up Front than As You Go, then the decrease in customization time is likely primarily because MICA is helping to increase the speed of (i) customization decision making, and/or (ii) interface operations necessary to select features in the customization interface. A second trend observed in our study is that users added slightly fewer features with the mixed-initiative interface than they did with the adaptable interface, which could mean that MICA helps users customize more selectively. In addition to verifying the above trends, an evaluation with a larger number of participants would also be required to validate the effects of MICA's support on performance, since these results reached only marginal significance in our study.

In terms of the design of the mixed-initiative interface, participants generally liked the way the recommendations were presented. There were comments, however, suggesting that participants wanted something similar to what is in the "System Recommendations" screen (i.e., a list of selectable recommendations) accessible from the "Add" screen. In other words, there was a desire to be able to view what was being recommended and quickly accept certain recommendations, without having to completely delegate the responsibility to the system. As discussed in chapter 4, our use of the yellow squares to indicate

recommendations, as opposed to highlighting the entire menu headings/items, was a compromise because of limitations in the available MSWord APIs. The evaluation confirmed that our solution is generally acceptable, since only one participant mentioned it as a usability issue and we did not observe participants having any particular difficulty locating recommendations.

Finally, while the evaluation provides support for MICA basing its recommendations on performance and for the general appropriateness of its recommendations, the evaluation also suggests that two assumptions embedded in MICA’s decision making require further exploration. First, MICA assumes that users will be willing to switch to the FI for less-frequently used features, but 22% of users indicated that having features “missing” from the recommendations was something they disliked about the system. It would be interesting to explore whether some users would prefer to use the PI solely, regardless of the performance impact, or whether better understanding this performance tradeoff would influence their preference. Second, MICA assumes users will be willing to delete features, which was not the case in the context of our study. Users were also reluctant to delete features in McGrenere *et al.*’s field study [91, 92].

5.6.2 Study Methodology

In addition to providing insight into the value of MICA’s mixed-initiative support, the evaluation provided insight into how to evaluate customization. In this section we reflect on some of our key evaluation design decisions, and we describe some of the lessons learned. We also discuss the study’s limitations.

Laboratory Study

In this study, we chose to evaluate MICA’s support for customization in a laboratory setting. An alternative with much higher ecological validity would have been to evaluate MICA with a field study. The laboratory study provided us with a number of benefits, one of which was the ability to create a controlled environment that permitted a direct comparison to the adaptable alternative on measures such as task performance. The second major benefit was the ability to evaluate MICA’s overall approach early in the development cycle. A field evaluation would have required a fully functional system and much more development time to ensure that the system could be used completely bug-free for long periods of time. It would be difficult to justify investing this amount and

type of effort prior to understanding the general benefits and drawbacks of the overall approach.

Having conducted the experiment in a laboratory environment and over a single session means that we do not know how users would respond to the mixed-initiative support if they were using the system on a day-to-day basis, performing tasks that they might feel have less obvious structure. McGrenere *et al.*'s field study [91, 92], however, provides evidence that most users saw enough regularity in their feature usage in a real working context to believe that customization would be beneficial. Thus, there is reason to believe that users would see the value of mixed-initiative support in such contexts as well.

Within-Subjects Design

We chose to employ a within-subjects design because we felt that it was critical to obtain direct preference data. This is also one of the key distinctions between our study and that of Debevc *et al.* [36], which compared a mixed-initiative interface to an adaptable alternative. An additional benefit of a within-subjects design is that it can account for variability owing to individual differences. Our results show that there was a large amount of individual variability among our participants, in terms of both task performance and customization behaviour.

The within-subjects design did have its disadvantages. As discussed in section 5.2.4, it was difficult (if not impossible) to design isomorphic tasks. First, the more straightforward approaches, requiring only superficial changes to the target application or the tasks themselves, did not meet the constraints of our study. Second, when trying to create different but similar task structures, we found it extremely difficult to balance all of the many relevant dimensions, while still minimizing the potential for transfer effects and having tasks with a logical flow. When using a commercial application, one notable challenge is predicting how participants' pre-existing expertise with the application affects task similarity (without detailed pre-screening). Depending on prior experience, aspects of a task that might be simple for one user, might be complex for another (and vice versa). In our experiment, the task differences had minimal effects, but this was partly luck. Given enough data, even seemingly small task differences would begin to show. Isomorphic tasks are not an issue with between-subjects designs because the same task can be used for all conditions.

The second main disadvantage of using a within-subjects design was the limited amount of data we were able to obtain on how participants interacted

with the mixed-initiative system. Given our desire to have a single-session experiment, we were only able to have participants complete one task in each condition.

Tasks and Customization

Our guided tasks and customization context allowed us to examine customization in a different light from approaches taken in the past. In this section we compare our tasks and customization context to two related evaluations: McGrenere *et al.*'s field study [92] and Findlater and McGrenere's laboratory study [41] (discussed in sec. 2.4.1). We compare the three studies along a number of dimensions that could impact willingness and ability to customize including: task load, customization timing, and customization permanence. While the evaluation by Debevc *et al.* [36] is also closely related, insufficient detail is provided on its design.

First, the three studies differ in the level of cognitive load present in their experimental tasks and the timing of customization. Higher levels of cognitive load and the need to take time away from tasks to customize are both likely to make customization less attractive to users and might affect users' abilities to do so effectively. Findlater and McGrenere's laboratory study used selection streams (i.e., users were asked to select a sequence of named menu items) and allowed participants to customize during a break between tasks. Selection streams, however, would (for the most part) have lower load than users' real tasks, and users would rarely have a designated break to customize. Our guided tasks had higher cognitive load and users had to take time away from their tasks to customize (at a point of their choosing). Therefore, we were able to examine customization in a more ecologically valid manner along these dimensions than Findlater and McGrenere. This increase in ecological validity, however, was accompanied by a decrease in internal validity. Findlater and McGrenere's study had a much more controlled environment with which to assess the effects of interface (in their case, adaptable, adaptive and static) on performance.

In McGrenere *et al.*'s field study, users were performing their own real-world tasks and it was up to them to decide when to take the time to customize. Since McGrenere *et al.*'s experiment was run in the field, it is clearly more ecologically valid than our study. The differences between the two studies in terms of task load and customization timing, however, are not clear cut. First, the specific amount of task load in the field would depend on the particular users and their

tasks. The same can be said for customization timing. In the field, some users might find time to customize during natural breaks in their real-world tasks, whereas others might have difficulty finding convenient times to customize.

In a laboratory setting, customizations are less permanent than in the field. As a result, special care must be taken to ensure that the experimental setup will motivate customization. Both Findlater and McGrenere’s study and our study used task repetition. Since our guided tasks did not permit us to have as many feature selections as the selection-stream approach, we used a larger number of repetitions (2 vs. 3) and also used deception (i.e., participants were told that the tasks would be repeated up to 5 times). Our strategy did motivate a large number of people to customize. The downside of having each task repeated three times was that a large percentage of the study session time was devoted to non-customization related behaviour, meaning that relatively small amounts of the behaviour observed during the three-hour session were actually of interest. In addition, the motivation was not sufficient to encourage customization by all participants in both conditions – we were able to obtain data to use in our within-subjects analysis from only eight of the twelve participants. Findlater and McGrenere’s study also motivated customization by having a financial incentive for performance. While designing the study, we explored the possibility of adding a financial incentive for performance in hopes of increasing our customization rate. We decided against it because we were worried that combining a financial incentive for performance with the expression of time savings within the rationale would bias participants towards accepting recommendations. Since running the study, we also believe that it could have caused some participants to be less willing to explore the customization mechanism.

Pre-screening According to the Feature Profile Scale

In this evaluation we pre-screened our participants according to McGrenere and Moore’s Feature Profile Scale [93] to avoid having a large number of participants who may have little opinion on the state of their interfaces (i.e., Feature Neutrals). This decision, however, led to considerable difficulty recruiting; a total of 75 participants completed the questionnaire before we reached our target number of study participants. Two factors contributed to this difficulty. First, 43 of the 75 participants were classified as Feature Neutral, making them ineligible for our study. As was eluded to in section 5.5.9 and we discuss in greater detail in appendix H, problems with the questionnaire may have caused

a larger than usual number of participants to be classified as Feature Neutral. Second, pre-screening introduced an additional communication step into our recruitment procedure, and we lost some participants at each step. Study Two, which we describe in the next chapter, did not include this pre-screening criterion. Recruiting in Study Two was substantially easier than in Study One and including Feature Neutrals in the study did not result in a noticeable impact on the quality of Study Two's data.

User Model Assessments

It remains to be seen whether or not our results will hold either in the laboratory environment or in the field when the User Model is performing on-line assessment of user expertise and expected feature usage. In our study there was some noise in the User Model's assessments. First, the settings used for the expected usages were not always accurate because some participants deviated from the task instructions or found unanticipated ways to accomplish a particular step. Second, we believe that not all participants accurately self-assessed their expertise. We did not quantify the above sources of noise because it would be laborious and difficult, particularly for expertise, and the vast majority of our results indicate that the model is functioning reasonably well. While there was some noise present in the settings used in our study, on-line User Model assessments are likely to be less accurate, which impacts the external validity of the study. When we were designing the study, we investigated the idea of deliberately introducing noise into the expected usages to simulate real world usage with a fully functional User Model, but decided against it for two reasons. First, understanding the effects of user modelling accuracy on the benefits of mixed-initiative support would be an entire study in itself. Second, we were concerned that inaccuracies in the laboratory environment, where tasks have such obvious structure, would stand out more than they would in a more realistic setting.

Additional Limitations

Our evaluation indicates that MICA's recommendations had a positive influence on the users' customization decisions, but it did not allow us to directly test the impact of all user-model and decision-making parameters. For example, the study did not assess the value of incorporating user expertise into the recommendations, or performing the on-line GOMS analysis. We felt that this

initial validation step, and the follow-up study described in the next chapter, were essential to justifying the considerable amount of research resources that would be required to run an evaluation, or series of evaluations, with human subjects that would fully test all aspects of the framework. Such evaluations, known as *ablation* studies, would involve disabling individual components of the User Model and CSM, and testing the resulting effectiveness of the system.

There are also limitations of our statistical analysis that deserve mention. First, as we discussed in section 5.6.1, our sample size did not provide enough statistical power to draw firm conclusions. Second, our use of multiple ANOVA to analyze the dependent measures as opposed to a single MANOVA might have increased our likelihood of making a Type I error. Finally, we presented the results of our analysis with Interface as the primary within-subjects factor; however, the experiment employed a 2 x 2 factorial design with Task as the additional within-subjects factor. For the sake of brevity, we did not present Task results since we did not find any significant or marginally significant main effects of Task and it was our intent that the tasks be similar. Despite the lack of main effects, there exists the possibility that some of our results could be attributable to task differences. Completely isolating the impact of the mixed-initiative and adaptable interfaces would require an experiment where participants performed the exact same task in each condition. A between-subjects design, using the identical task for each interface condition, would have alleviated the challenges we faced with designing isomorphic tasks for this study and would avoid the possibility of substantial carry-over effects. A between-subjects design, however, would require far more subjects than a within-subjects design to attain sufficient statistical power and would not permit direct preference information to be gathered.

Despite its limitations, we believe that our evaluation is an important first step. Showing that this type of mixed-initiative support can be beneficial motivates investigating appropriate on-line assessment techniques, in addition to conducting larger and more detailed evaluations, first in the lab and then in the field.

Evaluating the Utility of the Rationale

We did not create the right conditions in this experiment to motivate rationale viewing. A large number of participants stated that they were simply too focused on completing the tasks to look at the rationale. Customization, which

also requires a removal of focus from the tasks, has a much more obvious benefit than does viewing the rationale. The rationale is something we would expect users to want to see when they were not feeling time pressure or if they were feeling like exploring the system – conditions that are difficult to simulate in the laboratory environment. The rationale is the focus of Study Two, which we describe in the next chapter. In Study Two, we motivate rationale usage by requesting that participants look through the information during our pre-task instructions.

5.7 Summary

This chapter described the first evaluation of the MICA system, a within-subjects two-condition study comparing MICA to the purely adaptable alternative. The results indicate that users prefer the mixed-initiative support, at least in circumstances where the system has an accurate user model. The evaluation also provides initial evidence that MICA’s recommendations improve time on task and decrease customization time. Furthermore, the evaluation is one of the few direct comparisons of a mixed-initiative and adaptable interface, and thus it extends the body of knowledge pertaining to the value of mixed-initiative approaches. It is particularly encouraging that the mixed-initiative support was preferred to an adaptable interface that had performed well in a previous study.

Our evaluation did not provide any insight into the value of showing the user the rationale for the system’s customization suggestions, but did provide informal feedback on its effectiveness. The next chapter describes a follow-up evaluation designed specifically to test the effects of viewing the rationale on users’ qualitative impressions of the system and on their customization behaviour.

Chapter 6

Study Two: Understanding the Utility of the Rationale

In chapter 5, we described an evaluation aimed at testing the overall effectiveness of MICA’s mixed-initiative approach. Chapter 6 describes the second study we ran as part of this thesis work (Study Two)[16].¹⁵ The aim of the second study was to assess the utility of MICA’s rationale, including understanding why users do or do not want access to the information and for what reasons.

6.1 Study Objectives and Approach

The primary objective of Study Two was to better understand the qualitative impact of the rationale on users’ attitudes toward the system. In light of user feedback from Study One, we anticipated large individual differences along this dimension. Based on the outcome of Study One, we expected that some users would appreciate the information and find it useful, while others would find the rationale unnecessary. We wanted to better understand the reasons underlying different reactions, and the possible advantages and disadvantages of providing access to rationale information. We did not, however, expect the rationale to significantly impact customization decisions, since in Study One, participants already followed most of the system’s recommendations for additions to the PI (96%) without accessing the rationale. While there would be room for improvement in terms of having users follow recommendations for deletion, Study One and previous work [92] have shown users to be rather reluctant to delete features. Therefore, we anticipated most of the interesting findings to come from the qualitative data on user attitudes and preferences. A secondary objective

¹⁵An earlier version of parts of this chapter appeared as: A. Bunt, J. McGrenere and C. Conati. Understanding the utility of rationale in a mixed-initiative system for GUI customization. In *Proceedings of the International Conference on User Modeling (UM 2007)*, pages 147-156, 2007.

of Study Two was to obtain additional feedback on two assumptions within MICA’s framework: (1) that users are, in general, willing to delete; and (2) that users are willing to switch between the two interfaces, as opposed to using one interface exclusively.

As in Study One, we ran a two-condition within-subjects study. The two conditions compared in Study Two were two versions of the MICA system: one with and one without the rationale. The remainder of Study Two’s method was based on Study One. Prior to describing Study Two’s method and results, we first describe an iterative design and evaluation process that we used to improve the design of both the rationale and the customization interfaces prior to running Study Two.

6.2 Improving the Interface

Prior to running a full evaluation targeting the rationale, we engaged in an iterative design and evaluation process to ensure that we would be evaluating the best possible rationale and customization interfaces. First we improved these interfaces using feedback obtained during Study One. Next, we ran an informal evaluation of the changes, and further improved the design.

We first describe the structure of the informal evaluation. We then describe the final rationale and customization interfaces used in Study Two that resulted from feedback from both Study One and the informal evaluation. We note major changes and participant feedback that led to those changes.

6.2.1 Informal Evaluation

A total of eight participants, all of whom were computer science graduate students, participated in the informal evaluation. Our main reason for using computer science graduate students was that they were easy to recruit. An additional advantage, however, was that many of these students had experience with interface design and provided very detailed feedback. The structure of the evaluation was as follows: (1) The two-interface model was briefly explained. (2) Participants were asked to explore the mixed-initiative customization mechanism and to provide any feedback they had. Since participants did not use the two-interface model to perform tasks with the MSWord (as in Study One), participants were asked to imagine that the recommendations were tailored to them. (3) After participants spent time exploring and commenting on the in-

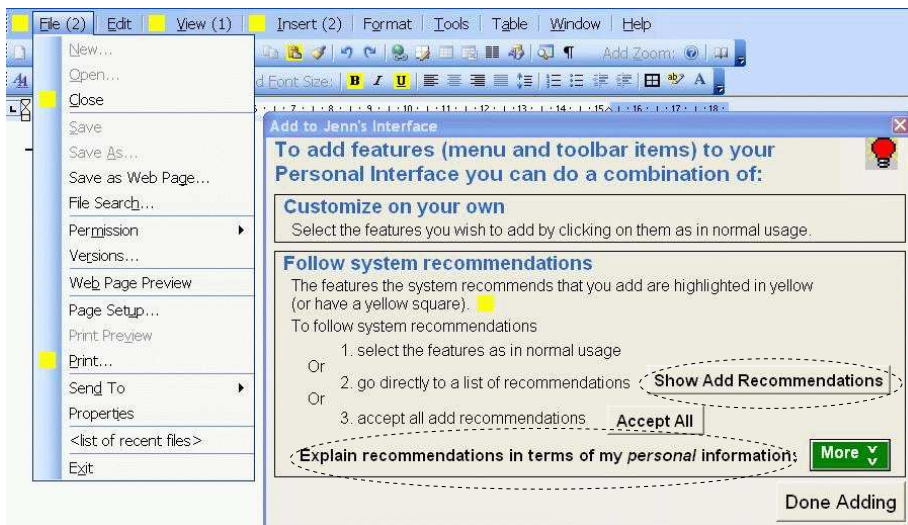


Figure 6.1: The customization interface for adding features used in Study Two. To see MICA’s recommendations within the menus and toolbars, the figure must be viewed in colour.

terface, they were asked to look through the rationale (if they hadn’t already done so). (4) Participants were interviewed, with a focus on the following issues: (i) wording clarity; (ii) missing or unnecessary information; (iii) whether it was clear where to access, and how to navigate through the rationale; and (iv) whether they could see themselves accessing the rationale in a study situation. The entire session lasted approximately 30 minutes.

6.2.2 Resulting Rationale and Customization Interfaces

Figures 6.1 - 6.5 illustrate the final customization and rationale interfaces that were evaluated in Study Two. The most significant changes made to these interfaces between Studies One and Two (circled in the figures) were as follows:

- We added a list of selectable recommendations to the Add/Delete screen (see figs. 6.1 and 6.2), similar to what is present in the Get System Recommendations screen (see fig. 6.3). We did so to address the complaint from Study One that one had to search through the recommendations to see what was recommended.
- We added icons (when available) next to the features in the recommen-

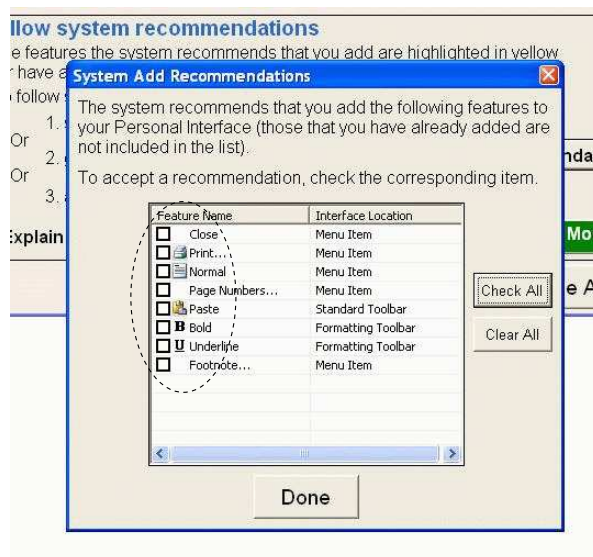


Figure 6.2: The list of recommendations accessible from the “Show Add Recommendations” button in figure 6.1.

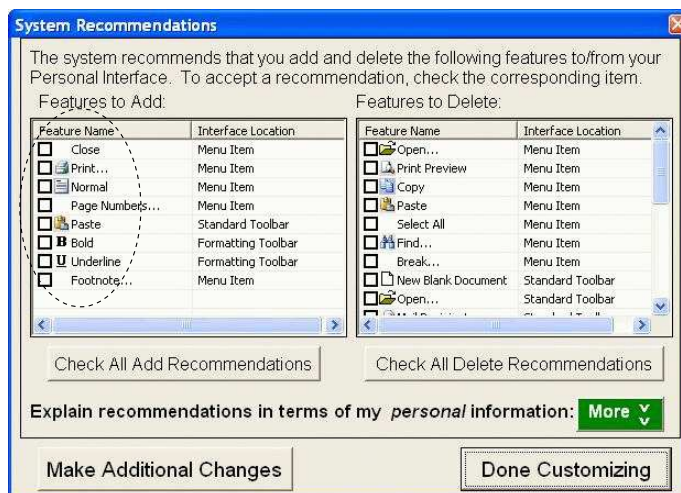


Figure 6.3: The screen accessible from the “System Recommendations” option in the main customization dialogue in Study Two.

