PRAGMATIC USER MODELLING FOR COMPLEX COMMERCIAL SYSTEMS

BY

LINDA NOREEN STRACHAN

A Thesis
Submitted to the Faculty of Graduate Studies
in partial fulfillment of the requirements of the degree of

Doctor of Philosophy

Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, Canada

0-612-45139-9

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION PAGE

Pragmatic User Modelling for Complex Commercial Systems

BY

Linda Noreen Strachan

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirements of the degree

of

Doctor of Philosophy

ABSTRACT

While user modelling has become a mature field, with demonstrable research systems of great power, comparatively little progress has been made in the development of user modelling components for commercial software systems. The development of minimalist user modelling components, which are simplified to provide "just enough" assistance to a user through a pragmatic adaptive user interface, is seen by many as an important step toward this goal.

This thesis describes the development, implementation, and empirical evaluation of a minimalist user modelling component for the Tax and Investment Management Strategizer (TIMS), a complex commercial software system for financial management. The experimental results demonstrate that a minimalist user modelling component improves the subjective measure of user satisfaction. Important issues and considerations for the development of user modelling components for commercial software systems are also discussed.

Dedicated to the memories of a dear friend,

Robin Jacqueline Crysdale
(1958-1997)

and my father,

Robert James Strachan
(1936-1997).

ACKNOWLEDGEMENTS

It does not matter how slowly you go so long as you do not stop. -- Confucius

First of all, I would like to thank my academic supervisor, Mark Evans, who has contributed to all aspects of this endeavour. He, together with David Scuse, introduced me to the challenging discipline of Artificial Intelligence and motivated me to conduct research with the highest possible standards. Mark's dedication to work, ability to focus, and quick intellect have always been an amazement to me. David Scuse initially triggered my interest in this area and has always provided me with careful and considerate guidance. His talent for remembering the exact reference in his library is renown. I am deeply indebted to them both for their patience, understanding, and friendship. I am also indebted to Jim Forest and Michael McTear who served as examiners of this thesis. Jim Forest deserves special mention for his expert theoretical and methodological input into the empirical study's content, design, and implementation. He spent a long summer providing me with a *personalized* Research Methodologies class!

During the course of my Ph.D. research, I have met and benefited from, either in person or through electronic communication, many other User Model(l)ing researchers. In particular, the program committees of the Doctoral Consortiums for the UM'94, UM'96 and UM'97 conferences are gratefully acknowledged for their valuable comments and guidance of my work throughout this project.

This research has allowed me to become involved with an exceptional organization, Emerging Information Systems (EISI). Without the underlying commercial system, TIMS, and access to the TIMS user base this research would not have been possible. EISI also provided financial support during the critical final stages of the thesis writing. Special thanks to Mark Evans, Geoff Crysdale, Darren Miller, and Paul Thiessen for taking the risk and becoming entrepreneurs. Both current and former EISI employees have been incredibly supportive and I thank them all. Murray Sneesby deserves special mention for enduring my constant design changes and for implementing the adaptive system, Gerald Adams for providing graphic design input to the project, Gwen Falkingham for ensuring that I didn't die from malnutrition and for listening, and Geoff Crysdale for being one of the best friends that I have ever had.

Graduate life is full of ups and downs. Over the years I have had the opportunity to meet some terrific people who shared these experiences with me. Of the many graduate students and staff that I consider to be my friends, several individuals deserve special mention. John Anderson deserves special recognition for his helpful comments and suggestions in reviewing several drafts of this thesis and in co-authoring several papers. He has provided many light moments and made the experience that much more tolerable. Ken Hotz has provided thesis guidance from the trenches and has proven to be a true friend. Tom Dubinski, Myrna Donald, Al Marshall, Gilbert Detillieux, and Gord Boyer encouraged junk food runs and much laughter. Lynne Romuld provided me with stress relief through running, long chats, and wine drinking (sometimes concurrently). I am also very appreciative of the opportunity given to me by Ralph Stanton and Peter King, the former and current department heads of Computer Science respectively, to teach undergraduate courses during my Ph.D. studies. Without this opportunity I may not have discovered that I loved to teach!

While graduate school can be an overwhelming experience, good friends and family are more important. The following individuals have always helped me to keep my life in context: Shelley McEvoy, Liz McAnanama, Sharon Evans and the Evans gang, Edina McDonald, the "Women of the WRC", my Marathon training group, the Loadman/Fultz family, and the "Stitch" group. Special thanks to Angela Reimer for providing feedback on the writing and great coffee.

I would like to thank my husband, Jim Hale, who has given me support and encouragement throughout my graduate career. And, to the most terrific daughter in the world, Samantha Hale, who has provided much needed distractions and a reminder that work is not the most important thing in life, thanks!

Finally, thanks to the Strachan, Hale, and Crysdale families for their love, understanding, endless patience and encouragement when it was most required. Yes, it is finally done!

I am certain that there are many other individuals that I have failed to mention. I have appreciated their encouragement and support throughout this effort.

TABLE OF CONTENTS

LIST OF FIGURES

List of Tables

LIST OF APPENDICES

CHAPTER I

INTRODUCTION

The reasonable man adapts himself to the world: the unreasonable one persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable man.

(George Bernard Shaw 1856-1950)

1.0 The Problem

Modern software systems are complex: they often support a wide variety of tasks and diverse groups of users who have widely differing problems and needs. Moreover, there is generally a trade-off between the power of a system and its ease-of-use. This situation has led to an increased focus on the role of the user in commercial software development and an increasing research focus on user modelling[1] and user-adapted interaction.

A user model is an explicit representation of properties of a particular user (Jameson *et al.*, 1997). The goal of user modelling as a research area is to create systems that are adaptive to an individual's needs, abilities, and preferences, and are responsive to the ongoing evolution of these traits. User modelling techniques have been developed and evaluated by researchers from a number of backgrounds, including artificial intelligence, education, psychology, linguistics, human-computer interaction, and information science.

[1] The term *modelling* is one with two common spellings. This work will employ the British spelling of *modelling* consistently. Exceptions are made for both quoted material and bibliographic titles.

In recent years, user modelling has matured as a research field. Current research systems show great promise in their ability to deliver a wide range of adaptability through user modelling techniques such as stereotyping and the separation of user knowledge into a user model database. In spite of the demonstrated capabilities of research systems, comparatively little progress has been made in supporting individualized interactions in commercial software systems. This lack of progress and a lack of emphasis on empirical studies are important concerns in modern user modelling research (McTear, 1993).

Still, some limited commercial successes in specific areas are becoming evident. The first prominent example is Microsoft[2] Office Assistant 1.0, a component of the Office 97[3] suite of products. The Office Assistant can answer questions, supply office tips, and provide Help on a variety of Office features. An animated icon provides information that is relevant to the task that the user is performing and monitors the current task to make suggestions about how to use the features of the program. An interesting feature of the Office Assistant is that the user can select from a range of choices that differ in their helpfulness levels and appearances. It is significant that a major software producer has chosen to bundle this technology with one of its most visible products (Kay & Fischer, 1997).

Intelligent agents (or personal assistants) are also a highly touted recent development in software because of the easy access to individuals and information through the World Wide Web, and the difficulty of dealing with the volume of this information (Maes, 1995; Norman, 1994). One company that is marketing software agents that actively assist the user in such situations is Firefly, Inc. Their Firefly passport[4] provides protected user profile information to participating web sites (Firefly, 1998). Concerns about the ease of gathering a large amount of information about individuals and the lack of protection for this information have led various groups to recognize the need for standards that will define the rules that govern the exchange of personal information. The World Wide Web Consortium (W3C) has instituted a Platform for Privacy Preference Project (P3P[5]) in

[2] Microsoft is a registered trademark of Microsoft Corporation.
[3] Microsoft Office 97 is a copyright of Microsoft Corporation.
[4] Firefly and Firefly passport are copyrights of Firefly, Inc.
[5] P3P is a trademark of World Wide Web Consortium.

order to provide a specification for web sites. This specification would inform web site developers of their responsibility to inform the user about the site's privacy practices and would allow the user to control his or her own preference information (P3P, 1998).

Personalized web search services and web site appearance tailoring are also becoming common. A good example of a customized web search engine is My Yahoo![6] (My Yahoo!, 1998). By setting up a user profile, individuals can personalize the site, tailor information such as customized news and sports team scores, and modify the site's appearance to suit their personal preferences. Advertisements can also be customized as the web becomes further commercialized. This will probably be one of the most lucrative applications of user modelling (or profiling) in this area. This customized advertising is often referred to as "one-to-one marketing". Another good example is Microsoft's default start page for their web browsers[7]. Many web sites such as this one give users the option to create simple user profiles that tailor their interactions.

Cookies are another mechanism used by web page developers to customize a user's interaction with a web page. Cookies are small text files containing details of a visit to a web page that are written to a machine's hard drive for later retrieval. Cookies can be used for a variety of tasks including: saving user preferences; remembering the pages that the user has visited, and when; greeting site visitors by name; notifying the visitor of changes made since his or her last visit; and storing data as the user navigates from one page to another (Goodman, 1996). More sophisticated sites make additional inferences that are based on information requests or other related user profiles, which are sometimes referred to as "affinity groups". Student modelling is also represented on the web, and a good example of this is a WWW-based LISP course that adapts to the student (Weber & Specht, 1997).

The academic community is certainly aware of these trends. The two major conferences that focus on user adapted interactions, the *International Conference on User Modeling* (UM) and the *International Conference on Intelligent User Interfaces* (IUI), have recently included forums that discuss how to translate the research developments into

[6] My Yahoo! Is a copyright of Yahoo, Inc. URL: http: //my.yahoo.com/

commercial practice. This issue was addressed in the workshops that were organized at two recent international conferences on user modelling (UM '96 and UM '97): *The Commercial Potential of User Modelling* (1996) and *User Models in the Real World* (1997). There are no signs that the academic conferences in this area feel that this issue has been resolved. Indeed, the overall theme of the IUI conference in 1999 will be the transfer of new scientific and technological advances to real world applications (IUI, 1999).

While current research momentum appears to be moving in the right direction, particularly so in the case of web-based applications, there is still a lack of commercial deployment in user modelling systems. Part of the reason for this is that there are important considerations in commercial software systems that are all but ignored in research systems. The most obvious consideration is the performance overhead that must be incurred when a user modelling system is employed. It is also important to consider the amounts of time and money that are involved in including a user modelling component, compared to the often unproven advantages that the added expense may bring. Less obvious but equally important factors include the design changes that may be necessary for the inclusion of a user modelling component, as well as the very real costs of training and support in an environment where a system's interaction is not necessarily consistent between one user and the next.

1.1 Pragmatic User Modelling

> ***prag·mat·ic***
> *2 : relating to matters of fact or practical affairs often to the exclusion of intellectual or artistic matters : practical as opposed to idealistic.*[8]

Despite the many details of design, implementation, maintenance, and support overhead, there is a much broader reason for the lack of commercial user modelling systems that has been speculated upon in the literature. Many of the approaches that are embodied in research systems are too complex or impractical for use in commercial software systems (Kobsa, 1993; McTear, 1993; Oppermann, 1994; McCalla *et al.*, 1996). The identification of this shortcoming has prompted some researchers to develop *pragmatic*

[7] URL: http://home.microsoft.com/
[8] WWWebster Dictionary Copyright © 1996 Merriam-Webster, Inc.

user modelling architectures and techniques that have been simplified from theoretically motivated approaches. As McTear has noted, "In some cases a theoretically motivated technique turns out to be inefficient, computationally expensive, or even infeasible, and some simpler and more cost-effective method may be more appropriate" (1996). From a commercial standpoint, this "good enough" user modelling involves the inclusion of a *minimalist* user modelling component that has some of the major advantages that have been demonstrated by large research systems, with minimal cost and commercial disruption. According to Totterdell and Rautenbach, "It is sometimes better to be dumb and fast than intelligent and slow" (1990, p. 69).

It is anticipated that minimalist approaches to user modelling will become more popular as the practical applications of user modelling receive greater consideration. McTear (1993) cites several systems that are designed for information filtering and adaptive hypertext documents, which employ simplified user modelling techniques that provide "good enough" user modelling in their respective domains. In a recent workshop report (Malinowski, 1996), the cost of building adaptive components for real-world systems was identified as a universal concern, and several of the contributed papers involve simplified techniques for use in commercial environments.

Recently, McCalla *et al.* (1996) employed a "slightly intelligent" design that they called *analogical user modelling* for document retrieval that was relevant to an individual. Here, the user model was not complete, but it captured just enough information to allow the relevant analogies to be found. The user was then required to guide and catch the system when it failed. A proof-of-concept system was empirically tested and was shown to be effective. McCalla *et al.* argued that the user and the system could work together symbiotically, and that, "minimalist approaches may be a pre-requisite for the successful application of AI techniques both inside and outside the domain of user modelling" (p. 19). The role of Artificial Intelligence (AI) techniques and the impact of these techniques on adaptive system development will be discussed in the next two chapters.

While the application of minimalist user modelling techniques to commercial systems shows promise, important questions remain. Many involve the efficacy and cost-benefit ratio of a minimalist user modelling component. These questions include the following:

- Can a relatively small number of user-specific adaptations help to support the user in navigating a complex computer application?
- Can these adaptations be employed practically in a commercial system?
- How can the sophistication and the cost of a user modelling system be balanced with the practical considerations of commercial software development?

There are also many practical issues, such as user involvement and management support, that must be addressed in supporting user modelling in ongoing commercial software development.

1.2 Thesis Overview

This thesis describes the design, implementation, and empirical evaluation of a minimalist user modelling component for the Tax and Investment Management Strategizer, or TIMS©[9], a complex commercial financial management software system. The addition of this component is intended to address the needs of novice users while minimizing its impact on expert users. This is achieved by reducing a novice's perception of the complexity of the system through the use of constraints and by exposing some of the system's underlying processing. A careful balance is required in order to support the novice users adequately, while trying to avoid irritating expert users with extraneous interactions.

This thesis is concerned with the whole development cycle of an adaptive system. It includes many subject areas, which range from informal task analysis, design, and implementation, to the evaluation of the final adaptive system. The pragmatic user modelling component that is described is designed to address many of the concerns that system developers have with applying user modelling in commercial systems. While

[9]TIMS is a copyright of Emerging Information Systems, Inc.

formal guidelines that support developers in the selection of user modelling techniques are not outlined, the experiences that are recounted will help to contribute to a general understanding of successful techniques that will support the introduction of user modelling to mainstream computing (McTear, 1996).

Chapter Two is a discussion of the general issues that surround Human Computer Interaction (HCI) research and user interface design and implementation. The purpose of this chapter is to make the reader aware of the pertinent HCI issues that have had an impact on the design and implementation of the adaptive system.

Chapter Three describes six classic modelling systems that illustrate the concepts that are present in the existing adaptive systems literature. The general issues within this area are examined in order to provide the background for an understanding of the origins of the various user modelling techniques that are described in the context of this research.

Chapter Four presents an analysis of the domain. The target domain for the system is a financial management software system, the Tax and Investment Management Strategizer (TIMS). TIMS is briefly described in terms of its functionality in order to provide a context for the discussions of the various adaptations. Two major functions, the strategies and the intelligent Assistants, were incorporated in the original TIMS system in order to assist novice users. These strategies and Assistant components are examined in detail because of the research's focus on the novice users and the attempt to control the complexity of the underlying system.

Chapter Five is a description of the functionality of the user modelling component and a description of the resulting adaptive system, Personalized TIMS (P-TIMS). An overview of the implementation is given in terms of how the interface works and then in terms of how the actual implementation was done (the system architecture, knowledge representation, etc.). For the sake of clarity of presentation, the presentation of P-TIMS is divided into four parts: the design and the three implementation stages.

This researcher arrived at the design basis for the P-TIMS system, the pragmatic or minimalist approach to user modelling, after considering several different intelligent interface research areas – primarily adaptive user interfaces and adaptive help systems. After describing how these areas motivated the decision to use a minimalist approach, this thesis goes on to provide a discussion of how the research team realized this approach in the design. This design provides practical adaptations with minimal disruption to the user, and it provides the user with a sense of control.

Chapter Six contains a discussion of the important issue of user control, a description of its relationship to the implementation, and a review of its effect on user satisfaction and acceptance of a system. The description and the results of an empirical study then identify the impact of the minimalist user modelling component on user satisfaction levels. This human factor experiment was an evaluation of the adaptive system in comparison with a non-adaptive variant of the same system. The final results suggest some specific application guidelines and supply empirical evidence that supports user modelling in this context.

Finally, Chapter Seven describes, from the standpoint of this research, some of the issues that are involved with applying user modelling to commercial systems. Some overall conclusions and topics for future research complete this chapter.

1.2.1 Contributions of the Thesis

The contributions of this thesis fall into two major categories. The most significant of these is the development and implementation of a minimalist user modelling component for a commercial software system. The model focused on taking validated techniques from user modelling research and simplifying them as much as possible in order to create a suitable architecture. This architecture was used to implement a practical module in a commercial software system that would support individualized interactions. Key issues for the architecture were practicality, increasing and decreasing support, user control over the adaptations, user acceptance, and the focus on the novice user.

The second category is the rigorous testing of the implemented system with the participation of actual users. The addition of this empirical study to the user modelling literature provides additional support for adaptive systems by providing statistically significant results of an improvement in user satisfaction. This evaluation addresses a definite need. Actual user involvement seems to be missing from the systems in the current literature; some of the adaptations that have been implemented are supported by research in cognitive science or psychology, and others are from the developers' own intuitions. This was highlighted at an Adaptive User Interfaces workshop where one of the overall conclusions was: "Furthermore, aspects of user acceptance and evaluation of adaptation, which have been neglected in the past, need far more attention." (Dieterich *et al.*, 1993).

The overall goal of this research is the support of the viability of commercial user modelling by the construction of a successful system. Although this does not guarantee the success of commercially viable adaptive systems, it does contribute to the acceptance of the concept.

CHAPTER 2

ISSUES IN HUMAN-COMPUTER INTERACTION

2.0 Introduction

The field of Human-Computer Interaction (HCI) encompasses the analysis, design, implementation, and evaluation of interactive computing. Researchers in HCI face the challenge of empowering users of computing technology to perform the tasks that they need to accomplish. This chapter provides a very brief overview of some sub-areas of HCI that place the following chapters in the proper context and provide enough background material to explain the motivation behind the decision to develop an adaptive system.

2.1 Knowledge-based Systems

Knowledge-based systems are programs in which domain knowledge is explicit and separate from the program's other knowledge. They create solutions by applying human knowledge in a specific area of expertise. The application of Artificial Intelligence techniques has allowed these types of systems to solve problems that conventional decision support programs could not handle.

Early expectations of Artificial Intelligence and expert systems were grandiose and many people were threatened by the terms themselves. During the past decade the hype has faded and, "The mysticism surrounding early expert systems have given way to recent successes in expert systems as problem solving tools" (Awad, 1996). This proven capability of addressing complex problems in combination with the growing use of shells and the development of hybrid (combining knowledge-based and conventional programming techniques) systems have established knowledge-based systems as relatively commonplace and successful in certain domains (Awad, 1996). More and more organizations have come to recognize the potential of this developing technology. The use of the term "expert systems" has created some overblown expectations; this thesis will continue the trend toward using the term "knowledge-based systems". This term seems more appropriate, given that most existing systems only embody traces of expert system technology.

Knowledge-based system development has experienced a tremendous growth rate in recent times. In the early 1970s only a handful of research systems were built. Most development took place on powerful workstations, using languages such as LISP and Prolog. During the 1980s, the technology was increasingly accepted by industry. New hardware and software advances, such as PCs and easy-to-use expert system shells, allowed individuals from other disciplines to develop knowledge-based systems. Available estimates of the number of developed expert systems over the last decade range from 50 per year (in 1985) to over 12,000 per year (in 1992) (Durkin, 1994). These numbers may not reflect the full impact of the technology because many recent systems use AI techniques but are not labelled as expert systems or knowledge-based systems.

A new problem emerges as these knowledge-based systems move away from research settings and their restricted environments, toward real-world practical applications. The overall complexity of these systems increases as the functionality expands to provide more comprehensive tools and more varied access to the valuable knowledge that is stored in the system. This increased complexity lessens the greatest potential benefit of knowledge-based decision making – the distribution of expertise (Hayes-Roth *et al.*, 1983). The use of these systems typically allows scarce and costly expertise to be distributed in a cost-effective manner to the less-experienced individuals of an organization. There is a risk that the complexity of the system itself will negate this potential benefit.

More complex systems result in a situation in which users must be quite knowledgeable about their domain in order to utilize the system properly (Waterman, 1986; Wexelblat, 1989). It is even more problematic because it requires a user to be at a higher level in two different dimensions. Not only must a user understand the intricacies of his or her domain but he or she must also be quite knowledgeable about the software facilities in order to perform the tasks in the system (Carroll & McKendree, 1987). Sullivan & Tyler emphasize that, "As the computer systems we build become more complex and begin to achieve the ability to reason and make decisions on their own (through the use of artificial intelligence technology), the role of the computer-human interface (CHI) is increasingly more critical to overall system performance" (1991, p. ix).

2.2 Human-Computer Interfaces

A human-computer interface has the potential to fulfill three key roles in support of a user: (1) it can help a user to accomplish his or her intended goals on the target system, (2) it can enable a user to understand how both the target system and the interface operate, and (3) it can magnify a user's capabilities (Tyler *et al.*, 1991). A good user interface should render the system usable by its users (Totterdell & Rautenbach, 1990). The final report of a research group of 10 top industrial representatives concluded that: "if the interface is ineffective, the system's functionality and usefulness are limited; users become confused, frustrated, and annoyed; developers lose credibility; and the organization is saddled with high support costs and low productivity" (Nolan, Norton & Co., 1992, p. 19).

The interface is becoming increasingly important because of the ever-widening range of computer system users. Both highly knowledgeable and quite naive users are often served by the same interface. This may pose a problem because each interactive computer system is designed for a particular type of user. The designer's view of a typical user's needs, preferences, and requirements are implicit in the design of an interface (Card *et* *al.*, 1987). This can be very effective if the interface is well designed and the users are homogeneous. However, user populations for systems are becoming more diverse and are requiring a wider range of capabilities that lead to an increase in the overall complexity of interactive systems. The underlying assumption that all users are essentially the same and that they all have the same requirements is rapidly proving to be false.

The basic guidelines for good interface design are well-known – provide direct access to the concepts that users associate with work in the domain, and provide appropriate feedback to users' actions (Norman, 1986). Despite simple principles, good user interfaces are becoming increasingly more difficult and time-consuming to build. Generally, as user interfaces become easier to use for the end user, they become more complex and harder to create for the UI designer (Bobrow *et al.*, 1986). Analysis of knowledge-based systems has shown that the acceptance and real use of knowledge-based systems depends on far more than a knowledge base and an inference engine and that the user interface is of crucial importance (Fischer & Reeves, 1995; Gordon, 1988). For many users, the user interface *is* the application. Not only is the user interface important to the user, but it is a significant effort for the developer. The user interface for an application is usually a significant subset of the code. User interface code as a percentage of the overall system code often ranges from 35-50% (Bobrow *et al.*, 1986). Browne *et al.* (1990b) also states that as much as 40% of code may be for user interface purposes and that a large percentage of all maintenance is purely for user interfacing requirements (p. 118). Recent reports estimate that over half of the total cost of new computer systems can be attributed to the user interface (Myers & Rosson, 1992).

2.3 User Involvement in Design

The easy-to-use, direct manipulation style interfaces that are popular with most modern systems are among the most difficult kinds to implement. The only reliable method for generating quality user interfaces is to test prototypes with actual end users and to modify the design based on the users' comments (Myers, 1991). This test and modify cycle is sometimes called iterative design, and it is repeated until the user interface is deemed acceptable (Nielsen, 1993). Iterative design goes hand in hand with the increasing focus on the user. User-centred design (Lewis & Norman, 1986) has become a trend in the industry for the following reasons (Miller, 1996):

- Explosive growth in interest in interfaces as a discriminator.
- Increasing product and domain complexity.
- Increasing competition.
- Decreasing ability to sell a "black box".
- Willingness to pay for development and support of UI.
- Increased emphasis on integration with other systems ("open" architecture), and with legacy systems and "training".

System design methods range from the formal to the strictly *ad hoc*. An *ad hoc* design process that is based on intuition and limited experience is insufficient for interactive systems that will be used by a large, diverse group of people. There are many different ways to get users involved in the design process and thereby improve the design beyond what can be achieved with common *ad hoc* methods. For example, it is not enough to simply use texts or manuals when performing task analysis during system design. Users can be included by: the direct observations of task performers, structured interviews with task performers, questionnaires, concurrent or retrospective protocols, tutorial sessions, prototype walk-throughs, and process model walk-throughs (Johnson *et al.*, 1988). Pilot studies and prototyping can also be used in the later stages of task analysis as confirmation or validation aids (Browne *et al.*, 1990b).

User requirements analysis may be regarded as a preliminary evaluation of the needs of the potential users of a system. A thorough analysis of the users' requirements is appropriate – what kind of information can be gathered about the users' needs, usage patterns, vocabulary, and surrounding environment? An examination of both user requirements and the available functionality may provide an indication of potential areas for building adaptation into a system (Totterdell & Boyle, 1990). In order to perform task analysis for an adaptive system, a developer must have a method for conducting the analysis, a notation for documenting the results of the analysis, and a mechanism for allowing the adaptive system to be aware of its tasks.

2.4 Evaluation

Design, implementation and evaluation were traditionally viewed as separate topics. Most software developers now view these activities as iterative and overlapping. Traditional evaluation techniques can be very effective in comparing two design alternatives, but conclusions can only be drawn if the results from the two systems are different enough and the number of test users is sufficient. One of the simplest approaches is a between-groups experiment. This approach uses two groups of test users, one of which uses version A of the system and the other uses version B. The experimenter tries to test if the typical value for version A is different from the typical value for version B, and by how much.

In order for an experiment to be valid, certain assumptions must be met. The most important assumption is that the users are drawn from a "normal distribution". This means that the data from both groups must come from distributions with the same standard deviation, or that the data falls within a "normal" range. Various statistical techniques are used in determining a normal distribution once the experiment is complete. Variability is controlled as much as possible in empirical studies, but the control of variability itself may remove possibly significant aspects of a typical system environment. That is, the results that are obtained may not reflect what will happen with the system in real life. One major source of variability is differences among test users – subjects can differ in how much they know about a task or about the system. This can affect how long it takes them to perform tasks and how likely they are to make errors, irrespective of the user interface involved. To help control this, users should be recruited from groups of individuals with similar backgrounds. Another method of controlling variation is to include a training component in the experiment that attempts to bring the users to some common level of preparation for their tasks. Differences in procedure, or how the test is conducted, will also add to variability. For example, users in one group cannot be assisted more than another group. Variability can also be introduced if the instructions and task descriptions used in the test are unclear. Researchers attempt to control these aspects of variability as much as possible in order to ensure that the results are statistically valid.

Pilot studies should be done as part of any usability test. These studies give the experimenters the opportunity to run through everything that is planned for the real study and to fix any problems. Normally several pilots are run, the first few with colleagues or friendly users to identify the biggest problems, and then some with actual users to test the experimental design and uncover potential flaws.

2.5 Commercial Pressures on Software Development

All but the most narrow of niche software producers has to balance two potentially conflicting demands from the marketplace. The first demand is for commercial software to provide ever-increasing breadth of functionality to satisfy the needs of an ever-widening range of users. The second demand is that each individual system must be easy to use. Henderson & Kyng recognize this market pressure: "When we are creating a product which will be purchased by many people, the need to design for many different situations of use is particularly clear, and in fact some of the techniques for allowing users to adjust systems in use have been motivated by market considerations; for example producers of "plastic" software who want to satisfy as many users as possible with a single product" (1995, p. 794). They argue for easier methods of tailoring systems for individual users, adaptable systems as opposed to adaptive systems. Their user-initiated solution is slightly different from researchers who prefer a system-initiated solution, but the underlying market pressure is the same.

Static, non-adaptive software may be unable to meet these two market demands. Henderson & Kyng (1995) argue that there are three main reasons why the system behaviour may need to change after the initial implementation. First, over the passage of time the needs of the user and the circumstances surrounding the system use may change, even if it is a good initial design. Second, it is virtually impossible to anticipate all of the possible issues in the initial design. Third, the product may need to be used by a variety of users for many purposes. Henderson & Kyng's goal is to provide a system that users can tailor to their own needs.

These marketplace demands affect users directly. As Höök (1996) puts it, "Systems are getting to be more and more *complex* and users experience great difficulty with keeping up with the recurrent releases of software and the new possibilities offered". If existing users are finding that their programs are becoming more difficult to use, then system developers need to especially concerned about the newest wave of users who have been forced by competitive pressure to adopt new technologies.

This pressure is occurring for a variety of reasons. Companies are attempting to maintain a competitive edge by dictating various types of software use by employees. Corporate software use is often expected to allow downsizing of existing staff. Not only will the employees be expected to quickly learn the software, but they are also expected to become much more efficient as a result. Individuals in the service sector are also experiencing increasing consumer demand for the new services that are made possible through these new complex programs. These users, who are often reluctant, are a special challenge for the developers of complex software.

2.6 Designing for Different Types of Users

Designers of interactive software systems have a variety of options when building a user interface. They can: (1) search for a general design that is adapted to most users' needs; (2) design the interface to incorporate the means for users to adapt the interface themselves; (3) devise and build in mechanisms able to determine a user's particular needs and automatically adapt the interface accordingly; or (4) "Adapt the users to the interface" by providing support for training users (Browne *et al.*, 1990). Research in adaptive user interfaces, including the work presented in this thesis, is concerned with the third method.

It is generally accepted that powerful computer systems are too hard for novices to use, and that easy-to-use systems are too clumsy and have too few features to be useful to experts. An experienced computer user has a deeper understanding of a software system and takes more responsibility in directing its actions. An expert does this in exchange for improved performance and extra functionality (Bonar & Liffick, 1991). Shneiderman (1992) makes the following point: "In designing a system for novices, every attempt should be made to make the user at ease, without being patronizing or too obvious". He adds that constructive messages and positive reinforcement produce faster learning and increase user acceptance.

Bonar & Liffick (1991) describe the ways in which designers can build interfaces that allow a graceful progression from a novice's use of a system to a more sophisticated use of a system. In a static system, developers could start beginners off with black boxes, which are simple, intuitive, and directly applicable to their particular tasks. Eventually, they might become frustrated with the lack of flexibility that is inherent in the fixed set of plans. This is the normal experience of advanced users of most computer software; the system cannot be tailored to their evolving needs. By providing many levels, developers can give users a smooth continuum from easy-to-use and intuitive to sophisticated and flexible. A user can become as sophisticated a computer user as he or she desires; the user is adapting the system.

Another solution is to have the system adapt to a much larger set of potential users by applying knowledge about the type of person that a user is, the understanding that a user is likely to have about the domain, and the functions that a user is likely to require (Benyon, 93b). Real systems need straightforward, easily implemented user models in order to control the complexity and to keep the user from being overwhelmed by the myriad of functions and options. As Judy Kay pointed out in her address to the International Joint Conference of Artificial Intelligence, "We are poised at the beginning of an era when vast resources of information and knowledge will be available to users mediated by machines. The magnitude and variety of those resources pose problems with users who are overwhelmed by the diversity and choice available. This is the type of domain where there will be benefits from user modelling in various roles" (Kay, 1995).

2.7 User Modelling

One phrase that is often used in human-computer interaction research is "Know Your User" (Thimbleby, 1983). This is sometimes referred to as the golden rule of HCI design (Browne *et al.*, 1990). User descriptions, profiles, and models each attempt to implement this philosophy in order to allow computer systems to communicate more effectively with individual users. A model of user behaviour can be achieved through systematic studies such as user performance analysis and interviews, in combination with borrowed concepts from various fields such as cognitive psychology (Thimbleby, 1983; Poor, 1987). Good models make it possible to predict user responses to new interfaces and to develop practical design guidelines (Sherman & Shortliffe, 1993).

In artificial intelligence research, the concept of a user model is not as general as it is in the fields of cognitive psychology or human-computer interaction. Donald Norman (1993) discusses three different types of user models in HCI research: "...the individual's own personal idiosyncratic model; the generalized *typical* user model that the designer develops to help in the formulation of the design model; or the model that an intelligent program constructs of the person with which it is interacting." It is the third type of user model that this research is focused upon.

In an attempt to clarify the various sources of information about user profiles, the user modelling research community typically asks several questions: What constitutes a good user profile? How can a designer of an interactive system use this information?, and How can a system be customized to an individual user?

The system designer must first categorize the various user groups and create a demographic profile: their age, sex, physical abilities, education, background, training, motivation, goals, and personality (Shneiderman, 1984, 1992). Specific considerations must also include the users' previous experience with computers, their level of comfort with the tasks involved, their typing ability, intelligence, flexibility, and possible anxieties about the technology. Users should also be tested for various skills such as boolean expression understanding or task-specific abilities if these are skills that are necessary for the operation of the system. For example, a system developed for an airline may require knowledge of airport codes.

The human factors literature indicates that there are many factors which influence user satisfaction and productivity (Strachan *et al.*, 1990). User satisfaction as a key attribute to software success will be focused on in Chapter 6. Screen layout, menu size, pointing devices, colour use and computer response time can have a large effect on error rates, learning time, and how different categories of users vary in their preferences (Raeder, 1985). For example, novice users generally prefer less dense screens, menus, and more extensive help and explanation levels. Expert users of the same system may want to have direct access to functions without stepping through menus, and they would find the help and explanation levels for a novice user to be too detailed. It would be helpful to have a system that could store an internal user model and could respond differently to novice and expert users.

How could this user model be constructed? Should the user be questioned at the beginning of a session, or should the system try to infer a user model through its use? The objective is to build the model so that it is useful to the system without overburdening it with the job of building and modifying the model (Rich, 1979).

The method that has the least impact on system design places the onus on a user for modifying the model. The easiest way to program user-specific changes to the system would be to have the users indicate their skill level and preferences just before entering the system. This could be as simple as a dialog that asks them questions about the following: their experience on the system, frequency of system use, dialog style preference (i.e. menu or commands), pointing devices or keyboard, colours, fonts, information presentation style (i.e. graphs or text), windows, and subsets of system functions that are required.

There are several problems with this approach: users are not always able to provide accurate information about themselves, it is time consuming, and it takes away from the actual purpose of using the system – getting the tasks completed. At the very least, these models should be retained from session to session and the users should be able to specify macros that can be executed to modify the system to suit them better (Rich, 1979). It might be desirable to have specific modifications for novice users such as more complete error messages, help presented automatically, and slower and consistent response times. Shneiderman (1992) reported that these characteristics have been shown to be better for inexperienced users. Some systems use the "training wheels design" approach for novice users by not allowing them initial access to the more advanced system functions.

Conversely, experienced users need complete functionality, the ability to go directly to the function that they need, and higher level explanations.

A more sophisticated system would infer a skill level based on how a user is using the system. For example, it would be safe to assume, at first, that the user was an experienced user if the first task that was invoked was the system modification function. If different skill levels were associated with the various system features and user errors are monitored and evaluated, this skill rating would be refined dynamically (Rich, 1979). Anderson & Olson (1987) implemented an animated help system using this technique. It used information such as the users' histories of command usage, their frequency of command usage, errors, and misconceptions in order to determine their level of expertise. This information was analyzed and used to make decisions about the conversation styles that would be appropriate for that level of user knowledge. Microsoft's Office Assistant 1.0 is a more recent example of a system that monitors the user's tasks to determine context-specific help and feedback.

These types of systems are examples of applied research in adaptive systems and learning which promises to provide interfaces that are more flexible, more tolerant of user error, more helpful, and more able to tutor their users when it is appropriate to do so (Rissland, 1987). It is not hard to imagine that the system overhead and development costs are much higher for these kinds of benefits and that these systems are still primarily in the research stage. The next chapter provides detailed information about adaptive systems and the structure and representation of the user modelling components that are necessary to support these types of systems. A number of classic user modelling systems are examined in order to demonstrate some of the common techniques used in the field.

CHAPTER 3

ADAPTIVE SYSTEMS

3.0 Introduction

A recent goal in human-computer interface research has been to have a computer system that caters to users' individual differences; meets user expectations; and achieves its full potential as a cooperative, helpful assistant in problem-solving, decision-making, and learning (Carberry & Kobsa, 1994). There are many different approaches to the adaptation of individual systems to individual users, and they range from simple *adaptable* systems to more complex *adaptive* systems.

An example of an adaptable system is one that allows the user to gain access to application customization functions such as preference menus and macro definitions. For example, a user may be restricted to simple changes such as colours, size, or positions of windows. These types of systems require a user to have knowledge about how the system could be customized and how to perform the customization. The amount of customization is normally quite limited.

An adaptive system monitors the user and recognizes the need for adaptation. The system then performs the adaptation automatically. Or, the system may prompt the user for permission to make changes. To date, there are few commercial products that can be classified as true adaptive systems. Chapter 1 described some of the more recent advances in the field. Adaptive system research is quite diverse and ranges over a wide variety of domains (Carberry & Kobsa, 1994), such as:

- Adaptive Databases that provide navigation aids that take the user's interests, plans, and goals into account (Calistri-Yeh, 1991).
- Hypertext systems with an adaptive component adapt the output text to the knowledge of the user (Brusilovsky, 1996; Höök, 1996).
- Online help systems that adapt advice to the user's goals (Höök *et al.*, 1995).
- Natural Language systems that adapt to an individual's use of a language and provide a context for interpretation (Carberry, 1989; Jameson, 1989; Wahlster & Kobsa, 1989).
- Computer Aided Instruction (CAI) systems which adapt their tutorials to a student's knowledge and misconceptions (Kass, 1989; Mark & Greer, 1993, 1995; Murphy & McTear, 1997).
- Expert or knowledge-based systems that tailor explanations to a user's level of expertise (Cawsey, 1993a, 1993b, 1993c; Kass & Finin, 1991).
- Adaptive User Interfaces (AUI) that adapt layout, interaction options, and interaction modes to the user's tasks, abilities, and preferences (Benyon, 1993a; Browne *et al.*, 90a; Browne, 1993; Schweighardt & Carter, 1988; Schneider-Hufschmidt, 1993).

In the words of Reinhard Oppermann, "the goal of adaptive systems is to increase the suitability of the system for specific tasks; facilitate handling the system for specific users, and so enhance user productivity; optimize workloads, and increase user satisfaction (1994, p. 4). Judy Kay emphasizes simply that the goal of adaptivity is, "to improve the efficiency and effectiveness of the interactions between users and systems" (1993). Most adaptive systems attempt to achieve one or more of the following goals: extend a system's lifespan, widen a system's user base, enable user goals, satisfy user wants, improve operational accuracy, increase operational speed, reduce operational learning, and enhance user understanding (Browne *et al.*, 1990).

All computer systems contain implicit assumptions of the type of user that is using the system, but applications that have an adaptive component have one common characteristic: a user model that is explicitly represented as a knowledge base separate from the application itself.

3.1 Components of User Modelling Subsystems

User models contain assumptions about the user, and they usually include some knowledge about the application as well. The contents of the user model may include: the amount of domain knowledge that a user knows, beliefs, goals and plans, abilities, attitudes and preferences, and personal characteristics.

The basic architecture of user modelling systems is shown in Figure 3-1. The user model (UM, the knowledge about individual users) and the user modelling component (UMC, the reasoning structure) comprise the user modelling module (UMM). The user modelling module's specific capabilities may vary from system to system. In general, it provides facilities that add, update, and delete information from the user model; draw inferences from the user model with input from both the application and the user; and maintain a consistent user model. Upon request, the user modelling module provides relevant information to the application about the user and the current context. The application then communicates with the user modelling module and performs the adaptations that are based on the information obtained from the UMC (Finin, 1989; Wahlster, 1991).

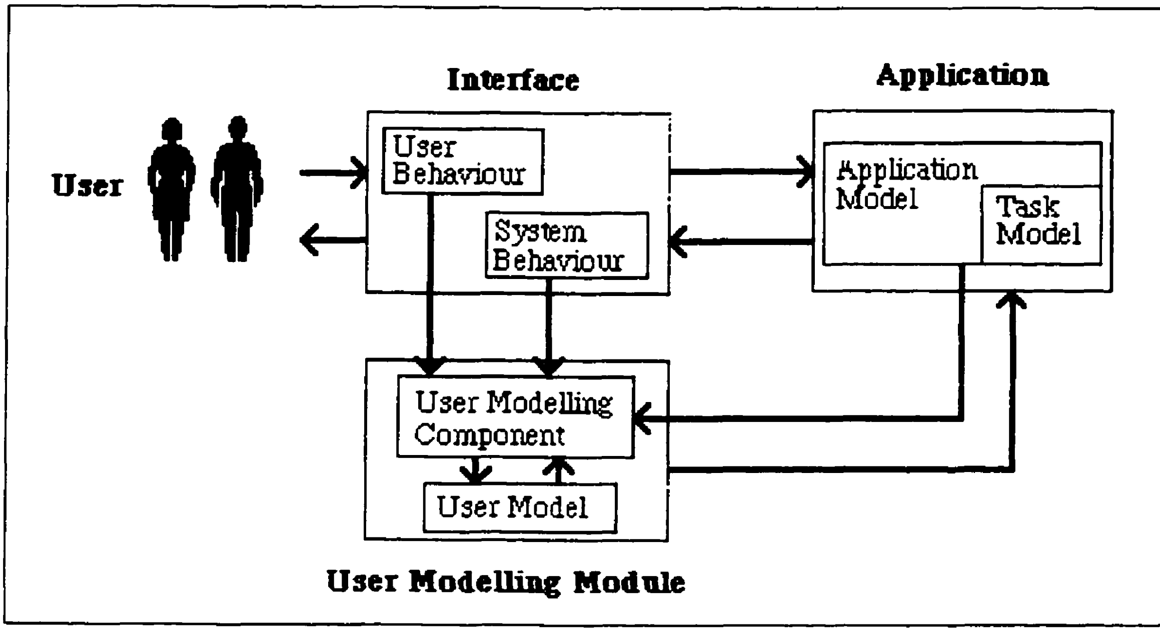


Figure 3-1. A Generic User Modelling Module (adapted from Finin, 1989).

A knowledge-based system approach is considered to be the most appropriate means to implement the User Modelling Module because the declarative representation that is inherent in this approach aids in the implementation and maintenance of the models. The separation of the knowledge bases from the inference mechanism provides an opportunity to replace one model of an application with another. Modifications and additions to the components of the module are easier in a declarative representation.

Many different representation methods have been employed to represent user models within the knowledge-based approach. The next section will review these basic techniques from the standpoint of the representation of an individual user. In the final section, six important user modelling systems that employ a variety of knowledge representation schemes will be briefly examined. The inferencing that is based on the content of the user models will not be addressed directly, but it will be mentioned where appropriate.

3.2 Representation Techniques

A representation is a set of conventions about how to describe things (Winston, 1977). Most early user modelling systems used traditional knowledge representation techniques of artificial intelligence. The choice of representation technique most often depends upon the purpose of the application and the expressiveness of the user model (Wahlster & Kobsa, 1989). As in other applications of knowledge-based systems, hybrids of various basic techniques are used in many systems. Because of this, attempts to create comprehensive surveys of representation use often present conflicting system categorizations, and indeed, often differ in the categories employed. In one recent attempt to categorize adaptive user interfaces into a taxonomy (Dieterich *et al.*, 1993), five categories were used for representational techniques: frame-based, object-oriented, formal logic, neural nets, and user modelling shells. An alternative classification scheme included additional categorizations such as fuzzy logic, connectionist networks, and functional translations (Carberry & Kobsa, 1994). Still another labelled the four main groups of techniques used in knowledge representation as: rule-based, frame-based, network-based and example-based (Dix *et al.*, 1993).