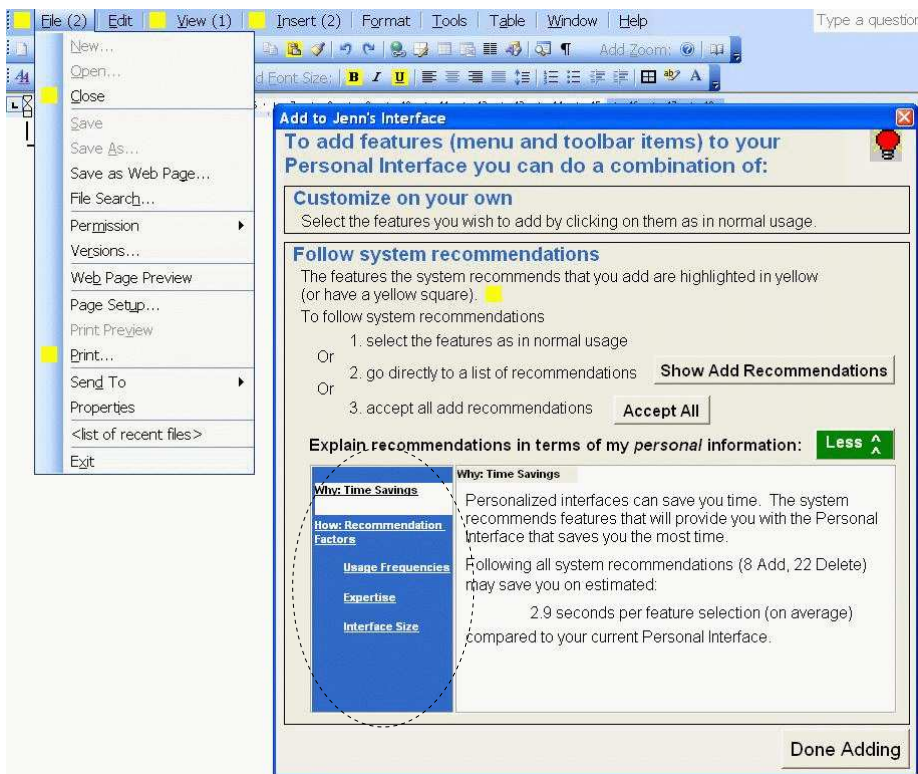
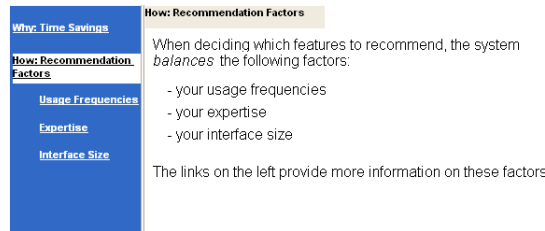
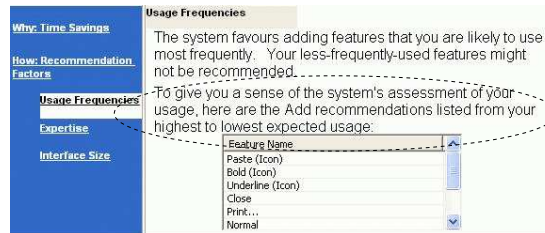


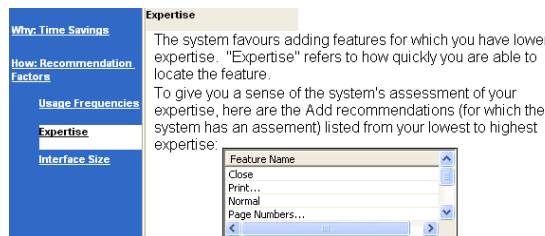
Figure 6.4: The customization interface for Adding features used in Study Two, with the rationale expanded. The “Why” component is displayed.



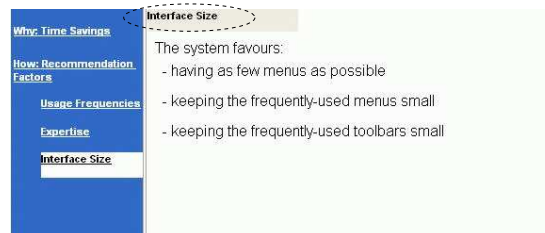
(a) How: Recommendation Factors



(b) Usage Frequencies



(c) Expertise



(d) Interface Size

Figure 6.5: The “How” component of the rationale used in Study Two.

dations lists (see figs. 6.2 and 6.3) and the confirmation dialogues, as a visual cue in hopes of diminishing the potential problem associated with the “Accept All” method of customization. In Study One, one user had difficulty remembering which features she had customized after using this method.

- We emphasized that the rationale contains personalized information both in the access point from the customization interfaces (see fig. 6.1) and within the rationale itself. With the interface used in Study One (see fig. 4.5), a number of pilot participants indicated that they expected the rationale to contain canned text, such as what can typically be found in a help file, and that they would be less likely to access this type of information.
- The mechanism to navigate the rationale was changed after participants commented that they found the version used in Study One either difficult to navigate or non-standard. Instead of the buttons used in Study One (see fig. 4.7), we used hyperlinks (see fig 6.4) since a number of participants tried to click on the text next to the buttons.
- We made numerous changes throughout the rationale to make the wording less technical and to improve its clarity. One notable modification in this regard, based on participants’ suggestions during the informal evaluation, was to change the name of the “Interface Characteristics” factor to “Interface Size” (see fig. 6.5 (d)). Since the term “Characteristics” was considered too technical, “Size” was suggested in order to simplify the language while still conveying the main message in the factor’s description, which is that the system tries to keep the menus (particularly those that are frequently used) and the toolbars small.
- We described why the ranked lists of recommendations (according to User Model assessments) are included within the rationale (see figs. 6.5 (b) and 6.5 (c)). Pilot participants expressed some confusion as to what the lists represented and why they were there.

After Study One and the subsequent informal evaluation, two open questions remained. The first was how much information to include in the ranked lists. Some participants wanted specific User Model assessments, while others mentioned (when asked) that this additional information would be too confusing or

would cause information overload. Second, feedback obtained during Study One indicated that the current expression of time savings might not be meaningful for some users. We decided not to explore alternative expressions of the time savings information within our informal evaluation because many participants in Study One did feel that the seemingly small amount of savings would add up and we felt that appreciating this information (in particular) requires having interacted with the two-interface model. None of the participants from the informal evaluation spontaneously commented on the number being too small.

6.3 Method

6.3.1 Participants

Sixteen participants completed Study Two (thirteen females, three males). Participants were recruited by posting signs around the university campus since, as in Study One, we wanted a relatively broad range participants. All participants were in the 18-29 age range, with the exception of two participants who were in the 30-39 age range. Thirteen were students, one was an administrative manager, one was a sales associate, and one was a secretary. Participants were paid \$10/hr.

In this experiment, we did not screen participants according to the Feature Keen/Shy classification. This decision was made to simplify recruiting and also because of the potential problems with the scale uncovered during Study One. Despite its potential limitations, we continued to ask participants to complete the Feature Keen/Shy questionnaire in case having the data turned out to be useful at some point.¹⁶ Table 6.1 summarizes the demographic information and Feature Keen/Shy classification for the 16 participants who completed the experiment.

We continued to screen for English language abilities, this time using the phrase “highly fluent in English” on our Call for Participation (see appendix I). Since we considered this criterion to be important, and we noted that a number of participants in Study One overestimated their English fluency, we also administered the North American Adult Reading Test (NAART) [120]. The NAART is a quick to administer test measuring verbal intelligence, which requires par-

¹⁶As in Study One, participants completed the questionnaire prior to signing up for the experiment. We were concerned that completing the questionnaire immediately before interacting with the rationale would cause participants to think about feature complexity in a manner that they wouldn’t normally, thus biasing our results.

#	Feature Keen/Shy	Gender	Age	Occupation	Average Expertise Level	MSWord usage per week (hrs)
S1	Neutral	F	18-29	Admin Manager	2.86	3-4
S2	Neutral	M	18-29	Student (Mechanical Engineering)	2.95	5+
S3	Keen	M	18-29	Student (Pharmacy)	2.02	2-3
S4	Shy	F	18-29	Sales Associate	2.38	3-4
S5	Neutral	F	18-29	Student (Business)	2.23	4-5
S6	Neutral	F	18-29	Student (Arts)	2.86	2-3
S7	Neutral	F	18-29	Student (Cognitive Science)	2.33	5+
S8	Keen	F	18-29	Student (Commerce)	2.62	1-2
S9	Neutral	F	30-39	Student (Human Kinetics)	2.02	0-1
S10	Neutral	F	18-29	Student (Arts)	2.29	3-4
S11	Shy	F	18-29	Student (Political Science)	2.50	2-3
S12	Keen	F	30-39	Secretary	2.95	4-5
S13	Keen	F	18-29	Student (Arts)	2.79	1-2
S14	Neutral	F	18-29	Student (English, Econ)	2.76	1-2
S15	Keen	M	18-29	Student (Microbiology, Immunology)	2.62	2-3
S16	Neutral	F	18-29	Student (Social Work)	2.90	0-1

Table 6.1: Description of the study participants. The “Average Expertise Level” is based on the participant’s self-assessments of each feature used in the experiment on a scale of 1-3 (see appendix B).

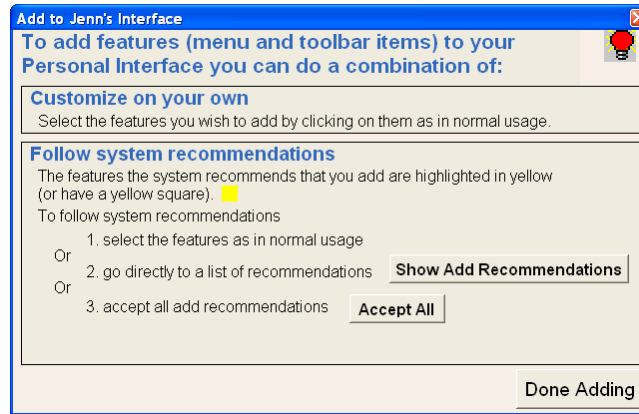


Figure 6.6: The main dialogue box in the “Add” screen for the No-Rationale version. Everything is the same as with the Rationale version, except the access point to the rationale is removed.

ticipants to read a list of 59 words across two pages increasing in difficulty. Using a somewhat arbitrary threshold, we accepted only participants who got a least 50% of the words on the first page correct. We also had participants complete the expertise questionnaire at this time.¹⁷ A total of six participants were disqualified after completing this pre-screening phase.

6.3.2 Design

The experiment used a within-subjects factorial design with Version type (Rationale vs. No-Rationale) as the primary factor. A screenshot of the No-Rationale version is displayed in figure 6.6. A within-subjects design was chosen primarily to elicit direct comparative statements.

As in Study One, participants completed two tasks (described below), one with each version of the interface. Therefore, Task was again a within-subjects control variable. Both Version Order and Task Order were between-subject controls. To account for carry-over effects we counterbalanced the order of version and task, resulting in four configurations.

¹⁷Based on our previous study and discussions with others who had administered the NAART, we were worried that ineligible participants would be hurt if they were told that the English-language fluency was too low. Given the sensitive nature of this issue, we instead told participants who did not meet our NAART cut-off criterion that their MSWord expertise fell into a category for which we already had too many participants.

6.3.3 Apparatus

The apparatus was the same as in Study One (see sec. 5.2.3).

6.3.4 Tasks

The tasks used in Study Two were based largely on those used in Study One, with two main exceptions described below. Using methodology and task design similar to Study One, interacting with the rationale is not an explicit experimental task. Instead, the majority of the session is spent performing pre-assigned word-processing tasks with the target application, MSWord. Alternatively, we could have required users to interact with the rationale for a period of time – for example, by having them complete a worksheet or questionnaire based on information in the rationale (e.g., [35] and [127]). We chose to build on our previous methodology, as opposed to designing tasks specific to the rationale, because we felt that it would generate more realistic feedback about when and why users might access the system’s rationale.

The first difference between the tasks used in Study One and Study Two is that, although we did not find any main effects of Task in Study One (see sec. 5.2.4), we attempted to make the tasks in Study Two more similar in terms of their potential effects on customization behaviour. In particular, we designed the tasks to have the same number of needed features missing from the starting PIs (in Study One there were 7 vs. 5) and the same number of extra features (in Study One there were 34 vs. 40). Second, we made the tasks slightly longer. We did so to increase motivation to customize and to generate more variability in the recommendations. A key component to generating variability in MICA’s recommendation is having the User Model believe that some features will be used much more frequently than others, which can cause the CSM to recommend that some needed features not be included in the PI (see sec. 4.3.1 for a description of the algorithm). If all needed features are used approximately the same number of times, the CSM will most often recommend that all needed features be included in the PI. To create more usage disparity, we had to increase the length of the tasks. The nature of the recommendations, however, would still depend on each individual participant’s expertise level for each feature in the experiment. Task details are summarized in table 6.2; the task instructions can be found in appendix K.

As in Study One, tasks were repeated three times and participants were told that the tasks would be repeated up to five times. The customization mechanism

Dimension	Task A	Task B
Needed features not in the starting PI	7	7
Features in the starting PI	39	39
Different features required	21	21
Feature invocations total	63	53
Task steps on instruction sheet	60	64

Table 6.2: A comparison of Task A and Task B along five dimensions.

was again enabled after the first repetition to allow participants to get a sense of their tasks before customizing. Unlike in Study One, however, if at the end of the second task repetition the participant had yet to customize, the experimenter asked her do so at a point of her own choosing during the third repetition. We did so in the hope of achieving a higher customization rate across both conditions than in our previous experiment, which was 67% without any prompting. All participants who customized in Study One did so prior to the second repetition of a given task. Waiting until the third repetition to prompt would, in theory, allow us to perform separate analysis on the participants who were prompted versus those who customized on their own initiative. Separate analysis would be necessary if it appeared as though those who were prompted behaved differently in the customization mechanism or reacted differently towards the rationale.

6.3.5 Procedure

The procedure for Study Two was very similar to that of Study One, with two exceptions that we discuss in this section. For the remainder of the study procedure please refer to section 5.2.5. The first difference concerned the introduction of the version (Rationale vs. No-Rationale) present in the second condition (step 5 in sec. 5.2.5). In this experiment, participants received a demonstration of this new version, regardless of which condition they completed first. The new version was opened and the differences were pointed out (i.e., the presence or absence of the rationale). The second difference involved motivating rationale usage. Our goal was to give participants as much autonomy as possible with respect to rationale usage; however, we did want participants to look at it. To balance these two objectives, we showed participants where to access the rationale during the initial interface demonstration of the Rationale version (during step 3 or step 5 in the procedure originally outlined in sec. 5.2.5) and requested that the participants “look through the information at some point.” Apart from this request, no prompting to look at the rationale was done during the experiment.

A session typically lasted 3 hours, but ranged from 3 to 4 hours. The sessions were longer in comparison to Study One, where they lasted 2 to 3 hours, because of longer tasks and a more in-depth interview.

6.4 Pilots

Prior to running the full experiment, the above protocol was pilot tested with three participants. Since the study protocol was so similar to that of Study One, we did not anticipate making any major changes. The pilot participants were used mainly to ensure that participants did not respond negatively to being prompted to customize after the second condition (which they did not). We were also interested in seeing whether any participants would choose to view the rationale, and found that one of the three participants did. While we were somewhat concerned with this low number, we decided to continue. We believed that it was likely that greater than 33% of participants would choose to view the rationale in Study Two, since one of the two pilot participants who did not view the rationale appeared unmotivated to complete the experiment.

6.5 Measures

6.5.1 Qualitative Measures

As we mentioned at the beginning of this chapter, our emphasis in Study Two was on qualitative measures. The post-treatment questionnaire gathered preference information. In particular, participants who viewed the rationale during the study were asked which version of the system they would choose to install (Overall Preference). The post questionnaire also asked participants to state which version they preferred, or whether they found the two equal, for the following five criteria: (1) agreeing with the system recommendations (Agreement), (2) trusting the system to make good recommendations (Trust), (3) understanding why the system was making specific recommendations (Specific Understanding), (4) understanding why the system was making recommendations in general (General Understanding), and (5) ability to predict future recommendations (Predictability). The questionnaire also asked participants to explain how they thought the recommendations were generated.

The interview gathered more detailed qualitative data on such topics as: (1) the influence of the study methodology on rationale viewing, (2) additional

reasons for viewing (or not viewing) the rationale, (3) the impact of the “Why” component on motivation to accept recommendations, (4) why participants did or did not access the “How” component and their impressions of its utility, and (5) whether participants would have liked any additional information from the rationale.

6.5.2 Quantitative Measures

Despite the focus on qualitative outcomes, we also analyzed a few quantitative measures that could be affected by the presence of the rationale. We measured the time spent viewing the rationale and included a number of measures pertaining to both conditions, which are listed below, grouped according to category. Our primary quantitative measures of interest are listed under the first category (“Percentage of Recommendations Followed and Methods used to Follow Recommendations”). For completeness, we also measured the impact of the rationale on our main quantitative measures from Study One (listed under the categories “Performance” and “Customization Behaviour”).

Percentage of Recommendations Followed and Methods Used to Follow Recommendations

We measured the percentage of recommendations followed (% Add/Delete Recommendations Followed). We also measured how participants used the available interface tools to follow system recommendations. In the interview, we also asked participants to explain their reasons for choosing particular methods.

Performance

We measured the impact of Version on Study One’s performance measures: Overall Performance and Task Performance (see sec. 5.4).

Customization Behaviour

We measured the impact of Version on Study One’s customization-related dependent measures: Customization Time, Customization Sessions, and Features Added/Deleted (see sec. 5.4).

6.5.3 Measures Pertaining to Framework Assumptions

In addition to the above quantitative and qualitative measures concerning the rationale, in the interview we asked questions designed to understand participants' general willingness to delete features and to switch between the two interfaces as opposed to just using the PI (or the FI).

6.6 Results

Similarly to Study One, 69% of the participants (11/16) customized in both conditions without any prompting. Once prompted, the remaining five participants (S4, S8, S12, S13, and S16) customized. Since separate analysis of those who were prompted versus those who were not failed to reveal any substantial differences, the remainder of the analysis includes data from all participants.

In the Rationale condition, 94% (15/16) of the participants accessed the rationale. Of these participants, 47% (7/15) accessed the "Why" component only, with an average viewing time of 15.1 seconds (SD: 9.6 seconds). The remaining 53% (8/15) accessed all of the rationale, with an average viewing time of 63.4 seconds (SD: 30.4 seconds).

We begin by describing the results pertaining to the qualitative dependent measures. Section 6.6.2 describes the analysis of the quantitative data and section 6.6.3 presents results pertaining to the validity of MICA's framework. To analyze the qualitative data, the interviews were first transcribed. Next, detailed coding was done by the thesis author, based on thorough analysis of the interviews and questionnaires. We report themes and trends that emerged from this analysis, along with the number of participants whose statements matched the given theme or trend. Our intention was not to prove or disprove hypotheses through statistical analysis, which, given the anticipated diversity of opinions, would have required a much larger number of participants.

6.6.1 Impact of Rationale on Qualitative Measures

Preference

Figure 6.7 depicts the preference data both overall and for each of the individual criteria, and indicates that, in general, the preference data was mixed. When forced to choose, the majority of participants (60%, 9/15) indicated that they would prefer to install the Rationale version, but the No-Rationale version also

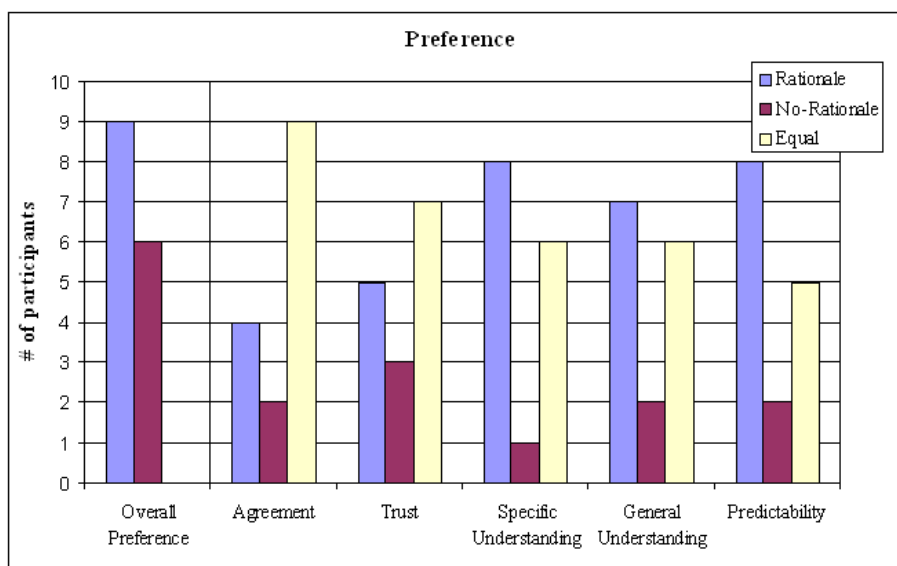


Figure 6.7: The Version preference data. For Overall Preference, participants were forced to choose. For the individual criteria, they were given the option to rate the two “Equal.” (N=15)

had reasonable support (40%, 6/15). For the individual criteria, participants were given the option of rating the two conditions “Equal.” Having the rationale appeared to have the largest impact on both Specific and General Understanding, as well as Predictability of the recommendations. While the Rationale version was preferred by some users for Agreement and Trust, the most popular response for these criteria was “Equal.”

At the end of the interview, participants were asked to explain in one or two sentences why they picked the version they did for each of the above individual criteria. Many comments echoed those expressed earlier in the interview, which we present in the upcoming sections. Some additional points that were brought forward are as follows: First, for all criteria, a number of participants commented that they rated the two versions “Equal” based on the fact that the recommendations were the same quality in both conditions or made by the same underlying system. Second, for the Specific/General Understanding and Predictability, a number of participants commented that as they gained more experience using the system they were better able to understand and predict its recommendations, independent of the rationale. Third, two of the three

users who chose No-Rationale for Trust, mentioned that condition order influenced their selection (both participants saw the No-Rationale version second). These participants trusted the system more in the second condition since at this point they had seen two sets of recommendations as opposed to just one. Thus, they trusted the system more in the No-Rationale condition, not because of the absence of rationale information, but because of their increased belief in the system's effectiveness. Finally, for Predictability, two users commented that this was not a criterion they cared about.

Influence of Study Methodology on Rationale Viewing

To understand whether users looked at the rationale solely because of the request during interface demonstration, users were asked why they looked at the rationale and to comment on the role that the request played in their decision. Out of the 15 users who viewed the rationale, 33% (5/15) said they were not influenced by the request. Another 20% (3/15) indicated that they were partially influenced by the request, but had additional reasons for accessing the rationale. The remaining 47% (7/15) said the request during interface demonstration was their sole reason for accessing the rationale. Just over half of these users (4/7) said that there would be circumstances where they would want the information, but that our particular study methodology did not provide the right motivating conditions. Finally, three users indicated that they had no interest in the rationale. Thus, 80% (12/15) of the participants either viewed the rationale for reasons other than our particular study methodology or could see circumstances outside of the study where they would want to view the rationale.

All participants who looked at the rationale on their own initiative preferred the Rationale version overall. For those who were only partially influenced by the request, two out of three preferred the Rationale version overall. For those participants who were solely influenced by the request, the majority (5/7) preferred the No-Rationale version. This suggests that there might be a positive correlation between desire to access the rationale and overall perception.

Additional Reasons for Viewing or Not Viewing the Rationale

Table 6.3 summarizes why participants did or did not view the rationale. Three reasons were given for viewing the rationale by the 53% (8/15) that accessed it for reasons other than the request during interface demonstration. The first was general curiosity (3/8). The second was to have the recommendations

Reasons for Viewing	Reasons for Not Viewing
<ul style="list-style-type: none"> • general curiosity (3/8) • to have recommendations explained (3/8) • to have interface explained (2/8) 	<ul style="list-style-type: none"> • mixed-initiative interaction (1/3) • embedded in a productivity application (1/3) • inherent trust in system (1/3)

Table 6.3: Reasons (other than the request) for viewing the rationale (8 participants) or not viewing the rationale (3 participants).

explained (3/8), e.g., “if something is customizing it for you...I want to have an understanding of why it is doing things” (S4). The third reason was to have an aspect of the interface explained (2/8), e.g., “I wasn’t sure about how the Personal Interface worked” (S5).

The three users who were not interested in the rationale gave unique reasons for why not. One felt that the rationale is unnecessary in a mixed-initiative system, since she could follow the recommendations if she found them useful or customize on her own if she did not. Another pointed to the fact that the rationale is embedded within a productivity application: “...when it comes to a program like Microsoft Word most of the time you only care about getting the job done. You don’t really care about why” (S11). The final participant expressed inherent trust in the system: “I just assume recommendations are because they are useful for you. That’s all really I need to know” (S14).

Effectiveness of Rationale: Impact of “Why” on Recommendation Acceptance

Of those who accessed the rationale, 93% (14/15) indicated that they actually read the “Why” component. Since its purpose is to illustrate the potential time savings that could result from accepting recommendations, we asked users to discuss whether or not this information was, in fact, motivating. Responses are illustrated in figure 6.8. Forty-three percent (6/14) of these users felt that the “Why” component motivated them to accept recommendations. Another 43% (6/14) were generally interested in having a PI that would save time, but were not motivated by the particular amount of time savings listed (labeled “Generally ‘Yes’ / Specifically ‘No’ ” in fig. 6.8). When asked if they could suggest a number that would be more motivating, one participant (S2) provided an estimate of 10 seconds or more. Another participant (S11) said that given the time necessary to read the “Why” component, the savings would have to be

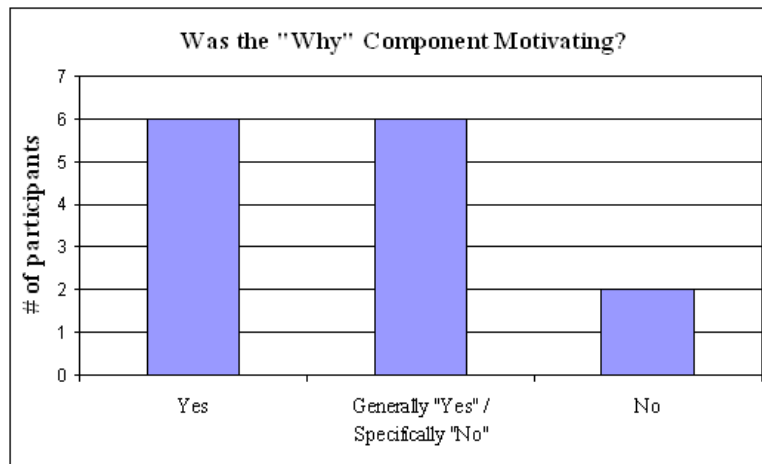


Figure 6.8: Whether or not participants who read the “Why” component of the rationale felt motivated to accept recommendations. (N=14)

on the order of minutes, not seconds. Finally, participant S15 felt the problem was more the expression of the time savings, which he found to be unintuitive: “I couldn’t relate it to the real world. It was like saying how fast you are driving in meters per second... .” The other participants couldn’t think of a number that would be more motivating. The remaining two users who read the “Why” component while customizing (14%) indicated that they were not motivated by time savings in general. One user (S6) felt that she could achieve the same results without system support, while the other (S14) said that she preferred the FI and that because of her high expertise, the additional features did not slow her down.

In this study, three users did delete features and did so after having viewed the rationale. Two of the three users indicated that the time savings was a motivating factor.

Effectiveness of Rationale: Usefulness of the “How” Information

Only 47% (7/15) of those who accessed the rationale indicated that they read the “How” component. The majority did so because of general curiosity (4/7). Two participants decided to read everything once they entered the rationale. The final participant wanted to confirm her hypothesis: “Because my first thought is that it is probably based on statistics of some sort and I wanted to see if it is true” (S1). The following reasons were given for not accessing or reading the

“How” Useful (10/16)	“How” Not Useful (6/16)
<ul style="list-style-type: none"> • Gained a better understanding (or confirmed) (5/10) • Recommendations more trustworthy or believable (3/10) • Simple explanation (1/10) • Could use knowledge to become more efficient (1/10) 	<ul style="list-style-type: none"> • Unnecessary or common sense (4/6) • Too technical (1/6) • Did not influence customization decisions (1/6)

Table 6.4: Reasons for finding the “How” component useful or not useful.

“How” component: (i) not liking the “Why” component (2/8), (ii) trusting the system to do things logically (1/8), or (iii) just wanting to complete the task (2/8). The remaining participants couldn’t recall why they did not read the information (2/8) or did not notice the additional information (1/8).

To obtain as much feedback as possible on the “How” component, during the interview we asked all 16 users to read through the information and comment on its usefulness. After reading the information 62% (10/16) found the information useful, including six of the seven users who read the information while customizing (as opposed to during the interview only); 38% did not find the information useful (6/16). Table 6.4 summarizes their reasons. The most popular reason for finding the information useful was gaining a better understanding of how the system makes recommendations or confirming their existing understanding. For those who did not find the information useful, the majority indicated that it was unnecessary or “just common sense.”

We also asked users to indicate which pieces of information, if any, were most or least useful. Figure 6.9 displays the results. While participants responded favourably to the Expertise and Usage Frequencies factors, 50% (8/16) found the Interface Size factor of little value. Many commented correctly that this factor wasn’t as personalized, or that having a small interface was the point of customization. Participants did not seem to understand that MICA balances this factor with both usage frequencies and expertise. This result is consistent with feedback from informal evaluation, which indicated that users respond most favourably to information that is personalized.

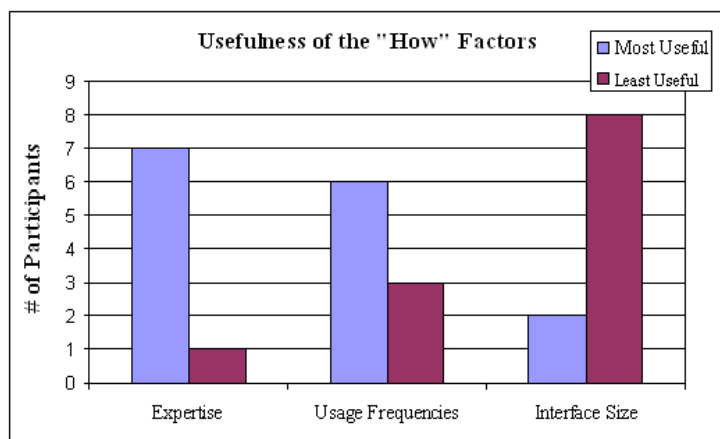


Figure 6.9: The usefulness of the individual “How” factors. (N=16)

Additional Information

Contrary to some opinions expressed during the informal evaluation described at the beginning of the chapter, only one participant (S2) requested specific user model assessments (for expertise). This user also wanted to know how the expertise was quantified, but only if the information was tucked away somewhere. A second participant (S15) wanted to know more about how the factors were balanced. Given that only two of the sixteen participants requested additional information, it appears that the quantity and type information presented within the rationale was generally sufficient.

Impact of Rationale on Understanding How the System Works

In the post-questionnaire, we asked participants to explain how the system made its recommendations, with the intention of correlating participants’ answers with rationale viewing. Our interview revealed that seven participants had read the “How” component of the rationale when they completed this portion of the questionnaire, while nine had not.

First, we discuss the responses of the seven participants who read the “How” component prior to completing the questionnaire. None of these participants indicated that the system considers the size or characteristics of the interface, and only two participants explicitly mentioned both usage frequencies and the time necessary to access features (i.e., expertise). The remaining answers either mentioned only usage frequencies or simply referred to usage in some way, e.g.,

“It looks at what you use” (S4). One exception was the response of participant S1, who thought the recommendations were based on a combination of data from the general population and her personalized usage: “General: what people generally use most often, i.e., open, save, build, underline, bulletin, print, etc. Specific: based on stats of my different function usage” (S1). This could be explained by the fact that the rationale indicates that the system considers which features the user is “likely to use” (see fig. 6.5 (b)), which could be interpreted as the system considering population data.

The responses of those who had not yet read the “How” component when they completed this portion of the questionnaire referred only to usage frequencies or usage in general. None of these participants mentioned either expertise or interface characteristics. Sample responses included:

- “I think the system records the features that is [*sic*] used when the ‘full interface’ is used, and would recommend them if it [*sic*] is [not] already part of ‘my interface’ ” (S7).
- “based on how frequently the functions are used” (S8).

Apart from the fact that the only two participants who mentioned expertise viewed the rationale before completing the questionnaire, the questionnaire did not reveal a large difference between those who read the rationale and those who did not in terms of their understanding how the system functions. Unfortunately, all participants’ answers were quite short and many were vague. Since we did not follow-up on their responses in the interview, it is difficult to assess whether they did or did not understand how the system works or simply did not know how detailed a response to give on the questionnaire. More in-depth probing, likely in an interview setting, would be required to get an accurate picture of the impact of viewing the rationale on understanding how the system functions.

Summary of Key Qualitative Results

Before describing how the rationale impacted our quantitative measures, we first summarize the qualitative results from this section. Our main results are as follows:

- **Preference:** The majority of our participants (60%) preferred the version with the rationale present, but a sizeable group preferred the version without the rationale (40%). On the individual criteria, the Rationale version

had the largest impact on system understandability and predictability. Two participants, however, indicated that predictability was not important to them.

- **Viewing the Rationale:** While a handful of participants were motivated to view the rationale solely because of the experimenter’s request at the beginning of the experiment, the majority of users (80%) had their own reasons for wanting to see the rationale either during this particular study or in general. Reasons for not wanting to view the rationale included the mixed-initiative nature of the system, the fact that it is embedded within a productivity application, and inherent trust in the system.
- **Motivation of the “Why” Component:** Most participants (86%) indicated that they were generally motivated to customize to save time. For half of these users, the time savings information expressed within the rationale made them feel motivated to accept recommendations, while the other half felt that the time savings should either be greater or be expressed differently. In addition, two of the three users who deleted features were motivated by the time savings expressed within the rationale.
- **Usefulness of the “How” Component:** The explanation of how the system makes recommendations was found to be “useful” by 62% of the participants, for reasons such as increased understandability and increased trust in the system. The other 38% of the participants generally felt that the information was either unnecessary or common sense. Participants liked the parts of the explanation that they perceived to be the most personalized and generally found the amount of information to be sufficient.
- **Impact of Rationale on Actual Understanding:** The post-treatment questionnaire indicated that rationale viewing led to a small amount of actual increase in understanding (as opposed to perceived increase in understanding). However, we found that our questionnaire did not obtain enough fine-grained information on this matter to support a strong inference.

6.6.2 Impact of Rationale on Quantitative Measures

The quantitative data were analyzed using repeated-measures ANOVA with Version (Rationale or No-Rationale) as the within-subjects factor. Version Order

Dependent Variable	Mean		SD		F(1,11)	p	η^2
	R	N-R	R	N-R			
Add Recs Followed (%)	94.2	93.5	9.0	21.6	0.001	0.982	0.000
Delete Recs Followed (%)	14.2	7.2	34.3	24.8	0.978	0.334	0.082

Table 6.5: A summary of the ANOVA results with Version as the primary within-subjects factor for the percentage of Add and Delete recommendations followed. (N = 15) R=Rationale, N-R = No-Rationale

and Task Order were included as between-subjects controls. Before analyzing the effects of Version (Rationale vs. No-Rationale), we checked for an effect of Task (A vs B). Contrary to Study One, this time we did find significant main effects of Task on two of our dependent measures: Overall Performance and Task Performance, which we discuss briefly later in this section (and in greater detail in appendix N). There were no effects of Task on the remaining quantitative measures. We again report effect sizes.

Percentage Recommendations Followed

As anticipated, table 6.5 shows that the percentage of Add recommendations followed was similar in both conditions, with participants following 94.2% of the Add recommendations in the Rationale condition, compared to 93.5% in the No-Rationale condition ($F(1, 11) = 0.001$, $p = 0.982$, partial $\eta^2 = 0.000$). In terms of Delete recommendations, three users did delete, leading to an average of 14.2% Delete recommendations followed in the Rationale condition compared to 7.2% in the No-Rationale condition, a difference which was also not statistically significant ($F(1, 11) = 0.978$, $p = 0.334$, partial $\eta^2 = 0.082$).

There was a marginally significant between-subjects order effect for the percentage of Add recommendations followed ($F(1,11) = 3.990$, $p = 0.071$, partial $\eta^2 = 0.266$). Participants who completed the Rationale condition first followed more Add recommendations overall (average: 99.1%, SD: 2.5%) than those who completed the Rationale condition second (average: 87.9%, SD: 14.3%). This order effect was anticipated; we expected knowledge that the system would make principled recommendations in the first condition to transfer to the second. This result suggests that the rationale might be most effective when viewed earlier rather than later and that frequent viewing isn't necessary.

Because the above order effect was anticipated, we performed some between-

subjects analysis on the version of the system (Rationale vs. No-Rationale) that participants interacted with first. Participants who interacted with the Rationale version first added an average of 98.2% of the recommendations in that condition (SD: 5.1%), while those who interacted with the No-Rationale version first added an average of 87.8% recommendations (SD: 29.2%). This difference, however, was not significant according to a one-tailed t-test ($t(14) = -0.995$, $p = 0.169$, Cohen's $d = 0.50$).¹⁸ This same between-subjects analysis indicated that participants also followed more Delete recommendations in the Rationale version (average: 14.4%, SD: 35.0%) than in the No-Rationale version (average: 0%, SD 0%), but the difference also was not significant ($t(14) = -1.166$, $p = 0.141$, Cohen's $d = 0.58$).¹⁹ These trends suggest that viewing the rationale might lead to slightly higher acceptance of system recommendations, but more data is needed before drawing conclusions.

Methods Used to Follow Recommendations

As in Study One, we looked at which methods participants used to follow the recommendations. We focus on Add recommendations because only three participants deleted features. Participants could follow recommendations using any combination of four methods. Three of the four methods were also available in Study One: Self Selection, Accept All, and Get Recommendations (see sec. 5.5.4). In section 6.2.2, we discussed a fourth method, added prior to Study Two, that allows users to select from a list of recommendations accessible through the “Show Add Recommendations” button in the Add customization screen (see figs. 6.1 and 6.2). We refer to this method as Show Add. Table 6.6 shows how participants in Study Two used the four methods. Participants used a variety of methods, with the most popular method again being Self Selection. The newly added method, Show Add, was also heavily used.

In the interview, participants were asked why they chose to customize in the manner that they did. Self Selection was perceived to be the primary customization method of 31% of the participants (5/16). Their reasons for relying primarily on this method differed. Some participants said that they wanted to be in control of the customization process (2/5) while others felt that it was important to see the placement of their selected additions within the menus and toolbars (3/5). For the 44% (7/16) who indicated that they used a combination

¹⁸Cohen's d [28] is a measure of effect size appropriate for t-tests. To interpret this value, 0.8 is considered to be a large effect size, 0.5 a medium effect size, and 0.2 a small effect size.

¹⁹The test for equal variances failed, and so equal variances are not assumed.

	Self Selection	Show Add	Accept All	Get Recs
% of recommendations added	44%	37%	11%	8%
# participants who used method	13	8	3	3
# participants who used method exclusively	4	2	1	0

Table 6.6: Summary of the methods used to follow the recommendations. (N=16)

of Self Selection and Show Add (or Get Recommendations), four participants indicated that they used one method to supplement the features selected from the other method. Two of the seven participants indicated that they started off using the Self Selection method and switched to the Show Add method once they started to trust the system recommendations, and one participant wanted to experiment with both methods. Two participants (13%, 2/16) indicated using Accept All as their primary method because it made customization faster. The final 13% (2/16) had difficulty articulating their customization strategy. These participants seemed not to understand what was possible within the customization interface. Interestingly, both these participants were prompted to customize, which might have influenced their willingness to read the instructions on the customization interface.

While none of the participants mentioned the rationale influencing their choice of method, one could imagine that seeing the reasoning behind the system’s recommendations could increase a user’s willingness to use the Accept All method. On average participants did rely on this method slightly more frequently in the Rationale version (average: 12.4%, SD: 29.0%) than in the No-Rationale version (8.6%, SD: 26.3%). This difference, however, was not significant ($F(1, 11) = 0.846$, $p = 0.377$, partial $\eta^2 = 0.071$).

Performance

As mentioned at the beginning of the section 6.6.2, Task did impact our two performance measures. Specifically, Task B appeared to be more difficult – it took participants longer to complete, despite having fewer total feature invocations (see table 6.2). For the sake of clarity and brevity, the analysis of the effects of Task on performance can be found in appendix N.

An overall summary of the effect of Version on both performance measures can be found in table 6.7, which shows that Version did not significantly impact

Dependent Variable	Mean		SD		F(1,11)	p	η^2
	R	N-R	R	N-R			
Overall Performance (minutes)	42:29	40:32	12:18	10:50	0.633	0.443	0.054
Task Performance (minutes)	39:48	38:53	11:57	10:56	0.083	0.779	0.007
Customization Time (minutes)	2:24	1:32	0:59	0:53	7.445	0.020	0.404
Customization Sessions	2.0	1.8	1.1	1.5	0.169	0.689	0.015
Features Added	9.5	8.7	3.6	2.7	0.443	0.519	0.039
Features Deleted	3.5	1.8	8.6	6.2	0.990	0.341	0.083

Table 6.7: A summary of the ANOVA results with Version as the primary within-subjects factor for measures pertaining to performance and customization behaviour. (N = 15) R=Rationale, N-R = No-Rationale

performance. These results need to be treated somewhat cautiously given the effect of Task discovered. There were also a number of significant or marginally significant interactions and between-subjects effects (see appendix N for the relevant summary tables). Given that the tasks were not isomorphic for these dependent measures, we omit a lengthy analysis and discussion of these results. Further evaluation will be needed to understand the impact (if any) of rationale viewing on performance in terms of time on task.

Customization Behaviour

Table 6.7 also summarizes the effects of Version on the remainder of the measures pertaining to customizing behaviour. Not surprisingly, there was a significant main effect of Version on Customization Time, with participants spending more time customizing in the Rationale version than the No-Rationale version ($F(1, 11) = 7.445$, $p = 0.020$, partial $\eta^2 = 0.404$). This is not a strong concern, since we do not expect that users will look at the rationale every time they customize. The rationale did not impact the remainder of the customization-related dependent measures.

Summary of Key Quantitative Results

Before describing the results pertaining to assumptions within MICA's framework, we first summarize the key results from the quantitative data:

- **Recommendations Followed:** The rationale did not significantly impact the percentage of recommendations that participants followed. Participants who viewed the rationale in the first condition, however, tended to accept more recommendations overall.
- **Methods Used to Follow Recommendations:** The rationale did not cause participants to begin delegating more responsibility to the system (i.e., through increased use of the Accept All method). Our newly added customization method “Show All” was widely used in both conditions.
- **Performance:** The rationale did not impact performance, but these results are complicated by task differences.
- **Customization Behaviour:** Participants spent more time customizing in the Rationale condition, but the rationale did not impact the total number of features customized or the number of customization sessions.

6.6.3 Further Exploration of Framework Assumptions

In addition to understanding the utility of the rationale, we also used Study Two to further explore the validity of two of MICA’s key assumptions. First, MICA assumes that users will want to delete features and second, MICA assumes that users will be willing to switch between interfaces for less frequently used features.

Willingness to Delete

While the rationale appeared to motivate at least two users to delete, the majority of participants still did not do so. A number of participants commented that they were aware of extra features in PI (38%, 5/13), but did not delete them because (a) they thought they might be useful at some point or (b) they were used to having these features in the interface. One participant commented that he felt more comfortable with the interface looking more similar to what he was used to. Three participants (23%) indicated that they did not feel the need to delete because they felt that the PI was sufficiently small and one participant did not want to make the effort. There were, however, participants who seemed unaware of the extra features (23%, 3/13), e.g., “I guess I wasn’t noticing what I didn’t need and it is more like I know what I needed so I just went for it” (S7).

Of those participants who did not delete during the study, 38% (5/13) felt that they would do so in the future. In particular, one participant explicitly

mentioned that he might employ a switching strategy, saying that he'd like to keep his interface minimal and could always switch to the FI to use a deleted feature. Some participants (24%, 3/13) said that they could not see themselves ever wanting to delete. Two of these participants were afraid of deleting features that they might need eventually. The third user felt that the PI would continue to be small enough that nothing would get in her way. The remaining 38% (5/13) were somewhere in the middle, feeling that they might delete, but only if features really started getting in their way or the PI grew to become very large.