Even if a suitable set of categories were agreed upon, there would still be some confusion about the nomenclature of representations themselves, particularly the hybrid techniques that combine several representations together. This review follows the categories used in the Dieterich *et al.* classification scheme, with some small modifications. As mentioned above, Dieterich *et al.* employed a five-category approach incorporating frame-based, object-oriented, formal logic, and neural net methodologies, as well as user modelling shells. Neural Nets will be discussed under a broader classification of example-based systems, Semantic Nets (often included by others in frame-based approaches) will be included under the network-based representations, and rule-based representations will be added, as specified in Dix *et al.* (1993). This is not a complete set of knowledge representation schemes, but it is representative of those employed in present user modelling systems.

3.2.1 Frame-based Techniques

The most popular and earliest representation technique was the use of frames, as proposed by Marvin Minsky (Shapiro, 1990). Frames are complex data structures that represent default knowledge and a prototypical, or commonly occurring situation. An example of a frame taken from (Shapiro, 1990) is as follows:

```
frame: PERSON
    birth-month: NUMBER constraint birth-month > 0 and birth-month < 13
    birth-year: NUMBER constraint birth-year > 1900
    father: PERSON
    eye-color: COLOR default = blue
    age: NUMBER procedure = COMPUTE-AGE

procedure COMPUTE-AGE ()
    return (CURRENT-YEAR () - (IF birth-month > current-month)
                                  (return current-person.birth-year + 1)
                                  (return current-person.birth-year))
```

This frame represents a person and has slots for birth-year, father, eye-color, and age, which are all related to the concept of person. The values of the slots can include simple values, frames, default values, constraints, and procedure calls. In this particular example, the slot for birth-year must contain a value greater than 1900, the slot for father must be a PERSON frame, the eye-color slot has a default value of blue, and the age slot contains a procedure that will calculate the age based on the current date.

Frames are often referred to by various names. They may be called concepts, schemas, scripts, units, or classes. In user modelling research, they are often referred to as stereotypes. Frames are often organized into networks, resulting in a significant overlap of frame-based techniques and other network-based approaches such as semantic networks.

3.2.2 Network-based Techniques

Networks represent knowledge in a labelled, directed graph whose nodes represent concepts and/or objects and whose arcs represent relationships between the objects and concepts. The network is a hierarchy and children nodes can inherit properties from their parents.

Semantic networks exemplify network-based techniques. A semantic network or a semantic net is a knowledge representation scheme that is based on a network structure. They were originally used to represent psychological models of human memory, and are more recently, commonly employed in Natural Language Understanding systems (Waterman, 1986).

Typical relationships between concepts in a semantic net are: is-an-instance-of (isa), is-a-kind-of (ako), and has-part. The simple semantic net in Figure 3-2 represents the statements, "Clyde is an elephant," "An elephant is a kind of animal," and "The nose is a part of an animal." From this net an inference can be made that Clyde has a nose even though this information is not explicitly stated. The inheritance hierarchy can save large amounts of storage space because only the exceptions for objects or concepts have been stated.

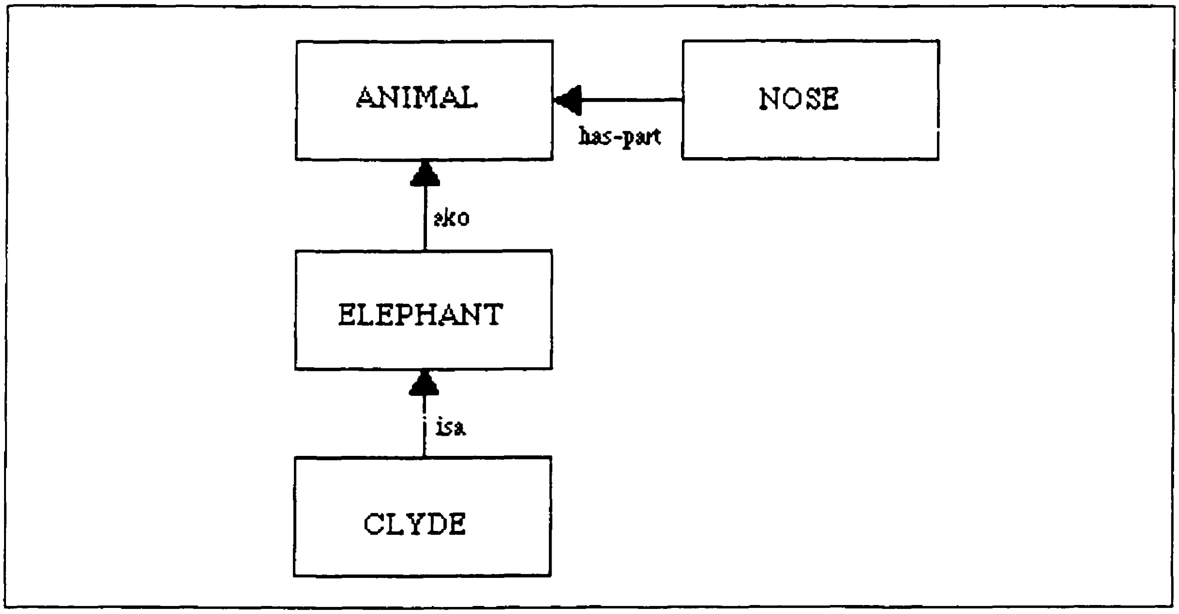


Figure 3-2. An example of a simple semantic net.

Semantic nets are useful in domains where one characteristic of the knowledge is well-established taxonomies. They are similar, in some ways, to relational databases (part-of relation) but they provide inheritance and typically do not represent behaviour properties of the nodes (i.e. methods). They tend to be very fine-grained as compared to frames and object-oriented systems.

3.2.3 Object-oriented Techniques

Object-oriented software techniques and development continue to be very popular in application development in the 1990's. In an object-oriented system, all information is represented in the form of objects and is organized in a network hierarchy. Each object is self-contained, has its own private data, and a set of public and private operations.

While object-oriented approaches are most often associated with conventional systems, knowledge-based systems are often object-oriented as well. (Indeed, the previous two schemes both contain elements of what is now considered the object-oriented approach, and also pre-date that approach significantly). If a knowledge-based system is object-oriented, then its knowledge base consists of objects that represent entities in the domain, which are divided into classes and instances. Each class is a template that describes information that is common to a group of objects. Classes can be further refined into any number of subclasses that identify additional characteristics, which are common to the particular subset of objects. The object instances that describe individual objects are at the lowest level of the hierarchy. This hierarchy mechanism provides default values that can be inherited or overridden at any level.

All of the interactions with an object take place through the operations, called messages, which act as an external interface to the object. All of the objects communicate with each other by sending and receiving messages. The object-oriented system kernel dispatches messages to objects that describe the processing that is required. When an object receives a message, it consults its database and procedures in order to decide what action to take. These procedures, which are called methods, may be stored directly with the object. They may also be stored in a higher-level object somewhere in the network hierarchy and may be accessed through the same inheritance mechanism that is used in accessing data. In most cases, the action implemented by a method involves sending new messages to other objects in the system and asking other objects to perform operations and return results.

Object-oriented database management systems use the same concepts, but they must also include additional characteristics in order to manage large, shared, persistent object stores. These characteristics include efficient processing over large secondary storage organizations, concurrency control, recovery facilities, and efficient processing of set-oriented requests or queries (Barker *et al.*, 1994). These systems have been developed in order to support applications in situations where the simple structure of conventional databases do not adequately represent real world entities or objects.

3.2.4 User Modelling Shells

Relatively early in the history of the field of expert systems, system developers realized that repetitive development efforts could be minimized (and component re-use maximized) through the collection of component elements common to many expert systems into a larger software development package – an *expert system shell.* While the likelihood that a new expert system project would precisely fit every element provided by a shell would be small, a significant portion of the functionality provided by the shell could still be applied.

In a similar fashion, one of the fastest growing research areas in user modelling is the development of user modelling shells. Some researchers argue that modelling tools that can support even minimal adaptation can achieve helpful levels of adaptation without incurring the high cost of building traditional user modelling systems (Kay, 1993).

A *user modelling component* consists of a domain-independent user modelling shell and application-dependent user modelling knowledge. A *user modelling shell* is a tool for creating, maintaining, and exploiting user models. The shell can be filled with domain-specific inference rules as well as domain-specific default assumptions about the user. In an expert system application this knowledge might be the user's goals, for example, while in a Computer Aided Instruction (CAI) system it might contain the student's knowledge of the subject.

Most user modelling shells allow the application to control the initial acquisition of the user model and the discovery of additional facts about a user (Finin, 1989). The shell itself is mainly responsible for incorporating new facts, checking consistencies, doing conflict resolution, and retrieving user knowledge for use in adaptations.

There are several different varieties of shells, and each has different underlying knowledge representations. GUMS (General User Modeling Shell) (Finin, 1989) uses default logic and stereotypes and it is implemented in Prolog. Two additional shell examples, BGP-MS (Belief, Goal, and Plan Maintenance System) (Kobsa *et al.*, 1994) and *um* (user modeller) (Kay 1993, 1994) are discussed in greater detail in the next section.

3.2.5 Rule-based Techniques

Many knowledge-based systems use rules to represent knowledge. Such systems are commonly called production rule-based systems. A production system consists of a database that describes facts that are associated with the application, a rule-base consisting of a set of rules, and a rule interpreter that selects and processes the rules that have been applied.

Rules have the form:

If CONDITION then ACTION

where CONDITION is a boolean expression that is evaluated to be either true or false. If the CONDITION is true then ACTION is performed. Valid actions include invoking commands or procedures, activating additional rules, and adding or deleting facts from the database.

For example (Waterman, 1986):

(1) If the patient was an insulator before 1965,
then the patient directly handled asbestos.

(2) If the patient directly handled asbestos and
the patient was exposed in confined spaces,
then the patient had a severe exposure.

In this set of rules, if a fact matches the condition in rule (1), the fact that the patient directly handled asbestos is added to the database. This will allow rule (2) to be activated. This data-directed process is called *forward chaining*, and results in an inference chain. This inference chain will ultimately derive some eventual conclusions, and can be used by knowledge-based systems to explain how the system reached those conclusions.

One major advantage in the rule-based approach is that rules are a natural representation for many problems where an algorithmic solution is not possible. They are often appropriate when the domain knowledge results from empirical associations that have been developed through years of problem solving in a specific area (Waterman, 1986).

3.2.6 Formal Logic

Formal logic is a very general representation method that subsumes all of the approaches that have been discussed in this section. It uses functions and predicates to describe individual entities and their relationships (Waterman, 1986). New facts are derived using a combination of numerous rules of inference; the one that is most commonly used is mathematical deduction (Rich, 1983). Because of its generality, unrestricted formal logic can be extremely inefficient computationally. There are many rules of inference that can be applied at any point in the search process, which can lead to an extremely large number of possibilities. Thus, restrictions are normally placed on formal logic in order to control the combinatorial explosion of potential states emerging from any point while still maintaining a high degree of expressiveness in representation. The most common restriction involves the use of resolution refutation as a rule of inference on restricted Horn clauses, as implemented by Prolog, the most common logic-based programming language.

Rules are represented in Prolog in the following form (Rich, 1983):

consequent :- antecedent-1, antecedent-2,...antecedent-n

Problem solving is performed by the Prolog interpreter, which attempts to match a goal to the consequents in the system. If it finds a match, then it must take each antecedent as a subgoal and try to prove that the subgoal is true by finding a match. A goal is true if all of the subgoals are true.

Predicate logic provides a powerful mechanism for representing declarative information that can be used to solve problems in a wide variety of domains, but it cannot easily represent information that is incomplete or inconsistent. Other techniques that are based upon formal logic and attempt to deal with the problems of uncertain and fuzzy knowledge are as follows: nonmonotonic logic, probabilistic reasoning, fuzzy logic, and the concept of belief spaces (Rich, 1983).

3.2.7 Example-based Techniques

Example-based techniques represent knowledge implicitly in a classification system. The decision structure is learned by examining training examples that have been presented to the system. The classifiers detect patterns within the examples, and they are able to use these to classify other input (Dix *et al.*, 1993).

Neural networks are the most commonly known example of this category of representation. Neural networks were originally developed as models of human thought processes, and they were based on research about the human brain. A neural network is comprised of many simple elements that are connected to each other. The overall pattern of activity over all of the elements is the system state. Learning algorithms are important in neural network research; they set the connection strengths between the elements to the correct values so that the input values will generate the correct output values (Shapiro, 1990).

The advantage of example-based techniques is that a large user model knowledge base is not necessary. Users can be categorized by the system as matching usage patterns from the training examples (Dix *et al.*, 1993).

As stated in the beginning of this section, various systems employ combinations of these methodologies in order to construct effective user models. The techniques themselves are very different, and have their own strengths and weaknesses. In order to reiterate the differences between these approaches in the context of user modelling, Figure 3-3 (Dix *et al.*, 1993) illustrates particular pieces of knowledge in four of the major representation techniques that were discussed in this section.

RULE: IF command is EDIT file1 AND last command is COMPILE file1 THEN task is DEBUG action is describe automatic debugger	**FRAME:** User Expertise level: novice Command: EDIT file1 Last command: COMPILE file1 Errors this session: 6 Action: describe automatic debugger
NETWORK: CC is an instance of COMPILE COMPILE is a command COMPILE is related to DEBUG COMPILE is related to EDIT Automatic debugger facilitates DEBUG	**EXAMPLE-BASED:** A trace of user activity: EDIT file1 COMPILE file1 trained as an example of a particular task, e.g., DEBUG

Figure 3-3. Examples of activity in four different representations (from Dix *et al.*, 1993).

3.3 Examples of Adaptive Systems Using Different Representations

The first part of this chapter described the main representational mechanisms that are employed in user modelling; the important systems that employ these techniques are now examined in the rest of this chapter. Eight systems have been selected, and the following section will discuss the nature of the user model representation in each. The first, the GRUNDY system developed by Elaine Rich in the late seventies, introduced the notion of the stereotype, which is a basic element in many user modelling systems today. The second example is Cécile Paris' work with the TAILOR system, a significant example of object-oriented knowledge representation techniques. David Chin's UC system demonstrates how double stereotypes can be used to represent user and the domain knowledge, while Alison Cawsey's EDGE system uses an object-oriented knowledge base and rules to achieve user-specific explanations. The PUSH project's POP system provides another example of an object-oriented approach combined with a WWW-interface with its adaptive help system.

Three examples of user modelling shells will be discussed and compared: Judy Kay's approach with the **um** modeller, which is primarily text-based; the BGP-MS system developed by Alfred Kobsa, which is primarily logic-based; and the GUMS system developed by Tim Finin, which views a user model as a deductive database. Finally, the chapter will present some of the approaches to representation used in Intelligent Tutoring Systems, an area that is closely related to User Modelling.

3.3.1 GRUNDY

A classic system that is often used as a first example in basic user modelling is GRUNDY (Rich, 1979, 1989). Rich's system acts as a library assistant by recommending appropriate books for individuals. This recommendation is based on assumptions drawn from information about personality characteristics supplied by the user. During the initial session with GRUNDY, users are asked to describe themselves in terms of keywords which may trigger stereotypes (e.g. athletic, feminist) (Figure 3-4).

A stereotype represents a collection of attributes that often co-occur in people (Rich, 1989). Stereotypes provide defaults that allow a user modelling system to operate with a small set of initial data that are further refined during interaction with the user.

FACET	VALUE	RATING	FACET	VALUE	RATING
Activated-by	Athletic-w-trig		Activated-by	Feminist-w-trig	
Genl	ANY-PERSON		Genl	ANY-PERSON	
Motivations			Genres		
Excite	800	600	Women	800	700
Interests		Politics	Liberal	700	
Sports	900	800	Sex-open	5	900
Thrill	5	700	Piety	-5	800
Tolerate-violence	4	600	Political-causes		
Romance	-5	500	Women	1000	1000
Education	-2	500	Conflicts		
Tolerate-suffering	4	600	Sex-roles	900	900
Strengths			Upbringing	800	800
Physical-strength	900	900	Tolerate-sex	5	700
Perseverance	800	600	Strengths		
			Perseverance	800	600
			Independence	800	600
			Triggers	Fem-woman-trig	
SPORTS-PERSON STEROTYPE			**FEMINIST STEREOTYPE**		

Figure 3-4. Two example stereotypes (from Rich, 1979).

Each stereotype contains facets and values, and a confidence-level and rationale that are arranged in a generalization hierarchy. The hierarchy is a Directed Acyclic Graph (DAG) which allows a stereotype to inherit attributes from more than one parent. For example, in Figure 3-5, the NUMERICAL-ANALYST stereotype inherits attributes from both the

MATHEMATICIAN and the COMPUTER-SCIENTIST stereotypes. The benefits of a DAG structure are that it avoids the duplication of information and that information from the more general stereotypes can be overridden farther down the hierarchy.

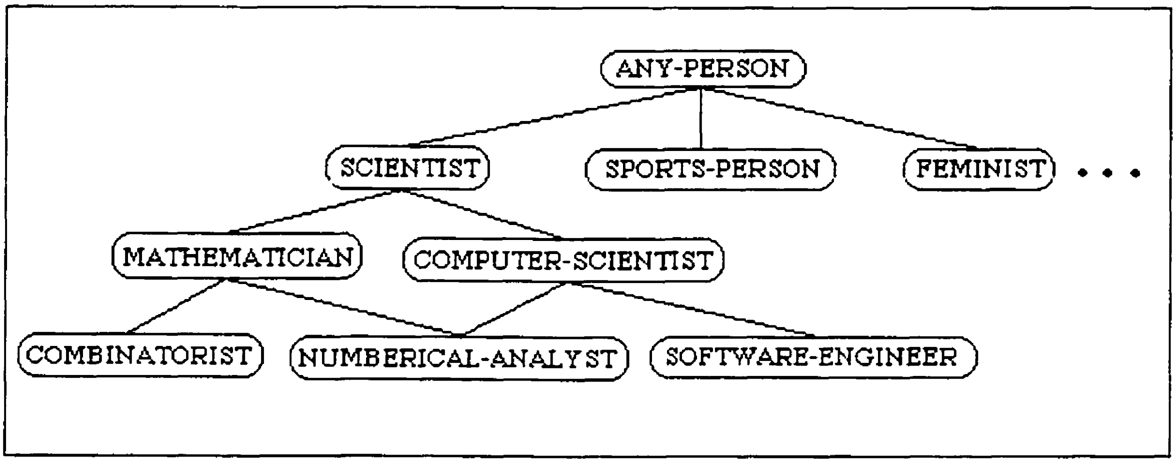


Figure 3-5. A section of the stereotype Directed Acyclic Graph (DAG).

The rationale, or why the system believes a facet to be a certain value, can consist of direct observations, or it can be derived from activated stereotypes. This is especially important for conflict resolution, which is necessary for all types of default reasoning. At the end of a dialogue session, the information is stored so that it can be retrieved and used as the initial user model in subsequent sessions.

Knowledge in GRUNDY is primarily represented as collections of objects and concepts that contain various properties or facets. These concepts may be related to one another by including other concept names as facet values. The user model, or user synopsis (USS), in GRUNDY is comprised of facts, inferences, and stereotypes. Facts and inferences are also concepts that contain the value of the facet, a confidence rating, and the name of the event that triggered them. Events in GRUNDY can add facts or inferences to the user model and can invoke stereotypes. For example, a user who told the system that a book suggestion was enjoyed triggered the first event shown in Figure 3-6.

EVENT 22	ISA	EVENT-INSTANTIATION
	WHAT	SUGGESTION7
	PERMANENT	USER-SAID-LIKES
	ACTION	REC-USER-LIKES
EVENT30	ISA	EVENT-INSTANTIATION
	PERMANENT	USER-SAID-FACT
	WHAT	FACT1
	ACTION	REC-USER-FACT

Figure 3-6. Sample events in Grundy (Rich, 1979).

INF54 ISA		INFERENCE
	FAC-KNOWN	GENRES
	GENRES	(LITERATURE 900 300 EVENT22)
FACT1 ISA		FACT
	FAC-KNOWN	POLITICS
	POLITICS	(LIBERAL) (1000) (EVENT30)

Figure 3-7. A sample fact and inference in Grundy (Rich, 1979).

This event activates procedures that include the inference that "since the book was a work of literature it is probable (900 300) that the user likes works of literature" (Figure 3-7). In the second event in Figure 3-6, the user stated a fact that will be recorded in the user model as FACT1 (a certainty of 1000 is a fact). The triggers are recorded in the user model for subsequent conflict resolution.

A portion of a user model (USS) is illustrated in Figure 3-8. The attributes, the associated value, the measure of belief of the value, and the source of the information are all represented. Note that the values of the attributes can be either text or one of two numerical scales. The first scale ranges between -5 and 5 (with 0 representing neutral) while the second scale ranges between 0 and 1000 (with 1000 representing absolute certainty). The value and the measure of belief of several attributes may be affected by contributions by more than one of the stereotypes. Specialized techniques for the combination of values and for conflict resolution are also integral elements of GRUNDY. However, these are separate issues from those of knowledge representation, and are not discussed in this chapter.

ATTRIBUTE	VALUE	MofB	INFORMATION SOURCE
Gender	female	1000	Inference-female name WOMAN
Nationality	USA	100	ANY-PERSON
Education	5	900	INTELLECTUAL
Seriousness	5	800	INTELLECTUAL
Piety	-3	423	WOMAN FEMINIST INTELLECTUAL
Politics	Liberal	910	FEMINIST INTELLECTUAL
Tolerate-sex	5	700	FEMINIST INTELLECTUAL
Tolerate-violence -5		597	WOMAN
Tolerate-suffering	-5	597	WOMAN
Sex-open	5	960	FEMINIST INTELLECTUAL
Personalities	4	646	WOMAN
	...		
Genres			
Literature	900	700	INTELLECTUAL
Women	800	700	FEMINIST
Political-causes			
Women	1000	1000	FEMINIST
	...		

Figure 3-8. Portions of a user model in Grundy (Rich, 1979).

A large number of subsequent user modelling systems have been based on Rich's research. While a significant early system, there are several major concerns with GRUNDY that serve as lessons for future systems. As Rich herself has pointed out, the system was never run as a large-scale performance program and only reflected limited testing (Rich, 1989). In addition, development overhead was very high in that the developer had to hand-code all the stereotypes in the system, and the initial stereotypes were developed strictly from intuition and not from empirical research.

3.3.2 TAILOR

Cécile Paris (1989) investigated differences in the requirements of dialogue systems in the generation of descriptions for users with varying levels of expertise. She compared encyclopedias for children to those for adults and discovered that physical objects were described differently and that the differences were much more complex than simply the amount of detail that each one provided. Paris' analysis suggested that a user's level of domain knowledge affects not only the amount but also the *kind* of information that descriptions provide. Novices needed more description of the processes that were associated with the object, whereas more experienced users needed information about the components involved and the properties of those components. TAILOR (Paris, 1989) was created as a question answering system that took a request for an object description and employed the user model to generate a description of the object that had been adapted to a user's level of expertise. The user model described the domain components in the knowledge base of the system. This allowed TAILOR to customize the descriptions for users who fell between the extremes of novice or expert.

The knowledge base of a program called RESEARCHER (described in Paris, 1989) contained patent abstracts about complex devices that were written in English. RESEARCHER translated a natural language request for the description of an object into queries to the knowledge base. The motivation for the development of TAILOR was that the RESEARCHER knowledge base was very large and detailed and that the response to a simple query would overwhelm the user with volumes of information that could not be easily assimilated. Paris felt that different types of users understood objects in different ways, and that each one desired a different level of detail.

The user model included explicit parameters to represent the user's understanding of the contents of the knowledge base. The parameters were a list of the objects and the basic concepts of the knowledge base that were known by the user (Figure 3-9).

The user has local expertise about the microphone and understands the concept of electricity:

Local Expertise:	microphone
Basic Concepts:	electricity

The user is a naive user:

Local Expertise:	nil
Basic Concepts:	nil

The user has local expertise about the microphone and the telephone and understands the concepts of electricity and magnetism:

Local Expertise:	microphone, telephone
Basic Concepts:	electricity, magnetism

Figure 3-9. An example of TAILOR user models (from Paris, 1989).

The *local expertise* attribute (as seen in Figure 3-9) contained actual pointers to the objects in the knowledge base. Once a description of an object was generated, the user model was updated. Each object in the knowledge base was a frame that had been organized by using a generalization hierarchy. In addition to the instance-of links there were part-of links and relations. These relations may include cause-effect relations, temporal relations, and analogical relations.

The actual algorithms that TAILOR used to generate the texts were quite complex, but Paris used two basic strategies to describe a part based on the user model. If a user knew a part, then a constituency schema was used that described the parts themselves in basic terms, and not the underlying concepts. If the part was unknown then a process trace was done, causing the part to be described in lower level concepts. The two strategies were then combined in the same explanation. Finally, English language text was generated.

In the TAILOR user modelling approach, the user's domain knowledge was not simply referred to as novice, intermediate, or advanced, but a single user's expertise was explicitly represented. No stereotypes were used; the descriptions of the objects in the knowledge base varied widely depending on the user.

3.3.3 KNOME

David Chin (Chin, 1989) developed a user modelling component for the UNIX Consultant (UC) system. UC is a natural language advisory system that answers questions about the UNIX operating system. KNOME (KNOwledge Model of Expertise), the user modelling component, was developed in order to assist the interpretation of the natural language input and to create better explanations by expressing concepts in terms that the user could understand. It also pruned unnecessary text.

KNOME used a *double stereotype* approach. A user's expertise was judged to fall into one of four categories: *novice, beginner, intermediate* or *expert*. The UNIX commands, command formats, terminology, and other types of information were rated by their difficulty and were assigned to four categories: *simple, mundane, complex* or *esoteric*. Their increasing levels of difficulty order the first three categories. *Esoteric* is the category for information that may have a special purpose and has not been learned at any specific time in the learning curve. Figure 3-10 is an outline of the relationships between the stereotypes and the difficulty levels. Each expertise stereotype contained assumptions about a user's knowledge of a specific level of UNIX-related information.

User	KNOWLEDGE DIFFICULTY LEVEL			
Stereotype	simple	mundane	complex	esoteric
expert	ALL	ALL	MOST	-
intermediate	ALL	MOST	AFEW	-
beginner	MOST	AFEW	NONE	-
novice	AFEW	NONE	NONE	NONE

Figure 3-10. The relationship between stereotypes and difficulty levels in KNOME (Chin, 1989).

A user also had specific information inferred about him or her, based on the natural language input. For example, if a user queried about a certain command, the system would infer that the command was unknown to him or her and the stereotype would then be updated by using that specific information.

Stereotypes in KNOME were represented in KODIAK, a network-style knowledge representation language that included objects and relations. It allowed for multiple inheritance, and it included default inheritance mechanisms. The double stereotype approach resulted in the categorization of users into stereotypes and the categorization of information about UNIX into stereotypes of levels of difficulty. An example of a graphical KODIAK representation of the relation, "Intermediates know most mundane facts", appears in Figure 3-11.

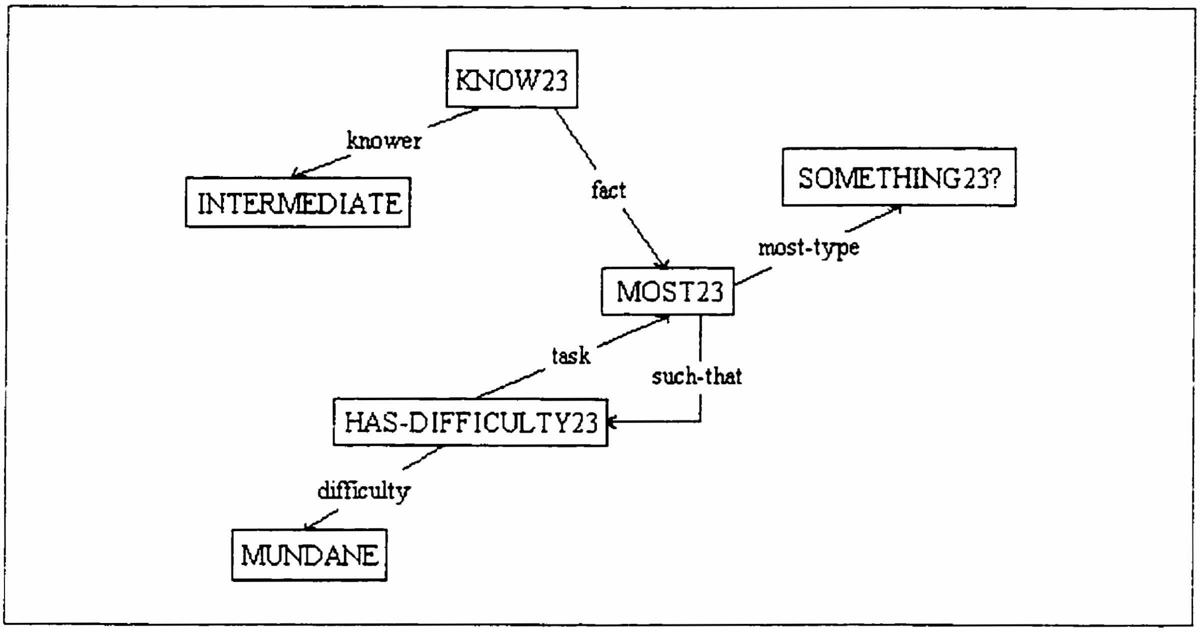


Figure 3-11. KODIAK representation of a relation (Chin, 1989).

The inferences used by KNOME are listed in Figure 3-12. There were also specific background rules that helped to determine what a user's initial stereotype would be. For example, if a user was a fourth year Computer Science student, then the user was likely to be an expert. The rules were implemented using *if-detected daemons*[10]. The system also contained meta-knowledge about its abilities to handle queries that were outside the scope of the system. That is, the system contained knowledge about its own boundaries, which it could use to judge its own effectiveness at handling a particular query.

[10] Daemons (demons) are procedures that are automatically triggered when designated events occur (Awad, 1996).

Inference Class	Kinds of Inference
Claim	user states that user does (not) know ?x -> user does (not) know ?x
Goal	user wants to know ?x -> user does not know ?x
Usage	user uses ?x -> user knows ?x
Background	user mentions his/her background -> user knows as much as the stereotype indicated by the background
Query-Reply	system asks if user knows, user replies -> user's reply
No-clarify	system uses new terminology, user does not ask for clarification -> user understands terminology

Figure 3-12. Inference about the user in KNOME (Chin, 1989).

The double stereotype approach allowed the system to predict what a user knew, based on a partial user model. The initial stereotype was not predefined. At first, a new user was considered a novice, and the actual stereotype would be inferred once a user had performed enough tasks that a clear candidate stereotype could emerge. Once the user stereotype had been determined, the system could predict other system concepts and commands that a user was likely to understand. All of the UNIX file manipulation commands, UNIX information gathering commands, UNIX concepts, and the UNIX terminology had been hard-coded as to their level of difficulty, and they were represented in the KODIAK knowledge base.

3.3.4 EDGE

Alison Cawsey's EDGE system, which is a tutorial system for electrical circuits, contains rules that adapt the explanations according to the specific model and the perceived goal of the interaction (Cawsey, 1993a, 1993b, 1993c). EDGE contains knowledge that can be used to recognize user knowledge and update the user model based on direct interaction with the user (Figure 3-13) (Cawsey, 1993a). EDGE also contains knowledge that indirectly infers user knowledge – through the ranked difficulty of concepts the user is involved with and relationships between concepts (Figure 3-14) (Cawsey, 1993a). The information that is extracted from the object-oriented knowledge base may be combined with the knowledge-based system output, which is known as a rule trace, in order to support three different types of queries about objects, constraints, or solutions.

Dialogue Exchange	Inference
System tells user X. User acknowledges	User maybe knows X
System asks user X. User replies correctly	User knows X
System asks user X. User replies incorrectly	User doesn't know X
System asks user if she knows X	(Whatever user's reply is)
User asks X	User doesn't know X

Figure 3-13. Direct Rules for Updating User Model.

Condition	Inference
All subconcepts known/unknown	Parent concept known/unknown
All subconcepts known or maybe known	Parent concept maybe known
Concept difficulty greater than level of expertise of user	Concept unknown
Concept difficulty less than level of expertise of user	Concept maybe known
Parent concept known or maybe known	Subconcepts maybe known
Parent concept unknown	Subconcepts unknown

Figure 3-14. Indirect Rules for Inferring What User Knows.

By explicitly representing the knowledge that a user has about a domain, an explanation can be adapted very specifically. Users who have demonstrated knowledge about certain concepts and constraints will receive a terse, more technical, presentation of the solution or solutions that are presented by the knowledge-based system. Less knowledgeable users may receive more general, almost tutorial, explanations with extra support from various presentation strategies (e.g. graphic animations). By using a complete domain model as an integral part of the user model, it is possible to support a single user whose expertise ranges from the first case to the second within the same domain.

3.3.5 PUSH

PUSH (Plan- and User Sensitive Help) is a joint project with Ellemtel, the Swedish Institute of Computer Science (SICS), NUTEK, Stockholm University, and Linköping University (Espinoza & Höök, 1996; Höök, 1997b; Höök, 1996; Höök *et al.*, 1996). The two main goals of this multi-year project were: to develop techniques for making systems more adaptive, and to empirically establish their usefulness. One of the results of PUSH is a prototype adaptive help system (POP[11]) for users of a large Intranet database. This database was used as an on-line manual for an object-oriented software development method called SDP. An adaptive information filtering system, with a WWW-interface, was created in order to reorganize this large, industry-produced manual into a highly

[11] PUSH Operational Prototype

interactive, multi-modal hypertext system (Figure 3-15). The interface includes static text, generated text, and generated graphics as output, and it accepts both direct-manipulation and free-form query input. Html-pages are not stored and downloaded by the user; each page is created dynamically in response to the user's actions and history of actions. Each dynamically-generated page, called an *answer* page, contains three frames: a textual description of the method, graphs that describe the relationship between the process that is being queried and its related objects, and a guide to the textual description that highlights the displayed sections.

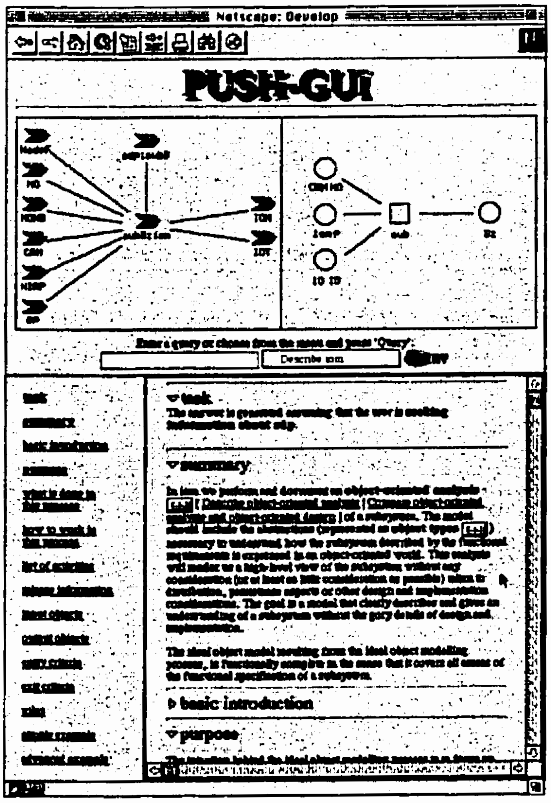


Figure 3-15. A screen dump from the PUSH system (Espinoza & Höök, 1996).

The graphs give the user an overview of the domain and a navigational tool. As the user clicks on an object, the view changes in order to focus on the new object, and the appropriate textual information is retrieved and displayed in the bottom frame. The textual information can be expanded or contracted through the guide frame by clicking on the appropriate heading. This is called *stretch-text* and it allows the user to create an answer page that is better suited to his or her needs. *Hotwords*, which are important concepts that have been marked with bold text, are embedded in the text in order to allow the users to ask follow-up questions. Users can also navigate through the database by composing questions through the query menus. This feature allows experienced users to go directly to the desired information.

The interactivity of the interface to the SDP database is very useful but problems with information overload can still occur if the answer page that was generated is too long. Höök *et al.* discovered that users tended to read no more than the first page of information (1997b). Users of the system were studied in order to determine some suitable information filtering techniques from their interactions with the system. The user's information-seeking task was discovered to be a good heuristic in determining the most relevant information. This knowledge about the current task enables the system to reduce the amount of information and to reduce the size of the answer page. The user initially sets the current task, and then the POP's plan inference algorithm is used to update the task. Simple rules connected a user's question with a task, in order to generate the most relevant information entities. This approach of using a combination of adaptive and non-adaptive techniques was intended to provide the user with concise explanations.

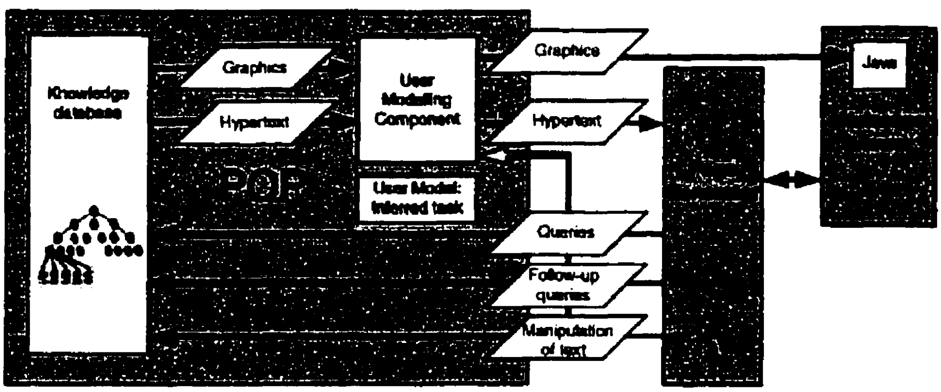


Figure 3-16. The architecture of the PUSH system (Espinoza & Höök, 1996).

This adaptive hyper-media system uses an object-oriented knowledge representation (Figure 3-16). The knowledge database is organized as objects that contain attributes. There is one object for each process or object in SDP. The attributes can contain information that relates the object to other objects, as well as information entities that describe the object. Information entities can be either prefabricated texts in the database, or other text that can be dynamically generated from the object information. The user can manipulate the output from system and can control the level of the adaptivity because one of the primary goals of the system was that the users have control over the adaptivity. The user model is separated from the information in the database, and the generation of the html-code is performed by Java applets and displayed through Netscape Navigator.

Comparative studies were done with the adaptive system and a non-adaptive variant of the system. The users preferred the adaptive system to the non-adaptive system. This project resulted in one of the first examples of a real-world adaptive system that was empirically tested.

3.3.6 um

um (Kay, 1993, 1994) is an experimental user modelling toolkit, which uses structured text files and a rule interpreter to provide user modelling to application programs. The toolkit is used to retrieve or update information contained in text files. Each entry in the text file contains a knowledge *component* and two information lists. The first of these is a list of information that supports the assumption that the component represents (with varying degrees of reliability), while the second of these lists information that negates the assumption. The system employs stereotypes, but there is no stereotype hierarchy.

The *um* toolkit is designed to keep the user model separate from the application that is using it. To date, the most extensive project using *um* has been a system that helps users to learn how to use the **sam**[12] text editor. Figure 3-17 shows an example of a user model (in English text format) that is used in conjunction with a **sam** coaching program.

[12]A powerful mouse and window-based text editor. Used at the University of Sydney, Australia.

Detailed knowledge of various aspects of the editor

1.1	believes that killing the window is a good way to quit
1.2	knows how to quit safely and sometimes does
1.3	has been told twice about the benefits of the safe quit method
1.4	probably does not know how to make multiple windows on a file
1.5	has been told once how to make multiple windows on a file

Some general attributes of the user

2.1	writes C programs
2.2	dislikes using a mouse
2.3	seem to only want to know the minimum about **sam**
2.4	claims to be a sophisticated user of the text editor **vi**
2.5	fast typist
2.6	prefers terse explanations and descriptions

User's current goal

3.1	currently typing a set of additions to a large program
3.2	currently adding code to a function (exprn) in the file parse.c

Figure 3-17. A portion of a user model in *um* (Kay, 1994).

The implementation of the user model contains attribute-value pairs that represent items that a user may know, believe, or prefer. Each component has values and explanations that are associated with it. The evidence that supports the value of the component can be an observation, stereotype activation, rule invocation, or a user entry. One of the biggest differences between this user modelling shell and other user modelling components is the user's access to the user model: all aspects of the user model are visible and may be inspected and changed by a user through a sophisticated graphical interface (Kay, 1994).

3.3.7 BGP-MS (Belief, Goal, and Plan Maintenance System)

BGP-MS is a more traditional user modelling shell system (Kobsa *et al.* 1994; Kobsa & Pohl, 1995). It acts like a black box for the application that is using the user modelling facilities, which are provided by the shell. The application may ask questions about a user, and the shell may provide information about the assumptions that it has about a user. In turn, the application must provide BGP-MS with information about the user that it is monitoring.

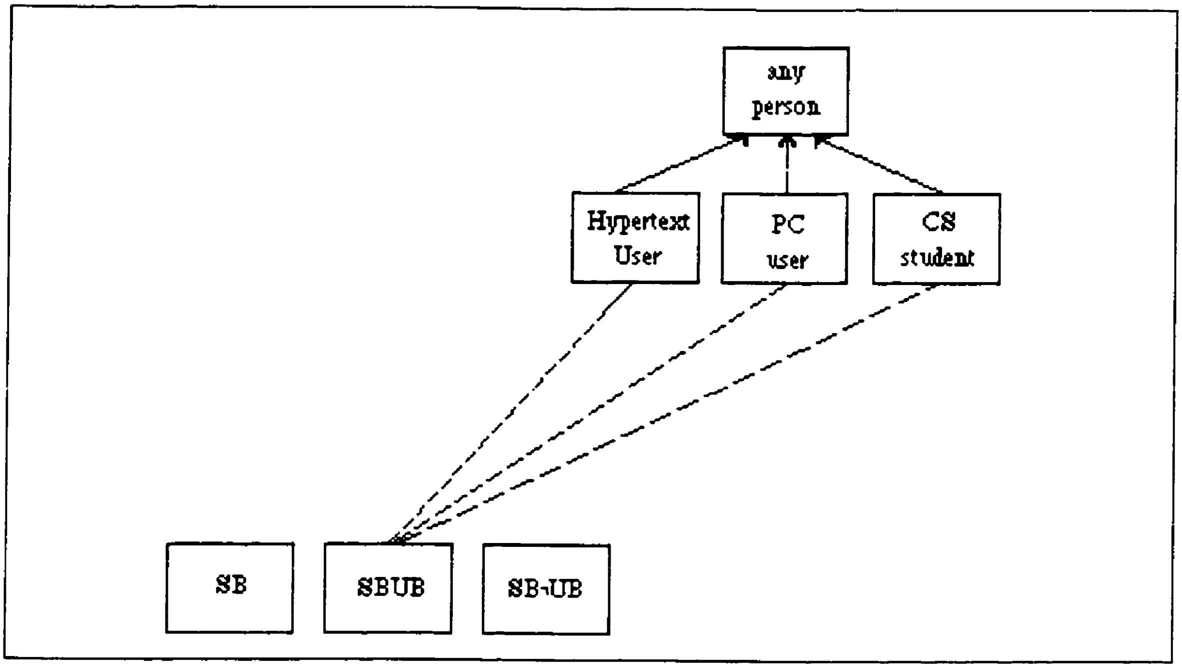


Figure 3-18. The partition hierarchy in BGP-MS (Kobsa *et al.*, 1994).

Figure 3-18 shows the user model architecture of BGP-MS (Kobsa *et al.*, 1994). The SB (System Believes) is the domain knowledge partition. SBUB (System Believes User Believes – assumptions about what the user knows) and SB~UB (System Believes User Doesn't Believe – assumptions about what the user does not know) partitions make up an individual user model. The SBUB partition is inherited from one or more stereotypes.