While it might seem that general willingness to delete could align well with Feature Keen/Shy classification, we did not find evidence that this was the case with our sample and the current version of the classification questionnaire. For example, the group that said they could not ever see themselves deleting consisted of one Keen, one Neutral and one Shy. We also looked for noticeable differences in expertise between those users who felt the PI was small enough already or weren't bothered by the extra features, and the remainder of the participants. Based on users' self-assessments, there was no obvious pattern; however, an experiment controlling for this factor could be an interesting avenue for future work.

Willingness to Use a Combination of the Two Interfaces

During Study One, a couple of participants commented, in an open-ended section of the post-treatment questionnaire, that features were "missing" from MICA's recommendation additions. Therefore, in Study Two, we explicitly asked if participants found this to be the case. Participants were then asked a series of questions designed to understand whether they would, in general, be willing to switch back and forth between the PI and the FI.

Just over half of the participants (56%, 9/16) did feel that there were features "missing" from MICA's recommendations. In only two of these cases, however, did MICA deliberately make the decision to omit a less-frequently used feature from its recommendations. The remaining cases might have been caused by noise in the User Model's assessment of expected usages. As we mentioned in section 5.6.2, not all participants followed the task instructions perfectly and some found alternative means of accomplishing certain steps. Thus, more detailed analysis would be necessary to determine whether MICA did "miss" these features because of inaccurate User Model settings.

In terms of general willingness to switch between interfaces, one quarter of

the participants (25%, 4/16) said that they would prefer to use one interface exclusively (in all cases the preferred interface was actually the FI), even if it made them faster overall to use a combination of the two. One participant commented that she did not like using both interfaces because absolute positional consistency is not maintained: “because when you switch the buttons move right, they sort of shift somewhere else I just sort of memorize where the spots are. I don’t really look at the pictures sometimes” (S14). The remaining three quarters (75%, 12/16) seemed open to the idea of using a combination of the two interfaces.

6.7 Discussion

6.7.1 Rationale Impact and Effectiveness

Our findings indicate that the majority of users preferred to have the rationale present, but that a non-insignificant group of users did not need or want the information. For some users, viewing the rationale led to increased trust, understanding, predictability, and motivation to accept recommendations. Some users, however, felt that the rationale was just common sense, or was unnecessary in a mixed-initiative system or productivity application. Others expressed an inherent trust in the system. These findings suggest that, contrary to previously stated guidelines [58], transparency and predictability might not, in fact, be important to all users in all contexts. Indeed, two participants explicitly mentioned that predictability was not something they cared about. On the other hand, since some users found the rationale to be just common sense, it might be that our particular design did not always succeed in improving transparency and predictability.

In terms of rationale design, feedback from our iterative design and evaluation process suggests that the personalized aspects of the rationale should be emphasized when possible. In addition, since reactions to the rationale were mixed, the information should be clearly visible for those who want it without disrupting those who don’t, which was the approach taken here. One user did comment, however, that he would have preferred the information to be even more hidden than it was. While the majority of the users found the information provided within the rationale to be sufficiently detailed, two users did request more detail. Therefore, future work could involve providing ways for interested users to access this information, without having it displayed in the main ra-

tionale interface. Finally, we might see different reactions to more lightweight graphical representations of the rationale. There could be ways to visualize the information so that users do not have to spend as much time reading text.

The rationale was a motivating factor for two of the three users who deleted features from their PIs, and participants who viewed the information in the first condition tended to accept more Add recommendations overall. In addition to the small quantitative impact the rationale had on deletion, it seems to have improved general attitudes towards deletion, in comparison to what was reported in McGrenere's field study [91]. Otherwise, the rationale had limited quantitative impact, mostly because users tended to follow the system's suggested additions even without looking at the rationale. Understanding whether the rationale could have a larger quantitative impact might require finding a target application where users are less inclined to accept recommendations without the rationale, for reasons such as recommendations being contrary to expectations or a higher cost associated with accepting recommendations. In applications that involve recommender or expert systems, accepting a system recommendation can sometimes lead to a financial investment or a high-stakes decision, whereas in our case it is the user's performance with the interface that is impacted. Alternatively, it might be the laboratory environment that led to such high acceptance of recommendations. The rationale might have a larger impact in the field, where users might have lower levels of trust in the system.

6.7.2 Study Methodology

In this section, we reflect on two key design decisions made in this study. The first was not to have explicit experimental tasks targeting rationale usage, but rather to have users performing tasks with the target application, leaving it up to them to decide when to customize and when to view the rationale. Until the post-session interview, the rationale was a secondary focus (at best), with users spending the vast majority of their time performing tasks with MSWord. This decision was made in an attempt to gain realistic insight into the effectiveness of the rationale (to the extent possible within a laboratory setting). We also wanted participants to have the opportunity to interact with tailored recommendations when viewing the rationale. The disadvantage is that our study sessions were long, and similar to Study One, only a small percentage of the time was devoted to behaviour that was of interest to our primary study objectives. Alternatively, we could have used a format similar to our informal

evaluation, which lasted 30 minutes per session as opposed to 3-4 hours. Using this format, we would ask participants to explore the customization mechanism and rationale, imagining that the recommendations were tailored to them, after which we would conduct a focused interview. It would be interesting to investigate to what extent the different study designs would impact the quality and type of the resulting qualitative data.

The second key design decision was to prompt participants to customize, if they hadn't already done so, after the second task repetition. This decision provided us with a larger quantity of customization and rationale usage data than we obtained in Study One, and separate analysis of those who were prompted versus those who weren't revealed no substantial differences. The main exception was that two of the participants who were prompted had difficulty understanding the range of options in the customization mechanism. This could simply be a coincidence, but it could also be an indication that participants who are prompted might be less willing to explore and learn the customization mechanism.

Finally, this experiment provided further evidence of the difficulties of designing isomorphic tasks in a within-subjects experiment using an application whose interface cannot be modified between tasks to achieve isomorphism (e.g., changing the names of menu items) and for which participants have pre-existing expertise. In many respects our tasks were more similar than in Study One, and so we might have been less fortunate with the manner in which participants' expertise was distributed amongst the features used in the experiment. In addition, it appeared as though some participants in Study Two were less familiar with "table" terminology, which was used in Task B (see appendix 6.3.4), and so spent more time trying to figure out the instructions as a result. The differences did not have a large impact on our study since our focus was primarily on qualitative measures and only two of our quantitative measures were affected. These challenges should be considered in future study design.

6.7.3 Validity of MICA's Assumptions

There is evidence that many users are willing to delete features, but also that adding and deleting are not necessarily symmetric operations in terms of awareness and subjective impressions of the operations. In terms of awareness, knowing what to delete requires users to make an effort to notice which features are in the PI but are not being used in the process of performing their tasks. Noticing

which features should be added is less difficult, since the awareness of needed features increases directly through usage. Users also appeared to be worried, despite continual and relatively easy access to the FI and to the customization facility, that deleting would turn out to be a mistake at some point down the road. The perceived difference in the risk associated with deleting features as compared to adding them might relate to findings from decision-theory research (see [26] for an overview). Studies have revealed similar asymmetries, where users are more risk averse when dealing with potential losses as compared to potential gains (e.g., [119]). Finally, in this study one participant also expressed a desire to keep the interface looking familiar.

Promoting deletion might require MICA to engage in more proactive support. For example, MICA could alert users to rarely used features, perhaps by highlighting them occasionally when the interface is first started. There could also be a facility within the rationale that provides users with the expected time savings that could result solely from deletion, to help users decide whether they feel that deletion is worth the risk.

The interview revealed that many users are open to employing a switching strategy for their rarely used features. Given that the general willingness is there, the next step would be to obtain more concrete evidence that users will, in fact, switch between interfaces given the right motivation (i.e., time savings) and that this strategy can be beneficial. A single-session laboratory study is likely not the right context to investigate this issue. Given that the savings resulting from having one or two rarely used feature reside solely in the FI is relatively small, particularly while one is still adjusting to the two-interface model, the desire to use just one interface for the duration of the study is understandable.

6.8 Summary

This chapter described the iterative design and the formal evaluation of MICA's rationale. Qualitative reactions to having this information varied across individuals. Many users preferred to have access to the information, commenting that it increased their understanding of the system, increased their trust in the system, and made the recommendations more predictable. In addition, the rationale motivated two users to delete features, which is an important component of maintaining a truly customized interface. Other users, however, did not find

the rationale necessary within this type of application or mixed-initiative interaction, and overall, the quantitative impact of the rationale was limited. While the evaluation revealed aspects of our rationale that could be improved, the most promising avenues of future research on the design of rationale might be to gain a more global understanding of when and why it can be useful. We present ideas for future evaluations in chapter 7, which summarizes the thesis contributions and discusses directions for future research.

Chapter 7

Conclusions

The goal of this thesis was to investigate a mixed-initiative approach to interface customization as a means of combining the positive aspects of both the adaptive and adaptable alternatives. In this chapter, we begin by summarizing the steps we took to satisfy this goal. We then describe the contributions of this thesis, which elaborate on the findings from these steps. This is followed by a description of promising avenues for future research.

Our first step in fulfilling the thesis goal was to examine the value of customization through an experiment with simulated users. Since our results showed that customization could be beneficial, but that some users might need adaptive support to customize effectively, we then designed the MICA system. Based on relevant user and interface characteristics identified in the simulation experiment, MICA helps the user customize by supplying tailored customization suggestions designed to maximize user performance. To increase interaction transparency and predictability, MICA provides the user with access to its rationale. The benefits of MICA's mixed-initiative approach were evaluated through two formal evaluations. Study One compared MICA to a purely adaptable alternative, while Study Two was targeted at the understanding the benefits of the rationale.

7.1 Thesis Contributions

7.1.1 Analysis of Factors Influencing the Effectiveness of User-Controlled Customization

Our experiment with simulated users analyzed the theoretical benefits of user-controlled customization. Prior to this work, there had been no evidence of how user customization impacts performance, or of what factors contribute to effective customization. In addition to illustrating that customization can be beneficial, our results identified properties of the user, his tasks, and the interface

itself that impact the effectiveness of a given customization strategy. Another contribution of this work is the manner in which we extended GOMS, the predictive performance measure used in our experiment, to incorporate expertise. Our extension allows GOMS analysis to account for the impact of varying levels of expertise on the time necessary to find items in a complex interface. GOMS has traditionally been used to model and predict only expert behaviour.

7.1.2 Framework for Providing Principled Customization Support

Using the results from the experiment with simulated users, we designed and built the MICA system. MICA employs a novel mixed-initiative approach to customization by relying on on-line GOMS analysis. While GOMS analysis is often used to assess the value of different interfaces off-line, in this work we illustrate how it can be used in a real-time setting to generate tailored customization suggestions. In performing this GOMS analysis, MICA considers not only usage frequencies, but also user expertise and interface characteristics. It is this formal and comprehensive reasoning that distinguishes MICA from the few other mixed-initiative systems for GUI customization that have been proposed to date.

7.1.3 Design of a Mixed-Initiative Customization Interface

In addition to MICA's underlying framework, we also investigated how to design a mixed-initiative interface to support user customization. MICA's support is delivered non-intrusively by recommending customizations through visual highlighting only when the user initiates customization. Our design also allows users to follow recommendations using a number of different methods, which permit users to delegate more or less responsibility to the system based on their desire for control. Our evaluations showed that users generally liked the format of the recommendations and that users did use multiple methods to accept recommendations.

7.1.4 Identification of the Benefits of a Mixed-Initiative Approach to Customization

Thorough evaluations of mixed-initiative approaches are rare. Even rarer are direct comparisons between mixed-initiative and adaptable (or adaptive) alternatives for GUI customization. The only known example is Debevc *et al.*'s study [36], which, in contrast to the evaluation in this thesis, was less in-depth. Since there is little empirical evidence demonstrating the value of mixed-initiative approaches, a contribution of this work is an illustration of the potential benefits and drawbacks of a mixed-initiative approach. In our first evaluation (Study One) we found that MICA's suggestions tend to improve performance and decrease customization time. We also found that users liked the way MICA presented its suggestions and that users prefer having the support to customizing on their own. In Study One, we evaluated our mixed-initiative approach by comparing it to an adaptive alternative that performed well in previous work. This also makes our study the first comparison to an adaptable system with a published and successful evaluation, ensuring that we were making a fair comparison. The results for our evaluation were very positive indicating the potential for mixed-initiative approaches to improve on purely-adaptable alternatives.

7.1.5 Identification of the Benefits of Rationale Within a System for GUI Customization

An additional contribution of the MICA system is its provision of rationale. Through our second evaluation (Study Two), we showed the benefits and drawbacks of incorporating this information within a mixed-initiative system for GUI customization. The results of the evaluation revealed interesting individual differences in terms of users' desire to have the information present. Many users felt that the rationale increased their trust in the system, increased their understanding of the recommendations, and increased the system's predictability. Perhaps more interesting, however, were the more negative reactions to the rationale. While the majority of users perceived the rationale to be beneficial, the evaluation identified characteristics of the context (i.e., a mixed-initiative system for GUI customization) and of the domain that made the rationale unnecessary for some, such as the fact that users were not forced to comply with the adaptive recommendations, and the fact that users were interacting with a

productivity application. The evaluation showed that not all users cared about predictability or transparency within this context, while others found the system to be sufficiently transparent and predictable to begin with.

7.1.6 Increased Understanding of How to Evaluate Customization in a Laboratory Environment

This thesis work also adds to the body of knowledge on how to evaluate customization. Customization is typically meant to be persistent and beneficial over longer periods of use – two properties that are more naturally evaluated in a longitudinal field study. As we discussed in previous chapters, laboratory evaluations are sometimes necessary or even preferable at certain stages of the research, yet little is known on how to design a laboratory experiment involving customization. Through Studies One and Two, we discussed a number of methodology issues investigated throughout this work, including how to motivate customization, task design, and within-subjects versus between-subjects tradeoffs. Lessons learned from these evaluations can be used in future studies of customization.

7.2 Directions for Future Research

7.2.1 Extensions to MICA

On-Line Assessment of Relevant User Traits

The most obvious extension to the MICA system would be to implement techniques within the User Model to permit it to perform on-line assessment of expertise and expected usages. The first step required to assess expertise will be to increase the amount of information about user interface actions available to the User Model, since it currently has access to only final feature-selection events. Other events that will likely be required to assess expertise on-line include: (i) menu-open events, (ii) menu and toolbar item mouse-over events, and (perhaps) (iii) events as a result of user actions in the help system. Prior to Study One, we discovered that some of this information can be obtained from the Microsoft Accessibility APIs. In fact, we used a logger based on this technology to capture data in Studies One and Two.²⁰ Future work would entail

²⁰The logger used in Studies One and Two was implemented by Jennifer Gluck, who completed an M.Sc. in the Computer Science Department at the University of British Columbia.

developing appropriate models of expertise and extending the User Model to capture and reason about the data collected by this logger on-line. We would also like to analyze the data collected during the evaluations in combination with the data from the expertise questionnaires to examine the range of behaviours exhibited by participants at each self-reported expertise level.

The primary challenge for assessing expected usages is how to handle situations where user's tasks are changing such that previously collected usage data is unrepresentative of the user's future tasks. First, MICA might want to reason about *when* a feature is expected to be used in addition to how often, both in deciding which recommendations to make and how to deliver them to the user. There might, however, be circumstances where even recent behaviour is not indicative of future behaviour. Dealing with these situations might require a dialogue between MICA and the user, to discuss how to customize for the user's upcoming tasks. An appropriate place for this dialogue facility could be within the rationale component, where the user could view and perhaps edit portions of the User Model assessments and ask the system to revise its recommendation accordingly.

Taking More of the Initiative

The second major area of future work concerning the MICA system would be to extend MICA's framework to allow it to take more of the initiative. Currently MICA waits for the user to initiate customization before providing any support. This type of support might not be sufficient for users who want to customize but are forgetting to do so, behaviour that was observed in our studies. MICA could take more of the initiative, by alerting users to the presence of customization suggestions while they are performing their primary tasks with the interface. Taking the initiative in this way would require both designing a method with which to alert the user and extending the framework to reason about the cost of interruption vs. potential customization benefits. Since MICA would have an estimate of how much customization suggestions would improve the user's performance, it could adapt its notification signals based on utility, an approach that was evaluated empirically with positive results by Gluck *et al.* [51].

Login Customization Support

The type of support that MICA currently provides assumes that the system has been able to observe the user interacting with the application for a period of

time (i.e., the User Model has information on the relevant factors). However, it might be desirable for MICA to help users create an initial Personal Interface when they first log into the system, since, as discussed in chapter 3, depending on expertise, some users will greatly benefit from having an interface customized right away. When a user logs onto the system, MICA could engage the user in a dialogue to help the user set up an initial Personal Interface that will suit her tasks. This type of support could be particularly beneficial for Novice users who have the most to gain from customization, yet might not be able to initiate customization early without system support.

7.2.2 Further Evaluation

Comparing MICA to an Adaptive Alternative

There is some indirect evidence that our mixed-initiative approach might be preferred to a purely adaptive alternative. In particular, MICA was compared (and preferred) to an adaptable alternative, and this adaptable alternative was previously compared (and preferred) to an adaptive interface. Thus, through transitivity, there is some evidence that users might prefer the mixed-initiative approach to a particular adaptive alternative. Fully understanding the advantages and drawbacks of the mixed-initiative approach relative to an adaptive alternative, however, requires a direct comparison. To isolate the effects of the mixed-initiative aspects of the interaction, it would be best to compare two versions of MICA: one with user control and one without. This would first require designing a way for MICA to deliver its support in a purely adaptive fashion, ideally minimizing usability problems that are commonly associated with adaptive interfaces. For example, previous evaluations have shown that frequently changing adaptive interfaces tend not to fare well. Therefore, one strategy could be to change the GUI at relatively infrequent intervals (e.g., once every 20-30 feature selections), providing the user with visual cues (such as temporary highlighting) to indicate the new system-controlled customizations. Prior to an evaluation comparing the adaptive and mixed-initiative approaches, appropriate frequencies and methods of highlighting changes would first have to be tested with users, to ensure as fair of a comparison as possible.

Exploring the Generalizability of MICA's Framework

While MICA's framework has the potential to generalize to a number of different types of graphical user interfaces, the techniques have been tested with only one application: a word processor. Exploring the generalizability of our approach would involve applying the techniques to different applications, differing in complexity and in the characteristics of the end users. In chapter 4 we discussed how MICA's techniques are designed to transfer easily to other applications. In addition to verifying that this is, in fact, the case, evaluations of these different interfaces could explore how the approach scales as the complexity of the interface increases and as the application-specific and computer-related expertise of the end user increases. Since the effort necessary to customize effectively and the potential savings might both increase as application complexity increases, users could be even more welcoming of the mixed-initiative support, including more experienced users.

Evaluating MICA's Approach Through Longitudinal and Field Studies

Our evaluations tested the mixed-initiative approach in a single session experiment in a laboratory environment, leaving more longitudinal studies both in the laboratory and in the field as avenues for future work. These types of evaluations would provide insight into how users respond to the mixed-initiative support when they are using MICA for longer periods of time and (in the case of field studies) performing their day-to-day tasks. A number of questions arise from the above circumstances including: (i) how will users' reliance on the mixed-initiative support change over time, and (ii) how appropriate will MICA's support be in terms of both perceived and actual benefit? As users gain experience with MICA, they might begin to trust the support more, a sentiment which was expressed in Study Two. For MICA's support to be beneficial over longer periods of time, however, it will probably be necessary to extend the framework to better handle users with changing tasks (as discussed above). Studies that are longer in duration and/or conducted in the field would also provide excellent opportunities to gather more information on rationale usage and impact.

Exploring Conditions Impacting Rationale Utility

As discussed in chapter 6, given the diversity of opinions expressed in Study Two, a promising direction of future research would be to gain a better understanding of when and why rationale is useful. In addition to observing rationale usage over longer periods of time, it would be interesting to evaluate how user variability, the target application's complexity, and the division of control between the system and the user affect the qualitative and quantitative utility of a system's rationale. Certain applications, such as decision-support systems and recommender systems in more complex domains, might exhibit adaptive behaviour that is inherently less predictable and transparent, which could make having access to rationale more crucial for users. Alternatively, it could be the amount of risk associated with accepting potentially inaccurate recommendations that effects the utility of the rationale.

7.3 Concluding Comments

This thesis has investigated the potential for mixed-initiative solutions to find an appropriate middle ground between the adaptive and adaptable approaches to customizing graphical user interfaces. Our evaluations indicate that a system that supports user customization while still maintaining control, transparency and predictability has many advantages over the purely adaptable alternative. The mixed-initiative solution described in this thesis is only one of number of different ways that a system and user could collaborate to produce a customized interface. The success of the approach developed here provides persuasive evidence that mixed-initiative solutions deserve further investigation.