The shell supports a sophisticated stereotype mechanism, which allows a user model developer to specify a stereotype hierarchy with rules that are associated with activating and retracting the stereotypes. The stereotypes are constructed using a graphical interface. This helps a system builder to see the relationships between the various stereotypes. The system also checks the consistency of the structures that have been created. Domain-specific inference rules may be executed upon the user model update (Carberry & Kobsa, 1994). The graphical interface provides a user model developer with partially completed rules and applicable options in order to make the creation of the inference rules easier.

The user models are implemented in a knowledge representation language (SB-ONE) and in predicate logic. SB-ONE is a language that is similar to KL-ONE (Brachman & Schmolze, 1985), a frame-based language that uses semantic networks. The BGP-MS shell supports backward reasoning (first-order resolution) and some limited forward reasoning.

3.3.8 GUMS

GUMS (General User Modelling System) (Finin & Drager, 1986; Finin, 1989) is a domain independent user modelling shell that is based on Prolog. The system views a user model as a deductive database that is acting in conjunction with an application system (Finin, 1989). The specific application is responsible for the acquisition of the initial stereotypes for users and the information gathered about users. GUMS maintains the model by using three types of default reasoning: stereotypical reasoning, explicit default rules, and failure as negation. Every application has a separate knowledge base that consists of a collection of the stereotypes that have been organized in a taxonomy and a collection of models for the individuals. Each stereotype is composed of definite and default facts and rules. If a definite fact is contradicted, then moving up the hierarchy chooses a new stereotype, and a default fact is simply updated. The rules can be used to derive new information, both definite and default, from the currently believed information about a user. An initial version of GUMS has been implemented, and several limitations including its inefficiency are being addressed (Finin, 1989).

3.3.9 Intelligent Tutoring Systems

User modelling in general software systems and student modelling in intelligent tutoring systems (ITS) are closely related; approaches to intelligent tutoring systems have significant bearing on user modelling in general. The goal in building intelligent tutoring systems is that a system will become aware of the knowledge it contains as well as what a student knows (Kass, 1989). The main difference between an ITS and a user modelling system is that a student does not normally know what he or she is seeking to achieve in an ITS. It is easier to go outside the capabilities of the system by answering a question incorrectly. The method that has been adapted by these systems is to use two models: a student model and an expert model. The system attempts to get the student model to match the system's expert model.

WUSOR III (Browne *et al.*, 1990b) is an advisor for the game (and toy AI problem) "Hunt the Wumpus". WUSOR III uses a simple technique known as the overlay model (Carr & Goldstein, 1977;). The student's knowledge is viewed as a subset of the expert's knowledge. Overlay modelling works well in systems that have relatively simple, surface level knowledge (Weber & Specht, 1997). Using the overlay modelling technique in user modelling systems results in several problems. The system authors must predefine all of the knowledge, including all of the possible links within the knowledge. The system is also unable to deal with knowledge outside of the expert model.

3.4 Conclusion

In much the same way that the Artificial Intelligence community has decided that no one knowledge representation technique is right for every problem, User Modelling researchers are arriving at the same conclusion. Every individual technique has certain advantages and disadvantages; the underlying characteristics of the system that is being developed will often dictate the form of the user modelling module. The work described in the remainder of this thesis focuses on adapting combinations of the techniques described in this chapter, which allow the system representations to function pragmatically.

Until very recently, the major question that has been posed to user modelling researchers has been "Can you represent the information that is needed in order to adapt the system in a way that the user would prefer, and would find to be helpful and natural?" This question forms the basis for the field of user modelling, and will not cease to be any less important in the future. However, a second question is only now beginning to be asked: "Can you keep the overall costs (development costs, time, and run-time performance) low enough to make the development effort viable?" Answers to this question will allow user modelling research to move out of the lab and into the marketplace. The adaptive systems described here, while all attempting to answer the first question, have all been almost entirely research-oriented, so the second question has been largely ignored. It is only recently that actual users have been involved in assessing the usability of systems at all. Actual user involvement in the system design has helped to clarify an individual system's goals and avoid developing sophisticated solutions to the wrong problems.

This chapter has briefly examined a few well-known adaptive systems to determine their underlying representations and basic methodologies. They not only demonstrate the range of approaches of user modelling but are also considered to be the "classics" of the field. Kobsa (1993) and McTear (1993) provide excellent overviews of relatively recent work in user modelling and adaptive systems for readers who wish to examine these issues in detail. Some later systems will be presented in the context of supporting design and implementation choices for the adaptive system described in Chapters 5 and 6. The following chapter sets the stage for the development of this adaptive system by examining the domain that has been selected and by describing the underlying application, the Tax and Investment Management Strategizer.

CHAPTER 4

TAX AND INVESTMENT MANAGEMENT STRATEGIZER (TIMS)

4.0 Financial Planning Software

Financial Planning software consists of tools that assist people with creating personal financial plans for their future. The products typically focus on either wealth accumulation or retirement planning, and they range in levels of sophistication from simple illustrators and calculators to comprehensive computerized plan generators that integrate data from other software packages. Object-oriented, knowledge-based financial planning software products are just beginning to enter the marketplace in order to keep pace with the growing demand for more comprehensive and powerful analysis tools. These products are becoming much more sophisticated and flexible. They often have significant graphics and reporting capabilities. Dynamic financial planning software products are used as a lifestyle modelling tool to explore various client "what if" scenarios in real time (Fossay & Sawchuk, 1994).

The increase in activity in sophisticated software development for this domain is partly explained by the effects of the deregulation of financial institutions. The boundaries of the four tiers of the financial services sector are blurring. Commercial banks, insurance companies, trust companies, and brokerage firms now find themselves offering very similar products and services. These institutions are facing several of the same issues:

- Intense competition from both domestic and foreign sources.
- An overload of dynamic data.
- A shortage of experienced staff.
- A complex regulatory environment.

These issues are driving a need for financial planning software that addresses all of the major areas of financial planning such as: retirement planning, insurance needs analysis, estate planning, and cash flow analysis. Consumers are increasingly demanding more comprehensive financial planning services as the affluent "baby boomers" reach their peak earning years and contemplate retirement. Software providers have recognized this need for solutions. Corporations that are seeking a competitive edge are expressing their need for simulation and modelling systems that model incomes, assets, loans, savings, redemptions, goals, and insurance. More sophisticated software products also incorporate heuristics that are gathered from financial planning experts. Software products that provide expert level analysis of these heuristics require the use of knowledge-base technology.

Financial institutions view financial knowledge-based systems as a way to manage the marketplace's issues and challenges by documenting valuable knowledge in an easy-to-use tool. Indeed, the documentation of an expert's knowledge and the distribution of this valuable and scarce knowledge to less experienced users are two of the largest benefits of knowledge-based systems.

This distribution of knowledge is an important consideration in the financial services industry. As forces in the financial marketplace move with increasing speed, corporations need to train people to meet new challenges. The shortage of highly trained people has become severe. Less experienced staff are required to take on more responsibilities. Once they are trained, junior staffs are also subject to high turnover. It is important for organizations to invest in systems that will help them to retain their investments in knowledge and training.

4.1 The Tax and Investment Management Strategizer (TIMS)

The Tax and Investment Management Strategizer (TIMS) system is a sophisticated commercial knowledge-based software system for financial planning that was developed by Emerging Information Systems, Inc. (EISI) of Winnipeg, Canada. The current version of this software is used in the Canadian and US marketplace by banks, insurance companies, trust companies and brokerage firms. TIMS allows a financial planner to analyze the financial status of a client and to demonstrate and evaluate various financial planning strategies in order to improve the client's financial situation. The planning process includes the use of information such as (1) assets, liabilities, incomes, and expenses; (2) a number of applicable planning strategies; (3) possible interactions among planning strategies; and (4) objectives set by the client. The system's objectives are to maximize the effectiveness of expert planners and to increase the effectiveness of non-experts in the application of financial planning strategies. The latter enables individuals who have limited training in financial planning to employ both simple and more complex financial planning strategies.

TIMS uses a desktop metaphor and operates with datasets known as *plans*, which represent the financial situation of a client at a particular time. A TIMS user creates a plan through a series of data entry dialogs, and the system creates a graphical display of the client's financial plan in a dialog window that is known as the *Plan Window*. A financial planner can then evaluate and implement numerous financial planning strategies to produce alternative plans.

Much of the user's interaction with TIMS is performed through system components known as *strategies*. Strategies implement the financial planning techniques that human planners suggest to their clients or those that clients already have in place that properly reflect their overall financial pictures. A simple example of a strategy is the Regular Savings strategy. Implementing this strategy allows the user to select the asset that the savings are made to, to determine when the saving occurs, and to enter the amount of the savings. For example, a client might save $100.00 per month to their stock portfolio from the present time until retirement.

Nineteen different strategies are implemented in the current version of TIMS, including a variety of savings strategies, debt reduction strategies, and asset redemption strategies. Each of these strategies is an abstraction of numerous individual system actions. That is, each of these financial planning strategies can also be implemented directly by a user who is familiar with the system when he or she inserts the appropriate transactions (Future Returns, Buy, Sell, or Transfer records) at the detailed level of input (Figure 4-1). Initially, novice users do not need to be exposed to this level of detail, as it is confusing to them. Strategies provide users with an intuitive method of implementing financial planning tasks, and they free users from excessively detailed input. Strategies are invoked at a high level and they automatically contribute their expertise by defining various low-level transactions, which are then automatically carried out upon a specific date or event.

The standard level of menus restricts access to the more complicated system functions and does not even provide access to the Detailed Assets dialog (Figure 4-1) that allows more detailed input. Instead, strategies provide a more structured environment that is more intuitive for both new TIMS users and less-experienced financial planners and which insulates these users from the underlying aspects of a very complex software system.

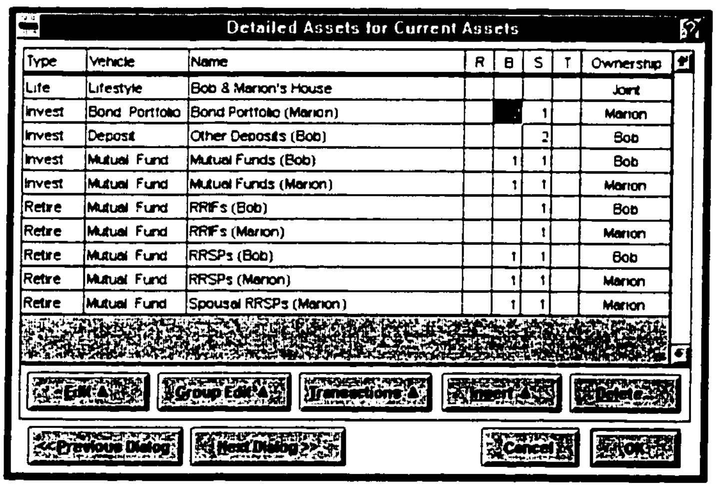


Figure 4-1. The Detailed Assets dialog.

A simple example illustrates the differences between implementing a series of data transactions through strategies and entering data at the detailed level of input. As mentioned earlier, one of the basic TIMS strategies is the Regular Savings strategy. At the Detailed Assets level, the user will enter a Buy record for the appropriate asset and define the interval and amount of the purchase through a series of dialogs. To perform the same task using a strategy, the Regular Savings strategy is selected and the appropriate asset is selected from a pull-down menu. The focus is on the *result* – the implementation of a savings program and not on the *process* – the insertion of a Buy record. The system inserts the Buy record automatically, so the actual processing that is done by the system is identical to performing the individual tasks at the Detailed Assets level. The Regular Savings strategy simply provides a different view of the data at a higher conceptual level.

Strategy dialogs also provide the user with a better overview of the total number of assets that are involved in the various strategies. In the Detailed Assets dialog (see Figure 4-1) there is an overall view of all of the assets and the appropriate Future Returns (R), Buys (B), Sells (S) and Transfers (T) for each asset. However, it is not possible to determine at a glance whether a Buy record is part of a Regular Savings program in which the client puts $150.00 into a Stock Portfolio every month or a Buy record that requires the client to use an inheritance for a lump sum purchase of that same asset. Through the use of strategies, all of the Regular Savings programs in effect would be displayed in the same dialog, along with the amounts and frequencies of each savings program (see Figure 4-2). The dialog displays two savings programs that will occur until the clients retire; Marion purchases $150/month of her Bond Portfolio and Bob purchases $1200/year of his Mutual Funds.

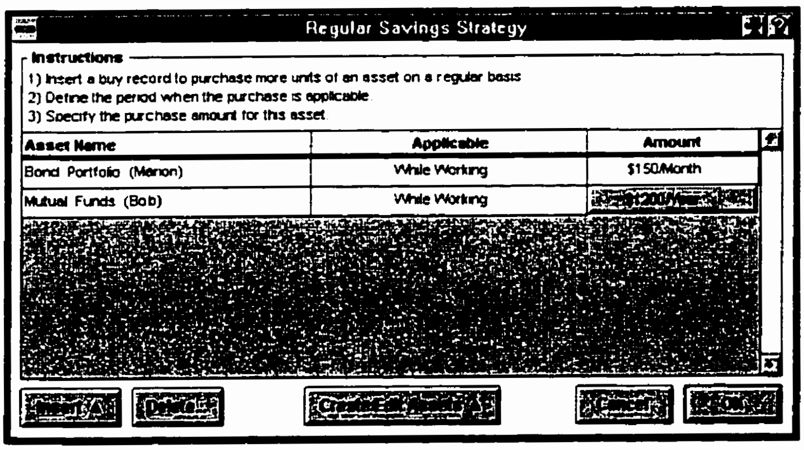


Figure 4-2. The Regular Savings strategy.

TIMS also provides three unique *Assistants*, which are knowledge-based components that provide intelligent support for system interaction. Assistants provide an even higher level of interaction by allowing the automatic analysis of plans and the generation of recommendations for system tasks. These assistants are the *Planning Assistant*, the *Cash Flow Assistant*, and the *Strategy Assistant*. Each assistant performs a unique supporting role. For example, users may analyze and implement financial planning strategies in a variety of ways. As described above, expert users may either invoke built-in strategies, or they may actually perform the detailed purchases and redemptions directly in the system. Novice TIMS users are encouraged to use the *Planning Assistant* (described in detail below) for the analysis of tentative strategies and for final plan creation. The Assistant components of the system require direction from users in order to implement strategies because the final decision rests with the planners. Planners would not accept a system that dictated generic solutions. Financial planning is very subjective and the system would be rejected if it were too intrusive or rigid.

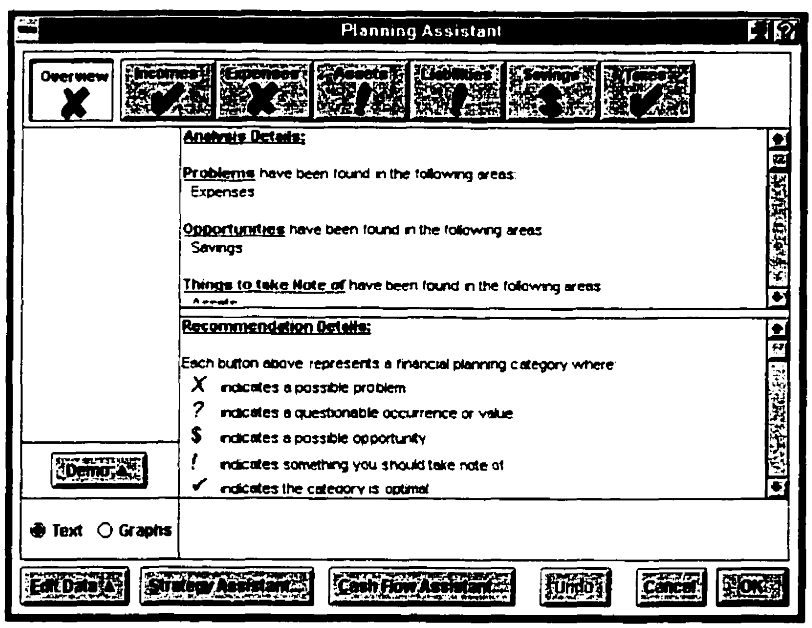


Figure 4-3. The Planning Assistant showing the overall analysis of a plan.

The Planning Assistant (Figure 4-3) provides a type of functionality that is unique to the financial planning software industry. It is designed to help automate common financial planning tasks, and it offers practical planning advice that is calculated to help a user maximize a client's overall financial situation. Each plan undergoes an analysis of its incomes, expenses, assets, liabilities, savings programs, and taxes. The analysis of any individual category is displayed in the top portion of the window while the customized recommendations are displayed in the bottom portion. If appropriate, strategies recommended by the Planning Assistant are displayed in quick access buttons in the panel below the recommendations. The user has access to a menu of data entry dialogs from within the Planning Assistant that allow him or her to modify either the client's data or the strategies that were used and to invoke the other Assistants that TIMS provides.

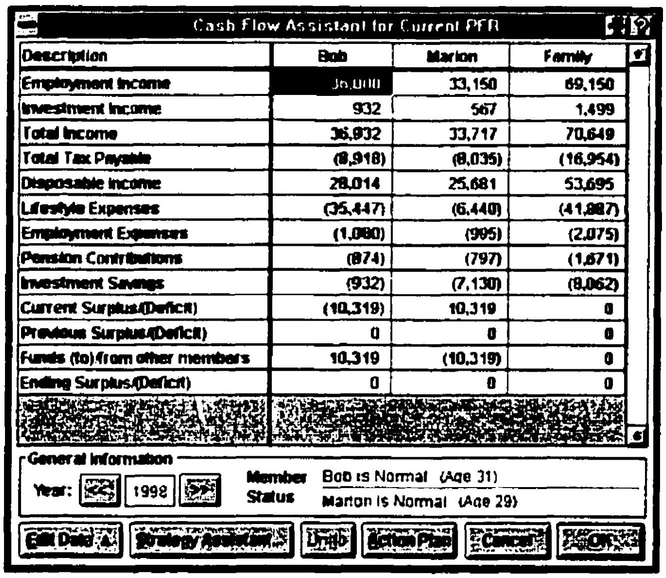


Figure 4-4. The Cash Flow Assistant is showing the detailed cashflow for a specific year.

TIMS also provides intelligent *Cash Flow* and *Strategy Assistants* that perform analogous functions for their particular aspects of the system. The Cash Flow Assistant (Figure 4-4) provides a complete analysis of a client's cash flow for any given year. This analysis can be based on the current plan year or it can be projected into the future. The Cash Flow Assistant allows a financial planner to make changes to the client's current plan (by entering data directly or through the use of strategies) and then view the results. Each cell represents the total value for a category, which can be selected to display the intermediate values that contribute to the total. It is also possible to display (through access to the action plan, described below) the ongoing transactions that are necessary to meet a user's investment objectives. These might be transactions that a user has entered into the system or transactions that have been inserted automatically as a result of implementing the TIMS financial planning strategies.

For example, a planner might implement a Registered Retirement Savings Plan (RRSP) Maximizer strategy for a client. In the Canadian tax system, money placed into Registered Retirement Savings Plans generates an immediate tax deduction and accumulates retirement savings tax-free until the savings are withdrawn at retirement. The Canadian government places strict guidelines on the annual contributions that individuals may make to RRSPs. These amounts are based on their incomes and other concurrent retirement savings. The RRSP Maximizer strategy determines the maximum contribution to an RRSP asset, providing the clients have some surplus cash. It tracks the RRSP deduction room, any RRSP overcontributions, and the amount of surplus cash that the client family has in each year. The user inserts the strategy and directs any surplus funds to one or more of the client's registered assets. Because the user does not specify the amount of these purchases, the system-calculated amounts are provided to the user in a report called the *Action Plan*.

The Action Plan provides a user with the savings programs, transfers, and redemptions that the clients must carry out in order to accomplish their overall financial goals. In Figure 4-5, an Action Plan for a particular year (1998) is shown. In order for this family to achieve the system-created financial plan, they would have to continue the Regular Savings that they currently have in place ($150/month), and follow the savings and RRSP contributions recommended by system ($167/month and $397/month, respectively).

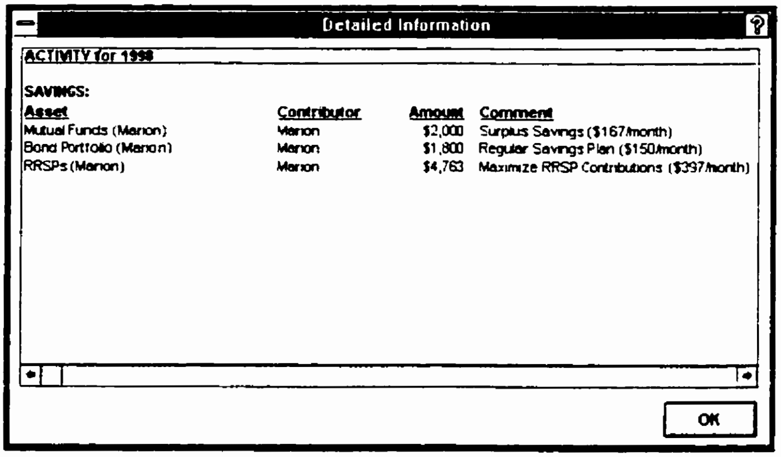


Figure 4-5. The Action Plan detailing the savings programs necessary for 1998.

The Strategy Assistant (Figure 4-6) is designed to help a user select and apply multiple financial planning strategies. Applying multiple strategies in a single dialog saves time by limiting the number of necessary recalculations, and it is also useful when a financial planner requires a quick overview of the number of strategies that are currently being applied in a particular plan.

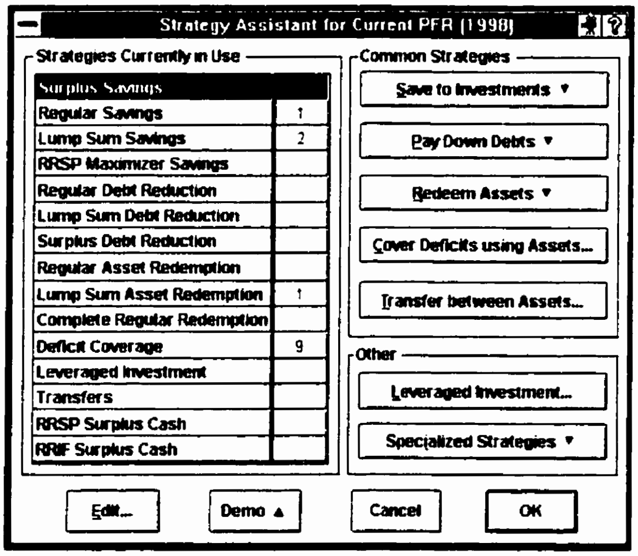


Figure 4-6. The Strategy Assistant.

In the example given in Figure 4-6, five strategies are in place for the Current PFR (1998) plan. Current Personal Financial Review (PFR) is often the first plan that is created in the system and it represents the client's current financial situation before any improvements are made. Three different savings programs are implemented: Surplus Savings, Regular Savings, and Lump Sum Savings. A Lump Sum Asset Redemption strategy and a Deficit Coverage strategy are also implemented in this plan. The column on the right side of the window lists the number of records in each strategy. For example, the Surplus Savings strategy contains two records. This means that two assets will be purchased with any surplus monies that the family has. The entry can be edited to determine the details of these transactions. The Surplus Savings strategy can also be entered through the Save to Investments button. Both methods open the Surplus Savings strategy dialog. Strategies can be added, deleted, or modified at any time.

Once the desired result has been achieved by implementing strategies and modifying the initial client data, a completed financial plan can be created by using built-in Document Packages. These Document Packages can be used to export one or more reports or graphs, which have been produced by the system, into a template to create a Rich Text Format (RTF) file. This type of file can be read by most current word processors. This complex process is made easier by allowing users to simply drag-and-drop icons that represent client plans onto icons that represent currently available Document Packages.

4.2 Summary

The basic construction of the TIMS program provides a sophisticated tool that can provide complete support to a computer literate, experienced financial advisor. However, one of the major difficulties with the system is that many TIMS users are not members of this group – and in fact, the group itself is a very select one. Like other commercial systems that are functioning in complex domains, TIMS must support a diverse range of users along two major dimensions. While many financial planners still have little computer expertise, some are sophisticated computer users who do not need lengthy demonstrations or extensive explanation facilities. Just as importantly, many TIMS users are expert financial planners with extensive knowledge of the domain, and they use TIMS mainly for convenience. Others will have only limited financial planning skills and will rely on TIMS to fill gaps in their own knowledge. Even among expert financial planners, the extent of their knowledge of the domain will vary from area to area. For example, a planner who is an expert in asset allocation may have less knowledge about life insurance and estate planning issues.

A study (Fossay & Sawchuk, 1994) that was conducted on the first commercial version of TIMS (released in 1995) confirmed the power and sophistication of the system but identified concerns with the complexity of the system and the impact of the complexity on novice users. Focus groups were formed from various representatives of the banking industry in order to confirm that the functionality of the program was comprehensive and that it fulfilled a need for financial planning software in the banking industry. Focus group participants generally felt that the level of planning offered by the program was beyond the expertise of their staff. One recommendation outlined in the report was that, "The product needs to be tailored to mid-tier and lower-tier needs in the minds of the bankers in our focus groups" (p. 30). While the functionality of the system received high praise, the participants in the study were concerned that potential users would be overwhelmed by the current scope of capabilities offered by TIMS.

This concern, along with the breadth of user experience that must be supported and the goals of TIMS itself, led the software developers to decide to develop an adaptive user interface for TIMS. Formative evaluations, which were possible as the result of a large number of first-time users going through an introductory TIMS training course, lent additional support to this decision. Selected users were observed and interviewed, and the difficulties that were experienced by the users were examined. There was a verified need for a user interface that would adapt to a user's level of sophistication and the complexity of the problems that were being addressed.

The use of a minimalist user modelling component to implement this adaptive user interface arose from commercial concerns and from the issues described in Chapters 1 and 2. These issues were essentially a desire to bring the benefits of user-adapted interaction to the system at minimal cost and with a minimum of commercial disruption. The design and implementation of this user modelling component, and the nature of its adaptations, are described in the following chapter.

CHAPTER 5

THE IMPLEMENTATION OF PERSONALIZED-TIMS[13]

5.0 Supporting User-Adapted Interaction in TIMS

The initial stages of developing a user modelling component for TIMS focused on the adaptations for novice users that would help them to navigate through the system and perform the required tasks. This was because TIMS was developed in collaboration with high-end expert financial planners and had already been structured to reflect the organization of material that was most helpful to them. This type of user recognizes that the system's power is essential and feels that it is worthwhile to spend time learning to use it. The researchers' focus on novices does not eliminate these expert users from consideration. The usability of the software by more experienced users must not be limited by the adaptations that are necessary for the support of novice users.

A number of areas where novice support could be increased in TIMS via a minimalist user modelling component were identified through interactions with users during the testing of the initial release of the system. Input was gathered through informal user

[13] P-TIMS was implemented by Murray Sneesby as part of his research program at the University of Manitoba, Canada.

interviews, training sessions with novice users, and telephone support calls. Two guidelines were used for the selection of adaptations for the TIMS user interface. The first was that the selected adaptations had to be intended to assist the novice user. The second guideline was that these adaptations were possible with only a minimal user modelling component to support them. The use of this selection process created four major categories of user adaptations:

- The addition of animated demonstrations of tasks considered to be more difficult for new users.
- A Strategy Interaction Detector.
- A Strategy Recommendation component.
- Assistance with the data entry process.

These adaptations were implemented in several stages and resulted in the development of Personalized-TIMS (P-TIMS), an adaptive version of the TIMS system.

5.1 A Minimalist User Modelling Component for TIMS

The basic architecture of the TIMS user modelling component (UMC) is depicted in Figure 5-1. It consists of a long-term user model (stored in a user model database); a task model containing descriptions of a subset of tasks in the system (such as rankings of the complexity of strategies and Assistants); and an application model containing information about the relationships between a subset of the tasks in the system (such as information about the strategies or combinations of strategies that are equivalent to each other). The inference engine contains simple rules about the relationships between these three models and the possible adaptations that can be performed on the application and the interface.

As Figure 5-1 illustrates, the P-TIMS user modelling component has a two-way exchange of information with both the user interface and the application. The UMC receives an ongoing description of the user's interaction with the system. From the interface, the UMC receives the user's interaction history, as well as explicit changes that have been made to the user model via the system's Preferences dialog, while the application's current status is received from the application itself. Information that is sent from the UMC to the other components causes the particular adaptations to be realized. These adaptations resulted in additional or modified dialogs, animated demonstrations to the interface, and modifications or limits to the functionality of the system.

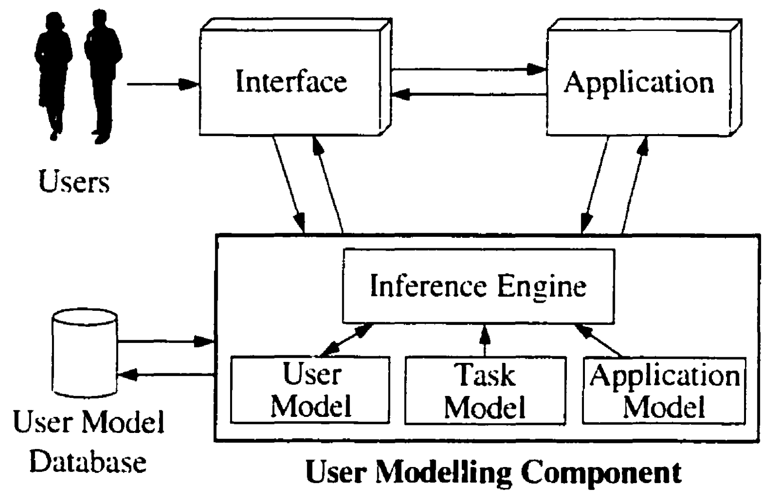


Figure 5-1. The user modelling component architecture of Personalized-TIMS.

Each P-TIMS user has a unique individual user model that is generated the first time the user logs into the TIMS program and is reused for subsequent entries into the system (Figure 5-2).

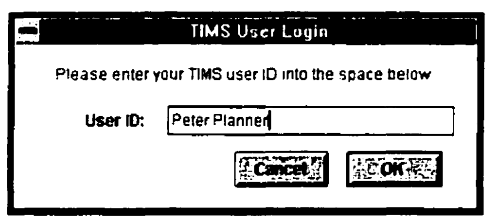


Figure 5-2. The initial entry to the P-TIMS system dialog.

The term *user model* is not used directly in the system because many users are already familiar with the concepts of a User ID and a User Preferences option through exposure to other software packages. This was done primarily to avoid introducing needless additional concepts to the user, but it also keeps any potential differences between the adaptive and static systems as small as possible. If the system does not have an entry for this particular user in the user model database, the system prompts the user to create one (Figure 5-3).

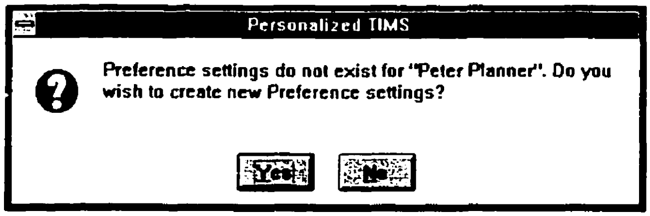


Figure 5-3. The prompt for the creation of a new user model.

The user model is initially based upon a set of stereotypes that are triggered upon a new user's first registration and login to the system. A simple novice or expert designation for a user was not adequate for the TIMS program. This was due to the nature of the domain: in the domain of financial planning, it is not uncommon to have users with widely disparate characteristics. Many very experienced planners are not computer literate, but this is changing rapidly due to increased consumer demand for more sophisticated planning and to changes in the corporate administrative infrastructures. At the same time, newer financial planners may have a lot of experience with other software packages and will be able to learn the mechanics of TIMS quickly, but they may not fully understand the underlying principles of financial planning.

The set of initial available stereotypes is based on the two-dimensional diversity of TIMS users. Users are classified as novice or experienced TIMS users (NTIMS, ETIMS) and as either novice or experienced financial planners (NFP, EFP). These four categories were chosen as a reasonable representation of the TIMS user population in order to keep the number of stereotypes to a minimum. Four possible generic user models can be created from these stereotypes, as illustrated in Figure 5-4.

	Financial Planning Expertise	
TIMS Expertise	*Novice*	*Experienced*
Novice	NTIMS + NFP	NTIMS + EFP
Experienced	ETIMS + NFP	ETIMS + EFP

Figure 5-4. User model created as a result of the initial user questionnaire.

One of the two stereotypes in each category is triggered during the registration process (Figure 5-5), and the two stereotypes are appended to form an initial working user model. During registration, new users identify themselves as Novice, Intermediate, or

Experienced Windows and TIMS users. Novice and Intermediate users are mapped to the NTIMS stereotype, while experienced users are mapped to the ETIMS stereotype.

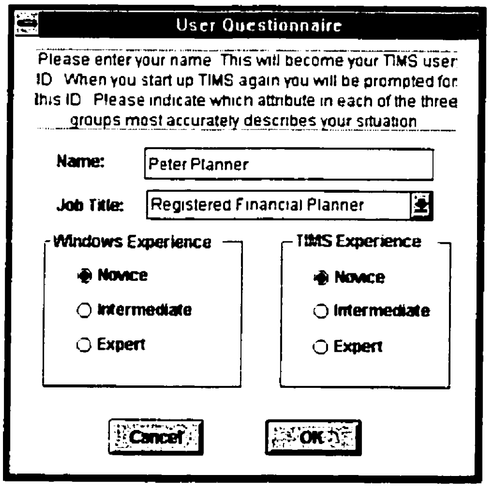


Figure 5-5. Questionnaire presented to a new user in the registration process.

Job Title	Stereotype
Certified Financial Planner	EFP
Registered Financial Planner	EFP
Chartered Accountant	EFP
Accountant	NFP
Investment Counselor	NFP
Marketing Assistants	NFP
Life Insurance Agent	NFP
Financial Services Representative	NFP
Tax Advisor	EFP
Help Desk	NFP
Other	NFP
TIMS Developer	EFP

Figure 5-6. List of job titles and the mapping to the stereotypes.

The assessment of financial planning experience and triggering of the EFP or NFP stereotype is done through the user's job title, which is selected during the registration process from a menu of generic job titles. Initially, the two attributes that vary between the two financial planning stereotypes are the User Complexity Rating and the Financial Planning Expertise. Figure 5-6 contains the current list of supported job titles for the P-TIMS system. The advantage of using a job title as an indicator is that it is a very simple parameter that is easily incorporated into a registration procedure. A generic job title that has been selected from a menu allows the system to make reasonable predictions about the financial planning expertise of the user, but it requires little additional effort from the user. Although job title was an appropriate trigger for P-TIMS, other simple variables, such as professional designation, location, or rank, might be appropriate in other systems.

One disadvantage of this scheme is that a job title may not be very specific. It is important to consider the implications of this: are users with the same job title a homogeneous group? This is clearly not the case. Limiting the number of stereotypes lessens this concern because a smaller number of stereotypes are, by their design, more general or coarse-grained. More elaborate user modelling systems with a larger number of stereotypes might have difficulty mapping the job title to one specific stereotype. The possible set of job titles could be extended, but this introduces another potential problem: different corporations have their own conventions for naming the employee's positions. A similar job title in two separate companies could easily have very different job requirements. A decision has been made to make the job titles relatively generic and to keep the number of stereotypes small, while understanding that misjudgments will occur. This disadvantage can be lessened by making adjustments to the user model based on user transactions and by allowing the user to exercise control over his or her user model.

The individual components of the Financial Planning and TIMS experience stereotypes are displayed in Figure 5-7, while Figure 5-8 illustrates an example of a complete initial user model. The user model contains information about the user's interaction with each of the TIMS strategies and Assistants. This allows a simple interaction history to be available to the user and to indicate the strategy and Assistant demonstrations that have been previously viewed. The user can also view specific settings for the various adaptations that the user modelling component supports.

	Experienced Financial Planner	Novice Financial Planner
User Name:	User supplied	User supplied
Job Title:	User supplied	User supplied
Frequency of Use:	0	0
Date of Last Access:	Today's date	Today's date
Last Update	Today's date	Today's date
User Type:	EFP	NFP
Financial Planning Expertise:	High	Low
TIMS Experience:	User supplied	User supplied
Windows Experience:	User supplied	User supplied
User Complexity Rating:	3	1
TIMS Strategies:		
strategy name: *		
counter:	0	0
date of last use:	Date	Date
demo counter:	0	0
date of last demo:	Date	Date
demo refusals:	0	0
Assistant Modules:		
name: **		
counter:	0	0
date of last use:	Date	Date
demo counter:	0	0
date of last demo:	Date	Date
demo refusals:	0	0

*For each individual strategy (19)
**For the Planning Assistant, Strategy Assistant and Cash Flow Assistant

	Experienced TIMS	Novice TIMS
User Settings:		
Show Task Demonstrations	False	True
Show Strategy Interactions	True	True
Suggest Strategies	True	True
Data Entry Assistance	False	True

Figure 5-7. The attributes of the four P-TIMS stereotypes.

The hypothetical user, Peter Planner, whose user model is depicted in Figure 5-8, was originally a novice TIMS user (user self-assessment) and a novice financial planner (triggered by the selection of the Company Representative job title). He has used the system for 11 days and has been upgraded by the system to an intermediate complexity rating. The UMC added information to the user model history when the user selected each of the TIMS strategies and Assistants. The user adaptations are still in place, with the exception of the task demonstrations. Notice that this user has viewed at least three demonstrations: two demonstrations for the RRSP Maximizer strategy and one for the Planning Assistant (the only two items visible in this figure) before terminating this option. The mechanisms providing user model maintenance and the various adaptations themselves will be described shortly.

User Model	
User Name:	Peter Planner
Job Title:	Financial Services Representative
Frequency of Use:	11 Days
Date of Last Access:	Apr 12, 1998
Last Update:	Mar 30, 1998
User Type:	NFP
Financial Planning Expertise:	Low
TIMS Experience:	2
Windows Experience:	1
User Complexity Rating:	2
TIMS Strategies:	
strategy name: *	RRSP Maximizer
counter:	8
date of last use:	Apr 1, 1998
demo counter:	1
date of last demo:	Mar 10, 1998
demo refusals:	0
Assistant Modules:	
name: **	Planning Assistant
counter:	8
date of last use:	Apr 12, 1998
demo counter:	1
date of last demo:	Mar 10, 1998
demo refusals:	0
User Settings:	
Show Task Demonstrations	False
Show Strategy Interactions	True
Suggest Strategies	True
Data Entry Assistance	True

*For each individual strategy (19)
**For the Planning Assistant, Strategy Assistant and Cash Flow Assistant

Figure 5-8. User model of a novice financial planner.

In addition to the individual user models, the UMC also incorporates *task* and *application* models. The task model (Figure 5-9) contains simple information about strategies and the more complex functions in the system. Although the primary focus is on the modelling of strategies and Assistants, other complex functions such as client report creation are also represented. The information that is recorded for each task in the model includes the following: an integer complexity rating, a Lotus ScreenCam[14] demonstration filename, two levels of explanation, and an appropriate index to the hypertext help system. Any function that is not represented in the task model is assumed to be at the base level of complexity (represented by the numeric value 1). Most novice users are expected to find these components to be reasonably straightforward once they have received a minimal amount of training and support. Complexity ratings were chosen as a simple tool to determine if a task was appropriate for the user's level of experience. In a more complex system, a task hierarchy, where only the expert users would have access to the entire hierarchy, might be more appropriate (Vassileva, 1994).

In the current implementation, all user levels are able to access all system functions, but the UMC has the ability to disable tasks based on the task model. This is useful for supporting task limitations that are based on user category in situations where it might be useful to disallow certain kinds of financial analysis. For example, a front-line bank employee might be permitted to perform simple analysis tasks for a client, but would be required to refer a more complex analysis to another employee with the appropriate expertise and authority.

Task Model*	
Name:	RRSP Maximizer
Complexity Rating:	1
Demo Available:	"RRSPMAX.SCM /SCH"
Explanations:	
Low-level	explain[rrspmax].low
High-level	explain[rrspmax].high
Availability to user type:	All
Help Screen:	2:4610 3025
* One entry for each strategy and Assistant	

Figure 5-9. RRSP Maximizer strategy representation in the task model.

The user modelling component's application model contains specific information that provides support for the financial planning strategy recommendations and the analysis of

[14]Lotus ScreenCam is a trademark of Lotus Corporation.

interactions between financial planning strategies. Figure 5-10 illustrates a portion of its contents (see Appendices A and B for complete listings). In the current version of the system, sixteen situations of equivalent strategies are represented, and seventeen possible combinations of strategy interactions can be detected. The detection of interactions is an important task because certain combinations of strategies are not a potential problem unless they are active for the same asset or assets and are applicable during an overlapping period of time.

Application Model (a portion)

Equivalent Strategies:

Location:	Detailed Assets
Transaction:	Future Returns
Selection:	N/A
Asset Type:	All
Recommendation:	Investment Returns Strategy
Location:	Detailed Assets
Transaction:	Buys
Selection:	Surplus
Asset Type:	RRSP
Recommendation:	Surplus RRSP Cash Strategy
Location:	Detailed Liabilities
Transaction:	Principal Modifier
Selection:	Periodic
Asset Type:	All
Recommendation:	Regular Debt Reduction Strategy

Strategy Interactions:

Strategy_combo:	Surplus Cash Savings & RRSP Maximizer & Surplus Debt Reduction
Asset Applicable Period Overlap:	No

Explanation:

"This combination of strategies will be performed in the following order: 1) RRSP Maximizer, 2) Surplus Debt Reduction and, 3) Surplus Cash Savings. There may be less surplus cash for debt reduction because of the RRSP Maximizer Strategy. Savings will take place only if there is a remaining surplus following debt reduction."

Strategy_combo:	(Surplus Cash Savings or Regular Savings or Lump Sum Savings) & Complete Regular Redemption
Asset Applicable Period Overlap:	Yes

Explanation:

"The asset(s), *asset_names*..... have both a savings strategy and the Complete Regular Redemption strategy associated with them. This may prevent these assets from being fully redeemed by the end of the period."

Figure 5-10. Representative entries in the application model.

The inference engine sub-component of the UMC contains the processing components that support the events that occur during system interaction with the user model. These include: user model creation; user model retrieval from, and storage to, the database; initialization and re-classification of the user; updates and modifications to the user model made during system usage; triggering of appropriate demonstrations and dialogs; and the interaction of the users with these dialogs. In order to simplify the processing as much as possible, both the task and application models are static and, they are not modified by the UMC. They are intended to provide information about a specific application's structure and functionality.

This simple architecture contains an important underlying assumption: *the user model is the property of the user, not the user modelling component.* One design objective of the adaptive system was to make the user model visible and modifiable by the user. This was done in order to take advantage of the benefits outlined by others (Cook & Kay, 1994) which include: the user's right to gain access to information about him or herself; the programmer's accountability; the user's ability to correct and verify information; and the importance of making complex systems more comprehensible. In P-TIMS, the user model attributes have been made accessible to the user through a Preferences dialog (Figure 5-11).

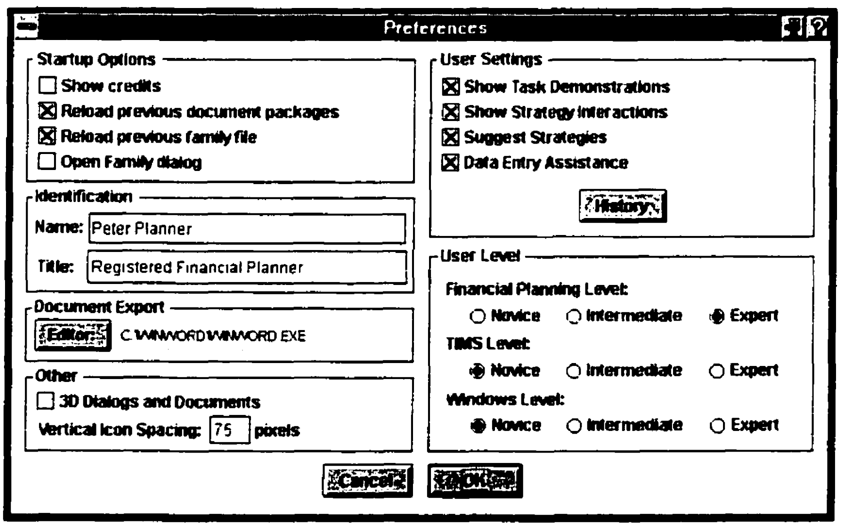


Figure 5-11. Preferences dialog showing all adaptations activated.

The Preferences dialog consists of: general user information, *user levels* for TIMS, Windows and Financial Planning experience; the *History* button containing the task interaction history, and the *user settings* that control the adaptations. The History dialog cannot be modified and is provided for informational use only (Figure 5-12). Each entry lists: tasks that were monitored by the UMC, the demonstrations that have been observed, the number of times that they have been observed, and the last date that they were triggered. The user settings can be individually activated or deactivated. Although the most significant updating of the user model occurs during the initialization phase of the system, the user can update his or her user model at any time.

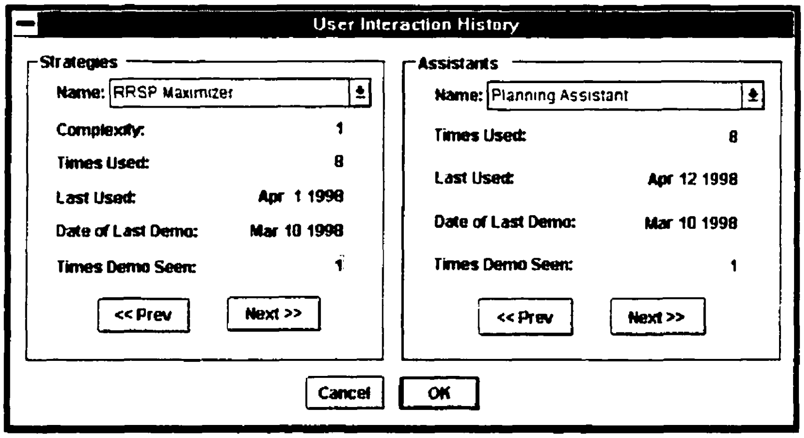


Figure 5-12. The user interaction history of the user model.

Other modifications occur when the user performs tasks that are represented in the task model. Reclassification of a user in the initialization phase is done only once per day in order to minimize potential problems with users repeatedly entering and exiting the program during initial experimentation. This design decision also incorporates an element of common sense, in that registering ten system uses over a period of ten days implies a different degree of system familiarity than the same number of uses in one day.

The user modelling component activates the user model maintenance procedures during the initialization of the system at the first session of each day (Figure 5-13). The user model is updated with a new access date, and the frequency of access attribute in the user model is incremented. The TIMS experience attribute is upgraded if the system has been used more than a threshold frequency and, it is downgraded if the system has not been used for a lengthy period of time. Other researchers have previously established that frequency of use was a major determinant of a subject's capability and that a significant decrease in user capability occurred after a break of some weeks (Maskery, 1984). The UMC then checks the history of the user's interaction with the system and updates the user model. For example, the user complexity rating for the user model will be increased if the user has implemented more than two strategies that were more complex than the last updated user complexity rating. The evolution of user models in this scheme will be described in further sections, within the context of the specific adaptations available in P-TIMS.

```
If the date has changed do the following:

    If the user has been away too long (currently 30 days)
        Downgrade TIMS experience level by 1 category
    If the user has the used the system long enough since the last update (currently every 10 sessions)
        Upgrade TIMS experience level by 1 category
    Examine all strategies/functions used recently (within the last 5 sessions) by the user
        If more than 2 are more complex than the user complexity level then
                Increase user complexity level by 1
        If all are less complex then
                Decrease user complexity level by 1
        Update financial planning expertise
    Update counter fields
```

Figure 5-13. The basic algorithm for update of the user model in the initialization phase.

Over time, the adaptive system diminishes into the original, non-adaptive system. That is, the adaptive system has a planned obsolescence. Because the user modelling component is focusing on the support of novice users through the system, the support is terminated with the user's permission once a user achieves a certain level of proficiency and has used the system for a predetermined period of time. The literature also supports this type of approach. Other researchers have suggested that "individualized systems are of the biggest importance for supporting a beginner in the learning phase and their importance fades when the user is able to apply the full range of options provided by the system" (Vassileva, 1996). This approach is also taken in the Microsoft Office '97 adaptive *Office Assistant*, which stops giving advice when the user has mastered the task at hand (Microsoft, 1998).

When the P-TIMS system determines that the user should be re-classified, a proposal is presented to the user that explains the implications of a reclassification and asks the user to confirm or reject the proposal. A comparable technique was used in an adaptive hypermedia system, HYNECOS; the system would only adapt at discrete points in time after the user had been informed (Vassileva, 1994). Because HYNECOS is a system in the medical domain it was felt that the user's acceptance of the system would be much better if the users had absolute confidence in the system – they knew what had changed in the system and why. This belief was reinforced through interaction with actual users in the domain, and the research team's preliminary experiences seem to strengthen this conclusion as well. In P-TIMS, different types of adaptations in the system have different termination points. The intermediate or expert user starts with only a few supports, which gradually erode once the system has been used for a period of time or once the user has decided to cancel the adaptations. In all cases, the user has ultimate control over the termination of supports.

The benefit of planned obsolescence of the adaptations in the P-TIMS system lies in its ability to minimize the problems that users have with systems that are inconsistent between users and over periods of time. These problems are minimized due to the fact that experienced users will tend to have what is essentially a non-adaptive system. These problems are significant and under-emphasized. They include the loss of shared experiences with other users when their system has personalized information other users may lack. For example, a common frame of reference would not exist with personalized newspapers that are filtered by an intelligent agent. One could not simply ask: "Did you see the article in the paper today?" Over time, readers would tend to become more and more knowledgeable about the topics that interest them and may not be as aware of the issues and topics that appear only on the periphery of their focus. A greater problem from a commercial prospective is that the user-specific adaptations could add a level of complexity to product support or help desk support of individualized systems. Even the informal networks of peer support, user communities, or "power users" that lend support to others may be jeopardized. If an adaptive system is designed to support collaborative work, the personalized views can't be too different or else they will effectively terminate the collaboration (Vassileva, 1994).

The architecture and basic functionality of the user modelling component have been described in general terms. A greater understanding is provided by examining the user modelling component within the context of the currently supported adaptations in P-TIMS. The adaptations were implemented in three major phases and are described in the following sections.

5.2 Phase 1: Animated Demonstrations

During training sessions and marketing presentations, it was noted that novice users, and particularly first-time users of TIMS, experienced a particularly high level of uncertainty while trying to perform new tasks. This uncertainty led to errors and wasted time and energy. Novice TIMS users and novice financial planners shared some characteristics, but they differed in the type of assistance that they needed in a system. Expert financial planners required assistance with mapping their domain knowledge into the system, understanding the types of functionality provided by the system, and learning how to perform specific tasks with it. Less experienced planners also needed some assistance in the form of suggestions on tasks to perform and extra confirmation on the successful completion of these tasks. These TIMS novices are best supported by demonstrations of tasks before they attempt them for the first time. For this reason, the first major phase of adaptations involved adding support for animated Lotus ScreenCam demonstrations.

This type of support is familiar to Microsoft Office users who have experienced the "Wizards" and animated Demos that are available in the help subsystems of software such as Microsoft Word[15] and Microsoft Excel[16]. HyPLAN, (Grunst, 1993) an adaptive hypermedia help system for Excel, found that demos of Excel operations recorded with an animation product were successful in assisting the user to learn the operation. It was anticipated that the result of this addition to the system would be a decreased number of errors, less uncertainty in performing a task for the first time, and increased user satisfaction with the system. An important consideration of this work was the validation of these hypotheses through empirical methods. Details of the experimental process and the findings that resulted from it are presented in Chapter 6.

[15] Microsoft Word is a copyright of Microsoft Corporation.
[16] Microsoft Excel is a copyright of Microsoft Corporation.

Lotus ScreenCam is a commercial software package that allows a user to record screen activity and save the result in a ScreenCam movie stand-alone executable file (Lotus, 1995). It was used to create animated demonstrations of significant TIMS tasks because of its ability to support recorded speech and text captions as well as visual presentation. Text captions were chosen over recorded speech as a tool for enhancing and clarifying the display. There were two main reasons for this: not all TIMS users have the necessary hardware support for audio presentations, and the use of recorded speech results in substantially larger files. The issue of file size was an important one due to the great variability in current and potential TIMS users' hardware (a common problem in delivering complex software systems). The complete adaptive system, which includes a relatively small number of movies, contains just over six megabytes of ScreenCam files. Narration would have made these files approximately 7.5 times larger.

Each animation provides for one of three possible supports: (1) the familiarization of the user with the overall structure of the TIMS system, (2) the familiarization of the user with the structure of a supporting TIMS component, and (3) a demonstration of how to perform a TIMS task. Each demonstration is generally less than two minutes in length and shows the user how to perform any function. The demonstrations were limited to two minutes because the pilot testing of early versions of the system revealed that this was the approximate maximum user attention span for a demonstration. A longer, more complete demonstration has the potential to bore a user or to cause the system to appear to be more complex.

The first of the above categories of support introduces the user to P-TIMS and the P-TIMS Preference's dialog. It includes a demonstration that explains the options in the adaptive system's Preferences dialog, which is the main controller of the adaptations made by the system. Its main purpose is to make the users aware of their direct control over the adaptations they experience. After the initial login session, the user is greeted by the Personalized-TIMS system dialog (Figure 5-14) which explains, very briefly, the features of the adaptive system.

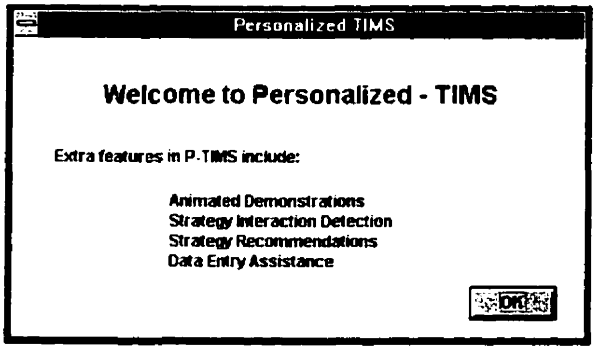


Figure 5-14. The initial greeting to the Personalized-TIMS system.

The user can then choose to view an animated demonstration about the Preferences dialog (see Figure 5-15). This demonstration briefly explains the differences between the original TIMS system and the P-TIMS adaptive system. Each of the different types of modifications, or User Settings, and the User Levels are explained, and the user is shown how to control them through the Preferences dialog.

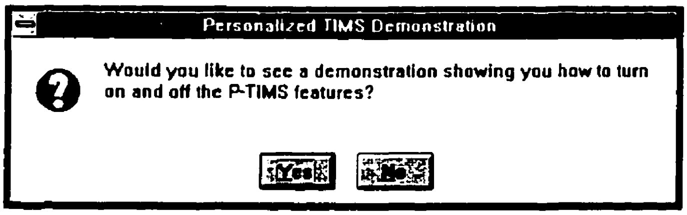


Figure 5-15. The dialog controlling the initial P-TIMS demonstration.

This first demonstration sets the tone; every subsequent demonstration follows the same format. Each one begins with a demonstration dialog and is followed by an introductory caption, a brief explanation of the component, an interactive demonstration of the method of use, the reactivation procedure, and the closing caption. See Appendix C for a sample of representative screen captures for a typical strategy demonstration. The user receives a consistent presentation from demonstration to demonstration.

Two additional demonstrations in this category, a new Windows user and a first-time TIMS user familiarization, were dropped from the final version of the adaptive system. These two demonstrations showed a new user basic information about dialog boxes, pull-down menus, icons, and the TIMS desktop. While all of these demonstrations are similar in that the user is presented with a dialog and can choose not to view the demonstration, it became obvious during the pilot testing of the adaptive system that a first-time P-TIMS user would be overwhelmed by the number of animated demonstrations that were immediately available. A better mechanism is needed in order to allow the user to make individual choices about the presentations.

The overall system structure demonstration category also includes two demonstrations that help to clarify the overall flow of data through the system. The UMC monitors two previously-identified, difficult transitions for new users and prompts the user at these points to view a demonstration that explains the next logical step in the user session. For example, following the initial data entry for the client, a user may view a demonstration that shows him or her how to copy an existing plan and invoke the Planning Assistant. A second demonstration is available to assist novices in performing the final export of data into a completed client report. These two demonstrations are elements of Data Entry Assistance, described in Section 5.4.

The second category includes demonstrations that are related to each of the Assistants described in Chapter 4. There are two important reasons to include animated demonstrations for each individual Assistant. First, each Assistant functions as a control centre for the system: The Planning Assistant, for analysis and plan modification; the Strategy Assistant, for implementing and monitoring strategies; and the Cash Flow Assistant, for monitoring the cash flow of a family. Demonstrations are also necessary because new users require brief explanations that help to familiarize them with the components' centralized-control interface.

The final category of demonstrations illustrates the application of each of the nineteen strategies that are implemented in the system. As discussed in Chapter 4, TIMS offers several strategy dialogs that allow the financial planner to assign transactions which can be automatically carried out at any time during the course of a family member's life. Each strategy allows the user to define various transactions that involve the member's assets and liabilities, and it then carries them out based on the applicable period defined, when the member's status changes, or on a specific date that the user has defined.

Demonstrations of each particular strategy (and Assistant) are available through the TIMS desktop menubar (Figure 5-16), the Strategy Assistant (Figure 5-17), or through the Planning Assistant (Figure 4-1).

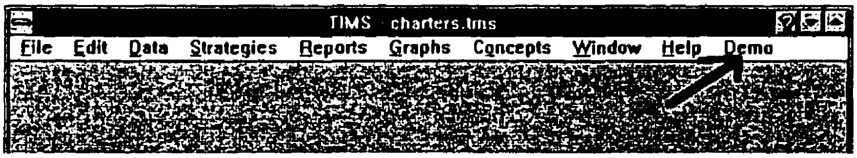


Figure 5-16. The Demo menu option in the TIMS system menubar.

The asterisks to the left of the entries in Figure 5-17 signify that the user has previously viewed the demonstration. Individual demonstrations are also available through a small movie camera icon, which is located in the appropriate dialog's title bar (Figure 5-18). The demonstration icons in the title bar are emphasized in each demonstration as a method of gaining animated, context-sensitive help for a particular task. The user is able to replay a demonstration at any time by either selecting it from one of the demonstration lists or by clicking on this icon in the title bar.

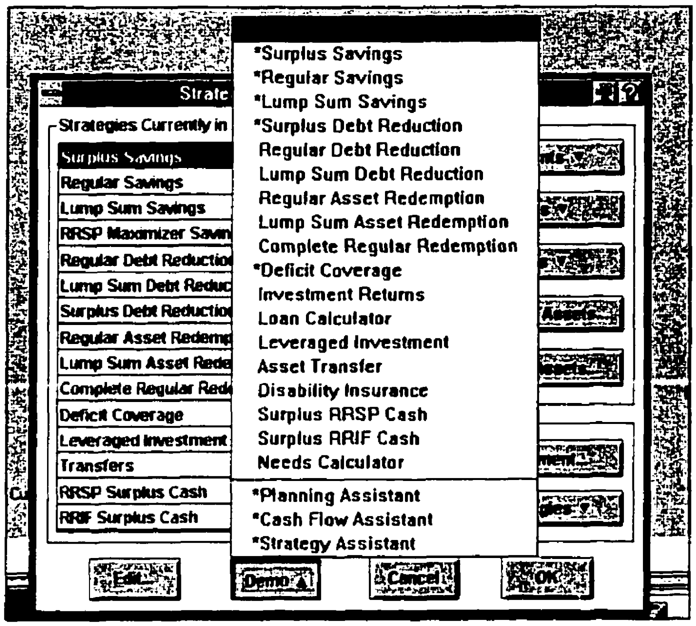


Figure 5-17. The Demo list with the previously viewed demonstrations.

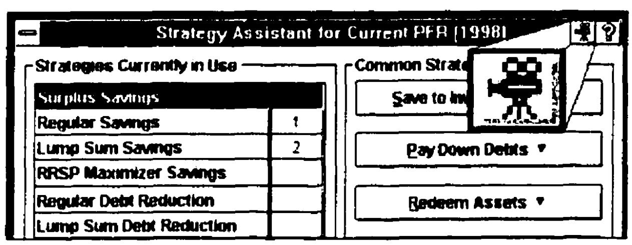


Figure 5-18. The ScreenCam movie icon in the titlebar (enlarged to show detail).

Like the other adaptations to be described, the demonstration facility is controlled directly by the user modelling component described in Section 5.1. When the user attempts to implement a TIMS strategy or to employ an Assistant, the UMC is invoked to determine if a demonstration should be shown to the user. The UMC determines if the Show Task Demonstrations attribute (visible in the Preferences dialog depicted in Figure 5-11) has been deactivated. This could occur through a variety of methods: by the user directly, by the user refusing further demonstrations, or by the user achieving an expert TIMS experience level. If the option is determined to be active, the user is asked if he or she would like to view a demonstration of that particular component (Figure 5-19).

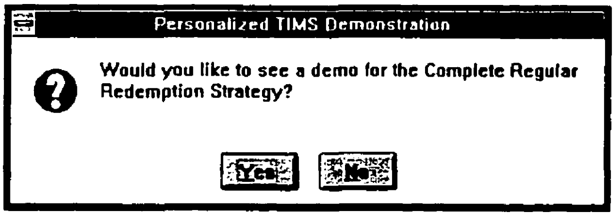


Figure 5-19. A prompt for the Complete Regular Redemption strategy demonstration.

If the user refuses a particular demonstration, a second dialog is presented which allows the user to discontinue all prompting regarding demonstrations (Figure 5-20). This deactivates the Show Task Demonstrations checkbox in the Preferences dialog, but a user can choose to reactivate it in the future.

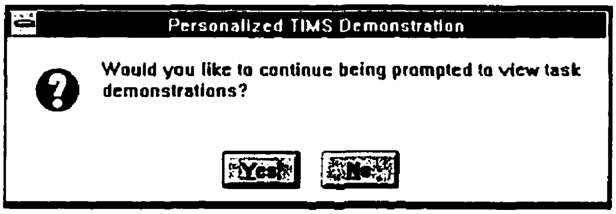


Figure 5-20. The prompt for the termination of the demonstrations.

P-TIMS' demonstration facility mainly supports the most inexperienced users of TIMS. The impact of this facility on users is minimal once they have passed the initial start-up phase of the system. Unfortunately, the first few sessions require the greatest amount of input from the user, but these sessions are also where the greatest level of assistance is required. Major changes for users of the adaptive system as compared to the original TIMS system include: the login procedure, the occasional viewing of animated demonstrations, and prompting to terminate demonstrations. In the worst possible case, users who were averse to any interruptions in their use of the system would be forced to respond to the P-TIMS demonstration query and the termination of demonstrations query immediately after entering the system. If the demonstration facility had been deactivated immediately, the user would only need to respond to the first dialog.

This amount of interruption was deemed manageable. Earlier versions of P-TIMS had triggered a demonstration the first time any individual strategy or Assistant was used. However, during the pilot testing of the original adaptive system, it became clear that this level of intrusion irritated some users. Their irritation increased because there was no obvious method for stopping, rewinding, or fast-forwarding a demonstration once it started. This led to a re-design of the animated demonstration triggering conditions. The next iteration of P-TIMS triggered only the first two distinct demonstrations. From a design standpoint, it seemed that two short demonstrations would be sufficient to help the new user to become familiar with the system and to establish how to invoke further demonstrations.

A review of user comments following the subsequent re-design and testing cycle clearly showed that users did not like any automatic triggering of the demonstrations and preferred to choose whether or not they viewed a demonstration. It was deemed particularly annoying because the ScreenCam mechanism that enabled them to exit from the demonstration was hidden in order to avoid confusing the users. Automatic triggering was especially confusing to new users who were not familiar with any other software packages; to this group it seemed almost magical. In one incident that occurred during pilot testing, several users were observed to be looking away from their monitors while the first automatically triggered demonstration was playing. When they were asked why they were doing this they replied, "We don't know what we did so we're waiting until it's over!" As a result of this misunderstanding, and the feedback from the second large pilot test of the P-TIMS system, the final design was modified to prompt the user before showing any demonstrations.

The first phase of adaptations for the TIMS program, the ScreenCam demonstrations, attempts to assist novice users in understanding the application. Animated demonstrations of some of the key components and the strategies were developed in order to give a new user a brief look at the system functionality. It was hoped that the demonstrations would essentially replicate a "TIMS guru," an experienced user showing the system to a new user. The automatic triggering of events has been minimized in order to avoid intimidating new users. Users must now chose the system component or strategy and then answer in the affirmative to a dialog if they want to view the demonstration.

A second modification was made to the demonstration itself in order to reassure the novice user. A warning is displayed at the beginning of every demonstration (Figure 5-21). It states that any open files will not be modified in any way by the demonstration. This warning is shown in order to combat fears that they, the users, have somehow triggered a sequence of events.

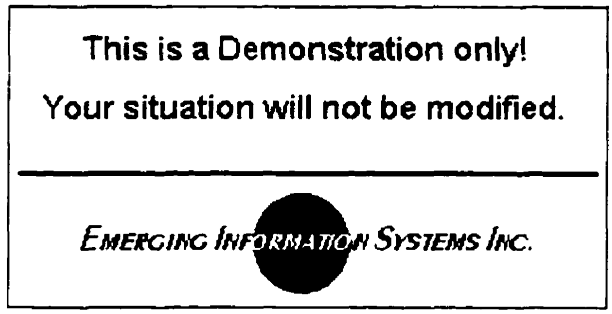


Figure 5-21. The initial dialog of the demonstrations.

5.3 Phase 2: Strategy Recommendations and Strategy Interaction Detection

The purpose of the second phase of the adaptations for the TIMS system is to provide additional help and support for functions and strategies that are rated as being more complex than the user's current level of skill. This support is provided in two ways: (1) suggesting simpler alternatives for more complex transactions, and (2) making the interactions between strategies more explicit. A good example of a strategy interaction in the TIMS system is the common occurrence of plans that include both savings and debt reduction strategies. Each strategy affects the client's cash flow and may cause deficits to occur in the plan. Despite these potential difficulties, strategies are the preferred method of implementing ongoing transactions, particularly for the inexperienced user. Strategies provide the user with the meta-task presented in his or her terms so that the user will not be concerned about the lower-level system details behind the strategy.

This set of adaptations relies heavily on the task and application models that are maintained in the UMC (Figures 5-9 and 5-10) described in Section 5.1. All of the strategies and the more difficult system functions are represented in the task model. This representation includes a task complexity rating (an integer range from 1 to 4, with 1 denoting "simple" and 4 denoting "very complex"). A user complexity rating is also maintained as part of the user model (Section 5.1). When a complex procedure is invoked, this rating is compared with the complexity rating of the task in the task model. The application model contains a mapping between complex tasks and low-level tasks (or sequences of low-level tasks). If a user's rating is lower than the complexity rating of the task that he or she has selected, the application model is checked for a simpler solution. If one exists then a recommendation is presented (Figure 5-22). The user may also choose to disregard any suggestions and deactivate subsequent suggestions at any time by removing the selected checkbox in the bottom left-hand corner of the dialog.

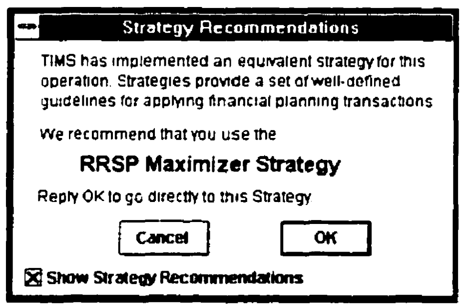


Figure 5-22. The Strategy Recommendation dialog.

In the example in Figure 5-22, the RRSP Maximizer strategy has been recommended to the user as a simple alternative to directly inserting a "Buy" of a registered asset[17] in the Detailed Asset dialog. Replying *OK* to this dialog will allow the user to go directly to the strategy.

Most of the available recommendations involve sequences of low-level detailed transactions that are equivalent to various strategies (see Appendix B). These strategies provide well-defined mechanisms that insulate the user from the details of the low-level transactions and provide the equivalent functionality. The strategies and functions and their associated complexity ratings, which are included in the task model, are outlined in Figure 5-23. In addition, the ScreenCam demonstrations that are described in Section 5.2 are also available to the user to illustrate the use of the strategy.

[17] Investments recognized by Revenue Canada which allow individuals to defer paying income tax on principal and earnings until the income is removed from the account.

The user complexity rating that forms the basis for the strategy recommendation adaptations is set when the financial planning stereotype is triggered, as described in Section 5.1. (The NFP stereotype assigns an initial value of 1, the EFP stereotype a value of 3.) However, like other components of the user model, it evolves over time as a result of the user's interaction history with the system (Figure 5-13). The user complexity level is increased by 1 if the user has invoked at least two functions or strategies that have a complexity rating higher than his or her current user complexity rating. This *ad hoc* method seems reasonable. If a user's abilities have been underestimated, a correction is made once two slightly more complex strategies or functions are used. This would normally take place in a relatively short period of time. In a case where a severe underestimation is made, it will take consistent complex feature invocations and at least ten sessions over the course of ten days (because the user model is only updated daily) to position the complexity rating at the proper level. Normally, this complexity level increase would occur gradually. A user would undergo training, gain experience, and increase his or her TIMS knowledge. This would result in the use of the more complex features of the system. The UMC would then increase the user's complexity level appropriately. In situations of severe underestimation, the user can still turn off undesired adaptations directly.

Strategies			
Investment Returns	3	Savings	
Future Returns	4	Regular	1
Loan Calculator	3	Lump Sum	1
Leveraged Investments	4	RRSP Maximizer	1
Transfers	3	Debt Reduction	
Disability Insurance	3	Surplus	1
Registered Cash Surplus		Regular	1
RRSP	2	Lump Sum	1
RRIF	2	Asset Redemption	
Periodic Needs Calculator	3	Regular	1
Deficit Coverage	2	Lump Sum	1
Targeted D. Coverage	3	Complete Reg. Sell	2
Functions			
Detailed Menus		Security	
Incomes	2	Transfers	2
Expenses	2	Remove Authorization	3
Assets	2	Concepts	
Liabilities	2	Needs Calculators	3
Additional Tax Info	2	Investments	3
Document Packages		Payouts	3
Export	2	Reports/Graphs	
Template Create	3	Custom	2
Plan Labels	2	Comparison	2

The complexity ratings are: 1=Simple: 2=Medium: 3=Complex: and 4=Very Complex.

Figure 5-23. TIMS strategies and functions included in the task model.

If the user's ability is overestimated, the user complexity rating in the user model will slowly decrease over time. Like the increase of the user's complexity level, this processing is performed only once a day. The user complexity rating is decreased by 1 if the user has not recently used any functions or strategies that are at least as complex as the user complexity rating. "Recent" is defined in the system as within the last five uniquely dated sessions, typically a one-week period. The reasoning behind this approach is that a user who has not performed complex tasks in this period of time has probably been misclassified as a more advanced user. A Last Update field in the user model allows the UMC to periodically check the user complexity. Again, individuals may increase or decrease their rating at some later date depending on their function use over time.

Although these adaptations allow the system to recommend particular strategies, interactions between a new strategy and previously applied strategies are still a major problem. Strategy interactions are a natural side effect of any financial planning that is performed using TIMS, and to a large extent, they reflect the conflicts inherent in the domain itself. The major advantage of employing strategies – the insulation of the user from low-level implementation details in the TIMS system – makes it difficult for users to foresee interactions. Even experienced financial planners find it difficult to pin down elusive combinations.

A typical interaction illustrates this problem. Consider the application of two strategies: Regular Savings and Deficit Coverage. This is a relatively common scenario, as most client plans involve saving money on a regular basis and monitoring the plan to ensure that their expenses do not exceed their incomes. The Regular Savings Strategy could be defined to contribute a set amount to a family's stock portfolio asset every month. The Deficit Coverage strategy automatically sells either portions of an asset, or an entire asset, in order to provide additional cash flow to pay expenses when required. In this example, if the level of savings were too optimistic, an excessive amount of the stock portfolio would be purchased, which would cause a deficit situation. A portion of this same asset may be sold in order to resolve the deficit (if it has an associated deficit coverage strategy). Purchase and redemption cycles may occur over and over again because TIMS projects a plan from the current year until the clients reach their projected life expectancy.

This example illustrates a fairly straightforward interaction between two strategies that affect a single asset. There are nineteen strategies with various options that, when activated, will affect any number of assets and liabilities. It is very difficult to fully understand all of the possible interactions because an individual asset could have potentially conflicting demands placed upon it (e.g. one or more saving and redemption strategies), or the overall client plan could have contradictory strategies in place that are intended to achieve a client's goal. A more complicated interaction scenario can be illustrated by considering a typical goal in TIMS: retirement planning. Clients may attempt to achieve this goal by: speeding up the reduction of one or more debts, embarking on registered and non-registered savings plans, and taking on debt in order to leverage[18] investments that will maximize their net worth. Each of these strategies can interact with the others because they each affect the family's cash flow.

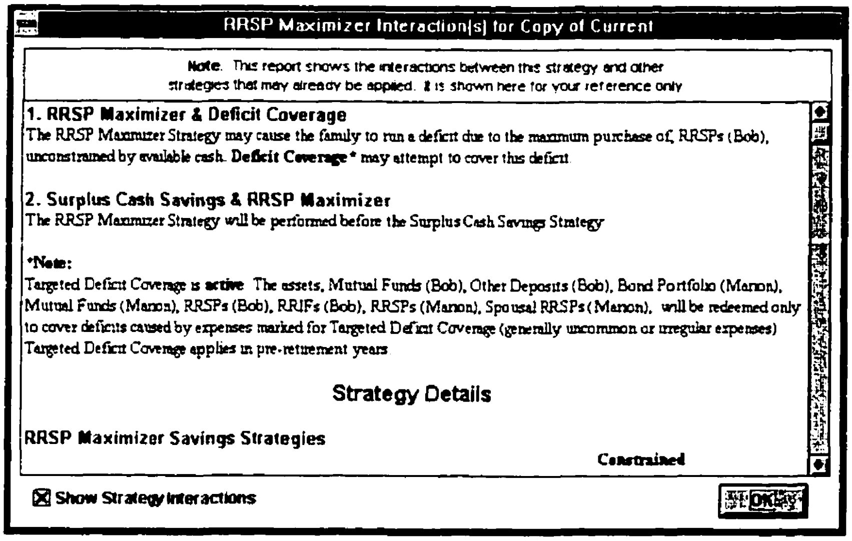


Figure 5-24. An example of a Strategy Interaction dialog.

The UMC helps to deal with this problem by checking for possible interactions that are recorded in the Application model (see Appendix B for a complete listing) whenever a strategy is implemented. A Strategy Interaction dialog is only presented if the Application model contains interactions for the strategy in the context of the current plan (Figure 5-24). This dialog explains the represented interactions and is simply informational; the user is not required to make any modifications to the plan. Like other adaptations, interaction detection can be deactivated and later reactivated if necessary.

In the example in Figure 5-24, the RRSP Maximizer strategy has just been implemented, and the application model lists two possible interactions between it and another strategy: Deficit Coverage and Surplus Cash Savings. The Strategy Interaction dialog explains that the RRSP Maximizer strategy is purchasing the registered assets that are specified in the strategy without regard to the amount of surplus cash in their cash flow. This could put the client into a deficit situation. The second point reminds the user that any surplus funds will be used to purchase the registered assets. The Surplus Cash Savings strategy will only dictate the asset or assets to purchase if there are surplus funds. A complete listing

[18] Leveraging means borrowing money to invest, thereby increasing the capital on which you earn money.

of all of the strategy information in the involved strategies is listed below the detected interactions. This dialog is simply informational; the user does not need to modify the plan. A specific note was added to the dialog to make this information clear to the user. The primary purpose of this dialog is to make the strategy interactions in the system explicit, but it could cause the user to reconsider his or her choice of strategies. A complete list of all of the strategy interactions that have been detected for a plan is also available in a report that is available to the user.

5.4 Phase 3: Data Entry Assistance

The third phase of implemented adaptations is intended to address concerns with the initial data entry process. One common method of simplifying interactive programs is to have two sets of menus: a standard set and an advanced or detailed set. The use of two menu sets is employed in TIMS, and it makes it easier for a novice to avoid the more complex dialogs that are not normally required for a simple financial plan. Once users find it necessary to employ some of the more advanced features, they are accustomed to the system and are not overwhelmed by the additional functionality.

The data entry path in the original TIMS system has been designed to make the entering of client data as straightforward as possible. A logical flow from the clients' demographic information to their incomes, assets, and liabilities is made explicit by providing Next and Previous Dialog buttons at the bottom of dialogs in this loop, in addition to the typical OK and Cancel buttons that terminate data entry (see Figure 5-25). By stepping from one dialog to the next, the user is assured that all of the pertinent data has been entered and that the clients' financial plan is ready to be analyzed once the data entry loop is complete.

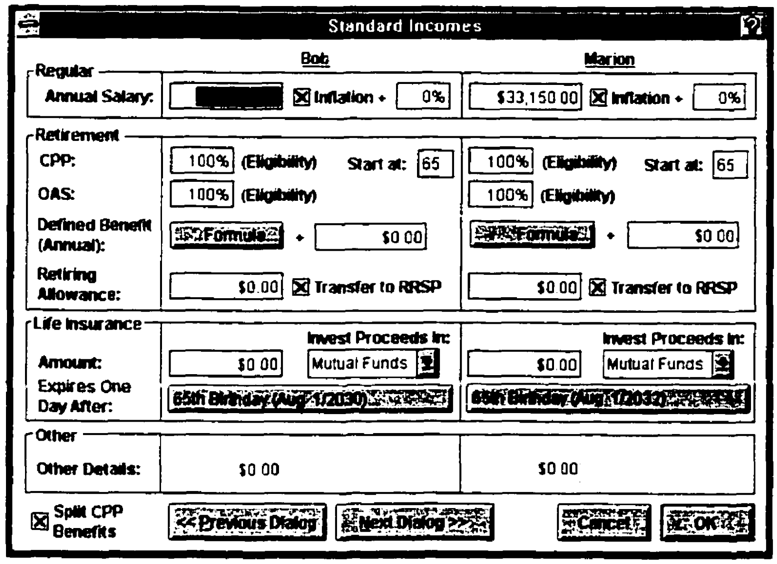


Figure 5-25. A typical dialog in the data entry loop.

Two levels of the data entry loop have been provided. One is for the standard data entry menus, and a second is for the detailed level of data entry menus. This provides a greater level of support for the novice user, but it is still possible to accidentally click on the *OK* button in the dialog and terminate the data entry loop. The user is subsequently returned to the TIMS desktop. This was not a great concern to the system developers as the loop can easily be re-entered by choosing the appropriate command from the Data pull-down menu (Figure 5-26). The problem that was detected with novice users was that they were unsure about the dialog they were on, how to return to the dialog, or even how they managed to get back to the desktop in the first place. This concern has been validated through the observations of hundreds of new TIMS users.

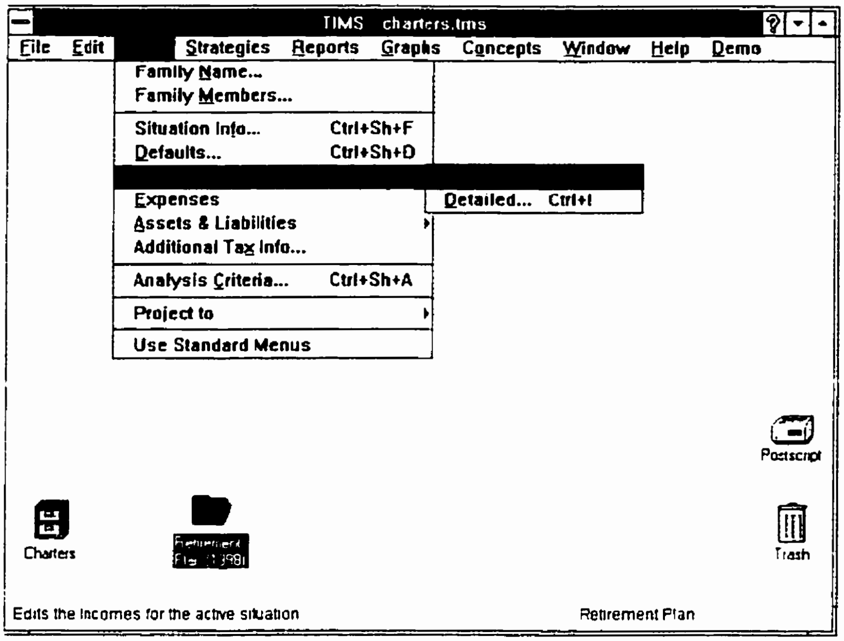


Figure 5-26. Re-entering the data entry loop through the desktop.

Although the data entry support mechanisms that were in place for newer users were adequate, and were at least on par with comparable commercial systems, the system's designers were open to suggestions for improvement. These suggestions occurred naturally as part of the system refinement process. Feedback from users progressed from bug identification and procedural concerns to an increased focus on usability issues. As a result of this feedback, two major modifications for new users have been introduced. The first identified concern was the early termination of the data entry loop by accidentally clicking on the OK button, as opposed to the Next Dialog button. A simple mechanism to solve this problem is to deactivate or "gray out" the OK button. This is useful but can be problematic in two ways: single dialogs must remain accessible in order to allow the modification of data, and experienced TIMS users must not be forced to complete the entire data entry loop.

These concerns were addressed through the user modelling component by prompting the novice user and allowing him or her to safely return to the data entry loop if it was accidentally terminated. This warning (Figure 5-27) was displayed only if the dialog was accessed as part of the initial data entry process, not if it was accessed individually. The use of this warning dialog to reduce accidental exits is directly linked to the user model so that it will not be displayed for experienced TIMS users.

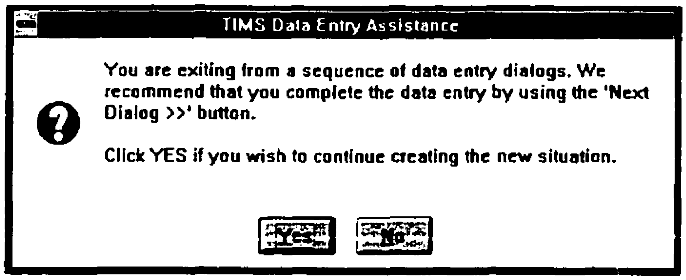


Figure 5-27. Attempting to exit from the data entry loop.

The second difficulty that was consistently reported by novice users was their feeling of uncertainty during navigation through the complex process of data entry, financial analysis, and the production of a final printed financial plan. TIMS uses a desktop metaphor which requires users to do the following tasks: enter data regarding clients; perform financial analysis by applying a variety of strategies (modifying client data and strategies appropriately); and produce a printed financial plan using document packages. The system was constructed to be as powerful and flexible as possible. Once data is entered, there is not necessarily one optimal solution. Compounding this lack of direction is the fact that financial planning is not an exact science; it relies on a large base of knowledge, heuristics, compromises, and individual preferences on the part of the advisor.

Experienced TIMS users, who have a good background in the financial domain, can take advantage of the flexibility of the system. An individual plan can be analyzed quickly, and financial planning strategies can be implemented using only the Detailed level of menus, which are primarily the Detailed Assets and Detailed Liabilities menus. The Plan Window can be used to graphically gauge the success of the strategies. Many members of the original group of users have gravitated towards performing the bulk of their analysis in this way and relying on the reporting mechanisms to judge the fit of the plan to the client.

The TIMS strategies and the intelligent Assistants have been constructed to provide the higher-level assistance for the financial planning process and they have been previously discussed. These facilities are available to all users, but once the data entry process has been completed the novice user should be guided towards the Planning Assistant as the focal point for all subsequent analysis. There, the user is presented with a detailed evaluation of the different aspects of the client's plan, the problems are identified, and a range of solutions is enumerated. Easy access to the recommended strategies is provided through a set of buttons that are dynamically reconfigured in the Planning Assistant.

Supporting the novice user through this process is not an easy task. The Assistants help a great deal, but they still leave gaps where the additional support is required. For example, once initial client data is entered, the system recommends the Strategy Assistant as the mechanism that may be used to enter the client's existing financial planning strategies. When the Strategy Assistant is closed, the user is returned to the TIMS desktop, and a graphical display of the client's financial plan is displayed in the Plan Window. At this point, the data entry phase is complete and a new user may be uncertain about the next logical step to perform in the system. Novice users could be confused because they would not have any clear conceptual design of the system. Once the Strategy Assistant is closed in the P-TIMS system, the novice user is queried if creation and analysis of a new plan is desired.

Usually, the user's final goal is the creation of a printed financial plan for a client. As mentioned in Chapter 4, TIMS uses Document Packages that accept data from the system and export it in the form of an RTF file. Each document package requires specific information in order to function. For example, the default document package, Retirement Plan, requires two plans: the initial financial plan and a new, improved plan. This improved plan is created as a result of the application of financial planning strategies to a copy of the first plan, and it assists the financial planner with implementing a plan that will provide the clients with a comfortable retirement. In the case of a user who has just finished entering the initial client data, the next logical step is the creation of this second financial plan. Likewise, the use of other Document Packages will have their own unique requirements that will dictate the next logical step after data entry.

In order to support the user through the various steps that follow data entry, the UMC maintains the requirements of the various Document Packages explicitly. At each step, the UMC examines the currently active document package and the requirements that remain incomplete. A dialog is displayed, and it asks the user if he or she wishes to follow what the UMC determines to be the next logical step (Figure 5-28). In the default example, this step would be the creation of a second retirement-oriented financial plan. The user may then view a ScreenCam demonstration that takes him or her through the proposed steps of duplicating the current plan, assigning a name, and invoking the Planning Assistant to perform analysis.

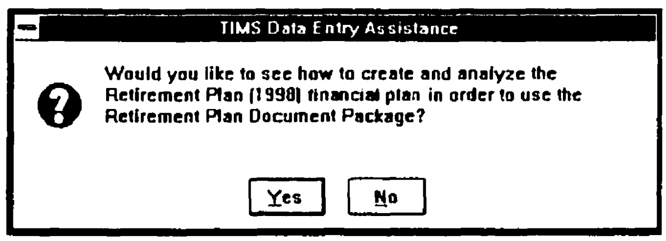


Figure 5-28. Data entry assistance dialog presented after the initial data entry.

When this analysis phase is complete, the next logical step is the final export of a financial plan via the Document Package. Once the user closes the Planning Assistant, the UMC checks to see if the active document package has met its data requirements. If it has, the user is asked if he or she needs assistance with navigating through the export procedure (Figure 5-29). The resulting ScreenCam demonstration illustrates the steps that are necessary for the export of a financial plan, as well as the steps that are necessary to activate the word processor, print the results of the export, and return to TIMS.

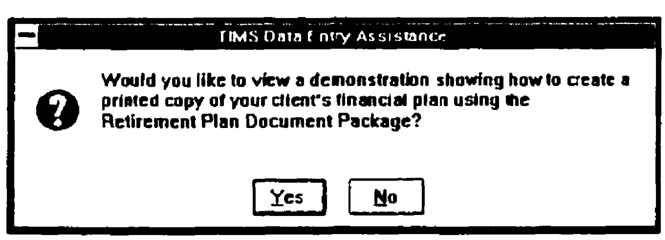


Figure 5-29. Data entry assistance presented after the analysis phase.