References

- [1] A. Aaltonen, A. Hyrskykari, and K. Raiha. 101 spots, or how do users read menus. In *Proceedings of ACM CHI Conference on Human Factors in Computing Systems (CHI'98)*, pages 18–23, 1998.
- [2] R. St. Amant and P.R. Cohen. Interaction with a mixed-initiative system for exploratory data analysis. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI'97)*, pages 15–22, 1997.
- [3] J.R. Anderson, A.T. Corbett, K.R. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4(2):167–207, 1995.
- [4] B.B. Bederson. Fisheye menus. In *ACM Symposium on User Interface Software and Technology (UIST 2000)*, pages 217–225, 2000.
- [5] D. Benyon. Accomodating individual differences through an adaptive user interface. In M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, editors, *Adaptive User Interfaces*, pages 149–165. Elsevier Science Publishers, 1993.
- [6] D. Billsus and M.J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10:147–180, 2000.
- [7] T. Bohnenberger, O. Jacobs, A. Jameson, and I. Aslan. Decision-theoretic planning meets user requirements: Enhancements and studies of an intelligent shopping guide. In *Pervasive Computing: 3rd International Conference*, pages 279–296, 2005.
- [8] D. Browne, P. Totterdell, and M. Norman, editors. *Adaptive User Interfaces*. Academic Press, San Diego, CA, 1990.
- [9] P. Brusilovsky. Adaptive navigation support. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web*

- Personalization, Lecture Notes in Computer Science, Vol. 4321*. Springer-Verlag, Berlin Heidelberg New York, 2007.
- [10] S. Bull and M. McKay. An open learner model for children and teachers: Inspecting knowledge levels of individuals and peers. In *Proceedings of ITS 2004, 7th International Conference on Intelligent Tutoring Systems*, pages 646–655, 2004.
- [11] A. Bunt, G. Carenini, and C. Conati. Adaptive content presentation for the web. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321*. Springer-Verlag, Berlin Heidelberg New York, 2007.
- [12] A. Bunt and C. Conati. Probabilistic student modeling to improve exploratory behaviour. *User Modeling and User-Adapted Interaction*, 13(3):269–309, 2003.
- [13] A. Bunt, C. Conati, M. Huggett, and K. Muldner. On improving the effectiveness of open learning environments through tailored support for exploration. In *Proceedings of AIED 2001, 10th International Conference on Artificial Intelligence in Education*, pages 365–376, 2001.
- [14] A. Bunt, C. Conati, and J. McGrenere. What role can adaptive support play in an adaptable system? In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2004)*, pages 117–124, 2004.
- [15] A. Bunt, C. Conati, and J. McGrenere. Supporting interface customization using a mixed-initiative approach. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 92–101, 2007.
- [16] A. Bunt, J. McGrenere, and C. Conati. Understanding the utility of rationale in a mixed-initiative system for GUI customization. In *Proceedings of the International Conference on User Modeling (UM 2007)*, pages 147–156, 2007.
- [17] M.D. Byrne, J.R. Anderson, S. Douglas, and M Matessa. Eye tracking the visual search of click-down menus. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)*, pages 402–409, 1999.
- [18] S. Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.

-
- [19] S.K. Card. Visual search of computer command menus. In H. Bouma and D.G. Bouwhuis, editors, *Attention and Performance X: Control of Language Processes*, pages 167–200. Mahwah, NJ, 1984.
- [20] S.K. Card, A. Newell, and T. P. Moran. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, 1983.
- [21] J.M. Carroll. Blocking learner error states in a training-wheels system. *Human Factors*, 26(4):377–389, 1984.
- [22] L. Chen and P. Pu. Hybrid critiquing-based recommender systems. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 22–31, 2007.
- [23] K. Cheverst, H.E. Byun, D. Fitton, C. Sas, C. Kray, and N. Villar. Exploring issues of user model transparency and proactive behaviour in an office environment control system. *User Modeling and User-Adapted Interaction*, 15(3-4):235–273, 2005.
- [24] L.G. Christiernin, F. Lindahl, and O. Torgersson. Designing a multi-layered image viewer. In *Proceedings of NordCHI 2004*, pages 181–184, 2004.
- [25] B. Clark and J. Matthews. Deciding layers: Adaptive composition of layers in a multi-layered user interface. In *Proceedings of the 11th International Conference on Human-Computer Interaction*, 2005.
- [26] R.T. Clement. *Making Hard Decisions*. Brooks/Cole Publishing Company, Pacific Grove, CA, 2nd edition, 1996.
- [27] A. Cockburn, C. Gutwin, and S. Greenberg. A predictive model of menu performance. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2007)*, 2007.
- [28] J. Cohen. *Statistical power analysis for the behavioral sciences*. Lawrence Earlbaum Associates, Hillsdale, NJ, 1988.
- [29] R. Cohen, C. Allaby, C. Cumbaa, M.Fitzgerald, K. Ho, B. Hui, C. L., F. Lu, N. Moussa, D. Pooley, A. Qian, and S. Siddiqi. What is initiative? *User Modeling and User-Adapted Interaction*, 8:171–214, 1998.

- [30] C. Conati and K. VanLehn. Toward computer-based support of meta-cognitive skills: a computational framework to coach self-explanation. *International Journal of Artificial Intelligence in Education*, 11:398–415, 2000.
- [31] J.A. Cote-Munoz. AIDA: An adaptive system for interactive drafting and CAD applications. In M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, editors, *Adaptive User Interfaces*, pages 225–240. Elsevier Science Publishers, 1993.
- [32] M. Coyle and B. Smyth. On the community-based explanation of search results. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 283–85, 2007.
- [33] A. Cypher. Eager: Programming repetitive tasks by example. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'91)*, pages 33–39, 1991.
- [34] Allen Cypher, editor. *Watch What I Do: Programming by Demonstration*. MIT Press, Cambridge, Massachusetts, 1993.
- [35] M. Czarkowski and J. Kay. How to give the user a sense of control over the personalization of adaptive hypertext? In *Proceedings of Adaptive Hypermedia and Adaptive Web-Based Systems (in conjunction with User Modeling 2003)*, pages 121–131, 2003.
- [36] M. Debevc, B. Meyer, D. Donlagic, and R. Svecko. Design and evaluation of an adaptive icon toolbar. *User Modeling and User-Adapted Interaction*, 6(1):1–21, 1996.
- [37] M. Debevc, B. Meyer, and R. Svecko. An adaptive short list for documents on the world wide web. In *Proceedings of Intelligent User Interfaces (IUI'97)*, pages 209–211, 1997.
- [38] H. Dieterich, U. Malinowski, T. Kuhme, and M. Schneider-Hufschmidt. State of the art in adaptive user interfaces. In M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, editors, *Adaptive User Interfaces*, pages 13–48. Elsevier Science Publishers, 1993.
- [39] F.J. Díez, J. Mira, E. Iturralde, and S. Zubillaga. DIAVAL, a Bayesian expert system for echocardiography. *Artificial Intelligence in Medicine*, 10(1):59–73, 1997.

-
- [40] L. Findlater. Comparing static, adaptive, and adaptable menus. Master's thesis, University of British Columbia, Vancouver, British Columbia, 2004.
- [41] L. Findlater and J. McGrenere. A comparison of static, adaptive, and adaptable menus. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2004)*, pages 89–96, 2004.
- [42] L. Findlater and J. McGrenere. Evaluating reduced-functionality interfaces according to feature findability and awareness. In *Proceedings of INTERACT*, 2007.
- [43] J. Fink, A. Kobsa, and A. Nill. Adaptable and adaptive information provision for all users, including disabled and elderly people. *The New Review of Hypermedia and Multimedia*, 4:163–188, 1998.
- [44] G. Fischer. Shared knowledge in cooperative problem-solving systems - integrating adaptive and adaptable components. In M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, editors, *Adaptive User Interfaces*, pages 49–68. Elsevier Science Publishers, 1993.
- [45] R. Freedman. Atlas: A plan manager for mixed-initiative, multimodal dialogue. In *AAAI-99 Workshop on Mixed-Initiative Intelligence*, 1999.
- [46] H.C. Frey and S.R. Patil. Identification and review of sensitivity analysis methods. *Risk Analysis*, 22(3):553–578, 2002.
- [47] K. Gajos, D. Christianson, R. Hoffmann, T. Shaked, K. Henning and J.J. Long, and D.S. Weld. Fast and robust interface generation for ubiquitous applications. In *Proceedings of Ubicomp*, pages 37–55, 2005.
- [48] K. Gajos and D.S. Weld. SUPPLE: Automatically generating user interfaces. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2004)*, pages 93–100, 2004.
- [49] K.Z. Gajos, M. Czerwinski, D.S. Tan, and D.S. Weld. Exploring the design space of adaptive graphical user interfaces. In *Proceedings of Advanced Visual Interfaces (AVI 2006)*, pages 201–208, 2006.
- [50] M. Gant and B.A. Nardi. Gardeners and gurus: Patterns of cooperation among CAD users. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'92)*, pages 107–117, 1992.

- [51] J. Gluck, A. Bunt, and J. McGrenere. Matching attentional draw with utility in interruption. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2007)*, pages 41–50, 2007.
- [52] Q. Gong and G. Salvendy. An approach to the design of a skill adaptive interface. *International Journal of Human-Computer Interaction*, 7(4):365–383, 1995.
- [53] R. Gong and D. Kieras. A validation of the GOMS model methodology in the development of a specialized, commercial software application. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)*, pages 351–357, 1994.
- [54] S. Greenberg and I.H. Witten. Adaptive personalized interfaces – a question of viability. *Behaviour and Information Technology*, 4(1):31–45, 1985.
- [55] S. Greenberg and I.H. Witten. Supporting command reuse: mechanisms for reuse. *International Journal of Man-Machine Studies*, 39:391–425, 1993.
- [56] F. Hayes-Roth and N. Jacobstein. The state of knowledge-based systems. *Communications of the ACM*, 37(4):27–39, 1994.
- [57] J. Herlocker, J.A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the ACM Conference on Computer-Supported Collaborative Work (CSCW 2000)*, pages 241–250, 2000.
- [58] K. Hook. Steps to take before intelligent user interfaces become real. *Interacting with Computers*, 12:409–426, 2000.
- [59] A.J. Hornoff and D.E. Kieras. Cognitive modeling reveals menu search is both random and systematic. In *Proceedings of the ACM Conference on Human factors in Computing Systems (CHI'97)*, pages 107–114, 1997.
- [60] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)*, pages 159–166, 1999.
- [61] E. Horvitz, J. Breese, and M. Henrion. Decision theory in expert systems and artificial intelligence. *Journal of Approximate Reasoning*, 2:247–302, 1988.

-
- [62] E. Horvitz, D. Herckerman, D. Hovel, and R. Rommelse. The Lumière project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of UAI'98, Conference on Uncertainty in Artificial Intelligence*, pages 256–265, 1998.
- [63] A. Hurst, S.E. Hudson, and J. Mankoff. Dynamic detection of novice vs. skilled use without a task model. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2007)*, 2007.
- [64] S.L. Jackson, J. Krajcik, and E. Solloway. The design of guided learner-adaptable scaffolding in interactive learning environments. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'98)*, pages 187–194, April 1998.
- [65] A. Jameson. Adaptive interfaces and agents. In J. Jacko and A. Sears, editors, *Human-Computer Interaction Handbook*, pages 305–330. Erlbaum, Mahwah, NJ, 2003.
- [66] A. Jameson and E. Schwarzkopf. Pros and cons of controllability: An empirical study. In *Adaptive Hypermedia and Adaptive Web-Based Systems: Proceedings of AH 2002*, pages 193–202, 2002.
- [67] H.B. Jimison, L.M. Fagan, R.D. Shachter, and E.H. Shortliffe. Patient-specific explanation in models of chronic disease. *Artificial Intelligence in Medicine*, 14:191–205, 1992.
- [68] B. E. John and D.E. Kieras. Using GOMS for user interface design and evaluation: which technique? *ACM Transactions on Computer-Human Interaction*, 3(4):287–319, 1996.
- [69] A.H. Jorgensen and A. Sauer. The personal touch: A study of users' customization practice. In *Proceedings of INTERACT*, pages 561–565, 1990.
- [70] S. Kaasten and S. Greenberg. Integrating back, history and bookmarks in web browsers. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2001), Extended Abstracts*, pages 379–380, 2001.
- [71] J. Kay. The um toolkit for cooperative user modelling. *User Modeling and User-Adapted Interaction*, 4(3):149–196, 1995.

- [72] J. Kay. Learner control. *User Modeling and User-Adapted Interaction*, 11:111–127, 2001.
- [73] R.M. Keller, S.R. Wolfe, J.R. Chen, J.L. Rabinowitz, and N. Mathe. A bookmarking service for organizing and sharing URLs. In *Selected Papers From the 6th International Conference on World Wide Web*, pages 1103–1114, 1997.
- [74] D. Kieras. A guide to GOMS model usability evaluation using GOMS and GLEAN3. Technical Report TR-98/ARPA-2, University of Michigan, 1998.
- [75] D.E. Kieras, S.D. Wood, K. Abotel, and A.J. Hornof. GLEAN: A computer-based tool for rapid GOMS model usability evaluation of user interface designs. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'95)*, pages 91–100, 1995.
- [76] J. Koenemann and N.J. Belkin. A case for interaction: a study of interactive information retrieval behaviour and effectiveness. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96)*, pages 205–212, 1996.
- [77] M. Krogsaeter, R. Oppermann, and C. Thomas. A user interface integrating adaptability and adaptivity. In *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*, pages 97–125. Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.
- [78] T. Kuhme, U. Malinowski, and J.D. Foley. Facilitative interactive tool selection by adaptive prompting. In *Proceedings of INTERACT'93 and CHI'93 Companion Conference on Human-Factors in Computing Systems*, pages 149–150, 1993.
- [79] T.K. Landauer and D.W. Nachbar. Selection from alphabetic and numeric menu trees using a touch screen: Breadth, depth, and the width. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'85)*, pages 73–78, 1985.
- [80] E. Lee and J. MacGregor. Minimizing user search time in menu retrieval systems. *Human Factors*, 27:157–162, 1985.
- [81] J.D. Lee and K.A. See. Trust in automation: Designing for appropriate reliance. *Human Factors*, 46(1):50–80, 2004.

-
- [82] N.B. Lesh, C. Rich, and C.L. Sidner. Using plan recognition in human-computer collaboration. In *Proceedings of the International Conference on User Modelling (UM'99)*, pages 23–32, 1999.
- [83] F. Linton, D. Joy, H. Schaefer, and A. Charron. OWL: A recommender system for organization-wide learning. *Educational Technology & Society*, 3(1), 2000.
- [84] F. Linton and H. Schaefer. Recommender systems for learning: Building user and expert model through long-term observation of application use. *User Modeling and User-Adapted Interaction*, 10:181–207, 2000.
- [85] A. Mabbott and S. Bull. Alternative views on knowledge: Presentation of open learner models. In *Proceedings of ITS 2004, 7th International Conference on Intelligent Tutoring Systems*, pages 689–698, 2004.
- [86] W.E. Mackay. Patterns of sharing customizable software. In *Proceedings of the ACM Conference on Computer Support Cooperative Work (CSCW'90)*, pages 209–221, 1990.
- [87] W.E. Mackay. Triggers and barriers to customizing software. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'91)*, pages 153–160, 1991.
- [88] I. Scott MacKenzie. Movement time prediction in human-computer interfaces. In *Human-Computer Interaction: Toward the Year 2000*, pages 483–492. Morgan Kaufmann Publishers Inc., 1995.
- [89] A. MacLean, K. Carter, L. Lovstrand, and T. Moran. User-tailorable systems: Pressing the issues with buttons. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'90)*, pages 175–182, 1990.
- [90] U. Malinowski. Adjusting the presentation of forms to users' behaviour. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI'93)*, pages 247–249, 1993.
- [91] J. McGrenere. *The Design and Evaluation of Multiple Interfaces: A Solution for Complex Software*. PhD thesis, University of Toronto, Toronto, Ontario, 2002.

- [92] J. McGrenere, R.M. Baecker, and K.S. Booth. An evaluation of a multiple interface design solution for bloated software. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2002)*, pages 163–170, 2002.
- [93] J. McGrenere and G. Moore. Are we all in the same "bloat"? In *Proceedings of Graphics Interface (GI 2000)*, pages 187–196, 2000.
- [94] T. Miah, M. Karageorgou, and R. P. Knott. Adaptive toolbars: An architectural overview. In *Proceedings of the 3rd ERCIM Workshop on User Interfaces for All*, 1997.
- [95] J. Mitchell and B. Shneiderman. Dynamic versus static menus: an exploratory comparison. *SIGCHI Bulletin*, 20(4):33–37, 1989.
- [96] K. Muldner and C. Conati. Evaluating a decision-theoretic approach to tailored example selection. In *Proceedings of IJCAI 2007, 20th International Joint Conference in Artificial Intelligence*, pages 483–489, 2007.
- [97] B. Myers, D.A. Weitzman, A.J. Ko, and D.H. Chau. Answering why and why not questions in user interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2006)*, pages 397–406, 2006.
- [98] K.L. Norman. *The Psychology of Menu Selection*. Ablex Publishing Corporation, Norwood, New Jersey, 1991.
- [99] OpenOffice. www.openoffice.org.
- [100] R. Oppermann. Adaptively supported adaptability. *International Journal of Human-Computer Studies*, 40:455–472, 1994.
- [101] S.R. Page, T.J. Johnsgard, U. Albert, and C.D. Allen. User customization of a word processor. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96)*, pages 340–346, 1996.
- [102] K.A. Papanikolaou, M. Grigoriadou, H. Kornilakis, and G.D. Magoulas. Personalizing the interaction in a web-based educational hypermedia system: the case of INSPIRE. *User Modeling and User-Adapted Interaction*, 13(3):213–267, 2003.

-
- [103] C. Price, G. McCalla, and A. Bunt. L2tutor: A mixed-initiative dialogue system for improving fluency. *Computer-Assisted Language Learning*, 12(2):83–112, 1999.
- [104] P. Pu and L. Chen. Trust building with explanation interfaces. In *Proceedings of the ACM Conference Intelligent User Interfaces (IUI 2006)*, pages 93–100, 2006.
- [105] M. Schneider-Hufschmidt, T. Kuhme, and U. Malinoswki, editors. *Adaptive User Interfaces: Principles and Practice*. North-Holland, Amsterdam, The Netherlands, 1993.
- [106] A. Sears and B. Shneiderman. Split menus: effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, 1(1):27–51, 1994.
- [107] B. Shneiderman. Promoting universal usability with multi-layered interface design. In *Proceedings of Conference on Universal Usability (CUU 2003)*, pages 1–8, 2003.
- [108] B. Shneiderman and P. Maes. Direct manipulation vs. interface agents. *interactions*, 4(6):42–61, 1997.
- [109] V.J. Shute and J. Psotka. Intelligent tutoring systems: Past, present, and future. In D. Jonassen, editor, *Handbook of Research on Educational Communications and Technology*, pages 570 – 600. Macmillan, New York, NY, 1996.
- [110] W. Stuerzlinger, O. Chapuis, D. Phillips, and N. Roussel. User interface façades: Towards fully adaptable user interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST 2007)*, pages 309–318, 2007.
- [111] S. Stumpf, V. Rajaram, L. Li, M. Burnett, T. Dietterich and. E. Sullivan, R. Drummond, and J. Herlocker. Toward harnessing user feedback for machine learning. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 82–91, 2007.
- [112] A.C. Stylianou, G.R. Madey, and R.D. Smith. Selection criteria for expert systems shells: A sociology technical framework. *Communications of the ACM*, 35(10):30–48, 1992.

- [113] R.L. Teach and E.H. Shortliffe. An analysis of physician attitudes regarding computer-based clinical consultation systems. *Computers and Biomedical Research*, 14:542–558, 1981.
- [114] L.G. Terveen and L.T. Murray. Helping users program their personal agents. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI'96)*, pages 355–361, 1996.
- [115] C.G. Thomas and M. Krogsoeter. An adaptive environment for the user interface of excel. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI'93)*, pages 123–130, 1993.
- [116] R. Trevelyan and D.P. Browne. A self-regulating adaptive system. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'87)*, pages 103–107, 1987.
- [117] S. Trewin. Automating accessibility: the dynamic keyboard. In *Proceedings of ASSESTS*, pages 71–78, 2004.
- [118] T. Tsandilas and m.c. Schraefel. An empirical assessment of adaptation techniques. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2005), Extended Abstracts*, pages 2009–2012, 2005.
- [119] A. Tversky and D. Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, 1981.
- [120] B. Uttl. North american adult reading test: Age norms, reliability, and validity. *Journal of Clinical and Experimental Neuropsychology*, 24(8):1123–1137, 2002.
- [121] A. Vandierendonck, R. Van Hoe, and G. De Soete. Menu search as a function of menu organization, categorization and experience. *Acta Psychologica*, 69(3):231–284, 1988.
- [122] W. Wang and I. Benbasat. Recommendation agents for electronic commerce: Effects of explanation facilities on trusting beliefs. *Journal of Management Information Systems*, 23(4):219–249, 2007.
- [123] J. Warren. Cost/benefit based adaptive dialog: Case study using empirical medical practice norms and intelligent split menus. In *Proceedings of IEEE 2nd Australian Conference on User Interface*, pages 100–107, 2001.

-
- [124] D.S. Weld, C. Anderson, P. Domingos, O. Etzioni, K. Gajos, T. Lau, and S. Wolfman. Automatically personalizing user interfaces. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2003)*, August 2003.
- [125] Z. Wen, M.X. Zhou, and V. Aggarwal. Context-aware information retrieval for investigative tasks. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2007)*, pages 122–131, 2007.
- [126] S.A. Wolfman, T. Lau, P. Domingos, and D.S. Weld. Mixed initiative interfaces for learning tasks: SMARTedit talks back. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2001)*, pages 167–174, 2001.
- [127] J.D. Zapata-Rivera and J.E. Greer. Interacting with inspectable bayesian student models. *International Journal of Artificial Intelligence in Education*, 14:127–163, 2003.
- [128] M.X. Zhou, Z. Wen, and V. Aggarwal. A graph-matching approach to dynamic media allocation in intelligent multimedia interfaces. In *Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2005)*, pages 114–121, 2005.

References

Appendix A

Preliminary Questionnaire

The preliminary questionnaire used in Studies One and Two is based on the one from McGrenere *et al.*'s field study [91, 92]. This questionnaire contains questions from McGrenere and Moore's the Feature Profile Scale [93], which are included here with the authors' with permission.

Microsoft Word Study

Please fill in this questionnaire if you are interested in participating in the Microsoft Word Study. The purpose of this questionnaire is to see if you will be a good fit for the study. Note that all information provided will remain strictly confidential. Completing the questionnaire should require no more than 10 minutes of your time.

*** Note that to participate in this study you must:

- Be 18 or older;
- Speak English as your native language;²¹ and
- Have used Microsoft Word 2003 at least once.