These two additional demonstrations assist novice users by providing direct paths through the TIMS program from the initial data entry through to the printing of the completed financial plan. They are especially helpful during the first few times that the user enters data into the program. These adaptations are targeted directly at novice users, and experienced users will not receive any of the modifications unless they explicitly turn on the option themselves. Over time, novices are expected to cancel this option once the path through the system is clear. Automatic cancellation of this option was not implemented because of the variability in user preference for the prompts. In terms of user comments this has been the most useful adaptation.

5.5 Summary

This chapter has provided an extensive description of the adaptive TIMS system, Personalized-TIMS (P-TIMS). Four major categories of user adaptations were implemented: animated demonstrations, a Strategy Interaction Detector, a Strategy Recommendation component, and assistance with the data entry process. This chapter has also described the minimalist approach to the user modelling component that is needed for the support of these adaptations and has provided several examples of typical interactions with a user. The following chapter focuses on the issues of user control and the measurement of the impact of the user modelling component on the user satisfaction levels of the P-TIMS users.

Chapter 6

Impact of a User Modelling Component on User Satisfaction Levels

6.0 Minimalism in the P-TIMS User Modelling Component

There are very clear advantages for the use of a simple user modelling architecture in complex systems such as TIMS. Issues that are critical in large, complex user modelling systems can be minimized or ignored in a simple system. For example, there is no need for a conflict resolution scheme, which would be absolutely necessary in a more complex approach. Conflicts can occur in user modelling systems when more than one stereotype is a potential candidate (based on the information gathered about the user) or if a fact is determined that makes the current stereotype selection inappropriate. In P-TIMS, users are assigned a single stereotype that has been chosen from a very small set of potential candidates. This stereotype is not revoked if further information changes the system's perception of the user's skills. Individual parameters of the user model are simply updated as new information is presented. The stereotype functions as a starting point by providing an approximation of the user's expertise in different aspects of the system.

In the P-TIMS system, the initial stereotype is chosen based on the job title and on the initial values of TIMS and computer experience that was selected by the user. The user modelling component then modifies the Financial Planning and TIMS expertise parameters by monitoring the tasks that this user has performed. If the initial stereotype is incorrect, the system gradually corrects itself through these interactions and the length of time that the system has been used. It is not necessary to revoke the current stereotype and trigger a stereotype that describes the user more accurately.

Similarly, neither sophisticated learning mechanisms nor uncertainty management schemes are necessary in a simplistic approach such as this. This architecture is simple and coarse-grained. The user models do not need to have certainty factors, which are extra parameters that represent the confidence level in facts that are not guaranteed to be 100% accurate. User models contain only a minimal set of user attributes, and each attribute is associated with an inherent risk of inaccuracy due to its general nature. More modest applications also have the advantage of being less likely to mislead the system into thinking that the system is smarter than it is (Spark Jones, 1989). The system learns basic information about users by monitoring user actions and by gathering user information from the users themselves. This information is composed primarily of: the tasks performed, the number of demonstrations viewed, and the user's direct modifications of the user model. As described earlier, the user model is refined on the basis of this interaction history and the length of time that the system has been used.

The most serious drawback of a simple user modelling mechanism is the potential for misjudging the users due to the generality of the attributes that are recorded. This exacerbates a common characteristic of virtually all adaptive systems, that they will make mistakes that will result in erroneous adaptations (Kay, 1994; Höök *et al.*, 1996). These types of errors have only a minimal effect on an individual user in this system. Consider the two most extreme examples: incorrectly categorizing a complete novice (a new TIMS user with limited financial planning skills) as an expert user, and vice versa. In the first situation, the user would experience system interactions that are similar to using a system with no user modelling component. Nothing would be gained through having an adaptive system, but nothing would be lost. As the user interacted with the system over time, the user complexity attribute in the user model would decrease and the necessary support would be available.

The second situation is somewhat more problematic. The user might take a week of system use to reach his or her true level, and during that time become irritated by extra prompts and reminders of known concepts. If it is clear to expert users that they have the ability to easily control the adaptations, perhaps through additional preference settings in the system, it is reasonable to expect them to access them, and then change them. In the P-TIMS system the user models are accessible, and users can modify them so that the impact of an initially inaccurate user model will be low.

6.1 User Control of the User Model

The accessibility of the user model is an important component of this approach and of minimalist user modelling in general. Previous research on adaptive systems suggests that user control over the user modelling component is important for user acceptance (Wahlster & Kobsa, 1989; Krogsæter & Thomas, 1994; Krogsæter *et al.*, 1994; Höök *et al.*, 1996). In particular, systems that act too independently may not be acceptable to users (Dietrich *et al.*, 1993; Meyer, 1994; Vassileva, 1994). Because users are often unfamiliar with adaptive behaviour in a system, care must also be taken so that users do not feel monitored. Users must also be able to inspect what the system knows about them (Cook & Kay, 1994). Kay (1995) believes that since the user model represents information about a person, it should ensure user control, access and privacy. Shneiderman (1992) believes that a satisfying system gives users the sense that they are in control.

User control over a system is an issue whenever any degree of "intelligence" or "autonomy" is introduced. Any system that forces users to give up control at some point may be met with some resistance. For example, a Programming by Example system called *Eager* monitors a user who is doing repetitive tasks in the Hypercard environment (Cypher, 1995). When a pattern of use is detected, Eager anticipates and highlights the next action. If the user clicks on the Eager icon, the task is completed automatically. A study that was conducted in order to prove the viability of the approach concluded that: "the most striking finding was that all subjects were uncomfortable with giving up control when Eager took over" (p. 808).

Krogsæter & Thomas (1994) concluded, in their study of the current state of adaptive systems, that care must be taken so that users do not feel dominated by a powerful adaptation component because adaptive behaviour by a system is unfamiliar to most users. They also felt strongly that users should not feel monitored by the system and that the users should have tools that will enable them to inspect what the system knows about them. One of the Flexcel (an adaptive system based on Microsoft Excel©) adaptation guidelines was that: "At any time, the user should be in complete control of the system; the system may only act as an assistant" (Krogsæter *et al.*, 1994, p. 101). They found through testing the system that "users refuse to be at a system's mercy" (p. 121).

Adaptive interfaces can cause users to feel a lack of control when the system takes over. Maskery (1984) found that "adaptive interfaces that change automatically can confuse or annoy" in an unsuccessful attempt at providing assistance. This finding was duplicated in another large scale study: Browne, Totterdell, and Norman (1990a) summarized a four-year Adaptive Intelligent Dialogues project. In their conclusions, they list lack of control as an outstanding research issue. They suggest that the users should be given an inspectable version of their user model, or alternatively, they should be allowed to turn off the adaptation. They went on to conclude that, "In this way they are being given ultimate control, the knowledge of which may be sufficient for them to be accepting of the adaptations" (p. 209).

Wahlster and Kobsa interpret a user's control over the user modelling component to be problematic (Kobsa & Wahlster, 1989). They feel that the user should be able to inspect his or her user model, but that it may be difficult because the number of assumptions may be very large or difficult to display in a meaningful way. Another issue arises if users are allowed to modify their models. The overall consequences of the modifications might not be clear or the user might actually change a correct model to reflect a more positive perception of themselves. Wahlster and Kobsa were not as convinced that the solution was to allow a user to "switch off" the user modelling component. They felt that whether or not it was possible to separate the user modelling component and the application in this way was an open technical question.

Oppermann (1994) notes that most current work proposes user control of the system's adaptations in order to overcome some of the problems of adaptive systems. For people to feel that they are in control of their computational systems, they must be comfortable, have an understanding of what is occurring, and have confidence in the result (Norman, 1994). Oppermann suggests several methods of achieving user control:

> a) Providing means for the user to activate and deactivate adaptation for the overall system or individual parts of the system before any adaptation is made or after a specific adaptation state has been reached;
>
> b) Offering the adaptation to the user in the form of a proposal that he can accept or reject, or enabling him to select among various possibilities of adaptation modification;
>
> c) Enabling the user himself to define, in an adaptation-resistant manner, specific parameters required for adaptation by the system;
>
> d) Giving the user information on the effects of the adaptation modification, which may protect the user from surprises; and
>
> e) Giving the user sole control over the use of his behavior records and their evaluation. (Oppermann, 1994, p. 7)

P-TIMS attempts to maximize the feeling of user control by using these five methods as guidelines for the system's design. Each of the five methods has been included, as is described below.

In order to provide the user with the means to control the system's adaptations, a "hard stop" mechanism for each individual adaptation has been provided. The user modelling component must allow the user to override any changes made by it. Check boxes in the Preferences dialog (previously shown in Figure 5-11) provide a means of examining the adaptations that are currently available. They are as follows: the Animated Demonstrations, the Strategy Interaction Detector, the Strategy Recommendation component, and the Data Entry Assistance. The user is free to activate and deactivate individual adaptations. The original design included one option that allowed the user to turn off the adaptive component and revert back to the original, non-adaptive system. This was deemed to be too inflexible. The current version gives the user the option of controlling each of the modifications; each group can be deactivated separately. The adaptive system is effectively removed when all of the options in the Preferences dialog are deactivated. The UMC monitors system usage in the background while the adaptive user interface remains disabled.

Proposals that offer the adaptation have been implemented in dialog boxes that prompt the user before displaying demonstrations. In several places in the system, if the user is determined to have little or no experience with a concept or function, he or she is asked if a context-sensitive presentation about this feature is desired. Once a suggestion is rejected, the user is given the option to terminate further prompts. In addition, the Strategy Interactions dialog and the Strategy Recommendations dialog contain check boxes that allow the user to immediately terminate these two adaptations. These features provide P-TIMS users with direct control over whether they will accept adaptations at a particular time and whether they will continue to experience adaptations in the future.

Adaptation-resistant definitions of parameters that are required by the system have been implemented by not overriding changes that the user has made directly, either through modifications of the user model in the Preferences dialog or by the user terminating an adaptation through one of the dialogs which were just described.

P-TIMS also attempts to give the user information about the effects of the adaptation modification through the use of informational dialogs. For example, if the UMC has determined that a user needs to be re-classified to a different user level, an appropriate dialog is posted with the suggested changes that are based on the user's new level. A constantly-adapting system might confuse the user, result in productivity lags, inhibit user exploration, and increase anxiety levels. Prompting the user about the changes gives the user the option of not adapting at all. The user can refuse to be re-classified at this point, and he or she may then choose to deactivate any further re-classifications. Users may re-classify themselves through the Preferences dialog at any time.

In a relatively simple manner, the P-TIMS system also allows users to have sole control over their user models. Each individual user model is stored as a file under the system directory. There is no attempt to share these files or extract information from them for any other purpose. A user could simply delete his or her user model file if desired. The user would not be expected to feel threatened by the existence of a P-TIMS user model since it does not contain any sensitive material. There are no error rates, task completion times, or other types of information that employers could use by to monitor their employees.

The minimalist user modelling component and the adaptations that were described in Chapter 5 have proven to be very useful in an informal way. However, there is much more to the evaluation of a human-computer interface than informal, subjective statements. Indeed, the very question of how one can evaluate one interface and be able to say that it is "better" or "more easily understood" is still the subject of much debate. While there are many possible ways in which the quality of a user interface can be measured, Shneiderman (1992) lists five (non-independent) significant factors: (1) speed, the time taken to complete tasks; (2) accuracy, the number of errors made; (3) the amount of time that is needed for learning; (4) retention of acquired knowledge; and (5) user acceptance or satisfaction.

Of these, user satisfaction or acceptance is one of the most neglected (Dietrich *et al.*, 1993). The reason for this neglect lies in the difficulty of measuring user satisfaction empirically, as opposed to the relative ease of measurement of the other factors. It is certainly not due to a lack of this factor's recognized importance. Early medical expert systems were not accepted by users in large part because of an inadequate attention to the user interface and communication with the user (Rennels & Shortliffe, 1987). Chin *et al.* (1987) believe that subjective satisfaction often determines the ultimate acceptance of any system. Indeed, systems can be more than adequate on all of Shneiderman's other measures but they may still fail if user satisfaction is not addressed (Shneiderman, 1992). Shneiderman (1992) felt that subjective satisfaction was the key determinant of success in office, home, and entertainment applications. He commented that, "We have come to expect that automobiles will have miles-per-gallon reports pasted to the window, appliances will have energy efficiency ratings, and textbooks will be given grade level designations; soon, we will expect software packages to show learning time estimates and user satisfaction indices from appropriate evaluation sources" (p. 495).

In fact, Shneiderman's other factors may be dependent on user satisfaction as well. A large meta-study provided evidence that user satisfaction or preference has a strong positive association with users' average task performance (Nielsen & Levy, 1994). Nielsen & Levy (1994) also reported that when choosing between systems "...one has a reasonably large chance of success if one chooses between interfaces based solely on users' opinions" (p. 75).

Because of the importance of user satisfaction and the desire for an empirical evaluation of the adaptations provided in P-TIMS, an investigation was performed in order to study the impact of the adaptive user interface on user satisfaction levels. The nature and results of this investigation are described in the next section.

6.2 Empirical Evaluation

In order to evaluate P-TIMS empirically, an experiment was initiated to investigate the impact of an adaptive user interface on user satisfaction levels. It was hypothesized that computer programs with user models would produce increased levels of user satisfaction over programs with no such user models. More specifically, it was hypothesized that the TIMS program with a user model (P-TIMS) would produce increased levels of user satisfaction over a TIMS program with no support for user models.

Originally, two adaptive TIMS systems were to be constructed and tested. The first TIMS system was to contain a user modelling component that adapted the user interface but was hidden from the user. The second version would contain the same user model and adaptation processes but the user would be able to easily view and change attributes to control them. During the experimental design phase a decision was made to only develop the adaptive system that the user would be able to directly control. This was for two major reasons. First, it was clear from the literature review that certain adaptive systems suffered from a lack of positive support from users because of the loss of their control, and second, it was felt that an adequate test of user satisfaction between three separate systems was not possible given the time and resources that were available.

6.2.1 Participants

Forty-four participants were selected from a pool of 3,500 employees of a large Canadian financial services company that had adopted the TIMS program as their financial planning software. A division manager from a large metropolitan area selected each participant from his staff. Ages ranged from 27–64, with a mean of 43.6 years. Five of the 44 participants were female and 39 were male. All of the participants had relatively high levels of education and moderate amounts of previous computer experience. More than 79% of the subjects had used, and were familiar with, word processors, and more than 77% had some experience with at least one financial planning software package. Minimal previous experience with the TIMS system was reported by 32% of the participants, while the majority (68%) reported no previous experience. All of the participants were judged by the experimenters to be novice TIMS users. A complete set of the demographic data is available in Appendix D.

6.2.2 Apparatus

The experiment took place over a period of four days at the Toronto, Ontario branch office of a large Canadian financial services company. The sessions were located in a conference room that had a large, centred rectangular table and electrical outlets around the perimeter. Two experimenters were present during the first training session of the TIMS version 3.0f software. One experimenter did all four sessions of the training, and the second experimenter acted as an assistant by walking around the room answering questions and helping the participants. A general overview of the TIMS system and an outline of the training program were presented on overhead transparencies that were projected onto a white screen at the front of the room. For the remainder of the session, a Texas Instrument TravelMate 4000E 486 WinDX2/40 MHz colour laptop was used to project the TIMS system image over a Proxima projection system. The projection system was used to provide the participants with a visual image that they could follow during the training session.

Each participant was asked to bring his or her own personal laptop computer to the session. The company had just initiated a company-wide conversion to new laptop computers for their representatives. With few exceptions, all of the participants used their new machines, which were 24 meg, IBM Pentium 120 Mhz ThinkPads with CD drives, removable floppy drives and printers, and built-in pointing devices (j-mouse). The remainder of the participants used an older generation of IBM ThinkPads, 486 33 Mhz machines, which were several years old. The Windows 3.1 operating system and TIMS version 3.0f were pre-installed on all of the machines. The participants in the two groups who were selected to use the adaptive system for the second session had their machines loaded with the second computer program (P-TIMS) by the experimenter just before the start of the second session.

6.2.2.1 Computer Software

The experiment used two computer programs: the Tax and Investment Management Strategizer (TIMS) (version 3.0f as a control) and an adaptive system that was based on this version of TIMS. Version 3.0f was the most current release being used by the financial service corporation's representatives across Canada, and it was discussed in the previous chapter. The adaptive system was called P-TIMS, or Personalized-TIMS, and it was targeted at novice TIMS users in order to assist them with managing the complexity of the TIMS system.

As detailed in the previous chapter, the adaptations added were: Animated Demonstrations, Strategy Suggestions, Strategy Interaction Detection, and Data Entry Assistance. Animated Demonstrations were available for all of the system's strategies and for several of the main system components. Strategy Recommendations were triggered for novice users who were selecting system functions that were deemed to be more complex than the user was expected to be able to handle. The Strategy Interaction Detection was invoked when a strategy was entered into the system; possible interactions with previously implemented strategies were outlined in a dialog presented on the screen. Two major types of Data Entry assistance were available. The first type of assistance helped to retain the novice user in the data entry loop that was implemented in the original TIMS program. The second type of assistance included two optional demonstrations that helped the transition between the data entry of a client, the analysis of the data, and the final printing of the client report in a word processing program.

The P-TIMS system required a user to log into the system, but individuals were not aware of this because they had been logged into the system by the experimenter prior to their arrival. The first visible difference between the two systems occurred after startup when the P-TIMS users were presented with a special welcome dialog. The user would notice very few changes between the P-TIMS system and the TIMS system at the desktop level. The underlying functionality of the two systems was the same, the P-TIMS user would simply experience occasional assistance from the adaptations in the process of having used the system.

6.2.3 Questionnaire

Generally, programs have been evaluated for user satisfaction through the use of questionnaires (Shneiderman, 1992). The problem with this methodology is that few established questionnaires have been developed to access users' perceptions of systems and few have been tested in a variety of conditions that established their reliability and validity. Reliability refers to consistency or stability; validity refers to the measurement of the appropriate item. Chin *et al.* have pointed out weaknesses in many of the subjective evaluations (Chin *et al.*, 1988). Their concerns included lack of validation, low reliabilities, allowances for the same response for questions, small sample sizes, non-representative populations, types of questions, and size of the studies. In their study, they established the validity and the overall reliability[19] of the Questionnaire of User Interface Satisfaction[20] (QUIS) version 5.0. This same questionnaire has also been independently proven to show reliability and validity in the behavioural measure of satisfaction in a study that compared various questionnaires done by Wong & Rengger (1990). QUIS has been highly reliable across many types of interfaces (Harper & Norman, 1993).

For these reasons the research team obtained a license to use the Questionnaire of User Interface Satisfaction (QUIS) as the instrument for measuring user satisfaction in this study. QUIS, which was developed by the Human-Computer Interaction Laboratory at the University of Maryland, was designed to evaluate a user's subjective satisfaction with

[19] QUIS's reliability was high. Cronbach's alpha, an estimation of reliability based on the average intercorrelation among items, was 0.94 (Chin *et al.* 1998).
[20] QUIS is a trademark of the Human-Computer Interaction Laboratory, University of Maryland at College Park.

the human-computer interface of an interactive computer system (Chin *et al.*, 1988; see also Shneiderman, 1992, chap. 13). QUIS was selected because it continues to be updated and refined as the focus of a long-term research project at the University of Maryland, and it has documented user interface evaluations in various industrial and academic environments (QUIS, 1998). Participants were given a modified short version of the Questionnaire for User Interface Satisfaction (QUIS) v5 (see Appendix E). This twenty-one item questionnaire is arranged in a hierarchical format and contains: (1) a demographic questionnaire; (2) six scales that measure overall reaction ratings of the system; and (3) four measures of specific interface factors: screen factors, terminology and system feedback, learning factors, and system capabilities. The short version of the questionnaire was chosen over the longer version because it provides the necessary information while not imposing an undue burden on the study participants.

The demographic section that appears at the beginning of the questionnaire was modified extensively. The description of the subjects was expanded to include questions about their level of education, financial planning background, previous experience with the TIMS system, and the helpfulness of the training sessions. The past experience section was modified to ask the subject the number of years that he or she had used a computer and to include hardware and software choices that were more appropriate to the group of subjects that were to be tested. In the remainder of the questionnaire, the Not Applicable (NA) option was removed from the Overall User Reactions section (the dependent variables) because these six factors would be analyzed in detail and missing data would make the subsequent analysis difficult. Ample space was provided for written comments. Several pertinent questions were also included from the longer version of the QUIS. These questions pertained to the amount and arrangement of information on the screen (4.3a and 4.3b), the predictability of an operation and the user control over the amount of feedback (5.5a and 5.5b), and the novice and expert usability of the system (7.5a and 7.5b). Two additional questions were also developed. These questions targeted the helpfulness of the Animated Demonstrations (6.7), and the adequacy of the system functionality (7.3).

PART 3: Overall User Reactions

Please circle the numbers which most appropriately reflect your impressions about using this computer system.

Overall reactions to the system:

3.1	terrible 1 2 3 4 5 6 7 8 9 wonderful	
3.2	frustrating 1 2 3 4 5 6 7 8 9 satisfying	
3.3	dull 1 2 3 4 5 6 7 8 9 stimulating	
3.4	difficult 1 2 3 4 5 6 7 8 9 easy	
3.5	ineffective 1 2 3 4 5 6 7 8 9 powerful	
3.6	rigid 1 2 3 4 5 6 7 8 9 flexible	

Figure 6-1. User satisfaction attributes in the overall reaction section of QUIS.

User satisfaction was measured in QUIS as a composition of six different attributes: (1) how impressed the user was with the system, (2) satisfaction with the system, (3) how stimulated the user was by the system, (4) ease of use, (5) perceived "power" of the system, and (6) flexibility of the system. The overall reaction section of QUIS lists the six attributes (see Figure 6-1), but for the purposes of simplifying this discussion a single label has been assigned to each attribute. The attribute labels are *Impression*, *Satisfaction*, *Stimulation*, *Ease of use*, *Power*, and *Flexibility*. Each attribute was rated on a scale of 1 to 9 and was anchored by two descriptive adjectives. This type of response format is referred to as a semantic differential (Ray & Ravizza, 1988). A low score for each attribute is associated with a negative term, and a high score for each attribute is associated with a positive term. For example, the first attribute, Impression (3.1), was rated on a scale between the terms terrible and wonderful. It was hypothesized that the adaptive TIMS system (P-TIMS) would score higher than the unmodified TIMS system in all six attributes of user satisfaction.

6.2.4 Procedure

Two programs were compared – the original TIMS system and an adaptive TIMS system (P-TIMS). The experiment took place over a period of four days. The 44 participants in this experiment were divided into four groups of between 10 and 12 subjects. The individual's availability affected the distribution of these participants into groups but they were made as randomly as possible. The experiment consisted of two parts: a training session and a test session. The training session was presented on the first day of the experiment, and the test session occurred two days later. One TIMS group and one P-TIMS group started the first session of the experiment on Day 1, and the second TIMS and P-TIMS groups started on Day 2. Day 1 participants returned to complete the experiment on Day 3; Day 2 participants returned on Day 4. The design produced a main effect for program, a main effect for day, and a program-day interaction. That is, the experiment attempted to measure whether or not there was a difference in user satisfaction levels (the dependent variable) for P-TIMS versus TIMS users and Day 1 versus Day 2 participants (the independent variables). The interaction of program and day attempts to measure if there was a difference in user satisfaction levels for the individual groups.

The first session of the experiment for all of the participants in the experiment consisted of one half-day TIMS introductory training session. The training sessions were done in the individual groups due to the training facility size and the participant's schedules but they were identical in content. This introductory training program was based on the "TIMS for Windows: The Financial Planning Expert Quick Start Manual Release 3.0", created for the TIMS system (EISI, 1996). The 76 page Quick Start manual provides the keystroke-by-keystroke instructions that are necessary for the users to perform representative tasks in the TIMS environment. Upon completion of the training, the users were expected to be able to enter client data, perform simple analyses of the client situation, and generate client reports.

All of the participants returned 2 days later for the second half-day session to perform the second component of the experiment. Participants were given descriptions of the procedures to be employed in the study, informed of the goals and purposes of the study, and asked for their informed consent to participate in the study. The participants worked alone on the specific system (TIMS or P-TIMS) determined by their group. Each individual was given a data worksheet (Appendix F) containing sample client financial data and a set of instructions (Appendix G) to assist them in performing certain tasks. The client data was fairly simple and was similar to the data that was entered during the TIMS training session. The worksheet detailed tasks that reflected typical system usage that the participants needed to perform in order to re-familiarize them with the TIMS system. For the group with the adaptive systems, it gave them an opportunity to experience the customization of their system. No extra assistance was given to the participants but they were encouraged to use the on-line help facilities. Each participant filled out an anonymous questionnaire (QUIS) following the completion of the worksheet.

6.2.4.1 Worksheet

The worksheet (Appendix F) that was presented to the participants in the second session of the experiment contained all of the data that was necessary to enter a new family file into the TIMS program. The minimal set of tasks that each participant performed were: (1) entering in a new client's data, (2) creating a new plan by copying the initial plan, (3) performing analysis and implementing improvements to the plan using the Planning Assistant, (4) exporting the client data into a Microsoft Word document, and (5) returning to the TIMS desktop and exiting from the TIMS program. These steps were familiar to the individuals from the first session of the experiment in which they had worked through the Quick Start manual and entered the sample *Jones* family.

Participants were free to perform any combination of strategies and data changes that they desired. The overall financial goals of the exercise were to provide the clients with the ability to retire early with a healthy retirement income, a reasonable estate, and to limit the impact to their current standard of living as much as possible. This lack of one clear winning set of strategies is characteristic of the financial planning field. Individual financial planners vary in their preferences for different strategies for the different types of clients.

The worksheet was based on the following hypothetical scenario. The Charters family filled out a 15 page Data Entry questionnaire (the worksheet) following their first visit to a financial planner (see Appendix F for the completed worksheet). This young couple, Bob (31 years old) and Marion (29 years old) already had a reasonably promising current financial situation. They had no dependents, a good income (collectively $69,150/yr), and good earning potential (they felt that their salaries would increase by 3% annually). Their annual lifestyle expenses were reasonably modest (about $25,000/yr) for this income level. Their conservative attitude was also reflected in the fact that they owned just one vehicle, which they planned to upgrade every 5 years with an additional $15,000, and they took relatively inexpensive annual vacations ($1,500).

They did have some potential problems with their current financial plan. Neither Bob nor Marion had pension plans at their places of employment. They had recently purchased a $165,000 home, on which the outstanding principal was $120,000 (at 10% over 25 years). As could be expected, based on the Charter's age and recent house purchase, they did not have a large accumulation of assets. Bob had almost $13,000 in mutual funds and deposits, but the deposits returned only 3% per year. Marion had about $8,000 in bonds and mutual funds, but was expecting to use her bond portfolio to finance an expensive holiday to Hawaii in January of 1999. Bob also had an outstanding short-term loan of $11,000. With their low asset return rates and their sizable debt, they felt that they needed some financial planning advice.

The biggest problem for this couple was saving for their retirement. Each of them had only a very small amount in savings, approximately $3,000 in registered assets. It is important to note that they did not have pension plans at work, and that many planners would not assume that individuals in this age category (early 30s) would benefit from Canadian government-sponsored pension incomes. However, they did have some limited savings programs in place, Marion saved $150/month towards the Hawaii trip in her Bond Portfolio, and they did put any surplus cash that they had accumulated at the end of the year into their respective mutual funds.

On the last page of the questionnaire, the couple was asked about their financial goals. Bob and Marion had both indicated that they would like to retire early, when Marion turns 55 years old. They would also like to maintain their current standard of living both now and throughout retirement, and both expect to live until they reached 90 years of age. They wanted to leave a reasonable estate but that was not their primary concern because they did not have any children.

Participants in the study were given a set of instructions (see Appendix G) in addition to the completed Data Entry Questionnaire. The instructions were to create a new file and to enter the client's data into the TIMS system. Once they had confirmed that the data was entered correctly, the client plan was to be copied, and a new plan was to be created. The new plan was to be analyzed and improved using the Planning Assistant component of TIMS. The planning phase was considered complete when the plan had been improved to the point where the participants felt that the client's goals were satisfied.

No direct guidelines about how to improve the plan were given. It was relatively simple to improve this particular client family's situation through a regular purchase of registered assets for retirement and a slight modification of living expenses or their proposed retirement date. Other solutions included modifying the hypothetical return rates on the client's investments or using leveraging for investment purposes. A wide range of acceptable solutions was expected, and this reflects the manner in which a financial planner would actually use the TIMS system with a client. Participants in the experiment were expected to perform the financial planning component of the experiment using both their financial planning expertise and their experiences with clients who were similar to the hypothetical client family.

Following the analysis of the client situation, the final step in the instructions was to use a template to perform the export of the client data to a word processing program, Microsoft Word 6.0. This step was identical to the export that was performed in the first training session.

All of the participants were given as much time as they needed to complete the worksheet. The worksheet was completed when the client family data was correctly entered, analyzed, and the solution implemented. After the experimenter confirmed that this step was correctly performed, the subjects were invited to take a ten-minute break. The QUIS questionnaire was handed out to all of the participants of a group at the same time, and a brief explanation about the questionnaire format was given. It was emphasized that no individual performance data or comments were to be released, and that the questionnaire was completely anonymous. Comments[21] were encouraged, and individuals were asked not to leave items blank but to instead select the Not Applicable option if it was appropriate. Once the questionnaire was completed, the participants were thanked and were asked if they had any comments or suggestions about the experiment.

6.2.5 Results

A 2 × 2 between-subjects multivariate analysis of variance (MANOVA) was performed on six dependent variables: the overall impression of the system (*Impression*), the subjective satisfaction (*Satisfaction*), whether the system was dull or stimulating (*Stimulation*), ease of use (*Ease of Use*), whether the system was ineffective or powerful (*Power*), and whether the system was rigid or flexible (*Flexibility*). Independent variables were Program (TIMS and P-TIMS) and Day (Day 1 and Day 2). Table 1 shows the means and standard deviations for all four groups.

Table 6-1. Means (and standard deviations) for the six dependent variables broken down into four groups.

	TIMS		P-TIMS	
Dependent variable	Day 1[a]	Day 2[b]	Day 1[c]	Day 2[d]
Impression	7.3 (1.3)	6.9 (1.2)	7.0 (1.2)	6.6 (1.3)
Satisfaction	6.4 (2.1)	5.7 (1.1)	6.1 (1.8)	6.7 (1.0)
Stimulation	7.4 (1.4)	6.5 (1.2)	7.6 (1.0)	7.2 (1.2)
Ease of Use	6.2 (2.1)	4.8 (1.8)	4.9 (1.8)	5.2 (1.8)
Power	7.9 (0.8)	6.9 (1.1)	8.2 (0.9)	7.9 (1.0)
Flexibility	7.4 (1.3)	6.3 (1.3)	7.0 (1.3)	6.8 (1.1)

Note. Maximum score = 9. [a]n = 11. [b]n = 11. [c]n = 10. [d]n = 12.

SPSS*[22] MANOVA was used for the analyses. All of the cases contained a complete set of data, which resulted in a total n of 44. There were no univariate or multivariate within-cell outliers. The assumptions underlying MANOVA were tested and found to be satisfied (Appendix H).

[21] Individual comments have not been included because the majority of them were comments about the underlying TIMS system.
[22] SPSS* is a trademark of SPSS Inc. SPSS was formally known as the Statistical Package for Social Sciences, it has been renamed to Statistical Product and Service Solutions.

Wilks' criterion for the combined DVs was significant for Program, $F(6, 35) = 2.73$, $p < .05$ but not for Day, $F(6, 35) = 1.73$, $p > .05$ or for the overall Program and Day interaction, $F(6, 35) = 0.99$, $p > .05$. Because the omnibus MANOVA shows a significant main effect for the Program variable, it is appropriate to do a further investigation of the nature of the relationships among the IVs and DVs.

Univariate analyses were done for each variable by Program, Day, and the interaction between Program and Day. *Power* was the only significant variable ($p < .05$), both for Program and Day but not for the interaction effect.

6.2.6 Discussion

In this experiment, the questions to be answered were whether or not the means of the six overall reaction factors measured by QUIS, which represented various aspects of subjective user satisfaction, differed as a function of the program used and the day on which the experiment was performed. These questions attempt to gather an understanding of the differences between the various user groups. Are there differences in the way that a user perceives each system in aspects such as overall user satisfaction, ease of use, power or flexibility? Also, are there differences between groups that performed the experiment on different days?

Four of the six user satisfaction factors were rated more highly by the P-TIMS groups than by the TIMS groups (*Satisfaction*, *Stimulation*, *Power*, and *Flexibility*). P-TIMS users rated the factor of *Power* significantly higher than the TIMS users. This was the only factor that showed a significant difference with a confidence level of 95%. Based on the informal comments by the participants in the experiment, the *Power* factor represented the impression that the system was more powerful because it could perform tasks for them (it was more helpful) and because it made it easier for them to see the capabilities of the system. While it would be more gratifying for the system with the user modelling component to show a higher rating for each of the six factors, a smaller effect was not unexpected. This is because the adaptations were narrowly focused on certain system components in order to allow a proper implementation of the system.

The research team also expected a smaller effect as a by-product of the experimental procedure. The group selection procedure was not completely random, and it was clear that the first TIMS group was composed of especially enthusiastic participants. One individual stood out as a cheerleader for the program. It was with some horror that the experimenters heard him exclaim repeatedly, "This program is so easy to use!" Many of the participants in the group were his subordinates and he appeared to be positively affecting the attitude of the group. Notice that the averages of the first group are higher than the second TIMS group (overall average of 7.1 versus 6.2), while the two P-TIMS groups are remarkably similar (overall averages of 6.8 and 6.7). In particular, the *Ease of Use* factor is much higher than the other three groups. If the first group's data was ignored (admittedly, this is not sound experimental procedure) then the P-TIMS system performs even better by comparison. Five of the factors are rated higher (*Satisfaction*, *Stimulation*, *Ease of Use*, *Power*, and *Flexibility*), and *Stimulation* is very close to being a statistically significant difference ($p < .06$) (Appendix I).

Two secondary analyses were planned for this experiment. Because the questionnaire captures subjective information, two quantitative measures were also planned. Worksheet completion time was to be recorded for each participant, and the actual solution that was generated by each participant was to be assigned a grade reflecting its overall quality. The time taken to complete the tasks was actually recorded, but this data was not analyzed because the actual time taken varied little across the subjects. Early finishers delayed announcing their completion of the worksheet because they were unsure that they had improved the client plan as much as possible. Once a critical mass of finishers had accumulated, they triggered a chain reaction that caused the rest of the participants to wrap up their analysis. Peer pressure appeared to play a large part in task completion time. The second set of data, the graded solutions, was not collected. It required copies of each participant's TIMS file to be saved to a floppy disk to allow the grading to be done at a later time. The office where the experiment took place had a computer virus that had affected many of the participants. It was not practical to save 44 files quickly without concern for cross-contamination of their systems, because 44 floppy disks were not available.

Appendix J contains a listing of the means and standard deviations for all of the items in the questionnaire. The six overall reaction factors were the only items that were subjected to rigorous statistical methods, but it was interesting to note some of the other findings as well as the informal comments and stories that were gathered as a result of this experiment. For example, as previously mentioned, the questionnaire was modified to include a question about the helpfulness of the Animated Demonstrations (6.7). Most of the TIMS-only groups rated this item instead of choosing the Not Applicable option. There was also a trend towards a participant choosing similar ratings for each item, particularly for the third section of the questionnaire.

Pilot studies are an opportunity to fine-tune an experiment before the actual participants are involved and before any real data analysis is performed. Three major pilot studies were undertaken as part of this research project. All pilots used employees from the same organization as that of the experiment. The Pilot I and II testing occurred in Regina, Saskatchewan. A total of forty people went through a TIMS Quick Start training program and filled out a questionnaire at the completion of training. Each training session was three hours in length, and approximately ten people were in each session. Pilot I participants (the first two sessions) used the TIMS system for their training, and Pilot II participants used the P-TIMS system and a modified Quick Start manual (Appendix K) for their training. These pilots established the training program and the QUIS data analysis procedures, and they provided invaluable feedback on the original set of adaptations.

Pilot III testing took place several months later in Dartmouth, Nova Scotia with twenty participants. These individuals participated in both a training session and a testing session with the worksheet. This pilot provided a dry run for the actual experiment.

6.2.7 Summary

The pilot studies and the final experiment provided a great opportunity to examine individual users and their initial experiences with the software. It has certainly reinforced the value of including users in the development and testing cycles. Each pilot study has modified the research team's perception of the most helpful adaptations to the novice user, the environment that the system was used in, and the best testing procedure for the system. The final experiment only served to reinforce the need to continue this work. The next chapter goes into detail about how these experiences will affect future studies. Perhaps the most important finding is learning to expect the unexpected. It is difficult to tightly control all of the extraneous variables for an experiment that is executed in a real world environment.

CHAPTER 7

ANALYSIS

7.0 Issues in User Modelling in Commercial Software Systems

There are many practical considerations that are involved when performing applied research in the commercial arena. Desmarais (1997) has recently emphasized several lessons that he learned as part of a research consortium performing joint research with commercial partners in real-world user modelling. He listed close contacts with potential clients/users, close contacts with potential commercial partners, good prototypes, an interdisciplinary team, and sufficient funding as being critical to the success of a project. A close affiliation with the commercial product has lessened some of these concerns in this situation, but this work reinforces the need for good prototypes and close contact with potential users. The most critical issues that arose during the course of this research were the following:

- *User involvement.* The adaptations that are chosen for a commercial system must reflect concerns that real users have with an existing system, and be discovered by studying real user's needs and problems. It is easy to make these types of decisions in isolation and to judge proposed changes on the basis of one's intuition. It takes much more time and effort to actually observe users in realistic situations and to take note of their questions and concerns. In this case, formal one-day TIMS system training programs presented excellent opportunities for the monitoring of hundreds of first-time users of the system as they became acquainted with it.

- *Quality of Adaptations.* There is an expectation of very high adaptation quality when experimental testing is performed on the real users of a commercial system. Normally, there is much more latitude when it is understood that the system being tested is a prototype or a mock-up of anticipated changes. If users believe that the experimental system that they are being shown will be the actual system in place at their desks, then they will expect a higher level of refinement. For this reason, the adaptive components in P-TIMS were limited to certain aspects of the system in order to bring them up to production level quality in a reasonable amount of time. The adaptations must appear seamless, and they must match the original system in look and in feel so that it is not obvious that modifications have been made.

- *Scheduling.* Scheduling users to perform the testing and feedback on the adaptive system created many problems and is likely to do so in any commercial environment. Final testing does not depend only upon a completed system. Involving users in the construction and testing of a software system involves anticipating any cycles of the domain in question. For the financial planning domain there is a reasonably predictable annual cycle. January and February are very busy months for Canadian financial planners due to the tax filing deadlines and other deadlines in Canadian financial legislation (e.g. tax-sheltered Registered Retirement Savings Plans). Corporate training is also less likely to be done in December, July, or August because these are traditionally the months when employees take their holidays. Because P-TIMS was to undergo empirical testing in a large Canadian financial service organization, some coordination with the TIMS program launch training programs was necessary.

  Scheduling is also an important issue on a more fine-grained temporal scale. Existing corporate training schedules are difficult to adapt to because a researcher has little control over the dates, times, and the number of users to be trained in any one session. This was an issue because of the training component of the experiment, which was intended to bring participants up to a minimum level of proficiency with the system.

- *Management Support.* The upper-level management of any company that is involved in the testing process must be convinced that their employees should be involved in the experiment. To ensure the project's success, it is invaluable to have supporters at the highest possible level. Eric Horvitz referred to this process as finding the "key evangelists" while recounting the Microsoft experiences with the initial prototype of the Office Assistant (1997). The P-TIMS research benefited from the support of an employee who had been involved in product development for a period of time and had championed the cause. Researchers must address concerns about employee productivity, and they must also ensure that the testing of the system will not have any adverse effects on the employees' normal duties.

- *Corporate Benefits.* Corporations expect, to some degree, that they will receive tangible benefits in return for their cooperation. One large, financial service corporation's participation in the experimental testing of P-TIMS was possible because it was interested in seeing if the proposed adaptations would be helpful to its employees. The TIMS program had recently been adopted as the financial planning software for its representatives across Canada, and the company's support was achieved by emphasizing a beneficial consequence of the experimental process – introductory TIMS training. As discussed in a previous chapter, introductory training was provided to all participants in order to ensure that the new users would be trained to a minimum standard before being asked to compare the adaptive and non-adaptive systems. This hands-on training is very expensive for geographically-distributed corporations, and through participation in the experiment described in this paper they obtained this training at a minimal cost.

  A second benefit for a corporation is feedback from employees about a software program. Every participant in the experiment described in this thesis filled out an anonymous questionnaire and was encouraged to make broad comments on various aspects of the system. Employees who do not fear reprisal can be expected to be more candid, and the result is higher quality feedback. Both the results of the statistical analysis and the anonymous compilation of anecdotal comments were released to the corporation. If continuing evolvement and corporate-level customization of a software package is expected, as it is in this case, then this information is invaluable for future development decisions.

- *Potential for Negative Transfer.* In a corporate environment, the researcher must also avoid sending users back to their normal working conditions with different training or system expectations (Meyer, 1994). This potentially negative transfer is a unique concern in the commercial environment. It was not a trivial task to coordinate the parallel development of the system containing the user modelling component (P-TIMS) with the latest release of the TIMS program. Because TIMS development is ongoing, there have been a number of new product releases throughout the period of this research. Some of the releases were primarily corrections of existing problems, but several major releases added extensive new functionality to the system.

  When an attempt is being made to compare an adaptive and a non-adaptive system, the adaptive system must be synchronized with the latest release in order for any test of preference to be valid. There are two reasons for this: first, to ensure that differences that are detected are based solely on adaptations, and second, because one cannot train new users on an older version of the software and then return those users to their working environment with a newer version. In order to address these concerns, the P-TIMS system was upgraded to the most recent TIMS release for testing. The development environment and existing internal company procedures assisted in this process by tracking the sections of code that had been modified between releases. Synchronizing the adaptive system without this type of information would be overwhelming on any non-trivial system.

- *Integration of Code.* Integrating a user modelling component with existing code can also pose challenges. A problem that is familiar to all maintenance programmers is the difficulty of fully understanding the coding of other programmers. The TIMS system is large and complex, and programmers require several months to become familiar enough with the structure of the system and the development environment to become productive. Ideally, if problems or questions about the primary system arise, the original developers should be consulted in order to avoid wasting large blocks of time. This requires taking those developers away from their own work to assist the researcher. This is a more difficult proposition in a commercial environment than it might be in a research environment because a commercial organization's goal is to get a product to market as quickly as possible.

In general, any developer of an adaptive component for a commercial product will be subject to constraints that do not normally exist in a research environment. For example, Meyer (1994) designed a prototype adaptive help system for the user of a retail point-of-sale device. Because system field-testing in a live retail environment was intended, there were many specific directions surrounding the design and testing of the system. It could not interrupt users during their work to request permission to change displays, and the existing procedures for operating the cash register could not be changed. In addition, the system was not allowed to give users significant additional tasks to perform. Meyer's solution was to use an adaptive multimedia help system on a computer that was installed beside the cash register. The system was intended to augment the limited help being given by the existing system and to detect and correct common operating errors. The assistance was modified to match the perceived level of knowledge of the user, who was free to carry on with activities even if information was being presented by the help system. This avoided affecting users who were involved in the experimental process, and it also avoided the potential for the negative transfer described above.

This research has resulted in a number of different findings in two broad categories. This chapter discussed the many practical implications of this research on user modelling in commercial software systems. The next chapter examines the important questions raised in the course of the experiment and suggests important areas for future work.

CHAPTER 8

CONCLUSIONS

8.0 Future Work

A system as complex as TIMS allows the user a great amount of flexibility and a variety of very sophisticated operations. While this is ultimately an incredible advantage to financial planners, it can also be a disadvantage. As previously mentioned, all of the adaptations described here have been targeted at novice users. In practice however, it was found that some of the adaptations have proven to be more useful to experienced users of the system. The best example of this is the highlighting of interactions between strategies. The current implementation focuses on the more frequently encountered or problematic interactions, which could be expanded to include many other combinations. A smaller, common subset is still the best approach, so that the processing overhead involved in checking all of the possible combinations is not too great. Although novice users require more assistance with understanding interactions at first, this extra information seems to confuse them. It is simply a case of information overload. A typical response was that the user felt that he or she had made an error. On the other hand, experienced users find interaction information helpful because it answers some of their questions and concerns by exposing some of the underlying processing in the system.

The Strategy Recommendation adaptation, which provides simple alternatives to complex procedures in the TIMS system, could be expanded to include other functions. The users who are expected to benefit most from this adaptation are not necessarily beginning users, but relatively competent TIMS users who do not yet realize the full capabilities of the system. Groups of users who had been using an earlier version of TIMS that did not contain strategies provided some insights into the habits of pre-existing users. These users had a predetermined method of data entry that they had used for over six months. They were not easily convinced that investing time and effort in learning new functionality would bring them significant results.

This is well documented in the literature (Mackay, 1990; Browne *et al.*, 1990). Users tend to prefer using old methods, even if they are recognizably inferior, to expending the effort to learn a new method. The major use of customization facilities was to make a new system resemble a previous system (Mackay, 1990). Convincing a user to expend time to learn new functionality when a user has a pre-existing, sub-optimal method is difficult. Most software users recognize their tendency to reuse old, substandard techniques. Often, observing a user use a better method of a frequently used task will encourage them to modify their practices. Recommending a context-specific alternative is a gentle method of introducing new functionality to these types of users.

For the next step, the focus of the adaptations will expand to include the implementation of adaptations that are specifically designed for different user types and groups. Examples of these include restricting the system functionality and simplifying the system. The system would only perform the functions that are necessary for the accomplishment of the user's tasks. Experienced and intermediate users who have had some experience using the TIMS system will also be solicited for helpful suggestions. It might be helpful to delay the introduction of certain modifications to the system until the user has reached a minimum number of system usages.

Additional parameters to the user model could be added as a fairly simple future modification. The production system has increased the number and type of default system parameters. These parameters could be readily stored in the user model and passed on to the TIMS application. This might be an advantage over standard defaults to a particular installation of a system because the defaults are portable. The two situations where this would prove beneficial are: cases involving a single user who is moving between different machines and cases where a single machine is supporting multiple users. The user's defaults would always be in place, regardless of the location of the system.

The empirical component of this work has re-emphasized the importance of the user in all aspects of the system development life cycle. Clearly, the assumptions made in the development environment need to be confirmed with actual users in their work environments. Introducing formal statistical analysis to this procedure was interesting because it caused the team to be more cautious about over-interpreting positive results that might have occurred by chance. At the same time, the informal results that were gathered as anecdotal stories and comments helped the researchers to gain a better understanding of the users and their environment.

Some unanticipated problems occurred in the empirical testing. The overall level of computer familiarization of the users was overestimated. As a result, the reactions to P-TIMS were very much also reactions to TIMS as well. During the experiment, users barely had time to digest the unmodified TIMS system and then they were asked to judge the adaptations. Many users were unfamiliar with Windows, their new hardware, and the new software (including Microsoft Office as well as TIMS). It is not surprising that the ease of use reaction parameter was not very high. One user commented, "I feel like I am learning to drive a Ferrari!" Making small changes to simplify a single program is not going to help a user who is feeling that overwhelmed.

A longitudinal study which tracked users of comparable versions of TIMS and P-TIMS over a longer period of time – perhaps several months – will suggest results that will be more significant and perhaps more meaningful. This would also allow the user model to make substantive changes, which is difficult to do over the course of a three-hour session. The termination of supports during the experiment was not observed. This did not surprise the research team because all of the users were still newcomers to the software. While the user model can be expected to gradually evolve to more closely match the user's capabilities and preferences, this will obviously take some time. It would be interesting to examine the user models to determine which attributes had been turned off explicitly, the supports that the system had terminated, and the state of the system that the user had gradually evolved into. It would also be interesting to see if different users had chosen to retain some of the adaptations, and if so, why this was done.

Longer-term experiments are lacking in the user modelling literature in part because of the difficulty and expense in conducting these experiments. This research has demonstrated the difficulty of controlling the environment in a two-day session in a commercial environment. Increasing the monitoring of the adaptive and non-adaptive systems over time would introduce many extra variables. Each user would have completely different experiences with the system, using it for different tasks and for differing amounts of time. Formal analysis of the data would prove to be impossible. Nonetheless, any information gathered would still be very valuable, particularly interviews with the users.

Conducting interviews with individual users could shed some light on the overall impression of the system. Future development efforts would be greatly helped by answers to the following questions. Did P-TIMS provide the right supports for the proper amount of time? Were the adaptations awkward or unwanted? Were they confused about what the system was doing? And, was it clear that the user could control the different adaptations directly through the Preferences dialog? As Höök concluded, "the real test for intelligent user interfaces is whether they continue to be used after the initial excitement is gone" (Höök, 1997a).

The user group that would be participating in this proposed longitudinal study would also have a higher base level of knowledge, over the passage of time, due to a corporation-wide training and familiarization campaign of the new hardware and procedures, and a company-dictated suite of software (of which TIMS is only one component). Initial resistance to this radical difference in the company's operating procedures would also be lessened. It is hard to determine user satisfaction from the length of time that the participants in the current study were exposed to the system. Subsequent research will expand the measures that were used during the evaluation of the system from just user satisfaction to the following: speed, the time taken to complete tasks, the accuracy, the number of errors, and the time to learn. The research team is also considering a suggestion to attempt to capture the "annoyance factor" by providing users with the ability to provide feedback by pressing a button if they felt bothered.

The information that was gathered as a result of the work presented here was intended from the outset to direct changes to the production version of TIMS. The experiences of the P-TIMS researchers are similar to that of a large, four-year adaptive user interface project (Browne *et al.*, 1990), in which investigators found that the adaptations were not simply an extra facility that could differentiate between the system and its competitors. The adaptations were an integral part of an iterative design process. In addition, the system developers also dealt with several problems that pointed toward the use of adaptive techniques by redesigning portions of the system. This further echoes warnings in the literature regarding the need to employ adaptive technology where it is truly warranted, as opposed to using it to overcome common system design problems (Höök, 1997a; Horvitz, 1997).

In the P-TIMS system, the three phases of adaptations all proved to be moderately successful, and the modifications that were based on them have been implemented. The modifications that were felt to be most helpful to the novice user were the animated demonstrations and the directing of the user to the next logical system function. The current release of the ScreenCam program was determined to be inflexible and inadequate for these purposes. This triggered the development of a macro-language demonstration component that is available on the desktop at all times. This component, called Hints, was implemented in the TIMS-Lite[23] version, and it actually performs the function while the demonstration is taking place so that the user can choose to accept the changes when the demonstration is complete. The demonstrations that are offered to the user are dependent on the current context of the system. Three options are available for all the entries: "Explain It", "Show Me", and "Do It." Demonstrations are currently available for various planning and analysis functions (e.g. How to create a new client),

[23] TIMS-Lite is a reduced functionality system that was designed specifically to be easy-to-use.

reporting functions (e.g. How to create a client report), and the concepts (e.g. Setting up an education plan).

Hints are context-sensitive. The selection of available demonstrations depends on the current state of the desktop. As a simple example, if a client file is opened that does not contain a plan then the Hints section will suggest a demonstration on the plan creation process. This is in contrast with suggestions for a demonstration detailing procedures on the analysis of the plan if a client file contains a plan. This sensitivity to the file contents accomplishes two goals: first, the user is insulated from a large selection of choices and second, the user is protected from selecting an inappropriate demonstration.

It is difficult to decide whether or not to spend the effort to "reinvent the wheel" when there is an existing product that could provide the functionality. Integrating TIMS with the ScreenCam program was a reasonable decision for this research project, but it quickly became apparent that a better solution was needed for a marketable application. A similar situation occurred with the exporting procedure in the original system. As discussed earlier, the client RTF file was exported to the Microsoft Word program as a customized client report where it could be edited and printed. TIMS launched the MS Word program directly as part of the export process. The users were observed to have some interesting problems with the integration of the two products. MS Word is an extremely powerful, complex application in its own right; the boundary between the two systems was not clear to the users in this study. This contributed to a perception that TIMS was even more complicated.

It is not safe to assume that users have the necessary expertise in the integrated program, or that they will even understand that they are using a second program. These difficulties, among others, were also reported during the testing of three different applications that extended their functionality by launching Microsoft Excel (Schroeder, 1996). Developers in this case commented that, "Complementing your application with another feature-rich application is like getting a drink of water from a fire hydrant." The TIMS product now includes a simple RTF viewer on the desktop that allows users to view the client report in the TIMS program[24]. Novice users and users who do not wish to modify the report can easily print the report. The capability to export and edit the RTF file remains, but it can remain hidden until this functionality is necessary or desirable.

Over the course of this research the original TIMS system has evolved into a suite of three progressively more complex applications. The first, relatively straightforward system consists primarily of financial planning calculators and illustrations that can easily be printed. The next system provides all of the necessary functions for performing financial planning for most types of clients. However, it makes a large number of assumptions and is fairly general so that the user can focus on providing clients with general planning information and answers to most of the questions that their clients may have about their financial situations. The third, extended version of TIMS allows the user to have as much power, flexibility, and control as they wish, but it also requires a much more intensive data entry process to fully utilize all the functions. This comprehensive version has also changed. Some of the supports that were specifically targeted to inexperienced users, like the simple and detailed menu options, have been removed. This is because the other two systems focus on these types of users. All client files are upward-compatible so that users can make the transition up through the products as their needs and capabilities increase.

Different types of adaptations may even be helpful for expert users in this case. An experienced user will encounter new features as he or she moves in an upward progression through the line of products. Supports could easily be put in place that could make suggestions to the user about employing the new features and could illustrate how to use them. The task model need only add a new feature flag that could be turned off once the initial suggestion had been made. Expert users could not be expected to be aware of features that were not in the system that they were familiar with, but they are not exactly novice users either. By adding a new parameter to the task model, new tasks could be recommended, even with low complexity levels, and the user's previously earned user complexity level could stay intact. It would be relatively simple to use the preexisting user model to take advantage of the information that had already gathered

[24] The latest version of the software also supports in-place editing of documents.

about the user. A similar kind of adaptation would also assist all types of users who were upgrading from an existing system to a new release of the same system.

Minimalist user modelling components appear to show much promise. This experience with a pragmatic user modelling component in a commercial system has shown that the time and expense of including an adaptive user interface did improve the user satisfaction with the overall system, although the difference was not very dramatic. More importantly from a minimalist standpoint, the user modelling component had virtually no impact on the system response time. The concept was sound but better decisions needed to be made about the actual modifications to the system. It appears that user models are destined to be perceived as just one more knowledge source in future knowledge-based systems. Certainly, more empirical work is necessary because of items that were not clear in the experiment. Keeping the user central to all of the design and testing efforts remains key!

BIBLIOGRAPHY

Awad, E. (1996). *Building Expert Systems: Principles, Procedures, and Applications.* West Publishing Company, College & School Division.

Anderson, N., & Reitman Olson, J. (1987). Methods for Designing Software to Fit Human Needs and Capabilities. In Baecker and Buxton (Eds.), *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, California: Morgan Kaufmann Publishers, 540-554.

Barker, K., Evans, M., & Anderson, J. (1992). Measuring Autonomy in Heterogeneous Cooperative Systems. *AAAI Workshop on Cooperation Among Heterogeneous Intelligent Systems*, San Jose, California, USA, 11-16.

Benyon, D., Innocent, P., & Murray, D. (1987). *System Adaptivity and the Modelling of Stereotypes* (NPL Report DITC 91/87). National Physical Laboratory, Teddington, Middlesex, UK.

Benyon, D. (1993a). Accommodating Individual Differences through an Adaptive User Interface. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 149-166.

Benyon, D. (1993b). Adaptive Systems: A Solution to Usability Problems. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 3(1): 65-86.

Bobrow, D.G., Mittal, S., & Stefik, M.J. (1986). Expert Systems: Perils and Promise. *Communications of the ACM*, 29(9), 880-894.

Bonar, J. & Liffick, B. (1991). Communicating with High-Level Plans. In Sullivan, J. & Tyler, S. (Eds.), *Intelligent User Interfaces*, New York: ACM Press, 129-156.

Brachman, R. & Schmolze, J. (1985): An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9, 171-216.

Browne, D. (1990). Conclusions. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 195-222.

Browne, D., Totterdell, P., & Norman, M. (Eds.) (1990). *Adaptive User Interfaces*. London: Academic Press Ltd.

Browne, D., Norman, M., & Adhami, E. (1990). Methods for Building Adaptive Systems. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 85-130.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 6(2-3): 87-128.

Calistri-Yeh, R. J. (1991). Utilizing User Models to Handle Ambiguity and Misconceptions in Robust Plan Recognition. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 1(4): 289-322.

Carberry, S. (1989). Plan Recognition and Its Use in Understanding Dialog. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 133-162.

Carberry, S. (1991). Second International Workshop on User Modeling. *AI Magazine*. Vol. 11, no. 5., 57-60.

Card, S. K., Moran, T. P., & Newell, A. (1987). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Laurence Erlbaum Associates.

Carberry, S., & Kobsa, A. (1994). *User Modeling and User-Adapted Interaction*. Tutorial presented at the Fourth International Workshop on User Modeling: UM94, Hyannis, MA.

Carr, B., & Goldstein, I. (1977). *Overlays: A theory of modelling for computer aided instruction*. (AI Memo 406). Cambridge, MA: Massachusetts Institute of Technology, AI Laboratory.

Carroll, J. M., & McKendree, J. (1987, January). Interface Design Issues for Advice-Giving Expert Systems. *Communications of the ACM*. Vol. 30, no. 1., 14-31.

Cawsey, A. (1993). *Explanation and Interaction: The Computer Generation of Explanatory Dialogues*. Cambridge, MA: The MIT Press.

Cawsey, A. (1993). Planning interactive explanations. *International Journal of Man-Machine Studies*. 38:169-199.

Cawsey, A. (1993). User Modeling in Interactive Explanations. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 3(3):221-248.

Chin, D. N. (1989). KNOME: Modeling What the User Knows in UC. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 74-107.

Chin, J. P., Diehl, V., & Norman, K. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. *CHI'88 Conference Proceedings: Human Factors in Computing Systems*, New York: Association for Computing Machinery, 213-218.

Chin, J. P., Norman, K., & Shneiderman, B. (1987). *Subjective user evaluation of CF Pascal programming tools.* Technical Report (CAR-TR-304). College Park, MD: Human-Computer Interaction Laboratory, Center for Automation Research, University of Maryland.

Cockton, G. (1993). Spaces and Distances: Software Architecture and Abstraction and their Relation to Adaptation. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice.* Amsterdam: Elsevier Science Publishers B.V., 79-108.

Cohen, R., & Jones, M. (1989). Incorporating User Models into Expert Systems for Educational Diagnosis. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems.* New York: Springer-Verlag, 313-333.

Cook, R. & Kay, J. (1994). The justified user model: a viewable, explained user model. In *User Modeling: Proceedings of the Fourth International Conference, UM94*, Hyannis, MA, 145-150.

Cote-Munoz, J. A. (1993). AIDA - An Adaptive System for Interactive Drafting and CAD Applications. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice.* Amsterdam: Elsevier Science Publishers B.V., 225-240.

Cypher, A. (1995). Eager: Programming Repetitive Tasks by Example. In Baecker, R., Grudin, J., Buxton, W., & Greenberg, S. (Eds.), *Readings in Human-Computer Interaction: Toward the Year 2000.* Morgan Kaufmann Publishers, San Francisco, CA, 804-810.

De Rosis, F., Pizzutilo, S., Russo, A., Berry, D. C., & Nicolau Molina, F. J. (1992). *Modeling the User Knowledge by Belief Networks.* In Kobsa (Ed.), *User Modeling and User-Adapted Interaction.* 2(4):367-388.

Desmarais, M. (1997). Real world applications of user models. *Proceedings for Workshop of the Sixth International Conference, UM97: User Models in the Real World*, Sardinia, Italy, 3-6.

Dieterich, H., Malinowski, U., Kühme, T., Schneider-Hufschmidt, M. (1993). State of the Art in Adaptive User Interfaces. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice.* Amsterdam: Elsevier Science Publishers B.V., 13-48.

Dix, A., Finlay, J., Abowd, G., Beale, R. (1993). *Human-Computer Interaction.* New Year, NY: Prentice Hall.

Durkin, J. (1994). *Expert Systems: Design and Development.* New York: Macmillan Publishing Company.

EISI (1996). *TIMS for Windows: The Financial Planning Expert Quick Start Manual Release 3.0.* Emerging Information Systems, Inc., Winnipeg, Canada.

Espinoza, F. & Höök, K. (1996). A WWW Interface to an Adaptive Hypermedia System. In *Proceedings of the 2nd International Conference on the Practical Application of Intelligent Agents and Multi-Agent Methods*, London, England.

Finin, T. W. (1989). GUMS - A General User Modeling Shell. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems.* New York: Springer-Verlag, 411-430.

Fischer, G. (1993). Shared Knowledge in Cooperative Problem-Solving Systems - Integrating Adaptive and Adaptable Components. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice.* Amsterdam: Elsevier Science Publishers B.V., 49-68.

Fischer, G. & Reeves, B. (1995). Beyond Intelligent Interfaces: Exploring, Analyzing, and Creating Success Models of Cooperative Problem Solving (Excerpt). In Baecker, R., Grudin, J., Buxton, W., & Greenberg, S. (Eds.), *Readings in Human-Computer Interaction: Toward the Year 2000.* Morgan Kaufmann Publishers, San Francisco, CA, 822-831.

Foley, J. D. & Van Dam, A. (1984). *Fundamentals of Interactive Computer Graphics.* Reading, MA: Addison-Wesley Publishing Company.

Firefly (1998). URL: http://www.firefly.com [1998, April 10].

Fossay, C., & Sawchuk, J. (1994). *Marketing Strategy for TIMS©: The Tax and Investment Management System.* KPMG Peat Marwick Thorne, Winnipeg, Canada.

Goodman, D. (1996). *Javascript bible* (2nd ed.). Foster City, CA : IDG Books Worldwide.

Gordon, S. E. (1988, January). The Human Factor in Expert Systems. *AI Expert*, Vol. 3, no. 1., 55-59.

Greer, J., MacKenzie, M., & Koehn, G. (1996). *User Models for Persuasion, Coercion and Sales.* ARIES Laboratory Research Report 96-1, Department of Computer Science, University of Saskatchewan, Saskatoon, Canada.

Grudin, J., Ehrlich, S. F., & Shriner, R. (1987). Positioning Human Factors in the User Interface Development Chain. *Proceedings SIGCHI+CHI: Human Factors in Computing Systems*. Toronto, Canada, 125-131.

Grüninger, C. & van Treeck, W. (1993). Contributions of a Social Science Based Evaluation for Adaptive Design Projects. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 319-330.

Grunst, G. (1993). Adaptive Hypermedia for Support Systems. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 269-284.

Hayes-Roth, F., Waterman, D. A., & Lenat, D. B. (1983). *Building Expert Systems*. Reading, MA: Addison-Wesley Publishing Company.

Harper, B. D. & Norman, K. L. (1993). Improving User Satisfaction: The Questionnaire for User Interaction Satisfaction Version 5.5. *Proceedings of the 1st Annual Mid-Atlantic Human Factors Conference*, Virginia Beach, VA, 224-228.

Henderson, A., & Kyng, M. (1995). There's No Place Like Home: Continuing Design in Use. In: Baecker, R., Grudin, J., Buxton, W., & Greenberg, S. (Eds.), *Readings in Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, San Francisco, CA, 793-803.

Höök, K., Karlgren, J., Wærn, A., Dahlbäck, N., Jansson, C. G., Karlgren, K., & Lemaire, B. (1996). A Glass Box Approach to Adaptive Hypermedia. *User Modeling and User-Adapted Interaction* **6**(2-3), 157-184.

Höök, K., Karlgren, J., & Wærn, A. (1995). A Glass Box Approach to Intelligent Help. *IMMI-1 First Workshop on Intelligent Multi-Modal Interaction*, Edinburgh, U.K.

Höök, K. (1996). A Glass Box Approach to Adaptive Hypermedia, Ph.D. Dissertation, Department of Computer and Systems Sciences, Stockholm University. October 1996

Höök, K. (1997a). *Steps to take before IUI becomes real*. [WWW Document] URL: http://www.sics.se/~kia/papers/reality_of_II.html

Höök, K. (1997b). Evaluating the Utility and Usability of an Adaptive Hypermedia System. In *Proceedings of the International Conference on Intelligent User Interfaces*, Orlando, Florida, January. ACM.

Hoppe, U. H. (1993). Intelligent User Support Based on Task Models. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 167-182.

Horvitz, E. (1997). Agents with Beliefs: Reflections on Bayesian Methods for User Modeling. Invited talk. *User Modeling: Proceedings of the Sixth International Conference, UM97*, Sardinia, Italy.

IUI (1998). *1999 International Conference on Intelligent User Interfaces*. Redondo Beach, Los Angeles, CA, URL: http://sigart.acm.org/iui99 [1998, April 10]

Jameson, A. (1989). But What Will the Listener Think? Belief Ascription and Image Maintenance in Dialog. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 255-311.

Jameson, A., Paris, C., & Tasso, C. (1997). Preface. In Jameson, Paris, & Tasso (Eds.), *User Modeling: Proceedings of the Sixth International Conference UM'97*. New York: Springer-Verlag Wien.

Jennings, A. & Higuchi, H. (1993). A User Model Neural Network for a Personal News Service. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 3(1):1-26.

Johnson, P., Johnson, H., Waddington, R. & Shouls, A. (1988). Task-related knowledge structures: Analysis, modelling and application. In D. M. Jones & R Winder (Eds.), *People and Computers: From research to implementation*. Cambridge: Cambridge University Press, 35-62.

Kantorowitz, E., & Sudarsky, O. (1989, November). The Adaptable User Interface. *Communications of the ACM*. Vol. 32, no. 11., 1352-1358.

Kass, R. (1989). Student Modeling in Intelligent Tutoring Systems - Implications for User Modeling. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 386-410.

Kass, R. (1991). Building a User Model Implicitly from a Cooperative Advisory Dialog. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 1(3):203-258.

Kass, R. & Finin, T. (1988). *The Need for User Models in Generating Expert System Explanations*. MS-CIS-88-37, University of Pennsylvania, Philadelphia, PA.

Kay, J. (1993). Pragmatic User Modelling for Adaptive Interfaces. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 129-148.

Kay, J. (1994). Lies, Damned Lies and Stereotypes: Pragmatic Approximations of Users. *Proceedings UM'94, Fourth International. Conference on User Modeling*, Hyannis, MA, 175-184.

Kay, J. (1995). Vive la difference! Individualised Interaction with Users. *In Proceedings of IJCAI '95*. Montreal, Canada, 978-984.

Kay, J., & Fischer, G. (1997). Introduction. *Proceedings for Workshop of the Sixth International Conference, UM97: User Models in the Real World*, Sardinia, Italy, 1-2.

Kobsa, A. (1989). A Taxonomy of Beliefs and Goals for User Models in Dialog Systems. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 52-69.

Kobsa, A. (1993). User Modeling: Recent Work, Prospects and Hazards. In Schneider-Hufschmidt, Kuhme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 111-128.

Kobsa, A., Müller, D., & Nill, A. (1994). KN-AHS: An Adaptive Hypertext Client of the User Modeling System BGP-MS. *In Proceedings of the Fourth International Workshop on User Modeling: UM94*, Hyannis, MA, 99-105.

Kobsa, A. & Pohl, W. (1995). The user modeling shell BGP-MS. In *User Modeling and User-Adapted Interaction* 2:59-106.

Kobsa, A. & Wahlster, W. (1989). In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 52-69.

Koller, F. (1993). A Demonstrator Based Investigation of Adaptability. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 183-196.

Krogsæter, M., Oppermann, R., & Thomas, C. (1994). A User Interface Integrating Adaptability and Adaptivity. In Oppermann, R. (Ed.), *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*. Hillsdale, NJ: Lawrence Erlbaum Associates, 97-125.

Krogsæter, M., & Thomas, C. (1994). Adaptivity: System-Initiated Individualization. In Oppermann, R. (Ed.), *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*. Hillsdale, NJ: Lawrence Erlbaum Associates, 67-96.

Lewis, C. & Norman, D. A. (1986). Designing for Error. In Norman and Draper (Eds.), *User Centered System Design*. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 411-432.

Lehman, J. F. & Carbonell, J. G. (1989). Learning the User's Language: A Step Towards Automated Creation of User Models. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 163-194.

Lotus Development Corporation. (1995). *Lotus ScreenCam 2.0 for Windows*. Lotus Development Corporation, 55 Cambridge Parkway, MA, URL: http://www.lotus.com

Mackay, W. (1990). Patterns of Sharing Customizable Software. *Proceedings CSCW'90*, ACM, 209-221.

Malinowski, U., & Schneider-Hufschmidt, M. (1993). Discussion Results. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 331-336.

Maes, P. (1995). Agents that Reduce Work and Information Overload. In: Baecker, R., Grudin, J., Buxton, W., & Greenberg, S. (Eds.), *Readings in Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers, San Francisco, CA, 811-821.

Malinowski, U. (1996). ABIS-95: GI Workshop on Adaptivity and User Modeling in Interactive Software Systems. *User Modeling and User-Adapted Interaction* **6**(1), 69-76.

Mark, M., & Greer, J. (1993). Evaluation Methodologies for Intelligent Tutoring Systems. *Journal of Artificial Intelligence in Education*, 4, 129-153.

Mark, M., & Greer, J. (1995). The VCR Tutor: Effective Instruction for Device Operation. *Journal of the Learning Sciences*, 4, 209-246.

Maskery, H. (1984). Adaptive Interfaces for Naive Users – An Experimental Study. In *Interact '84*, North-Holland, 343-349.

May, J., Barnard, P. J., & Blandford, A. (1993). Using Structural Descriptions of Interfaces to Automate the Modelling of User Cognition. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 3(1):27-64.

McCalla, G., Searwar, F., Thomson, J., Collins, J., Sun, Y., & Zhou, B. (1996). Analogical User Modelling: A Case Study in Individualized Information Filtering. *Proceedings UM'94, Fourth Int. Conference on User Modeling*, Hyannis, MA, 13-20.

McCoy , K. F. (1989). Highlighting a User Model to Respond to Misconceptions. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 233-254.

McTear, M. (1993). User Modelling for Adaptive Computer Systems: A Survey of Recent Developments. *Artificial Intelligence Review* 7, 157-184.

McTear, M. (1996). Introduction. *Proceedings for Workshop of UM'96: The Commercial Potential of User Modeling*, Kona, HI, 1.

Meyer, B. (1994). Adaptive Performance Support: User Acceptance of a Self-Adapting System. *Proceedings UM'94, Fourth International. Conference on User Modeling*, Hyannis, MA, 65-70.

Microsoft (1998). It's Your Microsoft Office 97. [WWW Document], URL: http://www.microsoft.com/office/office97/articles/off97_1.html [1998, April 10]

Miller, C. (1996). *Presentation at Workshop of UM'96: The Commercial Potential of User Modeling*, Kona, HI.

Moore, J. D. & Paris, C. L. (1992). Exploiting User Feedback to Compensate for the Unreliability of User Models. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 2(4):287-330.

Morik, K. (1989). User Models and Conversational Settings: Modeling the User's Wants. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 364-385.

Murphy, M. & McTear, M. (1997). Learner Modelling for Intelligent CALL. *User Modeling: Proceedings of the Sixth International Conference, UM97*. Vienna, New York: Springer Wien New York, 301-312.

Murray, D. M. (1988). *A Survey of User Cognitive Modelling*. NPL Report DITC 92/87, National Physical Laboratory, Teddington, Middlesex, UK.

Myers, B. A. (1988). *Tools for Creating User Interfaces: An Introduction and Survey*. Department of Computer Science, Carnegie Mellon University. CMU-CS-88-107.

Myers, B. (1991). Using AI Techniques to Create User Interfaces by Example. In Sullivan, J. & Tyler, S. (Eds.), *Intelligent User Interfaces*, New York: ACM Press, 385-399.

Myers, B.A. & Rosson, M.B. (1992). Survey on user interface programming. In: Bauersfeld, P.; Bennett, J. & Lynch, G. (Eds.) *CHI'92 Conference Proceedings: ACM Conference on Human Factors in Computing Systems*. Reading, MA: Addison-Wesley; 195-202.

My Yahoo! (1998). URL: http://my.yahoo.com [1998, April 10]

Nielsen, J. (1993, November). Iterative User-Interface Design. *Computer*, Vol. 26, no. 11, 32-41.

Nielsen, J., & Levy, J. (1994). Measuring Usability: Preference vs. Performance. *Communications of the ACM*. Vol. 37, No.4, 66-75.

Nolan, Norton and Co. (1992). *Managing End-User Computing*. Boston: Nolan, Norton & Co.

Norman, D. A. (1993). *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine*. Reading, MA: Addison-Wesley Publishing Company.

Norman, D. (1994). How People Might Interact with Agents. *Communications of the ACM*. 37(7), 69-71.

Norman, D.A. (1986). Cognitive engineering. In Norman, D.A. and Draper, S.W. (Eds.), *User centered system design*. Hillsdale, NJ: Erlbaum, 31-61.

Oppermann, R. (Ed.) (1994). *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*. Hillsdale, NJ: Lawrence Erlbaum Associates.

P3P (1998). *Platform for Privacy Preferences: P3P Project*. [WWW Document] URL: http://www.w3.org/P3P/P3FAQ.html [1998, May 17]

Paris, C. L. (1989). The Use of Explicit User Models in a Generation System for Tailoring Answers to the User's Level of Expertise. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 200-232.

Poor, A. (1987, August). Keyboards Beyond the Ordinary. *PC Magazine*, **6**(14), 99-102.

Quilici, A. (1989). Detecting and Responding to Plan-Oriented Misconceptions. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 108-132.

QUIS (1998). URL: http://www.lap.umd.edu/QUISFolder/quisHome.html [1998, April 10]

Raeder, G. (1985, August). A Survey of Current Graphical Programming Techniques. *Computer*, **18**(8), 11-25.

Ray, W., & Ravizza, R. (1988). *Methods Toward a Science of Behavior and Experience* (3rd ed.). Wadsworth Publishing Co., Belmont, CA.

Rennels, G. D., & Shortliffe, E. H. (1987). Advanced Computing for Medicine. *Scientific American*.

Rich, E. (1979). *Building and Exploiting User Models*. Ph.D. Dissertation, Department of Computer Science, Carnegie Mellon University.

Rich, E. (1983). *Artificial Intelligence*. New York, NY. McGraw-Hill Book Company, 436.

Rich, E. (1989). Stereotypes and User Modeling. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag.

Rissland, E. (1987). Ingredients of Intelligent User Interfaces. In Baecker and Buxton (Eds.), *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, California: Morgan Kaufmann Publishers, 703-708.

Sanderson, D. P., & Treu, S. (1993). Adaptive User Interface Design and Its Dependence on Structure. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 241-268.

Schneider-Hufschmidt, Kühme & Malinowski (Eds.) (1993), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V.

Schroeder, W. (1996, Nov/Dec). Integrating Other Applications. *Eye for Design*. North Andover, MA: User Interface Engineering.

Schweighardt, M. F. & Carter, J. A. (1988). *Explicit User Models for Adaptive User Interfaces*. Research Report, University of Saskatchewan, Canada.

Self, J. (1993). Model-Based Cognitive Diagnosis. In Kobsa (Ed.), *User Modeling and User-Adapted Interaction*. 3(1):87-106.

Shapiro, S. C. (1990). *Encyclopedia of Artificial Intelligence*. New York: John Wiley & Sons.

Sherman, E. H. & Shortliffe, E. H. (1993). A User-Adaptable Interface to Predict Users' Needs. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 285-316.

Shneiderman, B. (1984). Correct, Complete Operations and Other Principles of Interaction. In Salvendy (Ed.), Human-Computer Interaction, Amsterdam: Elsevier Science Publishers B.V., 135-146.

Shneiderman, B. (1992). *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (2nd edition). Reading, MA: Addison-Wesley.

Sparck Jones, K. (1989). Realism About User Modeling. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 341-363.

Strachan, L., Evans, M. & McAnanama, L. (1990). An Investigation into the Issues Surrounding Human-Computer Interaction. In *Proceedings of the National CIPS Congress 90*, Ottawa, Ontario.

Sukaviriya, P. N. & Foley, J. D. (1993). A Built-in Provision for Collecting Individual Task Usage Information in UIDE: the User Interface Design Environment. In Schneider-Hufschmidt, Kühme & Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice*. Amsterdam: Elsevier Science Publishers B.V., 197-222.

Sullivan, J. & Tyler, S. (Eds.). (1991). *Intelligent User Interfaces*. New York: ACM Press.

Thimbleby, H. (1983). User Interface Design: Generative User Engineering Principles. In Monk (Ed.), *Fundamentals of Human-Computer Interaction*. London: Academic Press, 165-180.

Totterdell, P. (1990). Introduction. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 1-14.

Totterdell, P. & Rautenbach, P. (1990). Adaptation as a Problem of Design. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 59-84.

Totterdell, P., Rautenbach, P., Wilkinson, A., & Anderson, S. O. (1990). Adaptive Interface Techniques. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 131-160.

Totterdell, P. & Boyle, E. (1990). The Evaluation of Adaptive Systems. In Browne, Totterdell and Norman (Eds.), *Adaptive User Interfaces*. London: Academic Press Ltd., 161-194.

Tyler, S., Schlossberg, J., Gargan, R., Cook, L., & Sullivan, J. (1991). An Intelligent Architecture for Adaptive Interaction. In Sullivan, J. & Tyler, S. (Eds.), *Intelligent User Interfaces*, New York: ACM Press, 85-109.

Vassileva, J. (1994). A Practical Architecture for User Modeling in a Hypermedia-Based Information System. *User Modeling: Proceedings of the Fourth International Conference, UM94*, Hyannis, MA, 115-126.

Vassileva, J. (1996). User Modeling in a Hypermedia Office Documentation System. *User Modeling and User-Adapted Interaction* **6**(2-3), 185-223.

Wahlster, W. (1991). User and Discourse Models for Multimodal Communication. In Sullivan, J. & Tyler, S. (Eds.), *Intelligent User Interfaces*, New York: ACM Press, 45-68.

Wahlster, W. & Kobsa, A. (1989). User Models in Dialog Systems. In Kobsa and Wahlster (Eds.), *User Models in Dialog Systems*. New York: Springer-Verlag, 4-34.

Waterman, D. A. (1986). *A Guide to Expert Systems*. Reading, MA: Addison-Wesley Publishing Company.

Weber, G. & Specht, M. (1997). User modeling and adaptive navigation support in WWW-based tutoring systems. In *User Modeling: Proceedings of the Sixth International Conference, UM97*, Vienna, New York: Springer Wien New York. URL: http://www.psychologie.uni-trier.de:8000/projects/ELM/elmart.html [1998, April 10]

Wexelblat, R. L. (1989, Fall). On Interface Requirements for Expert Systems. *AI Magazine*, **10**(3), 66-78.

Winston, P. H. (1977). *Artificial Intelligence*. Reading, MA: Addison-Wesley Publishing Company.

Wong, G., & Rengger, R. (1990). *The validity of questionnaires designed to measure user-satisfaction of computer systems*. National Physical Laboratory Report DITC 169/90, Teddington, Middlesex, UK.

APPENDIX A

APPLICATION MODEL

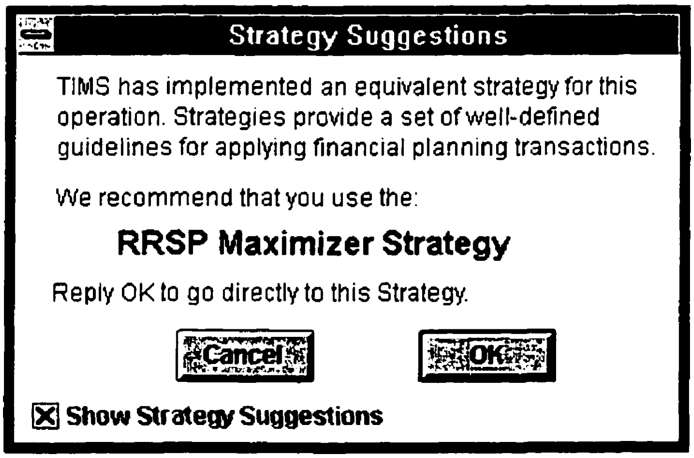
STRATEGY RECOMMENDATIONS

P-TIMS Application Model

Strategy Recommendations

Note: This text is presented in a Strategy Suggestions dialog following the detection of a complex task performed by a user with a complexity rating lower than the task complexity rating in the task model. Each recommended strategy is equivalent to the detected task.

The following text describes the conditions used in the detection of the complex task and the strategy recommendations presented in an English-text form.



Equivalent Strategies:

1.	Location:	Detailed Assets
	Transaction:	Future Returns
	Selection:	N/A
	Asset Type:	All
	Recommendation:	Investment Returns Strategy
2.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Surplus
	Asset Type:	RRSP
	Recommendation:	Surplus RRSP Cash Strategy
3.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Surplus
	Asset Type:	RRIF
	Recommendation:	Surplus RRIF Cash Strategy

4. Location: Detailed Assets
Transaction: Buys
Selection: Surplus
Asset Type: All
Recommendation: Surplus Cash Savings Strategy

5. Location: Detailed Assets
Transaction: Buys
Selection: Lump Sum
Asset Type: All
Recommendation: Lump Sum Savings Strategy

6. Location: Detailed Assets
Transaction: Buys
Selection: Periodic
Asset Type: RRSP
Recommendation: Regular Savings Strategy

7. Location: Detailed Assets
Transaction: Buys
Selection: RRSP Maximizer
Asset Type: RRSP
Recommendation: RRSP Maximizer Strategy

8. Location: Detailed Assets
Transaction: Sells
Selection: Lump Sum
Asset Type: All
Recommendation: Lump Sum Asset Redemption Strategy

9. Location: Detailed Assets
Transaction: Sells
Selection: Periodic
Asset Type: All
Recommendation: Regular Redemption Strategy

10. Location: Detailed Assets
Transaction: Sells
Selection: Deficit Coverage
Asset Type: All
Recommendation: Deficit Coverage Strategy

11. Location: Detailed Assets
Transaction: Sells
Selection: Complete Periodic
Asset Type: All
Recommendation: Complete Regular Redemption Strategy

12. Location: Detailed Assets
Transaction: Transfers
Asset Type: All
Recommendation: Transfers Strategy

13.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Periodic
	Asset Type:	All
	Recommendation:	Regular Debt Reduction Strategy
14.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Lump Sum
	Asset Type:	All
	Recommendation:	Lump Sum Debt Reduction Strategy
15.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Surplus
	Asset Type:	All
	Recommendation:	Surplus Debt Reduction Strategy
16.	Location:	Detailed Liabilities
	Transaction:	Asset Links
	Asset Type:	All
	Recommendation:	Leveraged Investments Strategy

APPENDIX B

APPLICATION MODEL

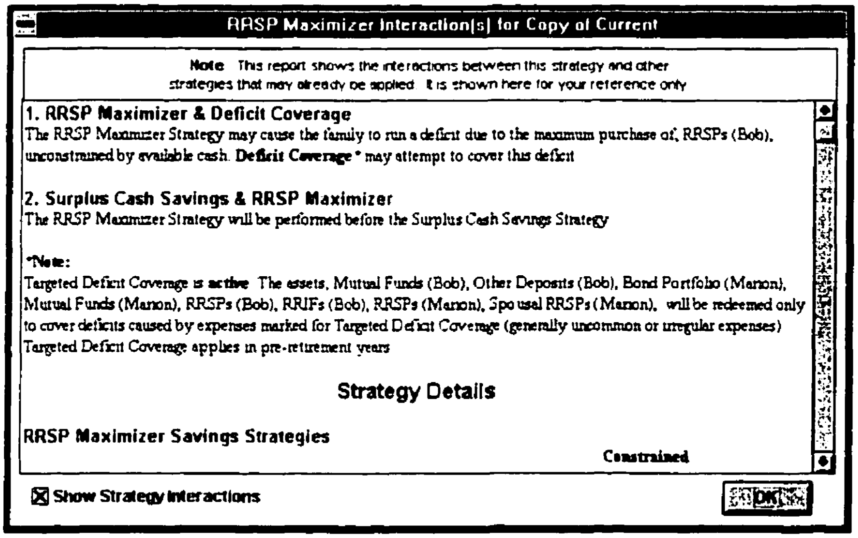
STRATEGY INTERACTIONS

P-TIMS Application Model

Strategy Interactions

Note: This text is presented in a Strategy Interactions dialog following the implementation of a strategy (the OK button is clicked in the strategy). The situation's currently implemented strategies and the interactions with the new strategy are checked and an informational message is presented only when an interaction is noted. The intent of these messages is to clarify the strategy interactions for the novice user since they may not be aware of the possible side effects of two or more strategies for a given situation. A complete list of all the strategy interactions detected is available from the Report pull-down menu in a Strategy Interaction Report.

The following text describes the rules used in the detection of the strategy interactions presented in an English-text form. Where appropriate the actual asset name(s) will be replaced in the text string and all of the detailed strategy information is presented after the detected interactions.



Surplus Cash Savings & RRSP Maximizer & Surplus Debt Reduction
SI-1:
This combination of strategies will be performed in the following order: 1) RRSP Maximizer, 2) Surplus Debt Reduction and 3) Surplus Cash Savings. There may be less surplus cash for debt reduction because of the RRSP Maximizer Strategy. Savings will take place only if there is a remaining surplus following debt reduction.

Surplus Cash Savings & RRSP Maximizer
SI-2:
The RRSP Maximizer Strategy will be performed before the Surplus Cash Savings Strategy.

Surplus Cash Savings & Surplus Debt Reduction
SI-3:
The Surplus Debt Reduction Strategy will be performed before the Surplus Cash Savings Strategy.

Surplus Debt Reduction & RRSP Maximizer
SI-4:
The RRSP Maximizer Strategy will be performed before the Surplus Debt Reduction Strategy. There may be less surplus cash for debt reduction because of the RRSP Maximizer Strategy.

The following (SI-5 => SI-17) have text changes. The appropriate note will be generated following the description of the interactions.

***Note:**
Targeted deficit coverage is **active**. The assets, asset_list, will be redeemed only to cover deficits caused by expenses marked for Targeted Deficit Coverage (generally uncommon or irregular expenses). Targeted Deficit Coverage only applies in pre-retirement years.

***Note:**
Targeted deficit coverage is **inactive**. The assets, asset_list, will be redeemed as necessary to cover any deficits occurring at year-end. Redemption ordering is based on tax implications.

Lump Sum Savings, Regular Savings & Deficit Coverage (non-registered)
Lump Sum Savings & Deficit Coverage (non-registered)
Regular Savings & Deficit Coverage (non-registered)
(Applicable periods match)
SI-5, SI-6, SI-7:
The non-registered asset, asset_name, used in the Lump Sum Savings Strategy and the non-registered asset, asset_name, used in the Regular Savings Strategy are also used by the Deficit Coverage Strategy. This may cause a problem if the savings level is too high and this causes the family to go into a deficit situation. Deficit Coverage* may in turn sell some of this asset to cover the deficit.

Lump Sum Savings, Regular Savings & Deficit Coverage (registered)
Lump Sum Savings & Deficit Coverage (registered)
Regular Savings & Deficit Coverage (registered)
(Applicable periods match)
SI-8, SI-9, SI-10:
The registered asset, asset_name, used in the Lump Sum Savings Strategy and the registered asset, asset_name, used in the Regular Savings Strategy are also used by the Deficit Coverage Strategy. This may cause a problem if the savings level is too high and this causes the family to go into a deficit situation. If the deficit is severe enough, Deficit Coverage* may in turn sell some of this asset if the registered asset is available for deficit coverage in the same period. This normally is not a desirable situation.