Please fill in every question as best you can. The fields with [required] are required fields.

Section 0: Contact Information

Firstname: [required] -----
Lastname: [required] -----
Phone number: [required] -----
E-mail address: [required] -----

²¹Changed to "Be highly fluent in English" for Study Two.

Section I: Essentials

A. [required] Have you ever used Microsoft Word, version 2003?

- Yes No

B. [required] For approximately how many months have you used Microsoft Word 2003?

- 0-1 month
 1-3 months
 3-5 months
 more than 6 months

C. [required] On average, how many hours a week do you spend word processing?

- 0-1 hours
 1-2 hours
 2-3 hours
 3-4 hours
 more than 5 hours
-

Section II: General Experience with MS Word and other MS Office applications.

If you have little or no experience with MS Office applications other than MS Word this is okay. Please continue to fill in the questionnaire.

It is important that you answer the questions in this section as best you can “off the top of your head”. Our intention is not to test your ability to look up answers.

Given you current computer setup, please indicate the extent to which you agree or disagree with the following statements:

SD	=	Strongly Disagree
D	=	Disagree
N	=	Neutral
A	=	Agree
SA	=	Strongly Agree

I can do my work more efficiently (i.e., faster) with each new version of MS Word. SD D N A SA

I can do my work more effectively (i.e., better) with each new version of MS Word. SD D N A SA

I am overwhelmed by how much new “stuff” there seems to be with each version of MS Word. SD D N A SA

I have a hard time finding the features I need unless I use them regularly. SD D N A SA

I feel that the time I spend learning a new version of MS Word makes me more efficient (i.e., faster). SD D N A SA

I feel that the time I spend learning a new version of MS Word makes me more effective (i.e., better) in doing my work. SD D N A SA

Each new version of MS Word becomes easier to use. SD D N A SA

I get annoyed when I can’t quickly find a feature that I’ve used previously. SD D N A SA

I feel that the time I spend learning a new version of MS Word makes it less frustrating to use. SD D N A SA

Each new version of MS Word is more complex than the previous version. SD D N A SA

I resent the time I spend learning a new version of MS Word. SD D N A SA

I get annoyed when I can’t quickly figure out how to do something that I need to do. SD D N A SA

Even though there is lots of “stuff” in each new version that I don’t use, I’m sure it is there for a good reason. SD D N A SA

Appendix A. Preliminary Questionnaire

In general, I think MS Word gets better with each new version. SD D N A SA

Even though there are functions found in the menus/toolbars that I do not use, it is reassuring to me that they are there. SD D N A SA

If it were up to me I would upgrade MS Word only when new functions that I need are added. SD D N A SA

I would prefer to have only the functions I use in the menus/toolbars - the other functions could be “tucked” away in the event that I might need them someday. SD D N A SA

I want the latest version of MS Word as soon as it is available. SD D N A SA

I am happy to pay for an application that is “fully loaded” even if I don’t use all the functions. SD D N A SA

I would like a word processor that gave me only the functions that I use. SD D N A SA

I prefer to continue using the version of MS Word currently on my machine for as long as possible. SD D N A SA

It is important to me that I continually discover new functions. SD D N A SA

Wading through unfamiliar functions can often be annoying/frustrating. SD D N A SA

Section III: Personal Information

1. In what age group are you?

- 17 and under
- 18-29
- 30-39
- 40-49
- 50-59
- 60+

2. Gender:

- Male
- Female

3. Do you speak English as your native language?²²

- Yes
- No

4. In terms of your current occupation, how would you characterize yourself?

- Writer
- Administrative Assistant
- Journalist
- Secretary
- Academic
- Professional
- Technical expert
- Student: department _____
- Designer
- Administrator/Manager
- Other, please specify: _____

Thank you for completing this questionnaire and your interest in the MSWord Study. We will contact you within one week's time to let you know if you are a good fit for our study.

If you have any questions do not hesitate to contact Andrea at bunt@cs.ubc.ca.

²²Changed to "Are you highly fluent in English?" in Study Two.

Appendix B

Expertise Questionnaires

Below is a part of the expertise questionnaire used in Studies One and Two, including the questionnaire instructions, and sample questions for one menu item and one toolbar item. There was one such question for each feature in the experiment plus a number of distracter questions. The “Access Levels” correspond to our three lowest expertise categories defined in chapter 3 (Thorough Search = Novice, Systematic Scan = Intermediate, and Local Scan = Expert). A fourth “Access Level” representing the highest category (Extreme Expert) was removed after Study One’s pilot (see sec. 5.3). We calculated the average expertise for each participant in Studies One and Two (see tables 5.1 and 6.1) as follows: Thorough Search = 1, Systematic Search = 2, and Local Scan = 3.

B.1 Menu Items

B.1.1 Instructions

Part I: Menu Items

Access Level

For each of the following features please circle one option for the “Access Level” To access the feature you:

Local Scan: know exactly which menu it is in and approximately where it is located within the menu.

Systematic Scan: know exactly which menu it is in but need to scan the menu one item at a time to find it.

Thorough Search: need to scan more than one menu from top to bottom to find it.

Knowing what a function does:

having general knowledge of the feature’s action.

B.1.2 Sample Question



Do you know what this function does? No Yes

Access Level: Local Scan Systematic Scan Thorough Search

You use the function:

- a) Never
 - b) Irregularly (at least a few times)
 - c) Semi-Regularly (once per month)
 - d) Regularly (weekly)
 - e) Frequently (multiple times per week)
-

B.2 Toolbar Items

B.2.1 Instructions

Part II: Toolbar Items

Access Level

For each of the following features please circle one option for the “Access Level”
To access the feature you:

Local Scan: know the approximate location of the feature

Systematic Scan: would have to look through the toolbar features one
at a time but would know when you found the feature

Thorough Search: would have to look at all toolbar items to find it.

Knowing what a function does:

having general knowledge of the feature’s action.

B.2.2 Sample Question



Do you know what this function does? No Yes

Access Level: Local Scan Systematic Scan Thorough Search

You use the function:

- a) Never
 - b) Irregularly (at least a few times)
 - c) Semi-Regularly (once per month)
 - d) Regularly (weekly)
 - e) Frequently (multiple times per week)
-

Appendix C

Study One: Call for Participation

Microsoft Word Study Participants Wanted

Pay: \$10/hour (study should take 2 - 3 hours)

You are invited to participate in an exciting word processing study being run in the Computer Science Department at the University of British Columbia! The study will take 2-3 hours and all participants will receive \$10/hour for participating.

We are looking for participants who:

- have used Microsoft Word 2003 at least once
- speak English as their native language
- are 18 years of age or older

What is involved?

Your involvement in this study will consist of one session lasting up to 3 hours. You will be asked to complete a number of short questionnaires and to use Microsoft Word to perform a number of tasks. We may also conduct an interview at the end of the study.

Interested in Participating?

If you are interested in participating we ask that you fill in a preliminary questionnaire. This will take no more than 10 minutes of your time. The preliminary questionnaire can be found at:

www.cs.ubc.ca/~bunt/study

If you have any questions or comments contact Andrea at bunt@cs.ubc.ca.

Appendix D

Study One: Instructions

D.1 Instructions Provided at the Beginning of the Session

The goal of this experiment is to obtain information on how people use a new version of MSWord called MSWord Personal.

MSWord Personal provides you with two interfaces:

1. Your Personal Interface. An interface that contains some, but not all of the Word Features.
2. The Full Interface. The standard interface that contains all of the features.

I will provide more information on these interfaces later in the study. To start the study, I'd like you to complete a questionnaire designed to understand your previous Word experience. You will then be asked to perform two tasks, each with a number of repetitions, followed by another questionnaire and a follow-up interview.

D.2 Instructions Provided Prior to the First Task

You will be asked to complete two tasks, each with a different version of the software and a different Personal Interface. Before describing the tasks, I will provide a brief demonstration of how the Personal Interface works using a sample Personal Interface and will also demonstrate how you customize your Personal Interface.

[The participant is given a brief demonstration of the two-interface model and the customization mechanism present in the first trial.]

For each task, you will be given an explicit list of instructions and a final product that following the instructions will produce. Consulting the final product will help you figure out how to accomplish the task. To get a sense of how you use the interface over time with the same task, you will be asked to repeat each task up to five times.

To start the first task, you will have the following Personal Interface. Before I give you your first task instructions, take a minute to look at what is in your Personal Interface.

[Participants are given a minute to look through the starting PI.]

After you complete the first repetition of the task, I will enable the “Modify” button, which you can use to customize your Personal Interface. You can customize as often as you like for the remainder of the task repetitions.

To complete a task:

- Follow the instructions step-by-step to produce the final document.
- If a step involves invoking a feature from the menus or toolbars, the instructions will indicate whether to use a menu or toolbar to invoke the feature. If the step says **Menu**, you are to invoke a menu item to complete the step. If the step says **Toolbar** you are to invoke a Toolbar item.

[Participants are shown an example of each type of step.]

- If you are stuck on how to complete a step you can ask the experimenter who may be able to provide additional assistance. The experimenter may also provide hints under certain circumstances.
- When you are ready to start the task press the “Start Task” button. When you are done the instructions, press the “Done Task” button.

D.3 Instructions Provided After All Repetitions of the First Task are Complete

I will now provide you with your Personal Interface for your second task. For this task, you will also be using a different version of the software.

[If it is the Adaptable condition:]

This time if you customize, there will not be any system recommendations. As with the first task, the customization mechanism will be enabled after the first repetition.

[If it is the Mixed-Initiative condition:]

This time if you customize, the system may also make recommendations.

[The participants are shown what the system recommendations look like on a sample PI.]

I will now give you a minute to look at the contents of your starting Personal Interface for your second task.

[Participants are given a minute to look through the starting PI.]

Appendix E

Study One: Tasks

E.1 Task A

E.1.1 Instructions

1. **Menu:** Open the document “mountainDesc.doc” from “My Documents\filesForTasks”
2. **Menu:** Open another document called “details.doc” from “My Document\filesForTasks”
3. **Menu:** Copy all of the text from “mountainDesc.doc”
4. **Menu:** Paste the text at the top of “details.doc”
5. **Menu:** Save the document “details.doc” under the name “tripReport.doc” in “My Documents\filesForTasks”
6. **Menu:** Close the file “mountainDesc.doc”

All modifications will now be made to the file “tripReport.doc”

7. At the top of the file, enter the text “details”
8. **Toolbar:** Change the size of “details” to 16 pt
9. **Toolbar:** Make it bold
10. **Toolbar:** Change the font to “Century”
11. **Toolbar:** Change the size of two newly-copied paragraphs to 12pt
12. **Menu:** Make the list of food items to bring into a bulleted list (include “Food to Bring” in the list)
13. **Toolbar:** Increase the indentation of the “breakfast” category
14. **Toolbar:** Increase the indentation of the “oatmeal” twice
15. **Toolbar:** Increase the indentation “lunch”
16. **Toolbar:** Increase the indentation of “apples” twice
17. **Toolbar:** Increase the indentation of “bagels” twice
18. **Toolbar:** Increase the indentation of “dinner”
19. **Toolbar:** Increase the indentation of “pasta” twice.
20. **Toolbar:** In the list of “other items”, decrease the indentation of “tents” twice
21. **Toolbar:** Decrease the “bugspray” once

22. **Menu:** Cut the text describing the directions (starting from the text “The hike is a short drive...” to the end of the numbered list)
23. **Menu:** Paste it under the “details” section
24. **Menu:** Cut text “How to Get There”
25. **Menu:** Paste it above the directions.
26. **Toolbar:** Change the size of “How to Get There” to 16
27. **Toolbar:** Make it bold
28. **Toolbar:** Change the font to “century”
29. **Menu:** Save the file
30. **Menu:** Cut text “Things to take on the trip”
31. **Menu:** Paste it above the food list
32. **Toolbar:** Change the size of “Things to take on the trip” to 16
33. **Toolbar:** Change the font to century
34. **Toolbar:** Make it bold
35. **Menu:** Save the file
36. **Toolbar:** Change the size of “Food to Bring” to 14
37. **Toolbar:** Change the size of “Other Items” to 14
38. **Toolbar:** Change the size of “oatmeal” to 10
39. **Toolbar:** Change the size of “apples” to 10
40. **Toolbar:** Change the size of “bagels” to 10
41. **Toolbar:** Change the size of “pasta” to 10
42. **Menu:** Save the document
43. Enter the text “Photos” at the bottom of the document
44. **Toolbar:** Change the size of “Photos” to 16
45. **Toolbar:** Make it bold
46. **Toolbar:** Change the font to “century”
47. **Menu:** Insert the photo “mountainPic” found in “My Documents\filesForTask”
48. **Menu:** Insert a caption under the photo “Figure 1: from near the summit”
49. **Menu:** Insert page numbers
50. **Toolbar:** Check the spelling (and fix the spelling mistakes)
51. **Toolbar:** Justify the text under “details”
52. **Menu:** See what the document would look like printed
53. **Toolbar:** Change the font of the “food” list and “other items” list to “times new roman”
54. **Menu:** Save the document
55. **Menu:** Print the document
56. **Menu:** Close the document (but do not exit the program)

E.1.2 Files Provided to Start the Task

details.doc

Food to Bring:
Breakfast
Oatmeal
Lunch
Apples
Bagels
Dinner
Pasta

The hike is a short drive from Vancouver and easy to find.

Directions:

1. Take highway 99 to just past Pemberton
2. Turn right on owl creek road

How to Get There

Things to Take on the Trip

- Other Items:
 - Sunglasses
 - Tents
 - Sunscreen
 - Bugspray

mountainDesc.doc

After a 9:00 am departure from Vancouver on Saturday morning, we started up the trailhead [780 m] at 12:00 pm for a 3:00 pm arrival at Wedge Mount Lake. After leaving most of our gear at the hut [1925 m], we left again at 3:30 pm headed around the western end of the lake and up snow ramps to the ridge [2240 m]. We scrambled up the east ridge of Wedge and reached its summit [2408 m] at 6:15 pm. Given the late hour, we opted not to climb up the opposite side of the ridge to the summit and instead descended to the lake to enjoy dinner in the last half hour of sunshine before sleeping near the hut under a starry sky.

Chris woke me at 5:30 am. Given our late day on the previous evening and the uncertain weather, we opted to climb the easier southeast ridge instead of the snow route we had intended on its north face. This meant we could leave rope, harnesses, crampons, pickets, etc. at the hut and enjoy a mellow day. We were hiking around the east side of the lake by 6:30 am. After alternating between snow and loose rock on the approach, we reached the ridge crest [2515 m] just before 9:00 am. From there we scrambled the ridge, crossing one snow patch, to reach the summit [2835 m] at 10:00 am. The descent was quick since much of the steeper sections below the ridge crest were on snow; Chris opted to glissade while Stephan preferred boot skiing. We returned to the hut at 12:30 pm, packed and left again at 1:00 pm, arriving at the car by 3:15 pm.

E.1.3 Final Product as a Result of Following the Instructions

Details

After a 9:00 am departure from Vancouver on Saturday morning, we started up the trailhead [780 m] at 12:00 pm for a 3:00 pm arrival at Wedge Mount Lake. After leaving most of our gear at the hut [1925 m], we left again at 3:30 pm headed around the western end of the lake and up snow ramps to the ridge [2240 m]. We scrambled up the east ridge of Wedge and reached its summit [2408 m] at 6:15 pm. Given the late hour, we opted not to climb up the opposite side of the ridge to the summit and instead descended to the lake to enjoy dinner in the last half hour of sunshine before sleeping near the hut under a starry sky.

Chris woke me at 5:30 am. Given our late day on the previous evening and the uncertain weather, we opted to climb the easier southeast ridge instead of the snow route we had intended on its north face. This meant we could leave rope, harnesses, crampons, pickets, etc. at the hut and enjoy a mellow day. We were hiking around the east side of the lake by 6:30 am. After alternating between snow and loose rock on the approach, we reached the ridge crest [2515 m] just before 9:00 am. From there we scrambled the ridge, crossing one snow patch, to reach the summit [2835 m] at 10:00 am. The descent was quick since much of the steeper sections below the ridge crest were on snow; Chris opted to glissade while Stephan preferred boot skiing. We returned to the hut at 12:30 pm, packed and left again at 1:00 pm, arriving at the car by 3:15 pm

How to Get There

The hike is a short drive from Vancouver and easy to find.

Directions:

1. Take highway 99 to just past Pemberton
2. Turn right on owl creek road

Things to Take on the Trip

- Food to Bring:
 - Breakfast
 - Oatmeal
 - Lunch
 - Apples
 - Bagels
 - Dinner
 - Pasta
- Other Items:
 - Sunglasses
 - Tents
 - Sunscreen
 - Bugspray

Photos



Figure 1: from near the summit

E.2 Task B

E.2.1 Instructions

1. **Menu:** Open a new document
2. **Menu:** Menu: Save it under the name “scholarships.doc” in “My Documents\filesForTasks”
3. **Menu:** At the top of “scholarships.doc” enter the text “Scholarships Received”
4. **Toolbar:** Format that text as Heading 1
5. **Menu:** Insert a table with 3 columns and 5 rows
6. **Menu:** Open the document “scholarshipData.doc” from “My Documents\filesForTasks”
7. **Toolbar:** Copy the text “Year” from “scholarshipData.doc”
8. **Toolbar:** Paste it into the first column of the first row
9. **Toolbar:** Make the text bold
10. **Toolbar:** Copy the text “Scholarship Name”
11. **Toolbar:** Paste it into the second column of the first row
12. **Toolbar:** Make the text bold
13. **Toolbar:** Copy the text “Value”
14. **Toolbar:** Paste it into the third column of the first row
15. **Toolbar:** Make the text bold
16. **Menu:** merge cells 2 and 3 in column 1
17. Enter the text “1999” in column 1, row 2
18. Enter the text “2000” in column 1, row 3
19. **Menu:** add one row to the end of the table
20. **Menu:** merge the last two cells in column 1
21. Enter the text “2001” in the last row of column 1
22. Enter the text “UGF” in column 2, row 2
23. **Toolbar:** Copy the text “Conference Travel Award” from “scholarshipData.doc”
24. **Toolbar:** Paste it into column 2, row 3
25. **Toolbar:** Paste it again into column 2, row 4
26. Enter the text “NSERC” into column 2, row 5
27. **Toolbar:** Copy the text “Computer Science Scholarship” from “scholarshipData.doc”
28. **Toolbar:** Paste it into the column 2, row 6
29. Enter the values \$16,000, \$400, and \$700 into rows 2-4 of column 3
30. **Menu:** Split the next cell in column 3 into two rows (to have a value for year 1 and year 2)
31. Enter the values \$19,000 and \$21,000 into rows 5 and 6 of column 3
32. Enter the value \$1,500 in the final cell.

-
33. **Menu:** Save the file
 34. **Menu:** Distribute the rows evenly in the table
 35. **Toolbar:** Center the text in the year column horizontally
 36. **Menu:** Center the text of the entire table in the vertical direction
 37. **Menu:** Make the header and footer visible.
 38. Add the text “Joe Smith’s Resume” as a header
 39. **Toolbar:** Align the text in the header to the right
 40. **Menu:** Add a footnote next to the text “NSERC” that says “2-year scholarship”
 41. **Menu:** Add page numbers
 42. **Menu:** Save the file
 43. **Menu:** Close the “ScholarshipData.doc” file (but do not exit the program)
 44. **Menu:** Print the file “scholarships.doc”
 45. **Menu:** Close the file “scholarships.doc” (but do not exit the program)

E.2.2 Files Provided to Start the Task

scholarshipData.doc

Headings:

Year

Scholarship Name

Value

Scholarship names:

UGF

Conference Travel Award

NSERC

Computer Science Scholarship

E.2.3 Final Product as a Result of Following the Instructions

Joe Smith's Resume

Scholarships Received

Year	Scholarship Name	Value
1999	UGF	\$16,000
	Conference Travel Award	\$400
2000	Conference Travel Award	\$700
2001	NSERC ¹	\$19,000
		\$21,000
	Computer Science Scholarship	\$1,5000

¹ 2-year scholarship

Appendix F

Study One: Post Questionnaire

The design of Study One's post-treatment questionnaire was informed by the questionnaire used in McGrenere's *et al.*'s field evaluation, which compared adaptive and adaptable versions of MSWord [91, 92].

1a). Did you customize at all during the study?

Yes

No

If your answer to 1a) is “Yes”, please answer parts b) and c). Otherwise, go straight to question 4.

1b). Please indicate the extent to which you agree or disagree with the following statements:

<p>SD = Strongly Disagree D = Disagree N = Neutral A = Agree SA = Strongly Agree</p>

I customized to reduce the number of features that I had to access using the Full Interface. SD D N A SA

I customized to make my Personal Interface as small as possible while still being appropriate for my tasks. SD D N A SA

I customized because I thought it would help me complete my tasks more quickly. SD D N A SA

1c). Did you customize for any reason(s) other than the ones given above? If so, please list the additional reason(s).

For the following questions:

NoRec = Word Personal without system recommendations

WithRec = Word Personal with system recommendations

Question 2

2a) Did you customize with the WithRec version of Word Personal?

Yes

No

**If your answer to 2a) is “Yes” please answers parts b), c) and d).
Otherwise go straight to question 3.**

2b) Are there particular aspect(s) the system recommendations that you *liked*?

2c) Are there particular aspect(s) of the system recommendations that you *disliked*?

2d). Please indicate the extent to which you agree or disagree with the following statements:

SD = Strongly Disagree
D = Disagree
N = Neutral
A = Agree
SA = Strongly Agree

With the WithRec version of the system, SD D N A SA
I trusted the system to make good recommendations.

With the WithRec version of the system, it SD D N A SA
was easy to tell which features the system was recommending.

With the WithRec version of the system, SD D N A SA
system recommendations were appropriate for my tasks.

With the WithRec version of the system, I SD D N A SA
understood why the system made the recommendations it did.

Question 3

3a). Did you customize with both versions of Word Personal (NoRec and With-Rec)?

Yes No

If the answer to 3a) is “Yes” please answer parts b) and c). Otherwise go straight to question 4.

3b). If you could choose only one of the versions of Word Personal to continue using, which would it be?

NoRec WithRec
Why?

3c). There are a number of criteria listed below. Please select the version that would be your 1st choice according to each of the criterion. If you really cannot make a choice for a given criteria please select “Equal”.

Criteria	1st Choice		
This software is easy to use.	<input type="radio"/> NoRec	<input type="radio"/> WithRec	<input type="radio"/> Equal
I find it easy to decide which features to add to my Personal Interface.	<input type="radio"/> NoRec	<input type="radio"/> WithRec	<input type="radio"/> Equal
I find it easy to decide which features to delete from my Personal Interface	<input type="radio"/> NoRec	<input type="radio"/> WithRec	<input type="radio"/> Equal
After I customize, the contents of the menus and the toolbars match my needs.	<input type="radio"/> NoRec	<input type="radio"/> WithRec	<input type="radio"/> Equal
Customizing doesn't take very much time.	<input type="radio"/> NoRec	<input type="radio"/> WithRec	<input type="radio"/> Equal

4. Are there any other comments you would like to make about the systems or the study?

5. How did you find out about the study?

Thank you for your participation!

Appendix G

Study One: Sample Interview Questions

Usage of the Personal Interface versus the Full Interface:

Which interface did you spend most of your time in? Why?

Customizing in General:

Did you customize? Why/why not?

Adding/Deleting

Did you add any features? Why/why not?

Did you delete any features? Why/why not?

System Recommendations

With the _____ version of the interface, the system made customization recommendations. Were these helpful to you? Why/why not?

[If the participant customized using the Add/Delete screens:] What did you think about the way in which the recommendations were communicated? Did the colouring help you? Could you recommend a better way of presenting recommendations?

Did you click on the “Accept All” button? Why/why not? Are there any circumstances under which you would have considered automatically accepting all recommendations?

What did you think of the recommendations themselves? Were they appropriate? Why/why not?

Did you ever go directly to the “Get System Recommendations Screen”? Why/why not?

Rationale

Did you notice the “more information button?” Did you click on the button? Why/why not? Did you consider finding out more information about how the recommendations are generated? Why/why not?

[If the participant didn't access the rationale, show him/her what happens when the button is clicked.]

Did you look at the expected time savings? *[If not, show him/her the screen and ask her/him to take a minute to look at the information.]*

Did it motivate you to accept the suggestions? Was the number too small?

Did you look at the explanation of how the system generates the recommendations? Why/why not? *[If not, show him/her the screens and ask him/her to take a minute to look at the information.]*

Did/do you find the information useful? Why/why not? Would you have liked to have any additional information?

Questionnaire ranking criteria:

[If applicable, the participant is asked to go through each criterion where they had to rank the systems and give a short explanation for their choice.]

I understood why the system was making the specific recommendations.

This software is easy to use.

I find it easy to decide which features to add to my Personal Interface.

I find it easy to decide which features to delete from my Personal Interface.

After I customize, the contents of the menus and the toolbars match my needs.

Customizing doesn't take very much time.

Appendix H

Analysis of Study One's Preliminary Questionnaires

During participant recruiting for Study One, 75 people filled out the on-line preliminary questionnaire (see appendix A), which contained the Feature Profile Scale (i.e., the Feature Keen/Shy classification) developed by McGrenere and Moore [93]. The Feature Profile Scale asks participants to indicate the degree to which they agree with 12 twelve statements (on a scale of 1 to 5). These twelve statement are divided into three sub scales: i) three statements on how participants feel about having many functions in the interface (Number of functions), ii) six statements on how much they want to have a complete version of their interface (Completeness), and iii) 3 statements on how up-to-date they would like their interfaces to be (Up-to-dateness). The ratings for each statement are normalized to produce a score between 0 and 1, which are summed to produce an overall score in the range [0, 3]. The individual statements are reproduced here with permission from the designers:

- Number of functions:
 1. (NOT) I have a hard time finding the functions I need unless I use them regularly.
 2. (NOT) I get annoyed when I can't quickly find a function that I've used previously.
 3. (NOT) I get annoyed when I can't quickly figure out how to do something that I need to do.

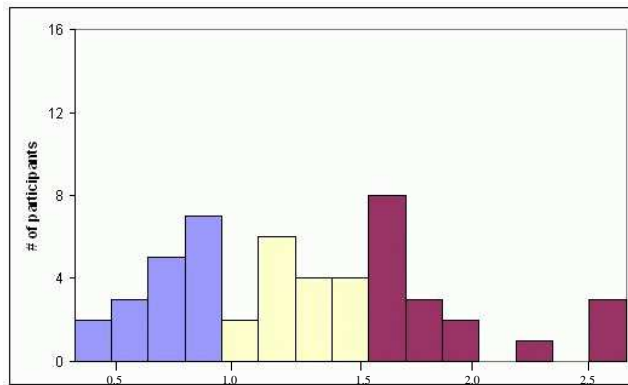
- Completeness
 1. Even though there are functions found in the menus/toolbars that I do not use, it is reassuring to me that they are there.

2. (NOT) I would prefer to have only the functions I use in the menus/-toolbars – the other functions could be “tucked” away in the event that I might need them at some point.
 3. I am happy to pay for an application that is “fully loaded” even if I dont use all the functions.
 4. (NOT) I would like a word processor that gave me only the functions that I use.
 5. It is important to me that I continually discover new functions.
 6. (NOT) Wading through unfamiliar functions can often be annoying/frustrating.
- Up-to-dateness
 1. If it were up to me I would upgrade MS Word only when new functions that I need are added.
 2. I want the latest version of MS Word as soon as it is available.
 3. I prefer to continue using the version of MS Word currently on my machine for as long as possible.

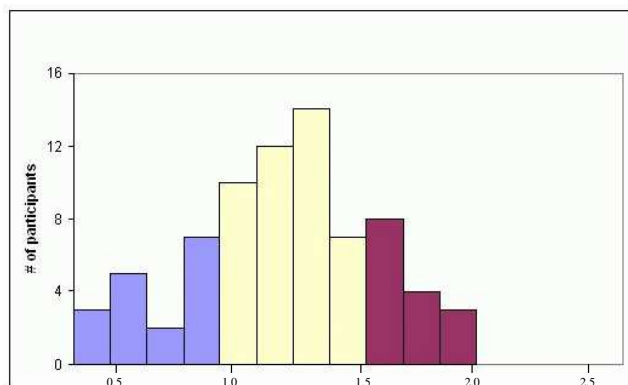
McGrenere and Moore classified users according to the following ranges: Shy [0, 0.92), Neutral [0.92, 1.58) and Keen [1.58, 3.00]. Using these ranges, figure H.1 shows the distributions obtained during the two studies (McGrenere and Moore's study and Study One). The main difference is the number of participants classified as Feature Neutrals. In Study One there was a much larger percentage of Feature-Neutral participants than in McGrenere and Moore's study.

We also the analyzed the questionnaires for reliability using Cronbach's α . Table H.1 presents the results for both McGrenere and Moore's study and Study One. The reliability of the Up-to-dateness sub scale decreased fairly substantially between the two studies. The remaining sub scales and the overall scale also decreased in reliability, but by smaller amounts. Decreases in reliability would have contributed to the increase in Feature Neutrals described above – the Feature Profile Scale classifies an individual as Feature Neutral if either they have little opinion on the state of their interfaces or their opinions are inconsistent.

The above results indicate that the classification ranges may need adjustment and/or the questionnaire may require further validation. In particular, the statements in the Up-to-dateness scale may need to be revisited. Towards the



(a) McGrenere and Moore's study [93]. (N=50)



(b) Study One. (N=75)

Figure H.1: The distribution of participants across the Feature Profile Scale in McGrenere and Moore's study [93] and in Study One.

Appendix H. Analysis of Study One's Preliminary Questionnaires

Sub scale	McGrenere and Moore's α	Study One's α
Functions	0.83	0.70
Completeness	0.76	0.64
Up-to-dateness	0.77	0.40
Overall	0.81	0.73

Table H.1: The reliability of the Feature Profile Scale using the data collected in McGrenere and Moore's study ($N = 50$), and the data collected during recruiting for Study One ($N = 75$).

end of recruiting for Study One, two participants provided feedback that they had difficulty completing parts of the questionnaire related to this scale. These participants said that they didn't have control over upgrades because they use group machines in shared laboratories.

Appendix I

Study Two: Call for Participation

Microsoft Word Study Participants Wanted

Pay: \$10/hour (study typically takes 2 - 3 hours)

You are invited to participate in an exciting word processing study being run in the Computer Science Department at the University of British Columbia! The study will take 2-3 hours and all participants will receive \$10/hour for participating.

We are looking for participants who:

- have used Microsoft Word 2003 at least once
- are highly fluent in English
- are 18 years of age or older
- have not participated in our previous Word study

What is involved?

Your involvement in this study will consist of one session lasting up to 3 hours. You will be asked to complete a number of short questionnaires and to use Microsoft Word to perform a number of tasks. We will also conduct an interview at the end of the study.

Interested in Participating?

If you are interested in participating we ask that you fill in a preliminary questionnaire. This will take no more than 10 minutes of your time. The preliminary questionnaire can be found at:

www.cs.ubc.ca/~bunt/study

If you have any questions or comments contact Andrea at bunt@cs.ubc.ca.

Appendix J

Study Two: Instructions

J.1 Instructions Provided at the Beginning of the Session

The goal of this experiment is to obtain information on how people use a new version of MSWord called MSWord Personal. I will provide with more details on this new version later in the study. To start the study I'd like you to complete a questionnaire designed to understand your previous Word experiences. Based on the results of the questionnaire, I will see if you fall into a category for which we still need participants. If so, we will continue with the second part of the study, which involves completing two tasks with the interface. I'll also ask you to complete another questionnaire and I'll interview you at the end of the session. If we do both parts of the study, the whole thing will last between 2-3 hours.²³ If not, I'll pay you for the first hour.

J.1.1 NAART Instructions

Before completing the questionnaire, I'd like you to read the words on this list. This is just a test of your knowledge of words. I'd like you to read them out loud one at a time going down the list by column. They get harder and harder as you go along. If you are not sure how to pronounce a word, just give it a try. Don't worry, most people don't know a lot of these words.

J.2 Instructions Provided Prior to the First Task

[The participant is given a brief demonstration of the two-interface model and the customization mechanism present in the first trial.]

²³In reality the study ended up lasting between 3 and 4 hours.

You will be asked to complete two tasks, each with a different version of the software and a different Personal Interface. Each Personal Interface will contain some, but not all of the features you need for the tasks.

For each task, you will be given an explicit list of instructions and a final product that following the instructions will produce. Consulting the final product will help you figure out how to accomplish the task. To get a sense of how you use the interface over time with the same task, you will be asked to repeat each task up to five times.

After you complete the first repetition of the task, I will enable the “Modify” button that you can use to customize your Personal Interface. You can customize as often as you like for the remainder of the task repetitions.

[If the first condition is the Rationale condition:]

I want to point out one additional feature of the customization mechanism. By clicking on this button, you can have the recommendations explained in terms of your personal information. We ask that you take a look at this explanation at some point if you customize during this task.

To start the first task, you will have the following Personal Interface, which contains some, but not all of the features you will need for the task. Before I give you your first task instructions, take a minute to look at what is in your Personal Interface.

[Participants are given a minute to look through the starting PI.]

To complete a task:

- Follow the instructions step-by-step to produce the final document.
- If a step involves invoking a feature from the menus or toolbars, the instructions will indicate whether to use a menu or toolbar to invoke the feature at the end of the step. If the step says **Menu**, you are to invoke a menu item to complete the step. If the step says **Toolbar** you are to invoke a Toolbar item. *[Show an example of each.]*
- Please follow the instructions as closely as possible. Since we are interested in menu and toolbar usage, the majority of the short-cut keys and the right-click menus have been disabled. You may know faster ways to accomplish the task, but for the purpose of this experiment please do follow the instructions, including the Menu/Toolbar specifications.
- If you are stuck on how to complete a step. Ask the experimenter who may be able to provide additional assistance. Under certain conditions

the experimenter may volunteer hints.

- When are you ready to start the task press the “Start Task” button. When you are done the instructions, press the “Done task” button.

J.3 Instructions Provided After All Repetitions of the First Task are Complete

For your second task, you will be using a different version of Word Personal.

[If the condition is the Rationale condition:]

I want to point out one additional feature of the customization mechanism. By clicking on this button, you can have the recommendations explained in terms of your personal information. We ask that you take a look at this explanation at some point if you customize during this task. *[Participant are shown where to access the rationale.]*

[If the condition is the No-Rationale condition:]

This time if you customize, you will not have access to the explanation of the recommendations. *[Participants are shown the No-Rationale interface.]*

To start this task, you will have the following Personal Interface. Before I give you the task instructions, take a minute to look at what is in your Personal Interface.

[Participants are given a minute to look through the starting PI.]

Appendix K

Study Two: Tasks

K.1 Task A

K.1.1 Instructions

1. Open the document “mountainDesc.doc” from “My Documents\filesForTasks” [Menu]
2. Open another document called “details.doc” from “My Document\files-ForTasks” [Menu]
3. Copy the text “Trip Summary” from “mountainDesc.doc” [Menu]
4. Paste it at the top of “details.doc” [Menu]
5. Save the document “details.doc” under the name “tripReport.doc” in “My Documents\filesForTasks” [Menu]
6. Copy the first two paragraphs from “mountainDesc.doc” [Menu]
7. Paste them under the heading “Trip Summary” in “tripReport.doc” [Menu]
8. Save “tripReport.doc” [Menu]
9. Close the file “mountainDesc.doc” [Menu]

All modifications will now be made to the file “tripReport.doc”

10. In the line below “Trip Summary” enter the text “details”
11. Change the size of “details” to 14 pt [Toolbar]
12. Change the font to “Century” [Toolbar]
13. Cut the paragraph at the bottom of the document [Menu]
14. Paste it after the text “Trip Summary” [Menu]

15. Save the document [**Menu**]
16. Make the list of food items to bring into a bulleted list (include “Food to Bring” in the list) [**Menu**]
17. Increase the indentation of the “breakfast” category [**Toolbar**]
18. Increase the indentation of the “oatmeal” twice [**Toolbar**]
19. Increase the indentation “lunch” [**Toolbar**]
20. Increase the indentation of both “apples” and “bagels” twice [**Toolbar**]
21. Increase the indentation of “dinner” [**Toolbar**]
22. Increase the indentation of “pasta” twice [**Toolbar**]
23. In the list of “other items”, decrease the indentation of “tents” once [**Toolbar**]
24. Decrease the “bugspray” once [**Toolbar**]
25. Save the document [**Menu**]
26. Cut the text describing the directions (from “The hike is a short drive..” to the end of the numbered list) [**Menu**]
27. Paste it under the two paragraphs in the “details” section [**Menu**]
28. Cut text “How to Get There” [**Menu**]
29. Paste it above the text describing the directions [**Menu**]
30. Change the size of “How to Get There” to 16 pt [**Toolbar**]
31. Make it bold [**Toolbar**]
32. Change the font to “century” [**Toolbar**]
33. Save the file [**Menu**]
34. Cut text “Things to take on the trip” [**Menu**]
35. Paste it above the food list [**Menu**]
36. Change the size of “Things to take on the trip” to 16 pt [**Toolbar**]
37. Change the font to century [**Toolbar**]

-
38. Make it bold [**Toolbar**]
 39. Save the document [**Menu**]
 40. Add a page break before “Things to take on the trip” [**Menu**]
 41. Change the size of “Food to Bring” to 14 pt [**Toolbar**]
 42. Change the size of “Other Items” to 14 pt [**Toolbar**]
 43. Change the size of “oatmeal” to 10 pt [**Toolbar**]
 44. Change the size of “apples” and “bagels” to 10 pt [**Toolbar**]
 45. Change the size of “pasta” to 10 pt [**Toolbar**]
 46. Save the document [**Menu**]
 47. Enter the text “Photos” at the bottom of the document
 48. Change the size of “Photos” to 16 pt [**Toolbar**]
 49. Make it bold [**Toolbar**]
 50. Save the document [**Menu**]
 51. After “Photos”, insert the photo “mountainPic” found in “My Documents\files-ForTask” [**Menu**]
 52. Insert a caption under the photo “Figure 1: from near the summit” [**Menu**]
 53. Insert page numbers [**Menu**]
 54. Check the spelling (and fix the spelling mistakes) [**Toolbar**]
 55. Justify the two paragraphs under “details” [**Toolbar**]
 56. See what the document would look like printed [**Menu**]
 57. Change the font of the “food” list and “other items” list to “times new roman” [**Toolbar**]
 58. Save the document [**Menu**]
 59. Print the document [**Menu**]
 60. Close the document (but do not exit the program) [**Menu**]

K.1.2 Files Provided to Start the Task

details.doc

Food to Bring:
Breakfast
Oatmeal
Lunch
Apples
Bagels
Dinner
Pasta

The hike is a short drive from Vancouver and easy to find.

Directions:

1. Take highway 99 to just past Pemberton
2. Turn right on owl creek road

How to Get There

Things to Take on the Trip

- Other Items:
 - Sunglasses
 - Tents
 - Sunscreen
 - Bugspray

Chris and I headed to Wedge Mount Lake on June 25 and 26, 2005, located near Whistler in the northern end of Garibaldi Provincial Park. Chris had been to the lake on several previous climbing trips while I had only been once to climb Wedge in 2001. Our goal for the weekend was to climb the north face. After climbing the east face on Saturday, low motivation and poor weather on Sunday led us to opt instead to climb the southeast ridge.

mountainDesc.doc

After a 9:00 am departure from Vancouver on Saturday morning, we started up the trailhead [780 m] at 12:00 pm for a 3:00 pm arrival at Wedge Mount Lake. After leaving most of our gear at the hut [1925 m], we left again at 3:30 pm headed around the western end of the lake and up snow ramps to the ridge [2240 m]. We scrambled up the east ridge of Wedge and reached its summit [2408 m] at 6:15 pm. Given the late hour, we opted not to climb up the opposite side of the ridge to the summit and instead descended to the lake to enjoy dinner in the last half hour of sunshine before sleeping near the hut under a starry sky.

Chris woke me at 5:30 am. Given our late day on the previous evening and the uncertain weather, we opted to climb the easier southeast ridge instead of the snow route we had intended on its north face. This meant we could leave rope, harnesses, crampons, pickets, etc. at the hut and enjoy a mellow day. We were hiking around the east side of the lake by 6:30 am. After alternating between snow and loose rock on the approach, we reached the ridge crest [2515 m] just before 9:00 am. From there we scrambled the ridge, crossing one snow patch, to reach the summit [2835 m] at 10:00 am. The descent was quick since much of the steeper sections below the ridge crest were on snow; Chris opted to glissade while Stephan preferred boot skiing. We returned to the hut at 12:30 pm, packed and left again at 1:00 pm, arriving at the car by 3:15 pm.

Trip Summary

K.1.3 Final Product as a Result of Following the Instructions

Trip Summary

Chris and I headed to Wedge Mount Lake on June 25 and 26, 2005, located near Whistler in the northern end of Garibaldi Provincial Park. Chris had been to the lake on several previous climbing trips while I had only been once to climb Wedge in 2001. Our goal for the weekend was to climb the north face. After climbing the east face on Saturday, low motivation and poor weather on Sunday led us to opt instead to climb the southeast ridge.

Details

After a 9:00 am departure from Vancouver on Saturday morning, we started up the trailhead [780 m] at 12:00 pm for a 3:00 pm arrival at Wedge Mount Lake. After leaving most of our gear at the hut [1925 m], we left again at 3:30 pm headed around the western end of the lake and up snow ramps to the ridge [2240 m]. We scrambled up the east ridge of Wedge and reached its summit [2408 m] at 6:15 pm. Given the late hour, we opted not to climb up the opposite side of the ridge to the summit and instead descended to the lake to enjoy dinner in the last half hour of sunshine before sleeping near the hut under a starry sky.

Chris woke me at 5:30 am. Given our late day on the previous evening and the uncertain weather, we opted to climb the easier southeast ridge instead of the snow route we had intended on its north face. This meant we could leave rope, harnesses, crampons, pickets, etc. at the hut and enjoy a mellow day. We were hiking around the east side of the lake by 6:30 am. After alternating between snow and loose rock on the approach, we reached the ridge crest [2515 m] just before 9:00 am. From there we scrambled the ridge, crossing one snow patch, to reach the summit [2835 m] at 10:00 am. The descent was quick since much of the steeper sections below the ridge crest were on snow; Chris opted to glissade while Stephan preferred boot skiing. We returned to the hut at 12:30 pm, packed and left again at 1:00 pm, arriving at the car by 3:15 pm.

How to Get There

The hike is a short drive from Vancouver and easy to find.