RRSP Maximizer & Deficit Coverage
(With unconstrained option on for RRSP Maximizer)
SI-11:
The RRSP Maximizer Strategy may cause the family to run a deficit due to the maximum purchase of RRSPs unconstrained by available cash. Deficit Coverage* may attempt to cover this deficit.

Surplus Cash Savings & (Regular Redemption Strategy or Lump Sum Redemption or Complete Regular Redemption)
(Applicable periods match)
SI-12:
The sale of the asset, asset_name, may result in funds in excess of those needed for the situation's expenses. Buys of the asset(s) specified in the Surplus Cash Savings Strategy may be performed.

Complete Regular Redemption & Deficit Coverage
(For the same asset)
SI-13:
The asset, asset_name, has both Complete Regular Redemption and Deficit Coverage* Strategies associated with it. This may cause this asset to be fully redeemed before the end of the period.

(Surplus Cash Savings or Regular Savings or Lump Sum Savings) & Complete Regular Redemption
(For the same asset)
(Applicable periods overlap)
SI-14:
The asset, asset_name, has both a savings strategy and the Complete Regular Redemption Strategy associated with it. This may prevent this asset from being fully redeemed by the end of the period.

Life Insurance Payout & Surplus Cash Savings
(No lump sum strategy)
SI-15:
The proceeds of a life insurance policy result in surplus cash and the subsequent purchase of assets, asset_list, specified in the Surplus Cash Savings Strategy. In addition, normal expenses may be significantly reduced by the loss of mortgage payments through mortgage insurance coverage. This will occur only if the life insurance policy is still in effect at the time of death.

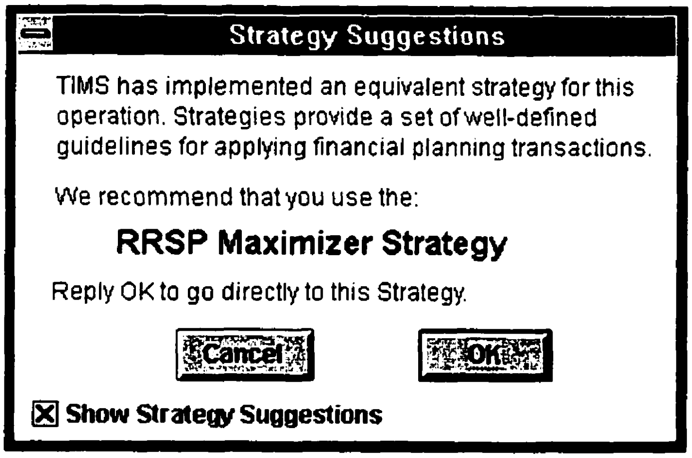
Life Insurance Payout & Lump Sum Savings
(Lump sum strategy)
SI-16:
The proceeds of a life insurance policy will be used to purchase the asset, asset_name, specified in the Lump Sum Savings Strategy that was inserted automatically by the

system. This will occur only if the life insurance policy is still in effect at the time of death.

Surplus Cash Savings & Deficit Coverage
SI-17:
The Surplus Cash Savings strategy will take surplus funds and purchase the assets, asset_names. In years where a deficit occurs these assets will not receive funds. Deficit Coverage* may attempt to cover this deficit.

Note: This text is presented in a Strategy Suggestions dialog following the detection of a complex task performed by a user with a complexity rating lower than the task complexity rating in the task model. Each recommended strategy is equivalent to the detected task.

The following text describes the conditions used in the detection of the complex task and the strategy recommendations presented in an English-text form.



Equivalent Strategies:

1.	Location:	Detailed Assets
	Transaction:	Future Returns
	Selection:	N/A
	Asset Type:	All
	Recommendation:	Investment Returns Strategy
2.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Surplus
	Asset Type:	RRSP
	Recommendation:	Surplus RRSP Cash Strategy

3.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Surplus
	Asset Type:	RRIF
	Recommendation:	Surplus RRIF Cash Strategy
4.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Surplus
	Asset Type:	All
	Recommendation:	Surplus Cash Savings Strategy
5.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Lump Sum
	Asset Type:	All
	Recommendation:	Lump Sum Savings Strategy
6.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	Periodic
	Asset Type:	RRSP
	Recommendation:	Regular Savings Strategy
7.	Location:	Detailed Assets
	Transaction:	Buys
	Selection:	RRSP Maximizer
	Asset Type:	RRSP
	Recommendation:	RRSP Maximizer Strategy
8.	Location:	Detailed Assets
	Transaction:	Sells
	Selection:	Lump Sum
	Asset Type:	All
	Recommendation:	Lump Sum Asset Redemption Strategy
9.	Location:	Detailed Assets
	Transaction:	Sells
	Selection:	Periodic
	Asset Type:	All
	Recommendation:	Regular Redemption Strategy
10.	Location:	Detailed Assets
	Transaction:	Sells
	Selection:	Deficit Coverage
	Asset Type:	All
	Recommendation:	Deficit Coverage Strategy
11.	Location:	Detailed Assets
	Transaction:	Sells
	Selection:	Complete Periodic
	Asset Type:	All
	Recommendation:	Complete Regular Redemption Strategy

12.	Location:	Detailed Assets
	Transaction:	Transfers
	Asset Type:	All
	Recommendation:	Transfers Strategy
13.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Periodic
	Asset Type:	All
	Recommendation:	Regular Debt Reduction Strategy
14.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Lump Sum
	Asset Type:	All
	Recommendation:	Lump Sum Debt Reduction Strategy
15.	Location:	Detailed Liabilities
	Transaction:	Principal Modifier
	Selection:	Surplus
	Asset Type:	All
	Recommendation:	Surplus Debt Reduction Strategy
16.	Location:	Detailed Liabilities
	Transaction:	Asset Links
	Asset Type:	All
	Recommendation:	Leveraged Investments Strategy

APPENDIX C

SCREENCAM DEMONSTRATION

Figure 1.

All ScreenCam™ Demonstrations start with a disclaimer to reassure users that they have not inadvertently triggered a sequence of events that might damage their situation.

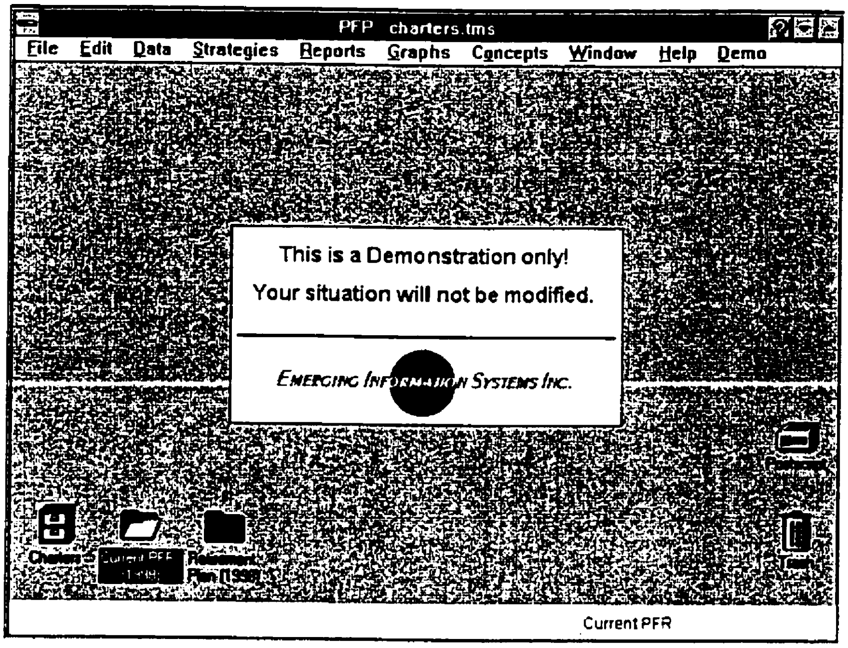


Figure 2.

Show the title of the demonstration.

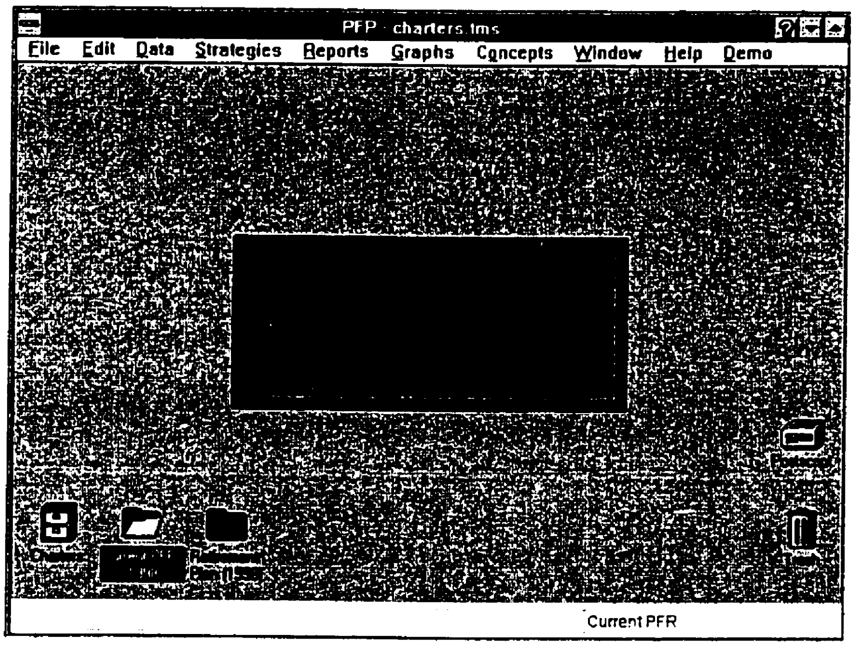


Figure 3.

Showing the location of the strategy command in the menu bar. The RRSP Maximizer Strategy is one of the Savings strategies.

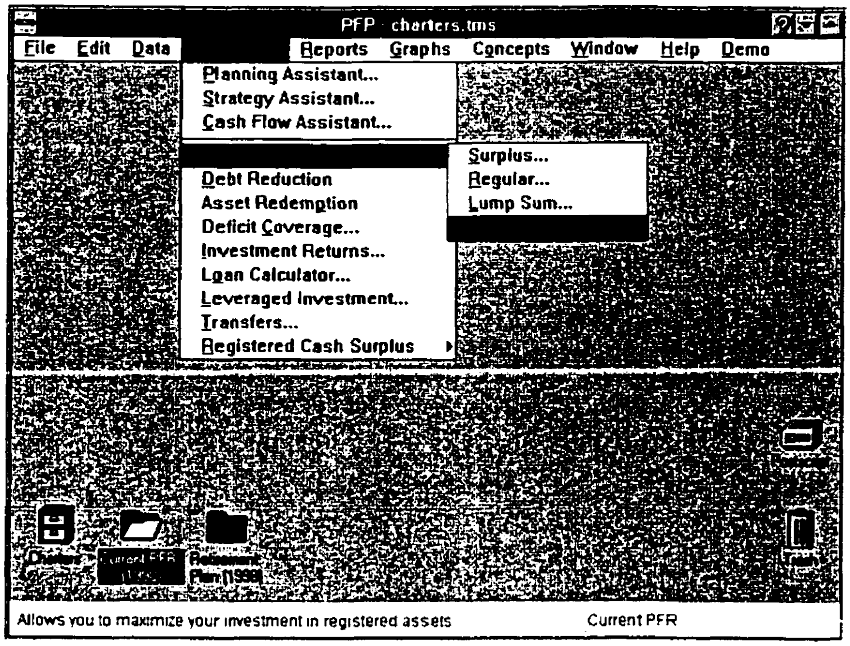


Figure 4.

Initiate the strategy and briefly explain the purpose of the strategy.

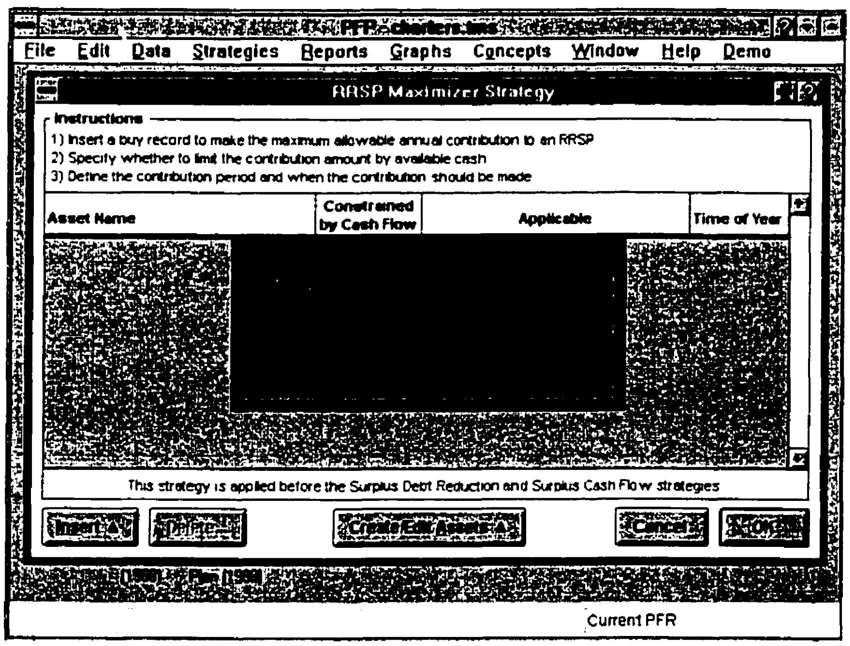


Figure 5.

Demonstrate the operations of the strategy. The first step in the RRSP Maximizer Strategy is to insert the registered assets that the user wishes to maximize.

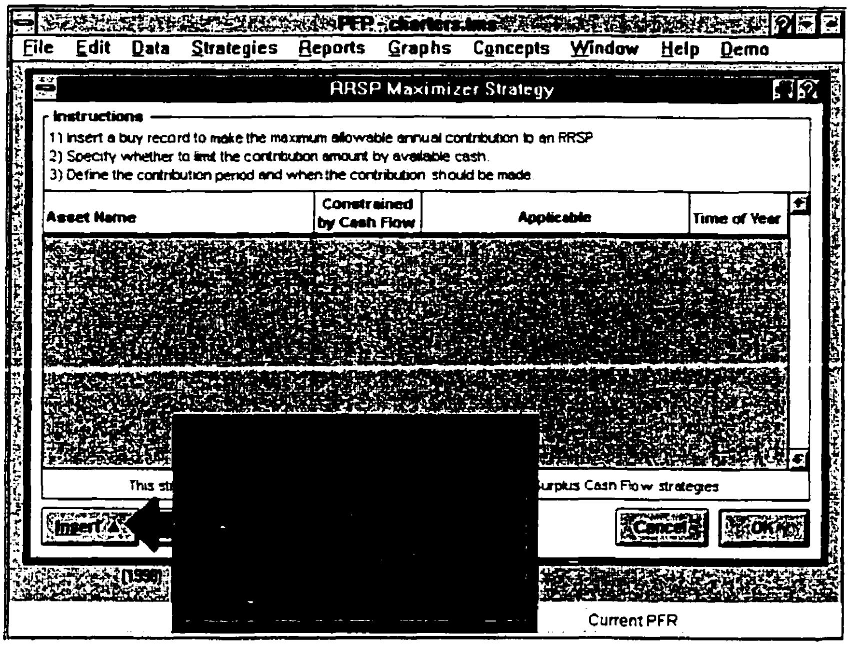


Figure 6.

Select one of the current family's registered assets.

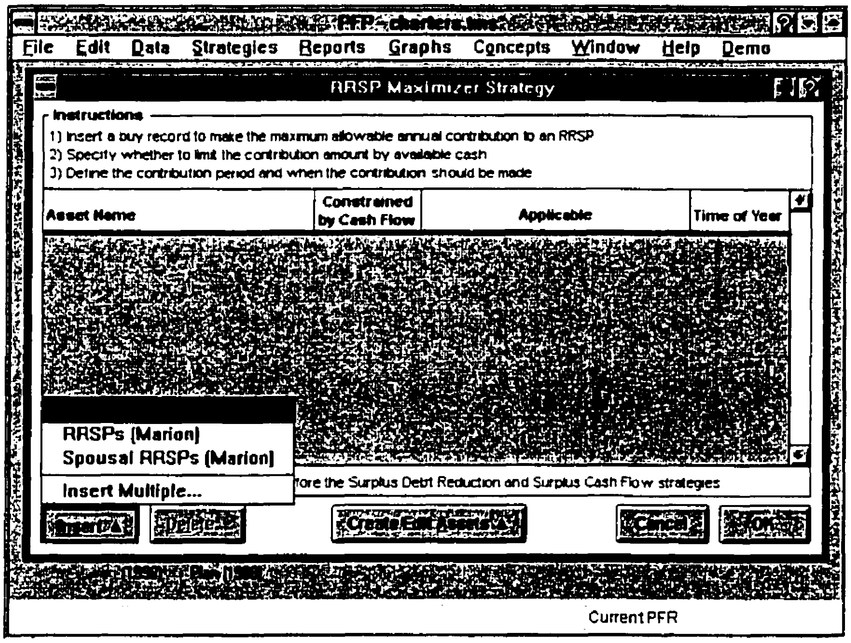


Figure 7.

A brief explanation of the common options is given.

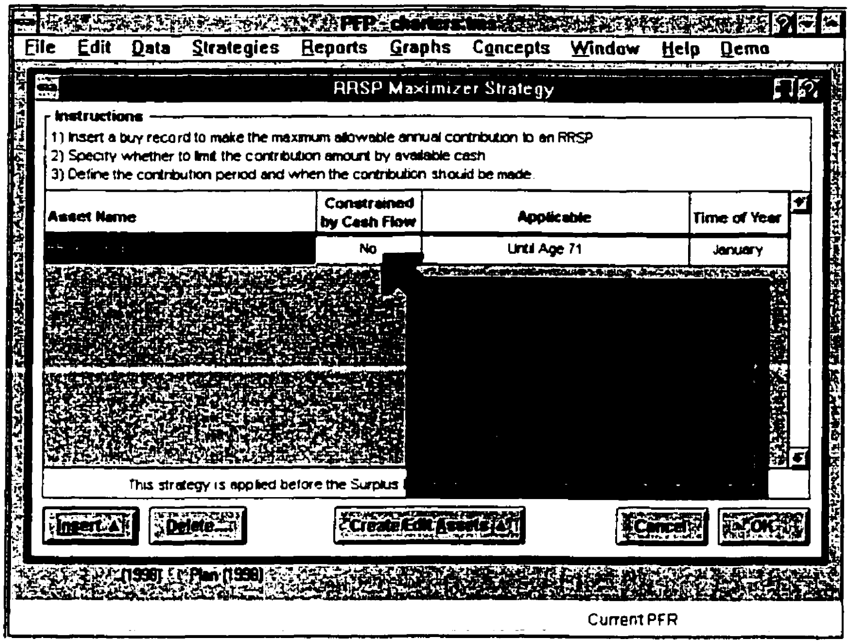


Figure 8.

Show the pull-down menu of options.

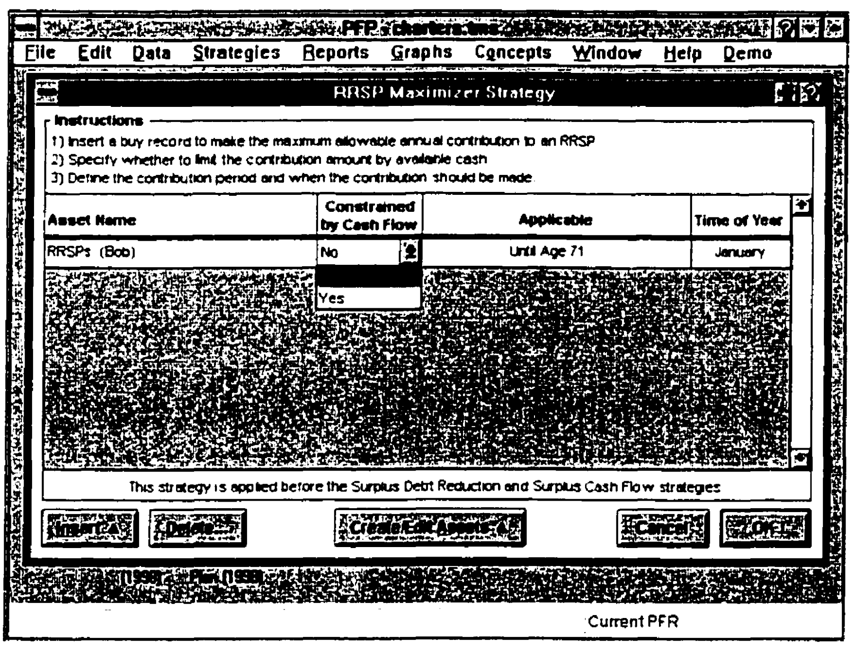


Figure 9.

Select the period of time when the strategy will be active. The RRSP Maximizer strategy default is until the owner of the registered asset (Bob) reaches the age of 71.

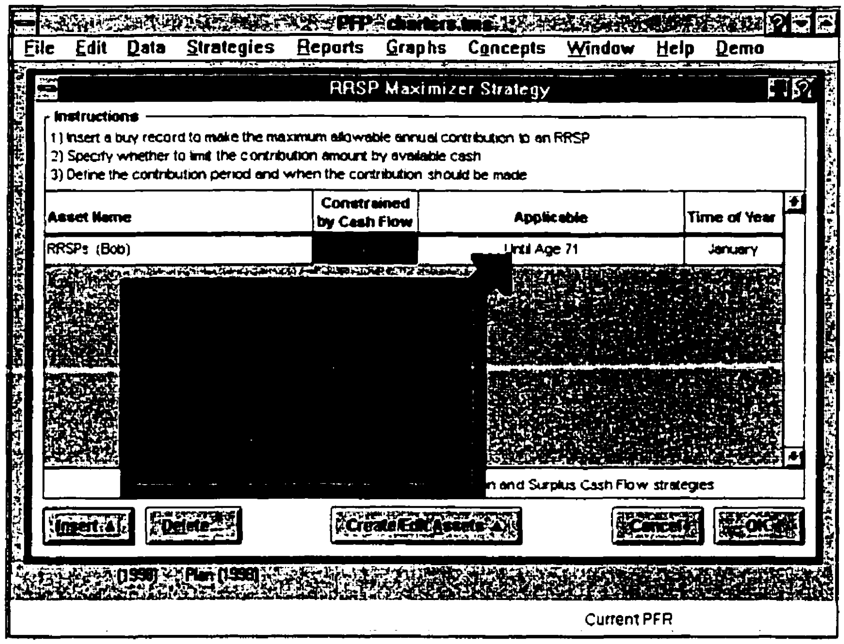


Figure 10.

Other choices include only while the owner of the asset is working or a specific date that the user specifies.

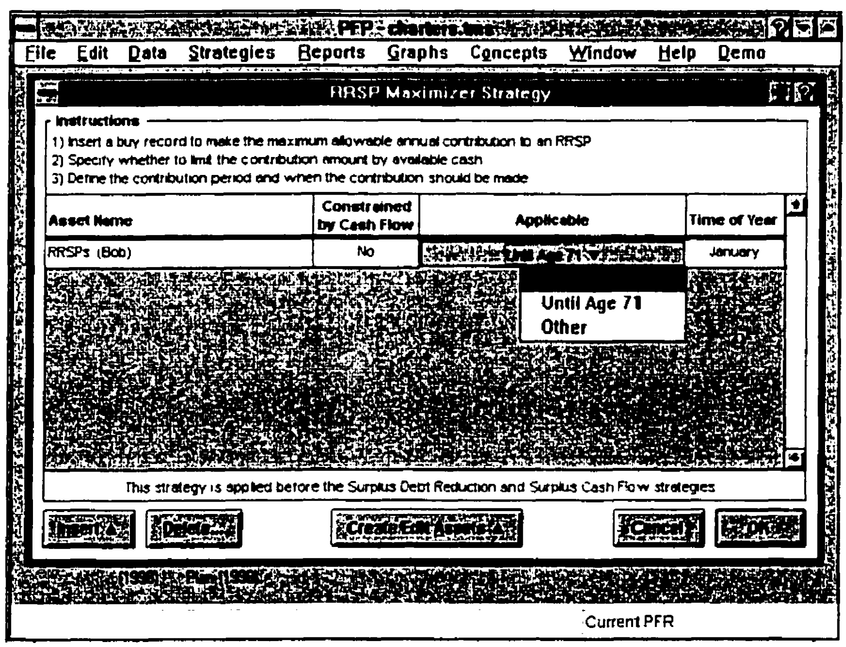


Figure 11.

Insert any number of assets into the strategy. Only one asset is inserted in the demonstration.

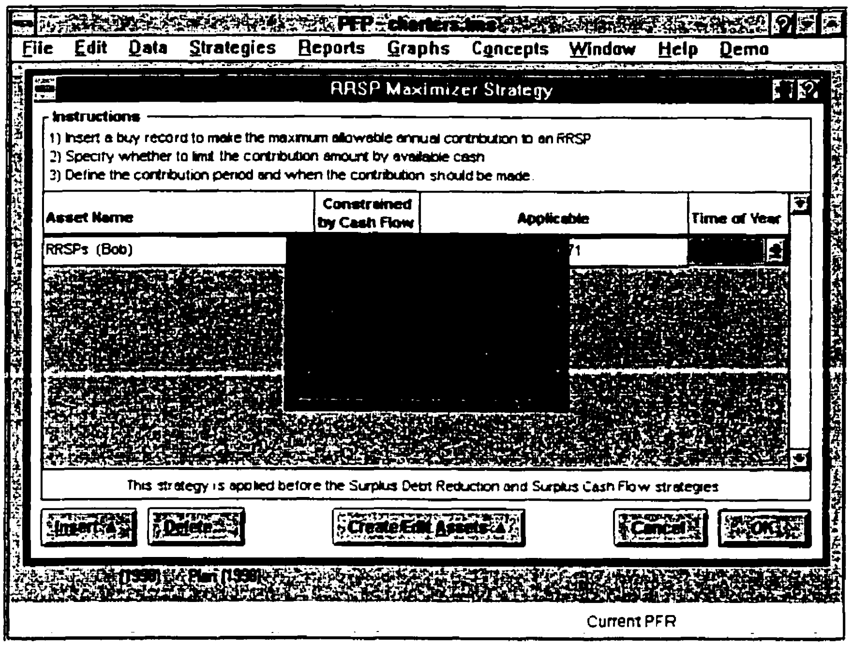


Figure 12.

The ScreenCam™ demonstration for this particular strategy can be re-triggered from this movie icon in the title bar.

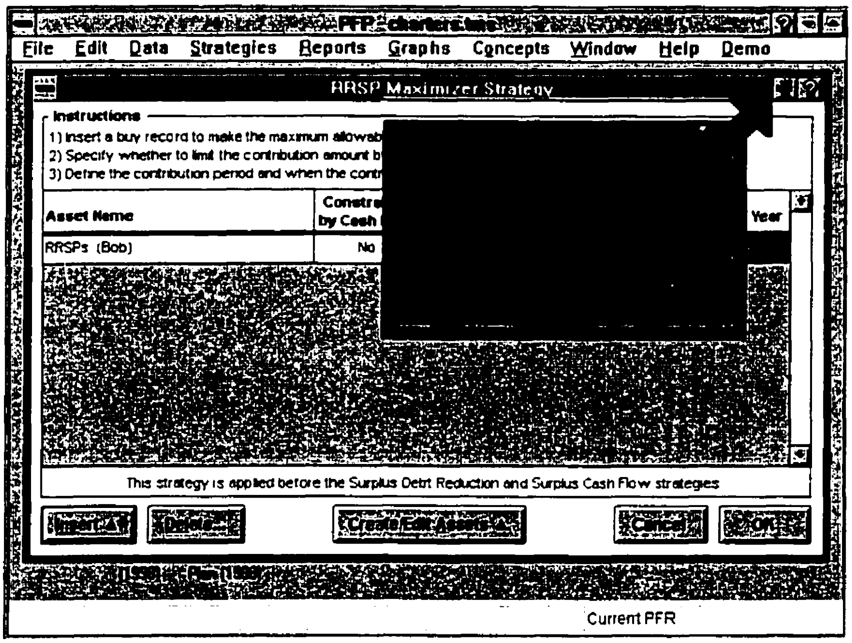


Figure 13.

Show the method of completing the strategy and applying the strategy to the active situation.

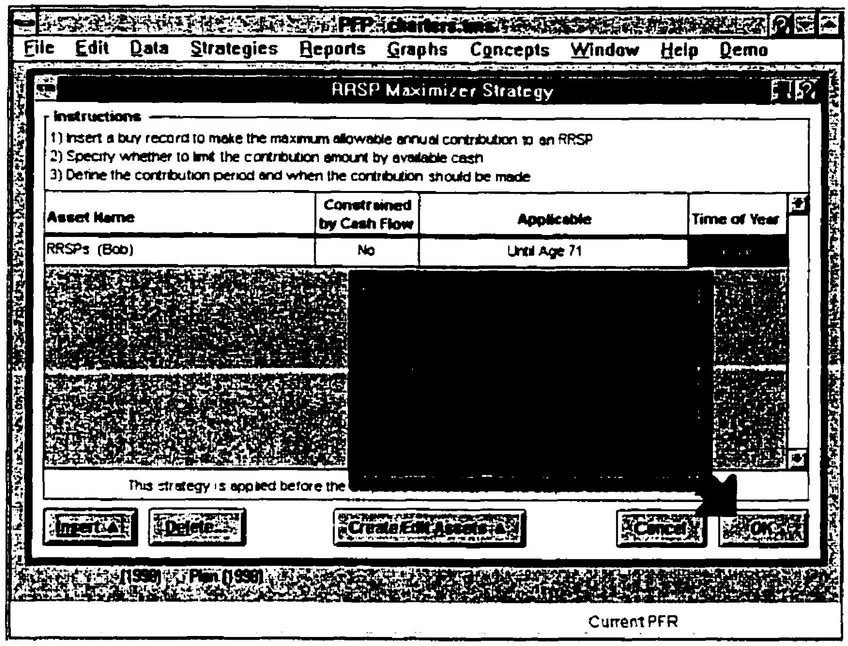


Figure 14.

All available demonstrations are listed in the Demo pull-down menu in the TIMS menubar.

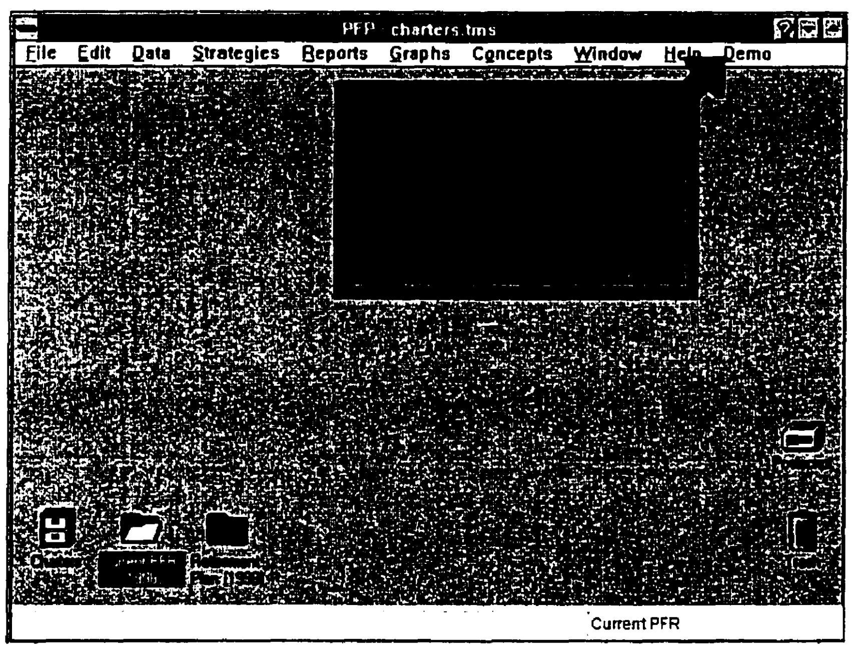
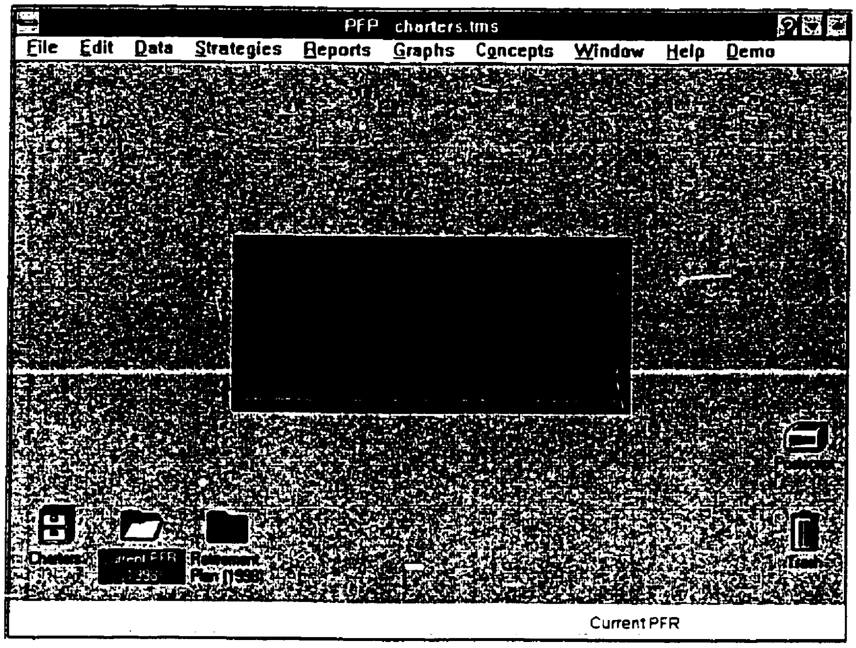


Figure 15.

A final caption indicates the completion of the strategy demonstration.



APPENDIX D

DEMOGRAPHIC INFORMATION

The first section of the Questionnaire for User Interaction Satisfaction (QUIS) gathered information about the subjects. It was intended to contribute information that would be useful in understanding the results of the questionnaire. A brief summary of the important information is presented below. Following the summary is a chart of the raw demographic information.

PART 1: Description of Subject

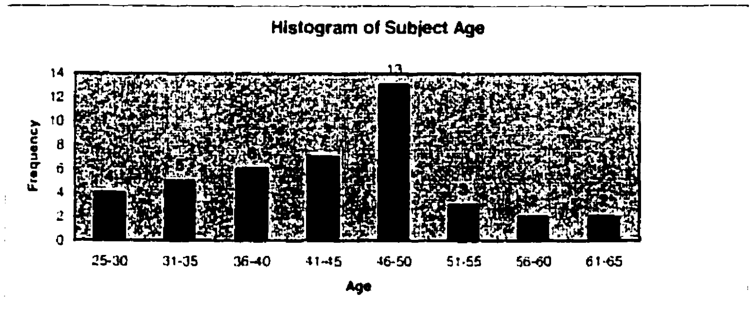


Figure D-1. **Age of Subjects.**

The average age of the subjects was 43.7 years. The youngest participant was 27 years old, the oldest was 64 years old.

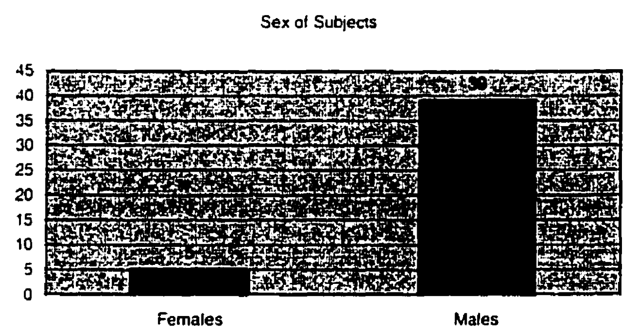


Figure D-2. **Sex of Subjects.**

89% of the subjects were male, 11% were female.

1.1 Highest Education Level achieved?

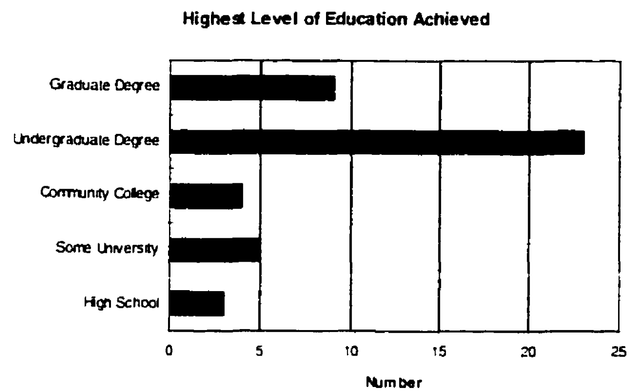


Figure D-3. **Highest level of education achieved by subjects.**

Education levels were high and fairly consistent with over 72% of the subjects attaining at least an undergraduate degree.

1.2 Check all applicable descriptions that correspond to your financial background.

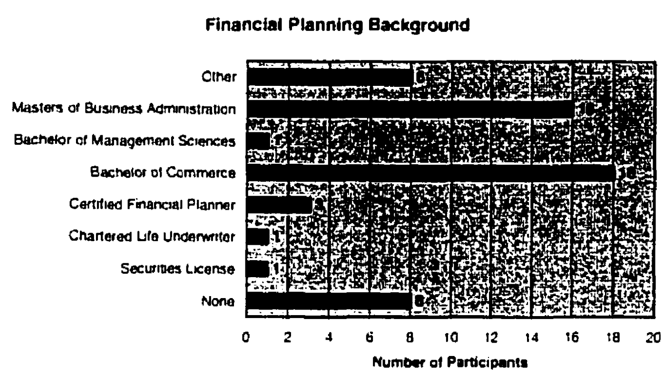


Figure D-4. **Subjects selected descriptions of their financial planning background.**

Only 18% of the participants had not reported any additional financial planning education.

1.3 Did you have any previous experience with the TIMS system?

A minimal amount of previous experience with the TIMS system was reported by 32% of the participants, while the majority (68%) reported no previous experience at all.

1.4 Did you feel this training session helped with your understanding of the TIMS system?

A rating of 1 corresponded to "not very helpful" while a rating of 9 corresponded to "very helpful". One participant did not answer this question. The average rating for this question was 7.3.

1.5 On the average, how much time do you expect to spend per week on this system?

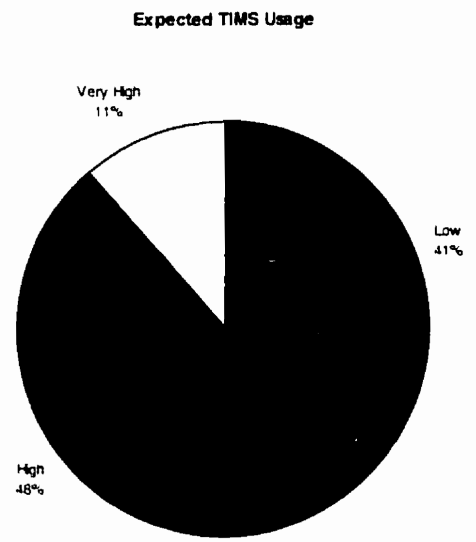


Figure D-5. Expected TIMS Usage.

After the completion of the training program, 41% of the participants expected to spend less than 4 hours per week with the TIMS system, 48% expected to spend between 4 and 10 hours per week, and 11% expected to spend more than 10 hours per week.

PART 2: Past Experience

2.1 How many years have you used a computer?

Respondents differed in their level of computer experience; 9% had no previous experience, 46% had used computer systems between 1 and 5 years, and 45% had used computers for more than 5 years. Because the participants were not normally using computer systems in their day-to-day work tasks, this was interpreted as an overall moderate level of experience.

2.2 Of the following devices, software, and systems, check those that you have personally used and are familiar with:

This impression that the participants in the study had an overall moderate level of experience was confirmed through the reported number of hardware devices and software systems that the individuals were familiar with and had used. The average number of devices that the participants reported having used and were familiar with was 6.66, and the number of software packages was 3.77. Only 9% of the subjects were judged to be brand-new computer users, which were not familiar with many hardware devices and had not used any word processing or financial planning software systems. More than 79% of the subjects had used and were familiar with word processors, and more than 77% had some experience with at least one financial planning software package.