Directions:

1. Take highway 99 to just past Pemberton
2. Turn right on owl creek road

Things to Take on the Trip

- Food to Bring:
 - Breakfast
 - Oatmeal
 - Lunch
 - Apples
 - Bagels
 - Dinner
 - Pasta

- Other Items:
 - Sunglasses
 - Tents
 - Sunscreen
 - Bug spray

Photos



Figure 1: from near the summit

K.2 Task B

K.2.1 Instructions

1. Create a new document [Menu]
2. Save it under the name "scholarships.doc" in "My Documents\filesForTasks" [Menu]
3. At the top of "scholarships.doc" enter the text "Scholarships Received"
4. Format that text as Heading 1 [Toolbar]
5. Insert a table with 3 columns and 5 rows [Menu]
6. Open the document "scholarshipData.doc" from "My Document\files-ForTasks" [Menu]
7. Copy the text "Year" from "scholarshipData.doc" [Toolbar]
8. Paste in into the first column of the first row [Toolbar]
9. Make the text bold [Toolbar]
10. Copy the text "Scholarship Name" from "scholarshipData.doc" [Toolbar]
11. Paste it into the second column of the first row [Toolbar]
12. Make the text bold [Toolbar]
13. Copy the text "Value" [Toolbar]
14. Paste it into the third column of the first row [Toolbar]
15. Make the text bold [Toolbar]
16. Save the file [Menu]
17. Merge cells 2 and 3 in column 1 [Menu]
18. Enter the text "1999" and "2000" into rows 2 and 3 of column 1
19. Add one row to the end of the table [Menu]
20. Merge the last two cells in column 1 [Menu]
21. Enter the text "2001" in the last row of column 1

22. Save the document [**Menu**]
23. Enter the text “UGF” in column 2, row 2
24. Copy the text “Conference Travel Award” from “scholarshipData.doc”
[**Toolbar**]
25. Paste it into column 2, row 3 [**Toolbar**]
26. Paste it again into column 2, row 4 [**Toolbar**]
27. Enter the text “NSERC” into column 2, row 5
28. Copy the text “Computer Science Scholarship” from “scholarshipData.doc”
[**Toolbar**]
29. Paste it into the column 2, row 6 [**Toolbar**]
30. Save the file [**Menu**]
31. Enter the values \$16,000, \$400, and \$700 into rows 2-4 of column 3
32. Split the next cell in column 3 into two rows (to have a value for year 1
and year 2) [**Menu**]
33. Enter the values \$19,000 and \$21,000 into rows 5 and 6 of column 3
34. Enter the value \$1,500 in the final cell.
35. Save the file [**Menu**]
36. Distribute the rows evenly in the table [**Menu**]
37. Center the text in the year column [**Toolbar**]
38. Save the file [**Menu**]
39. Make the header and footer visible [**Menu**]
40. Add the text “Joe Smith’s Resume” as a header
41. Align the text in the header to the right [**Toolbar**]
42. Close the header and footer
43. Add a footnote next to the text “NSERC” that says “2-year scholarship”
[**Menu**]

-
44. Insert a new column to the right of the “Value” column [**Menu**]
 45. Save the file [**Menu**]
 46. Enter the text “Location” at the top of the new column
 47. Copy the text “UBC” from “scholarshipData.doc” [**Toolbar**]
 48. Paste it into column 4, row 2 [**Toolbar**]
 49. Copy the text “Hawaii” from “scholarshipData.doc” [**Toolbar**]
 50. Paste it into column 4, row 3 [**Toolbar**]
 51. Copy the text “Toronto” from “scholarshipData.doc” [**Toolbar**]
 52. Paste it into column 4, row 4 [**Toolbar**]
 53. Merge the next two cells into one [**Menu**]
 54. Enter the text “UBC” into this cell.
 55. Split the next cell into in column 4 into two rows [**Menu**]
 56. Enter the text “UBC” and “SFU” in these two new rows
 57. Save the file [**Menu**]
 58. Center the “Value” column [**Toolbar**]
 59. Add a footnote next to the text “Computer Science Scholarship” that says “Held in two locations” [**Menu**]
 60. Add page numbers (close the footnote panel first if it is open) [**Menu**]
 61. Print the file “scholarships.doc” [**Menu**]
 62. Save the file [**Menu**]
 63. Close the file “scholarships.doc” (but do not exit the program) [**Menu**]
 64. Close the “scholarshipData.doc” file (but do not exit the program) [**Menu**]

K.2.2 Files Provided to Start the Task

scholarshipData.doc

Headings:

Year

Scholarship Name

Value

Scholarship names:

UGF

Conference Travel Award

NSERC

Computer Science Scholarship

Locations:

UBC

Hawaii

Toronto

K.2.3 Final Product as a Result of Following the Instructions

Joe Smith's Resume

Scholarships Received

Year	Scholarship Name	Value	Location
1999	UGF	\$16,000	UBC
	Conference Travel Award	\$400	Hawaii
2000	Conference Travel Award	\$700	Toronto
2001	NSERC ¹	\$19,000	UBC
		\$21,000	
	Computer Science Scholarship ²	\$1,500	UBC SFU

¹ 2-year scholarship

² Held in two locations

Appendix L

Study Two: Post Questionnaire

1a). Did you customize at all during the study?

Yes

No

If your answer to 1a) is “Yes”, please answer parts b) and c). Otherwise, go straight to question 4.

1b). Please indicate the extent to which you agree or disagree with the following statements:

SD = Strongly Disagree
D = Disagree
N = Neutral
A = Agree
SA = Strongly Agree

I customized to reduce the number of features that I had to access using the Full Interface. SD D N A SA

I customized to make my Personal Interface as small as possible while still being appropriate for my tasks. SD D N A SA

I customized because I thought it would help me complete my tasks more quickly. SD D N A SA

1c). Did you customize for any reason(s) other than the ones given above? If so, please list the additional reason(s).

1d) Are there particular aspect(s) the system recommendations that you *liked*?

1e) Are there particular aspect(s) of the system recommendations that you *disliked*?

1f) Please explain how you think the system made its recommendations:

For the following questions:

NoExplanation = Word Personal without system recommendations

WithExplanation = Word Personal with system recommendations

Question 2

2a) Did you view the system's explanation of recommendations in the WithExplanation version? (circle one)

Yes

No

If your answer to 2a) is "Yes" please answers parts b) and c). Otherwise go straight to question 3.

2b) Are there particular aspect(s) the explanation that you liked?

2c) Are there particular aspect(s) of the explanation that you disliked?

Question 3

3a). Did you customize with both versions of Word Personal (NoExplanation and WithExplanation) and look at the explanation in the WithExplanation version?

- Yes No

If the answer to 3a) is “Yes” please answer parts b) and c). Otherwise go straight to question 4.

3b). If you could choose only one of the versions of Word Personal to continue using, which would it be?

- NoExplanation WithExplanation

Why?

3c). There are a number of criteria listed below. Please select the version of Word Personal that would be your 1st choice according to each of the criterion. If you really cannot make a choice for a given criteria please select “Equal”.

Criteria	1st Choice
I agreed with the system’s recommendations.	<input type="radio"/> NoExplanation <input type="radio"/> WithExplanation <input type="radio"/> Equal
I trusted the system to make good recommendations.	<input type="radio"/> NoExplanation <input type="radio"/> WithExplanation <input type="radio"/> Equal
I understood why the system made those <i>specific</i> recommendations to you.	<input type="radio"/> NoExplanation <input type="radio"/> WithExplanation <input type="radio"/> Equal
I understood why the system was making recommendations <i>in general</i>	<input type="radio"/> NoExplanation <input type="radio"/> WithExplanation <input type="radio"/> Equal
I could predict how the system would make recommendations in the future.	<input type="radio"/> NoExplanation <input type="radio"/> WithExplanation <input type="radio"/> Equal

4. Are there any other comments you would like to make about the systems or the study?

5. How did you find out about the study?

Thank you for your participation!

Appendix M

Study Two: Sample Interview Questions

Usage of the Personal Interface versus the Full Interface:

Which interface did you spend most of your time in? Why?

Customizing in General:

Did you customize? Why/why not?

Adding/Deleting:

Did you add any features? Why/why not?

Did you delete any features? Why/why not?

[If the information has not already been conveyed:] Would you ever want to delete features? Why/why not? Under what circumstances?

[If the information has not already been conveyed:] Did the design of the customization mechanism influence your decision not to delete any features?

Recommendations:

Did you make use of the system recommendations when you customized? Were the recommendations helpful to you? Why/why not?

[If the information has not already been conveyed:] Did you find the recommendations appropriate? Why/why not?

Willingness to Switch Between Interfaces:

Did you feel that any features were missing from the recommendations?

Would you prefer to use one interface exclusively? If so, which interface?

What if it made you faster overall to use a combination of the two interfaces? For example, you could spend the majority of your time working in the Personal Interface, and switch to the Full Interface to use features that you don't use very frequently.

Customization Methods:

[If the participant customized using the Add/Delete screens:] Did you add or delete features by selecting them as in normal usage? Did you follow the colouring in the menu and toolbars to guide your selections?

Did you click on the "Show Add/Delete" Recommendations?

Did you click on the "Accept All" button?

[If the information has not already been conveyed:] Why did you choose to customize in that manner? *[The question is tailored to the manner in which the participant customized.]*

Did you ever go directly to the "Get System Recommendations" Screen? Why/why Not?

Rationale:

In the ----- version of the system that you used, did you notice the explanation of the recommendations? Did you click on the "More" button? Why/why not?

[If the participant didn't look at rationale or the answer to the above question doesn't mention me asking him/her to look at the rationale:] Did you remember me asking you before you started the task to look at the information at some point?

[If the participant didn't look at rationale or did only because I asked him/her to:] Would you ever want to find out more information about how the recommendations are generated? Why/why not?

[If the information has not already been conveyed:] Are there any circumstances that would cause you to want to have the recommendations explained? *[Only if the participant's answers indicate some interest in the rationale]* What if the recommendations had been less appropriate? What if the system was automatically adding and deleting features to and from your Personal Interface without your input?

[If the participant didn't access the rationale, show him/her what happens when the button is clicked and ask him/her to look through the information.]

Did you look at the expected time savings? *[If the answer is no, show him/her the screen and ask him/her to take a minute to look at the information.]*

In general, is having a Personal Interface that would save you time something that would motivate you to accept recommendations? Why/why not? If not, is there something that would motivate you to accept recommendations?

Did this particular amount of time savings motivate you to accept recommendations? Why/why not?

[If the information has not already been conveyed] Would/did it motivate you to delete features? In general, would it make you more willing to delete features?

[If the participant indicates that the time was not motivating for following recommendations and/or deleting:] Is there an amount of time savings that would be large enough? Why/why not? Can you give me an estimate of what this amount might be?

Did you look at the explanation of how the system generates the recommendations? Why/why not? *[If the answer is no, show him/her the screens and ask him/her to take a minute to look at the information.]*

Did/do you find the information useful? Why/why not?

Which parts were the most useful? Why?

Which parts were the least useful? Why?

Would you have liked to have any additional information?

Questionnaire ranking criteria:

[If applicable, the participant is asked to go through each criterion where they had to rank the systems and give a short explanation for their choice.]

I agreed with the system's recommendations.

I trusted the system to make good recommendations.

I understood why the system was making the specific recommendations.

I understood why the system was making recommendations in general.

I could predict how the system would make recommendations.

Appendix N

Study Two: Further Performance Results

N.1 Effects of Task on Performance Measures

In Study Two, there were significant main effects of Task (A vs. B) on two of our performance measures. In particular, table N.1 shows that it took participants longer to complete Task B, both for Overall Performance and Task Performance. Table N.2 indicates that there was also a marginally significant interaction effect between Overall Performance and Task Order ($F(1, 12) = 4.386$, $p = 0.058$, partial $\eta^2 = 0.268$). Figure N.1 displays the interaction, indicating that the main effect was primarily driven by those who performed Task B first. Not surprisingly, table N.3 shows a similar interaction effect between Task Performance and Task Order ($F(1, 12) = 4.838$, $p = 0.048$, partial $\eta^2 = 0.287$).

N.2 Effects of Version on Performance

As we described in section 6.6.2, there were no significant main effects of Version (Rationale vs No-Rationale) on our two performance measures. Tables N.4 and N.5, however, shows significant and marginally-significant interactions.

Dependent Variable	Mean		SD		F(1,11)	p	η^2
	A	B	A	B			
Overall Performance (minutes)	38:00	44:07	10:33	11:22	15.324	0.002	0.561
Task Performance (minutes)	36:01	42:04	10:12	11:17	19.868	0.002	0.623

Table N.1: A summary of the ANOVA results with Task (A vs. B) as the primary within-subjects factor for the performance measures. ($N = 16$)

Appendix N. Study Two: Further Performance Results

Source	SS	df	MS	F	Sig.	partial η^2
Task	1080083	1	1080083	15.324	0.002	0.561
Task*TO	309095	1	309095	4.386	0.058	0.268
Task*VO	44925	1	44925	0.637	0.440	0.050
Task*VO*TO	37060	1	37060	0.526	0.482	0.042
Error	845769	12	70481			

Table N.2: The univariate repeated-measures ANOVA for Overall Performance with Task as the within-subjects factor. TO = Task Order and VO = Version Order (N = 16)

Source	SS	df	MS	F	Sig.	partial η^2
Task	1052701	1	1052701	19.868	0.001	0.623
Task*TO	256328	1	256328	4.838	0.048	0.287
Task*VO	42778	1	42778	0.807	0.387	0.063
Task*VO*TO	2628	1	2628	0.500	0.828	0.004
Error	635813	12	52984			

Table N.3: The univariate repeated-measures ANOVA for Task Performance with Task as the within-subjects factor. TO = Task Order and VO = Version Order (N = 16)

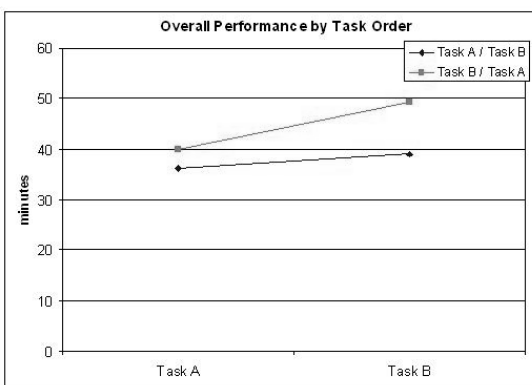


Figure N.1: The interaction between Overall Performance and Task Order. (N=16)

Source	SS	df	MS	F	Sig.	partial η^2
Version	47829	1	47829	0.633	0.443	0.054
Version*TO	56350	1	56350	0.746	0.406	0.064
Version*VO	250410	1	250410	3.315	0.096	0.223
Version*VO*TO	930755	1	930755	12.321	0.005	0.528
Error	830969	11	75543			

Table N.4: The univariate repeated-measures ANOVA for Overall Performance, with Version as the within-subjects factor. TO = Task Order and VO = Version Order (N = 15)

Source	SS	df	MS	F	Sig.	partial η^2
Version	4739.3	1	4739.3	0.083	0.779	0.007
Version*TO	47656.2	1	47656.2	0.831	0.382	0.070
Version*VO	217937.6	1	217937.6	3.800	0.077	0.257
Version*VO*TO	933488.3	1	933488.3	16.278	0.002	0.597
Error	630825.1	11	57347.7			

Table N.5: The univariate repeated-measures ANOVA for Task Performance, with Version as the within-subjects factor. TO = Task Order and VO = Version Order (N = 15)

There were additional significant and marginally between-subjects effects for both Task Order and Version Order for both dependent measures. Because of the differences in tasks described above, we did not try to interpret these interactions and order effects. We leave understanding how the provision of rationale could impact performance for as an area for future work.

Appendix O

UBC Research Ethics Board Certificates

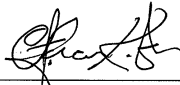
This appendix includes all Certificates of Approval for the research conducted in this thesis administered by the UBC Research Ethics Board.

Appendix O. UBC Research Ethics Board Certificates



The University of British Columbia
Office of Research Services and Administration
Behavioural Research Ethics Board


Certificate of Approval

PRINCIPAL INVESTIGATOR Conati, C.		DEPARTMENT Computer Science	NUMBER B03-0144
INSTITUTION(S) WHERE RESEARCH WILL BE CARRIED OUT UBC Campus ,			
CO-INVESTIGATORS: Bunt, Andrea, Science; Findlater, Leah, Computer Science; McGenere, Joanna, Computer Science			
SPONSORING AGENCIES Natural Science Engineering Research Council			
TITLE: Adaptive and Adaptable Information Technology			
APPROVAL DATE 03-04-03 <small>(y/m/d)</small>	TERM (YEARS) 1	AMENDMENT: July 30, 2003, Co-Investigator	AMENDMENT APPROVED: AUG 13 2003
<p>CERTIFICATION:</p> <p>The protocol describing the above-named project has been reviewed by the Committee and the experimental procedures were found to be acceptable on ethical grounds for research involving human subjects.</p> <p style="text-align: center;"></p> <p style="text-align: center;"><i>Approval of the Behavioural Research Ethics Board by one of the following:</i> Dr. James Frankish, Chair, Dr. Cay Holbrook, Associate Chair, Dr. Joe Belanger, Associate Chair</p> <p>This Certificate of Approval is valid for the above term provided there is no change in the experimental procedures</p>			



The University of British Columbia
Office of Research Services and Administration
Behavioural Research Ethics Board

Certificate of Approval

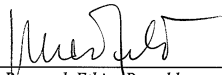
PRINCIPAL INVESTIGATOR McGrener, J	DEPARTMENT Computer Science	NUMBER B03-0144
INSTITUTION(S) WHERE RESEARCH WILL BE CARRIED OUT UBC Campus ,		
CO-INVESTIGATORS: Bunt, Andrea, Science; Conati, Cristina, Computer Science; Findlater, Leah, Computer Science		
SPONSORING AGENCIES Natural Science Engineering Research Council		
TITLE: Adaptive and Adaptable Information Technology		
APPROVAL RENEWED DATE OCT - 4 2004	TERM (YEARS) 1	
CERTIFICATION: <p>The protocol describing the above-named project has been reviewed by the Committee and the experimental procedures were found to be acceptable on ethical grounds for research involving human subjects.</p>  <p><i>Approval of the Behavioural Research Ethics Board by one of the following:</i> Dr. James Frankish, Chair, Dr. Cay Holbrook, Associate Chair, Dr. Susan Rowley, Associate Chair Dr. Anita Hubley, Associate Chair</p> <p>This Certificate of Approval is valid for the above term provided there is no change in the experimental procedures</p>		

Appendix O. UBC Research Ethics Board Certificates



The University of British Columbia
Office of Research Services and Administration
Behavioural Research Ethics Board


Certificate of Approval

PRINCIPAL INVESTIGATOR McGrener, J		DEPARTMENT Computer Science	NUMBER B03-0144
INSTITUTION(S) WHERE RESEARCH WILL BE CARRIED OUT UBC Campus ,			
CO-INVESTIGATORS: Bunt, Andrea, Science; Conati, Cristina, Computer Science; Findlater, Leah, Computer Science			
SPONSORING AGENCIES Natural Science Engineering Research Council			
TITLE: Adaptive and Adaptable Information Technology			
APPROVAL RENEWED DATE OCT 13 2005	TERM (YEARS) 1	AMENDMENT: Oct. 4, 2005, Recruitment method / Procedures / recruitment letter / Consent form	AMENDMENT APPROVED: OCT 13 2005
<p>CERTIFICATION:</p> <p>The protocol describing the above-named project has been reviewed by the Committee and the experimental procedures were found to be acceptable on ethical grounds for research involving human subjects.</p> <p style="text-align: center;"></p> <p style="text-align: center;"><i>Approval of the Behavioural Research Ethics Board by one of the following:</i> Dr. Peter Suedfeld, Chair, Dr. Susan Rowley, Associate Chair</p> <p>This Certificate of Approval is valid for the above term provided there is no change in the experimental procedures</p>			



The University of British Columbia
Office of Research Services and Administration
Behavioural Research Ethics Board

Certificate of Approval

<small>PRINCIPAL INVESTIGATOR</small> McGreener, J	<small>DEPARTMENT</small> Computer Science	<small>NUMBER</small> B03-0144
<small>INSTITUTION(S) WHERE RESEARCH WILL BE CARRIED OUT</small> UBC Campus ,		
<small>CO-INVESTIGATORS:</small> Bunt, Andrea, Science; Conati, Cristina, Computer Science; Findlater, Leah, Computer Science		
<small>SPONSORING AGENCIES</small> Natural Science Engineering Research Council		
<small>TITLE:</small> Adaptive and Adaptable Information Technology		
<small>APPROVAL DATE</small> 05-10-13 <small>(yr/mo/day)</small>	<small>TERM (YEARS)</small> 1	<small>AMENDMENT:</small> Jan. 30, 2006, Advertisement / Consent form / Interviews / Questionnaires
		<small>AMENDMENT APPROVED:</small> MAR 23 2006
<small>CERTIFICATION:</small> The request for continuing review of an amendment to the above-named project has been reviewed and the procedures were found to be acceptable on ethical grounds for research involving human subjects. <div style="text-align: center;"> _____ <i>Approved on behalf of the Behavioural Research Ethics Board by one of the following:</i> Dr. Peter Suedfeld, Chair, Dr. Susan Rowley, Associate Chair Dr. Jim Rupert, Associate Chair Dr. Arminee Kazanjian, Associate Chair</div> This Certificate of Approval is valid for the above term provided there is no change in the experimental procedures		



The University of British Columbia
Office of Research Services and Administration
Behavioural Research Ethics Board

Certificate of Approval

PRINCIPAL INVESTIGATOR McGrener, J	DEPARTMENT Computer Science	NUMBER B03-0144
INSTITUTION(S) WHERE RESEARCH WILL BE CARRIED OUT UBC Campus ,		
CO-INVESTIGATORS: Bunt, Andrea, Science; Conati, Cristina, Computer Science; Findlater, Leah, Computer Science		
SPONSORING AGENCIES Natural Science Engineering Research Council		
TITLE: Adaptive and Adaptable Information Technology		
APPROVAL RENEWED DATE OCT 26 2006	TERM (YEARS) 1	
CERTIFICATION: The request for continuing review of the above-named project has been reviewed and the procedures were found to be acceptable on ethical grounds for research involving human subjects. <i>mg</i> <hr/> Approved on behalf of the Behavioural Research Ethics Board by one of the following: Dr. Peter Suedfeld, Chair, Dr. Jim Rupert, Associate Chair Dr. Arminee Kazanjian, Associate Chair Dr. M. Judith Lynam, Associate Chair This Certificate of Approval is valid for the above term provided there is no change in the experimental procedures		