Date	Group	Subject	Age	Sex	1.1 Education
23-May	A	1	51	M	Undergraduate Degree
23-May	A	2	30	M	Undergraduate degree
23-May	A	3	38	M	some University
23-May	A	4	50	M	Graduate degree
23-May	A	5	42	M	High School
23-May	A	6	28	M	Community College
23-May	A	7	48	M	Undergraduate degree
23-May	A	8	49	M	Graduate degree
23-May	A	9	34	M	Undergraduate degree
23-May	A	10	41	M	Graduate degree
23-May	A	11	33	M	Undergraduate degree
23-May	B	12	48	M	Community College
23-May	B	13	42	M	Undergraduate degree
23-May	B	14	50	F	Graduate degree
23-May	B	15	37	M	Some university
23-May	B	16	56	M	High School
23-May	B	17	46	M	Graduate degree
23-May	B	18	33	M	Undergraduate degree
23-May	B	19	48	M	Undergraduate degree
23-May	B	20	38	M	Undergraduate degree
23-May	B	21	37	F	Undergraduate degree
24-May	C	22	62	M	some University
24-May	C	23	33	M	Undergraduate degree
24-May	C	24	47	F	Undergraduate degree
24-May	C	25	56	M	some University
24-May	C	26	27	M	Undergraduate degree
24-May	C	27	50	M	Community College
24-May	C	28	42	M	Undergraduate degree
24-May	C	29	43	M	Undergraduate degree
24-May	C	30	41	M	Undergraduate degree
24-May	C	31	49	M	some University
24-May	C	32	34	M	Undergraduate degree
24-May	D	35	44	M	Community College
24-May	D	36	27	F	Undergraduate degree
24-May	D	37	49	M	High School
24-May	D	38	54	F	Undergraduate degree
24-May	D	39	64	M	Undergraduate degree
24-May	D	40		M	Graduate degree
24-May	D	42	36	M	Graduate degree
24-May	D	43	40	M	Undergraduate degree
24-May	D	44	52	M	Undergraduate degree
24-May	D	45	46	M	Undergraduate degree
24-May	D	46	53	M	Graduate degree
24-May	D	47	49	M	Graduate degree

1.2 Financial Bkgnd	1.3 Previous TIMS Experience	1.4 Training Helpful?
CFP	No	9
Securities License	No	8
CFP, Securities License, CIM	Yes	7
CFP	Yes	6
none	No	8
EET	Yes	9
CFP	No	7
CFP, Securities License, P.Eng	No	7
none	No	7
CFP	Yes	7
CFP, Securities License, BMaS	Yes	9
none	Yes	
CFP, B.Comm.	Yes	6
Securities License, Insurance License	No	9
Securities License	No	9
Securities License	No	7
CFP, BA, BEd	No	8
CPA	No	6
BComm	No	7
CSC	No	7
CFP, Securities License	Yes	8
Securities License	No	6
CFP	No	7
CFP	No	6
CFP	Yes	9
none	No	6
NONE	No	9
Securities License	No	5
P.Eng	No	7
CFP	No	8
Securities License	No	9
B.Comm, CFP, Securities License	Yes	7
none	Yes	8
CSC	No	7
CFP	Yes	7
Securities License	Yes	5
CLU, Securities License	No	9
Securities License	No	5
Portfolio Manager	Yes	5
Securities License	No	8
none	No	7
CFP	No	9
MBA	No	7
ACIB, Post Grad diploma in Management Studies (UK)	No	7

1.5 Avg Time on System Expected	2.1 Computer Experience
one to less than 4 hours	over 5 years
4 to less than 10 hours	over 5 years
4 to less than 10 hours	2 to 5 years
one to less than 4 hours	over 5 years
one to less than 4 hours	less than one year (none)
one to less than 4 hours	2 to 5 years
4 to less than 10 hours	one to less than 2 years
4 to less than 10 hours	over 5 years
one to less than 4 hours	over 5 years
one to less than 4 hours	over 5 years
one to less than 4 hours	2 to 5 years
one to less than 4 hours	over 5 years
4 to less than 10 hours	over 5 years
over 10 hours	over 5 years
4 to less than 10 hours	2 to 5 years
over 10 hours	over 5 years
one to less than 4 hours	2 to 5 years
4 to less than 10 hours	over 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	over 5 years
4 to less than 10 hours	less than one year
one to less than 4 hours	less than one year
over 10 hours	over 5 years
one to less than 4 hours	2 to 5 years
4 to less than 10 hours	over 5 years
over 10 hours	over 5 years
over 10 hours	over 5 years
less than one hour	2 to 5 years
one to less than 4 hours	2 to 5 years
4 to less than 10 hours	one to less than 2 years
4 to less than 10 hours	over 5 years
one to less than 4 hours	2 to 5 years
one to less than 4 hours	over 5 years
one to less than 4 hours	over 5 years
one to less than 4 hours	2 to 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	over 5 years
4 to less than 10 hours	2 to 5 years
4 to less than 10 hours	2 to 5 years
one to less than 4 hours	less than one year
4 to less than 10 hours	2 to 5 years

2.2 Familiar with the Following Devices and Software

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect, IG Options, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, network, MS Windows 3.1, MS Word, WordPerfect, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Word, Quicken, Wealth Creater, Destiny
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect, Quicken, IG Options
none
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Word, IG Options, Destiny
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, WordPerfect, Quicken, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, network, MS Windows 3.1, MS Word, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, network, MS Windows 3.1, MS Word, WordPerfect, Quicken, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Windows 95, MS Word, WordPerfect, PlanPlus, IG Options, Destiny, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, WordPerfect, Quicken, IG Options, Destiny, Maximizer
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Windows 95, MS Word, WordPerfect, MS Money, Quicken, PlanPlus, IG Options, Destiny, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, WordPerfect, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, WordPerfect, IG Options
personal computer, keyboard, mouse, printer, hard drive
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Windows 95, IG Options, electronic mail
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect, IG Options, Destiny
personal computer, keyboard, mouse, printer, color monitor, floppy drive, Destiny
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect, Quicken, PlanPlus, IG Options, Destiny
personal computer, keyboard, mouse, printer, floppy drive, MS Windows 95, MS Word
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, IG Options
personal computer, keyboard, mouse, printer, IG Options
MS Windows 3.1
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, network, MS Windows 3.1, MS Windows 95, MS Word, WordPerfect, IG Options, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, floppy drive, hard drive, WordPerfect, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, OS/2, WordPerfect, AmiPro/WordPro, IG Options, electronic mail
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, WordPerfect, Destiny
personal computer, keyboard, mouse, printer, color monitor, hard drive, MS Windows 3.1, IG Options
personal computer, keyboard, mouse, printer, MS Windows 3.1, MS Word, electronic mail
personal computer, keyboard, mouser, printer, color monitor, floppy dirve, hard drive, compact disk (CD) drive, MS Windows 3.1, OS/2, MS Word, Quicken, IG Options, electronic mail, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, WordPerfect, IG Options
personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact (CD) drive, Internet/WWW
personal computer, keyboard, mouse, printer, color monitor, hard drive, MS Windows 3.1, MS Word, Quicken, Destiny

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, MS Money

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Word, IG Options

personal computer, keyboard, mouse, printer, hard drive, MS Windows 3.1, MS Word, WordPerfect, IG Options, Destiny

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, MS Word, WordPerfect

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, compact disk (CD) drive, MS Windows 3.1, MS Word, WordPerfect, MS Money, Quicken, Destiny, Internet/WWW

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, WordPerfect, IG Options

personal computer, keyboard, mouse, printer, floppy drive, MS Windows 3.1, MS Word, IG Options, Destiny

personal computer, keyboard, mouse, printer, color monitor, WordPerfect

personal computer, keyboard, mouse, printer, color monitor, floppy drive, hard drive, MS Windows 3.1, AmiPro/WordPro, Wealth Creater, Destiny

APPENDIX E

QUIS© QUESTIONNAIRE

Tax and Investment Management Strategizer™

Questionnaire for User Interaction Satisfaction©

The Confidential Data for:

Identification Number:* ____________________

Date: ____________________

* This data is confidential and is the property of Emerging Information Systems Inc. (EISI). Summarization and analysis of the findings may be released but will not be associated with any individuals.

PART 1: Description of Subject

Age: ______ Sex: __ male __ female

1.1 Highest Education Level achieved?

__ did not complete High School
__ High School
__ some University
__ Community College
__ Undergraduate degree
__ Graduate degree

1.2 Check all applicable descriptions that correspond to your financial background.

__ none
__ CGA
__ CMA
__ other (specify) ______________
__ CA
__ CPA
__ CLU
__ CFP
__ RFP
__ Securities License

1.3 Did you have any previous experience with the TIMS system?

__ Yes __ No

1.4 Did you feel this training session helped with your understanding of the TIMS system?

not very helpful very helpful
1 2 3 4 5 6 7 8 9

1.5 On the average, how much time do you expect to spend per week on this system?

__ less than one hour
__ one to less than 4 hours
__ 4 to less than 10 hours
__ over 10 hours

PART 2: Past Experience

2.1 How many years have you used a computer?

__ less than one year
__ one to less than 2 years
__ 2 to 5 years
__ over 5 years

2.2 Of the following devices, software, and systems, check those that you have personally used and are familiar with:

	personal computer		compact disk (CD) drive		MS Money		network
	keyboard		MS Windows 3.1		Quicken		electronic mail
	mouse		MS Windows 95		PlanPlus		Internet / WWW
	printer		OS/2		MS Word		
	color monitor		WordPerfect		Wealth Creater		
	floppy drive		AmiPro/WordPro		Destiny		
	hard drive				IG Options		

PART 3: Overall User Reactions

Please circle the numbers which most appropriately reflect your impressions about using this computer system.

3.1	Overall reactions to the system:	terrible 1 2 3 4 5 6 7 8 9 wonderful
3.2		frustrating 1 2 3 4 5 6 7 8 9 satisfying
3.3		dull 1 2 3 4 5 6 7 8 9 stimulating
3.4		difficult 1 2 3 4 5 6 7 8 9 easy
3.5		ineffective 1 2 3 4 5 6 7 8 9 powerful
3.6		rigid 1 2 3 4 5 6 7 8 9 flexible

Please leave any general comments about the system here:

PART 4: Screen

Not Applicable = NA.

4.1	Characters on the computer screen	hard to read — easy to read 1 2 3 4 5 6 7 8 9	NA
4.2	Was the highlighting on the screen helpful?	not at all — very much 1 2 3 4 5 6 7 8 9	NA
4.3	Were the screen layouts helpful?	never — always 1 2 3 4 5 6 7 8 9	NA
a)	Amount of information that can displayed on screen	inadequate — adequate 1 2 3 4 5 6 7 8 9	NA
b)	Arrangement of information on screen	illogical — logical 1 2 3 4 5 6 7 8 9	NA
4.4	Sequence of screens	confusing — clear 1 2 3 4 5 6 7 8 9	NA

Please leave any comments that you have about the screen here:

PART 5: Terminology and System Information

5.1 Use of terms throughout system	inconsistent								consistent		
	1	2	3	4	5	6	7	8	9	NA
5.2 Does the terminology relate well to the work you are doing	unrelated								well related		
	1	2	3	4	5	6	7	8	9	NA
5.3 Messages which appear on screen	inconsistent								consistent		
	1	2	3	4	5	6	7	8	9	NA
5.4 Messages which appear on screen	confusing								clear		
	1	2	3	4	5	6	7	8	9	NA
5.5 Does the computer keep you informed about what it is doing?	never								always		
	1	2	3	4	5	6	7	8	9	NA
a) Performing an operation leads to a predictable result	never								always		
	1	2	3	4	5	6	7	8	9	NA
b) User can control amount of feedback	never								always		
	1	2	3	4	5	6	7	8	9	NA
5.6 Error messages	unhelpful								helpful		
	1	2	3	4	5	6	7	8	9	NA

Please leave any comments that you have about terminology and system information here:

PART 6: Learning

6.1 Learning to operate the system — difficult 1 2 3 4 5 6 7 8 9 easy — NA

6.2 Exploration of features by trial and error — discouraging 1 2 3 4 5 6 7 8 9 encouraging — NA

6.3 Remembering names and use of commands — difficult 1 2 3 4 5 6 7 8 9 easy — NA

6.4 Can tasks be performed in a straight-forward manner? — never 1 2 3 4 5 6 7 8 9 always — NA

6.5 Help messages on the screen — confusing 1 2 3 4 5 6 7 8 9 clear — NA

6.6 Training materials — confusing 1 2 3 4 5 6 7 8 9 clear — NA

6.7 Animated demonstrations — not helpful 1 2 3 4 5 6 7 8 9 helpful — NA

Please leave any comments that you have about learning the system here:

____

____

____

____

____

____

____

PART 7: System Capabilities

7.1 System speed	too slow								fast enough			
		1	2	3	4	5	6	7	8	9		NA
7.2 How reliable is the system?	unreliable								reliable			
		1	2	3	4	5	6	7	8	9		NA
7.3 System functionality	not adequate								adequate			
		1	2	3	4	5	6	7	8	9		NA
7.4 Correcting your mistakes	difficult								easy			
		1	2	3	4	5	6	7	8	9		NA
7.5 Are the needs of both experienced and inexperienced users taken into consideration?	never								always			
		1	2	3	4	5	6	7	8	9		NA
a) Novices can accomplish tasks knowing only a few commands	with difficulty								easily			
		1	2	3	4	5	6	7	8	9		NA
b) Experts can use features/shortcuts	with difficulty								easily			
		1	2	3	4	5	6	7	8	9		NA

Please leave any comments that you have about the system capabilities here:

Appendix F

Data Worksheet

Tax and Investment Management Strategizer™

Data Entry Questionnaire

The Confidential Personal
and Financial Data for:

Client Name: CHARTERS

Prepared by: ____________ / ____________
Planner Designation

Before You Begin...

This *Personal and Financial Data Questionnaire* is designed to help you gather all the required information for your customized financial plan. The questionnaire's easy-to-follow format will allow you to enter your required personal data and financial details. These items are necessary so we can create a complete and thorough picture of your *current* and *future* financial situation.

Please have the following items on hand to help you complete this questionnaire:

1] Previous Year's Tax Returns.
2] Monthly Budgets.
3] Future Capital Expenditures.
4] Complete Financial Debt Information.
5] Lifestyle Asset and Balance Sheet.
6] Employment Benefits, Group Benefits, Pension Benefits, etc.
7] Financial Goals, Lifestyle as well as Retirement.
8] Capital Growth Investment Assets; purchase dates, expected growth rates, maturity dates, etc.
9] RRSPs, DPSPs; purchase dates, expected growth rates, maturity dates, etc.
10] Personal Life Insurance and/or Disability Policies.
11] Other Pertinent Information.

Keep In Mind...

The more information you provide, the more realistic your financial plan will be. If you are unsure of an exact value for any piece of information, please give it your best estimate.

PERSONAL INFORMATION SHEET

Family Head Information CHARTERS

CLIENT

Name: BOB

Date of Birth: AUG. 5, 1965
Month Day Year

Street Address: 123 MAIN STREET

City: WINNIPEG Province: MANITOBA

Country: CANADA Postal Code: R3T 2N2

Home Phone: # (204) 444-5555

Fax Phone: # —

Business Phone: # (204) 488-1199

SPOUSE

Name: MARION

Date of Birth: AUG. 11, 1967
Month Day Year

Street Address:

City: Province:

Country: Postal Code:

Home Phone: #

Fax Phone: #

Business Phone: #

Children / Dependants NONE

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

Child Name:

Date of Birth:
Month Day Year

RETIREMENT GOALS AND PAST FINANCIAL DATA

Retirement / Other Objectives * WE WOULD BOTH LIKE TO RETIRE WHEN MARION TURNS 55!

	CLIENT		SPOUSE
Retirement Date: (Month Day Year)	(WHEN MARION TURNS 55)	Retirement Date: (Month Day Year)	AGE 55
Life Expectancy: (Age / Year)	90	Life Expectancy: (Age / Year)	90

RRSP Information

	CLIENT		SPOUSE
Previous Year's Earned Income:	$ 36,000.	Previous Year's Earned Income:	$ 33,150.
Previous Year's Pension Adjustment:	$ 1,418.	Previous Year's Pension Adjustment:	$ 0
RRSP Overcontribution Balance:	$ 0	RRSP Overcontribution Balance:	$ 0
RRSP Deduction Carryforward:	$ 13,269.	RRSP Deduction Carryforward:	$ 11,406.
Homebuyer's Loan Balance:	$ —	Homebuyer's Loan Balance:	$ —
Years Left to Repay:	—	Years Left to Repay:	—

Defaults

Inflation Rate that will be used in all financial projections: % 3.0

INCOME SOURCES

CLIENT	SPOUSE

Employment

Annual Salary: $ 36,000. | Annual Salary: $ 33,150

Is this salary indexed to inflation? Yes ☑ No ☐ | Is this salary indexed to inflation? Yes ☑ No ☐

Retirement

Canada Pension Plan and Old Age Security

Estimated CPP Eligibility: % 100 | Estimated CPP Eligibility: % 100

At what age will these benefits begin? 65 | At what age will these benefits begin? 65

Do you expect to split these benefits with your spouse? Yes ☑ No ☐

Estimated OAS Eligibility: % 100 | Estimated OAS Eligibility: % 100

Employer Defined Benefit Pension Plans DO NOT HAVE ANY AT THIS POINT.

Expected Annual Pension: $ ______ | Expected Annual Pension: $ ______

What age will these benefits begin? ______ | What age will these benefits begin? ______

Is pension indexed to inflation? Yes ☐ No ☐ | Is pension indexed to inflation? Yes ☐ No ☐

Is pension integrated? With CPP ☐ With OAS ☐ | Is pension integrated? With CPP ☐ With OAS ☐

Retiring Allowance $ ______ | Retiring Allowance $ ______

Will it be transferred to an RRSP? Yes ☐ No ☐ | Will it be transferred to an RRSP? Yes ☐ No ☐

Other Income

Life Insurance

Death Benefit (Face Amount): $ 125,000. | Death Benefit (Face Amount): $ 125,000.

What age will this coverage expire? 65 | What age will this coverage expire? 65

Miscellaneous *(Use the back of this sheet to enter additional details)*

CLIENT		SPOUSE	
Employment Bonus:	$	Employment Bonus:	$
Self-Employment Earned:	$	Self-Employment Earned:	$
Self Employed Commission:	$	Self Employed Commission:	$
Professional Fees	$	Professional Fees	$
Tax Free Income	$	Tax Free Income	$
Other	$	Other	$

EXPENSES

	CLIENT	SPOUSE

Non-Tax Related

Lifestyle *(If you share your common lifestyle expenses, just enter the total for one person)*

CLIENT	SPOUSE
Annual Lifestyle: $ 25,000.	Annual Lifestyle: $
By what percentage will this expense increase/decrease by when you enter retirement? % 25	By what percentage will this expense increase/decrease by when you enter retirement? %

*(The following section is optional and can be used if you wish to break down the **Annual Lifestyle** total above)*

CLIENT		SPOUSE	
Housing: $	Food: $	Housing: $	Food: $
Transportation: $	Personal: $	Transportation: $	Personal: $
Entertainment: $		Entertainment: $	

Semi-Regular Expenses *(Optional)*

CLIENT	SPOUSE
I plan to upgrade my *automobile* every: 5 years.	I plan to upgrade my *automobile* every: ____ years.
I usually spend approx. $ 15,000. for the upgrade.	I usually spend approx. $ ____ for the upgrade.
I will wait: 5 years before my next car upgrade.	I will wait: ____ years before my next car upgrade.
I plan to take a *vacation* every: 1 years.	I plan to take a *vacation* every: ____ years.
I usually spend approx. $ 1,500. for the vacation.	I usually spend approx. $ ____ for the vacation.
I will wait: 1 years before taking my next vacation.	I will wait: ____ years before taking my next vacation.

Tax Related

CLIENT		SPOUSE	
Charitable Contributions:	$ 1,000.	Charitable Contributions:	$
Medical / Dental:	$ 230.	Medical / Dental:	$
Child Care:	$	Child Care:	$
Carrying Charges:	$	Carrying Charges:	$
Professional / Union Dues:	$	Professional / Union Dues:	$
RPP - Defined Benefit:	$	RPP - Defined Benefit:	$
Other:	$	Other:	$

Detailed Lifestyle Expenses

Name: TRIP TO HAWAII Amount: $ 7,000. every/on JAN. 1/99

ASSETS

Your House and Mortgage

House *(Principal residence only - Real Estate assets will be defined later)*

Purchase Amount: $ 165,000. Purchase Date: DEC. 31/95

Who is listed as the house's primary owner? Client ☐ Spouse ☐ Joint Ownership ☑

What is its current market value? $ 165,000. What is its expected growth rate? % 2

Mortgage *(For the Above Residence)*

Original Principal: $ 120,000. Mortgage Start Date: DEC. 31/95 (Month Day Year)

Who is listed as the mortgage's primary owner? Client ☐ Spouse ☐ Joint Ownership ☑

Mortgage's interest rate: % 10 Mortgage is amortized for: 25 years.

I/we make regular principal and interest payments on a MONTHLY basis. (e.g. Weekly, Monthly, Yearly, etc.)

As of: DEC. 31/95 (Month Day Year) there is still: $ 120,000 outstanding on the mortgage.

Is the mortgage insured? Yes ☑ No ☐

Non-Registered Assets

(Enter each asset type's ***aggregate*** *value as of the start of the year. Break down each asset's expected return rate based on its return component - e.g. If a Mutual Fund asset is earning 2%* ***Interest****, 2%* ***Dividends****, 4%* ***Capital Gains*** *and 4%* ***Deferred Capital Gains****, you would enter* ***2-I, 2-D, 3-CG, 4-DCG****).*

CLIENT

GICs:	$		Return: %
Bonds:	$	ACB: $	Return: %
Mut. Funds:	$ 7,800.	ACB: $ 6,000.	Return: 10-CG %
Stocks:	$	ACB: $	Return: %
Deposits:	$ 5,000.		Return: 3-I %

T-Bills (F.V.): $	M. Value: $	ACB: $	Maturity:
Real Estate: $	M. Value: $	ACB: $	P. Date:
$	M. Value: $	ACB: $	P. Date:
Lifestyle: $	Return: %	ACB: $	S. Date:

Non-Registered Assets (con't)

*(Enter each asset type's **aggregate** value as of the start of the year. Break down each asset's expected return rate based on its return component - e.g. If a Mutual Fund asset is earning 2% **Interest**, 2% **Dividends**, 4% **Capital Gains** and 4% **Deferred Capital Gains**, you would enter **2-I, 2-D, 3-CG, 4-DCG**).*

SPOUSE

GICs:	$ ______		Return: ______ %
Bonds:	$ 5,000.	ACB: $ 5,000.	Return: 7- I %
Mut. Funds:	$ 3,000.	ACB: $ 2,800.	Return: 5- I %
Stocks:	$ ______	ACB: $ ______	Return: ______ %
Deposits:	$ ______		Return: ______ %

T-Bills (F.V.):	$ ______	M. Value: $ ______	ACB: $ ______	Maturity: ______
Real Estate:	$ ______	M. Value: $ ______	ACB: $ ______	P. Date: ______
	$ ______	M. Value: $ ______	ACB: $ ______	P. Date: ______
Lifestyle:	$ ______	Return: ______ %	ACB: $ ______	S. Date: ______
Other:	$ ______	Return: ______ %	$ ______	Return: ______ %

JOINT

GICs:	$ ______		Return: ______ %
Bonds:	$ ______	ACB: $ ______	Return: ______ %
Mut. Funds:	$ ______	ACB: $ ______	Return: ______ %
Stocks:	$ ______	ACB: $ ______	Return: ______ %
Deposits:	$ ______		Return: ______ %

T-Bills (F.V.):	$ ______	M. Value: $ ______	ACB: $ ______	Maturity: ______
Real Estate:	$ ______	M. Value: $ ______	ACB: $ ______	P. Date: ______
	$ ______	M. Value: $ ______	ACB: $ ______	P. Date: ______
Lifestyle:	$ ______	Return: ______ %	ACB: $ ______	S. Date: ______
Other:	$ ______	Return: ______ %	$ ______	Return: ______ %

Registered Assets

*(Enter each asset type's **aggregate** value as of the start of the year. To be complete, enter return rates for the eventual transfer or conversion of an asset. E.g. Enter an expected RRIF return rate.)*

CLIENT				SPOUSE			
RRSPs:	$ 3,000.	Return:	8 %	RRSPs:	$ 2,000.	Return:	8 %
Spousal:	$	Return:	%	Spousal:	$ 1,000.	Return:	8 %
LIRAs:	$	Return:	%	LIRAs:	$	Return:	%
Money Pur. RPPs:	$	Return:	%	Money Pur. RPPs:	$	Return:	%
RRIFs:	$ 0	Return:	8 %	RRIFs:	$ 0	Return:	8 %
LIFs:	$	Return:	%	LIFs:	$	Return:	%

Fixed Term Annuities *(Enter additional Fixed Term annuities on a separate sheet)*

Annuity #1

The annuity pays: $ _______ Annually ☐ Monthly ☐ | The annuity pays: $ _______ Annually ☐ Monthly ☐

The annuity ceases payments on: _______ (Month Day Year). | The annuity ceases payments on: _______ (Month Day Year).

Is this a registered annuity? Yes ☐ No ☐ | Is this a registered annuity? Yes ☐ No ☐

Annuity #2

The annuity pays: $ _______ Annually ☐ Monthly ☐ | The annuity pays: $ _______ Annually ☐ Monthly ☐

The annuity ceases payments on: _______ (Month Day Year). | The annuity ceases payments on: _______ (Month Day Year).

Is this a registered annuity? Yes ☐ No ☐ | Is this a registered annuity? Yes ☐ No ☐

Life Annuities *(Enter additional Life annuities on a separate sheet)*

Annuity #1

The annuity pays: $ _______ Annually ☐ Monthly ☐ | The annuity pays: $ _______ Annually ☐ Monthly ☐

The payments are guaranteed for: _______ years. | The payments are guaranteed for: _______ years.

Is this a registered annuity? Yes ☐ No ☐ | Is this a registered annuity? Yes ☐ No ☐

Annuity #2

The annuity pays: $ _______ Annually ☐ Monthly ☐ | The annuity pays: $ _______ Annually ☐ Monthly ☐

The payments are guaranteed for: _______ years. | The payments are guaranteed for: _______ years.

Is this a registered annuity? Yes ☐ No ☐ | Is this a registered annuity? Yes ☐ No ☐

Is this a prescribed annuity? Yes ☐ No ☐ | Is this a prescribed annuity? Yes ☐ No ☐

LIABILITIES

Personal Loans

(Enter additional loans on a separate sheet)

Loan #1

Original Principal: $ 11,000. Who owns the loan? Client ☑ BOB Spouse ☐ Joint Ownership ☐

Loan Start Date: DEC. 31/95 (Month Day Year) Payment Type: PRINCIPAL & INTEREST (e.g. Interest Only, P.I.T., etc.)

Interest Rate: % 10.00 Payment Frequency: MONTHLY (e.g. Monthly, Weekly, etc.)

This loan is amortized for 5 years

There is: $ 11,000. still outstanding on this loan as of: DEC. 31/95 (Month Day Year).

Is this loan insured? Yes ☑ No ☐

Other information: ____________________

____

Loan #2

Original Principal: $ ______ Who owns the loan? Client ☐ Spouse ☐ Joint Ownership ☐

Loan Start Date: ______ (Month Day Year) Payment Type: ______ (e.g. Interest Only, P.I.T., etc.)

Interest Rate: % ______ Payment Frequency: ______ (e.g. Monthly, Weekly, etc.)

This loan is amortized for ______ years

There is: $ ______ still outstanding on this loan as of: ______ (Month Day Year).

Is this loan insured? Yes ☐ No ☐

Other information: ____________________

____

CURRENT PLANNING STRATEGIES

Use this section to tell us about any planning strategies that you are currently applying (e.g. Regular Savings to an investment, additional payments towards a loan's principal, etc). Use the back of this sheet to enter additional plans.

Regular Savings Plans *(Savings made on a regular, periodic basis)*

Asset Name	Amount $	Frequency (e.g. monthly, weekly)	Indexed to Inflation?	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)
BOND PORTFOLIO	$150.	MONTHLY	YES	WHILE WORKING
(MARION)				

Lump Sum Savings Plans *(Savings made in one or more lump sums)*

Asset Name	Amount $	Indexed to Inflation?	When is this Transaction Applicable? (Upon Retirement, Upon Disability, Upon Death, Other - *e.g. Jan. 15, 1998*)

RRSP Maximizer Plans *(Maximum allowable contributions made to RRSPs each year, whenever possible.)*

Asset Name	Constrained by Cash Flow?	Time of Year	When is this Transaction Applicable? (While Working, Until Age 71, Other - *e.g. Jan 1, 1995 to Dec 31, 2025*)

Surplus Cash Plans *(Savings made using surplus cash when available.)*

Asset Name	Percentage of Surplus Cash to be Used in Purchase %	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)
MUTUAL FUNDS (BOB)	100%	WHILE WORKING & RETIRED
MUTUAL FUNDS (MARION)	100%	WHILE WORKING & RETIRED

Regular Debt Reduction Plans *(Any payments made directly toward the principal of an existing loan. These are over and above the required payments that are automatically defined for the liability.)*

Liability Name	Amount $	Frequency (e.g. monthly, weekly)	Indexed to Inflation?	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)

Lump Sum Debt Reduction *(Payments made in one or more lump sums toward the principal of an existing loan. These are over and above the required payments that are automatically defined for the liability.)*

Liability Name	Amount $	Indexed to Inflation?	When is this Transaction Applicable? (Upon Retirement, Upon Disability, Upon Death, Other - *e.g. Jan. 15, 1998*)

Surplus Cash Debt Reduction

(Payments made toward the principal of an existing loan using surplus cash. These are over and above the required payments that are automatically defined for the liability.)

Liability Name	Maximum Dollar Amount that can be Paid Down $	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)

Regular Asset Redemption Plans

(Any redemptions from an asset made on a periodic basis.)

Asset Name	Amount $	Frequency (e.g. monthly, weekly)	Indexed to Inflation?	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)

Lump Sum Asset Redemption

(Redemptions from an asset in one or more lump sums)

Asset Name	Amount $	Indexed to Inflation?	When is this Transaction Applicable? (Upon Retirement, Upon Disability, Upon Death, Other - *e.g. Jan. 15, 1998*)
BOND PORTFOLIO	ALL OF IT!	NO	JAN. 15/99
(MARION)			(TRIP TO HAWAII)

Complete Regular Asset Redemption *(Redemptions from an asset that will completely use it up at the end of a defined period)*

Asset Name	Frequency of Sells (e.g. Annually, Monthly,)	Indexed to Inflation?	When is this Transaction Applicable? (While I'm Working, While I'm Retired, Both, Other - *e.g. Jan. 1990 - Dec 2025*)

Asset Transfers *(The transfer of funds from one asset to another)*

Source Asset	Destination Asset	Amount $	When is this Transaction Applicable? (Upon Retirement, Upon Disability, Upon Death, Other - *e.g. Jan. 15, 1998*)

Leverage Investments *(Any loans taken out for the purposes of investing)*

Liability Name	Asset Name	Amount $	Transaction Date	Units to Buy (*Original* if asset is new or *Additional* if asset already exists)

OTHER INFORMATION

Use this page to enter any other information that you feel would be relevant to your financial plan.

- WE WOULD LIKE TO MAINTAIN OUR PRESENT STANDARD OF LIVING AND RETIRE EARLY.
- WE DO NOT NEED AN ENORMOUS ESTATE SINCE WE DO NOT (CURRENTLY) HAVE ANY CHILDREN.
- WE ARE WILLING TO BE FLEXIBLE AND TAKE YOUR ADVICE — DO WHAT YOU CAN!

Appendix G

Worksheet Instructions

The Experiment

Thank you for participating in this study. All responses and information gathered are not associated with specific individuals and will be used only for the purposes of this experiment. If you have any questions please ask them now, once the experiment starts questions cannot be answered. Feel free to use the extensive on-line help system (press F1) which is available to you throughout the TIMS program.

We suggest that you use the information in the TIMS Data Entry Questionnaire (we will refer to this as a *worksheet*) to strive for as accurate a plan as possible, given a reasonable time frame. You may have as much time as necessary to complete the worksheet. Read through the worksheet quickly before starting to get a sense of this particular family's situation.

Following completion of the worksheet you will be asked to complete a questionnaire about the TIMS system.

Note:
Your version of TIMS has been modified slightly to be personalized to you. You may notice some changes in the system since you completed the Quick Start manual in your Introductory TIMS training program. The system functions in the program remain the same. You may control the changes by adjusting the values in the *Preferences* dialog under the **Edit** menu.

Instructions

Your clients are arriving later today for their second appointment where you will present them with a completed financial plan created with the Tax and Investment Management Strategizer (TIMS) program. On their first visit you discussed their current financial situation and asked them to take home and complete the TIMS Data Entry Questionnaire and return it to you. Attached is their completed worksheet containing all of the necessary information needed to create the "Personal Financial Plan" using the *Retirement Plan* document package included in your version of the TIMS program.

Note: You may refer to the Quick Start manual or use the on-line help for assistance.

Please perform the following steps in order:

1) Create and save a new family based on their personal data from the worksheet.
2) Create an initial situation for this family and name it *Current PFR*.
3) Enter the data and current planning strategies for this family from the worksheet.
4) Duplicate this situation and label it *Retirement Plan*.
5) Modify the *Retirement Plan* situation given the following objectives:
    - Your first task is to balance the client's cashflow for a realistic projection. In the TIMS system a straight line in the upper window of the **Plan Window** (the surplus and deficit graph) indicates a balanced cash flow situation.
    - The client's overall goals are to retire early and have a comfortable retirement. Since the clients are young with good future earning potential they would like to retire together when she turns 55 years old. They are willing to follow any of your suggestions that would help them accumulate as much net worth as possible.
    - Find an outstanding problem(s) with the situation using the **Planning Assistant**.
    - Solve the problem(s) by following the suggestions in the **Planning Assistant**.
6) Generate a copy of the family's Financial Plan by using the *Retirement Plan* document package and save it onto a floppy disk.
7) Close the TIMS program.
8) Notify the tester.

Appendix H

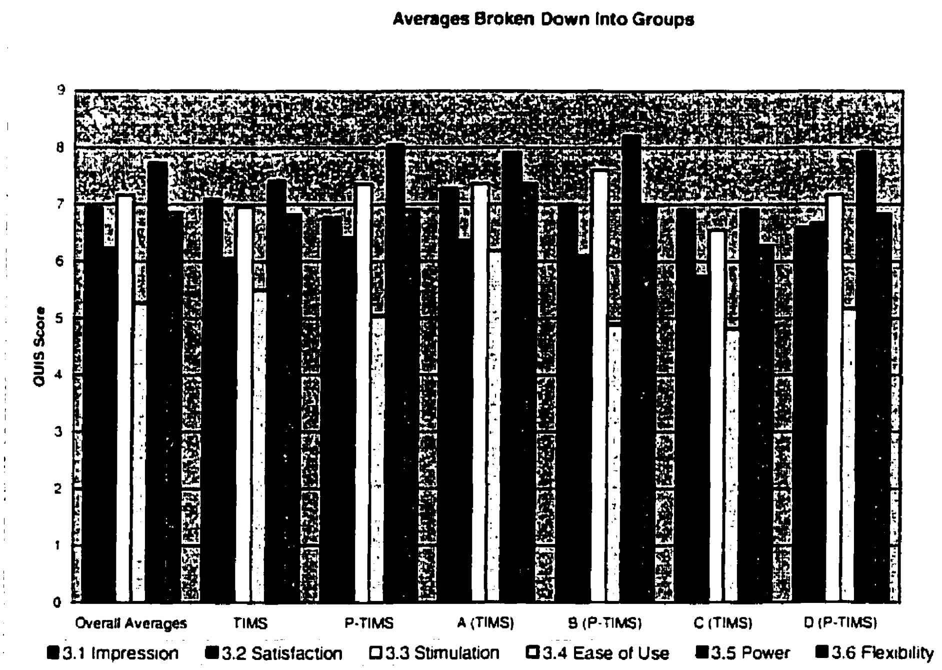
Data Analysis

Six Attributes of User Satisfaction Measured by the QUIS Questionnaire

3.1 Impression
3.2 Satisfaction
3.3 Stimulation
3.4 Ease of Use
3.5 Power
3.6 Flexibility

Averages for Each Individual Group

	3.1	3.2	3.3	3.4	3.5	3.6		
A	7.3	6.4	7.4	6.2	7.9	7.4	Group A Average	7.1
B	7.0	6.1	7.6	4.9	8.2	7.0	Group B Average	6.8
C	6.9	5.7	6.5	4.8	6.9	6.3	Group C Average	6.2
D	6.6	6.7	7.2	5.2	7.9	6.8	Group D Average	6.7



TIMS Users Rating (Scale of 1-9)

3.1T	3.2T	3.3T	3.4T	3.5T	3.6T	Group	Average
9	9	9	9	9	9	A	9.0
8	5	7	5	7	7	A	6.5
8	7	8	7	8	8	A	7.7
6	4	6	6	8	5	A	5.8
6	3	6	2	8	7	A	5.3
8	7	9	6	8	6	A	7.3
5	5	7	4	6	6	A	5.5

8	8	8	8	8	8	A	8.0
6	5	5	5	8	8	A	6.2
7	8	7	7	8	8	A	7.5
9	9	9	9	9	9	A	9.0
9	8	8	5	8	6	C	7.3
7	5	6	3	8	6	C	5.8
7	5	7	7	6	6	C	6.3
9	7	7	3	7	9	C	7.0
7	6	6	6	6	6	C	6.2
7	6	5	7	9	8	C	7.0
7	4	9	6	7	4	C	6.2
6	6	6	3	7	7	C	5.8
6	5	7	2	7	6	C	5.5
5	5	5	5	5	5	C	5.0
6	6	6	6	6	6	C	6.0

P-TIMS Users Ratings (Scale of 1-9)

3.1P	**3.2P**	**3.3P**	**3.4P**	**3.5P**	**3.6P**	Group	Average
5	6	6	5	6	6	B	5.7
7	4	7	6	8	7	B	6.5
8	6	8	6	9	9	B	7.7
7	5	9	3	9	5	B	6.3
9	9	9	9	9	9	B	9.0
8	3	8	3	9	6	B	6.2
6	6	7	4	8	6	B	6.2
7	7	8	4	8	8	B	7.0
6	7	7	5	8	7	B	6.7
7	8	7	4	8	7	B	6.8
8	7	9	3	8	7	D	7.0
5	7	7	5	7	6	D	6.2
6	5	6	6	6	6	D	5.8
6	6	6	5	9	8	D	6.7
5	6	8	5	9	6	D	6.5
5	6	5	6	7	5	D	5.7
8	7	7	4	8	7	D	6.8
7	7	7	3	8	7	D	6.5
7	7	8	7	9	8	D	7.7
9	9	9	9	9	9	D	9.0
7	7	7	3	7	7	D	6.3
6	6	7	6	8	6	D	6.5

Comparisons of TIMS Groups vs. P-TIMS Groups for each individual factor.

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.1T	22	156	7.09091	1.61039
3.1P	22	149	6.77273	1.51732

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	1.11364	1	1.11364	0.71211	0.40352	4.07266
Within Groups	65.6818	42	1.56385			
Total	66.7955	43				

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.2T	22	133	6.04545	2.71212
3.2P	22	141	6.40909	1.96753

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	1.45455	1	1.45455	0.62165	0.43486	4.07266
Within Groups	98.2727	42	2.33983			
Total	99.7273	43				

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.3T	22	153	6.95455	1.75974
3.3P	22	162	7.36364	1.19481

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	1.84091	1	1.84091	1.24615	0.27063	4.07266
Within Groups	62.0455	42	1.47727			
Total	63.8864	43				

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.4T	22	121	5.5	4.16667
3.4P	22	111	5.04545	3.09307

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	2.27273	1	2.27273	0.62612	0.43323	4.07266
Within Groups	152.455	42	3.62987			
Total	154.727	43				

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.5T	22	163	7.40909	1.20563
3.5P	22	177	8.04545	0.9026

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	**P-value**	*F crit*
Between Groups	4.45455	1	4.45455	4.22587	**0.04606**	4.07266
Within Groups	44.2727	42	1.05411			
Total	48.7273	43				

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
3.6T	22	150	6.81818	1.96537
3.6P	22	152	6.90909	1.41991

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.09091	1	0.09091	0.05371	0.81786	4.07266
Within Groups	71.0909	42	1.69264			
Total	71.1818	43				

Data validation check for independence of attributes.

Correlations for Overall Data

	3.1	3.2	3.3	3.4	3.5	3.6
3.1	1.00					
3.2	0.51	1.00				
3.3	0.65	0.44	1.00			
3.4	0.35	0.50	0.28	1.00		
3.5	0.48	0.31	0.50	0.22	1.00	
3.6	0.59	0.61	0.27	0.39	0.52	1.00

Correlations for Group A

	3.1	3.2	3.3	3.4	3.5	3.6
3.1	1.00					
3.2	0.79	1.00				
3.3	0.81	0.80	1.00			
3.4	0.78	0.91	0.70	1.00		
3.5	0.65	0.55	0.39	0.63	1.00	
3.6	0.63	0.70	0.37	0.59	0.60	1.00

Correlations for Group B

	3.1	3.2	3.3	3.4	3.5	3.6
3.1	1.00					
3.2	0.11	1.00				
3.3	0.80	0.09	1.00			
3.4	0.43	0.52	0.17	1.00		
3.5	0.84	-0.08	0.85	0.08	1.00	
3.6	0.58	0.56	0.26	0.74	0.27	1.00

Correlations for Group C

	3.1	3.2	3.3	3.4	3.5	3.6
3.1	1.00					
3.2	0.65	1.00				
3.3	0.51	-0.03	1.00			
3.4	-0.01	-0.08	-0.04	1.00		
3.5	0.50	0.30	0.11	-0.11	1.00	
3.6	0.44	0.59	-0.34	-0.23	0.41	1.00

Correlations for Group D

	3.1	3.2	3.3	3.4	3.5	3.6
3.1	1.00					
3.2	0.73	1.00				
3.3	0.63	0.67	1.00			
3.4	0.07	0.24	0.11	1.00		
3.5	0.32	0.43	0.55	0.26	1.00	
3.6	0.76	0.69	0.57	0.29	0.64	1.00

Appendix I

Group A Exclusion

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.1T	11	80	7.27273	1.81818
3.1P	22	149	6.77273	1.51732

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	1.83333	1	1.83333	1.13563	0.29481	4.15962
Within Groups	50.0455	31	1.61437			
Total	51.8788	32				

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.2T	11	70	6.36364	4.25455
3.2P	22	141	6.40909	1.96753

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.01515	1	0.01515	0.0056	0.94082	4.15962
Within Groups	83.8636	31	2.70528			
Total	83.8788	32				

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.3T	11	81	7.36364	1.85455
3.3P	22	162	7.36364	1.19481

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	2.3E-13	1	2.3E-13	1.6E-13	1	4.15962
Within Groups	43.6364	31	1.40762			
Total	43.6364	32				

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.4T	11	68	6.18182	4.56364
3.4P	22	111	5.04545	3.09307

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	9.4697	1	9.4697	2.65447	0.11338	4.15962
Within Groups	110.591	31	3.56745			
Total	120.061	32				

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.5T	11	87	7.90909	0.69091
3.5P	22	177	8.04545	0.9026

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.13636	1	0.13636	0.16344	0.68878	4.15962
Within Groups	25.8636	31	0.83431			
Total	26	32				

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
3.6T	11	81	7.36364	1.65455
3.6P	22	152	6.90909	1.41991

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	1.51515	1	1.51515	1.01307	0.32196	4.15962
Within Groups	46.3636	31	1.4956			
Total	47.8788	32				

Appendix J

Questionnaire Results

Means and Standard Deviations of QUIS items.

	TIMS Mean	TIMS SD	P-TIMS Mean	P-TIMS SD
Part 3: Overall Reactions to the System:				
1. terrible–wonderful	7.09	1.27	6.77	1.23
2. frustrating–satisfying	6.05	1.65	6.41	1.40
3. dull–stimulating	6.95	1.33	7.36	1.09
4. difficult–easy	5.50	2.04	5.05	1.76
5. ineffective–powerful	7.41	1.10	8.05	0.95
6. rigid–flexible	6.82	1.40	6.91	1.19
Part 4: Screen				
1. Characters on the computer screen hard to read–easy to read	7.41	1.14	7.14	1.28
2. Highlighting on the screen helpful not at all–very much	7.10	1.17	6.86	1.17
3. Screen layouts helpful never–always	6.91	1.27	7.09	1.31
a) Amount of information displayed inadequate–adequate	6.91	1.69	7.27	1.12
b) Arrangement of information illogical–logical	7.09	1.48	7.55	1.18
4. Sequence of screens confusing–clear	6.55	1.82	7.32	1.49
Part 5: Terminology and System Information				
1. Use of terms inconsistent–consistent	7.10	1.18	7.14	1.21
2. Terminology relates to your work unrelated–well related	6.91	1.27	7.14	1.36
3. Messages appearing on screen inconsistent–consistent	6.91	1.34	7.27	1.16
4. Messages appearing on screen confusing–clear	6.91	1.57	7.00	1.41
5. Computer keeps user informed never–always	6.36	1.79	6.60	1.47
a) Predictable results from operations never–always	6.27	1.58	6.27	1.28
b) User can control feedback never–always	5.95	1.75	5.91	1.34
6. Error messages unhelpful–helpful	6.18	2.51	6.10	1.92
Part 6: Learning				
1. Learning to operate system difficult–easy	5.05	1.94	5.10	1.67
2. Exploration of features discouraging–encouraging	5.82	1.40	6.18	1.62
3. Remembering names and commands difficult–easy	5.68	1.46	5.59	1.84
4. Tasks performed in straight-forward manner never–always	5.95	1.73	6.14	1.49
5. Help messages confusing–clear	6.27	1.70	6.43	1.47

6. Training materials				
confusing–clear	6.90	1.34	6.55	1.41
7. Animated demonstrations				
not helpful–helpful	6.86	1.41	6.68	1.29
Part 7: System Capabilities				
1. System speed				
too slow–fast enough	7.05	1.21	7.52	1.75
2. System reliability				
unreliable–reliable	7.05	1.21	7.48	1.33
3. System functionality				
not adequate–adequate	7.09	1.15	7.38	1.32
4. Correcting mistakes				
difficult–easy	5.55	1.97	6.20	1.88
5. Needs of all ranges of users considered				
never–always	5.76	1.87	5.90	1.59
a) Novices can accomplish tasks				
with difficulty–easily	4.43	1.83	4.95	1.84
b) Experts can use features/shortcuts				
with difficulty–easily	7.00	1.29	7.00	1.63

APPENDIX K

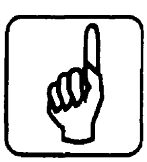
QUICK START MANUAL MODIFICATIONS

Personalized TIMS

This Quick Start manual is modified slightly to use with an experimental TIMS system that is personalized to each individual user. If you are a previous user of the TIMS program you will notice several differences in the operation of the system. These changes are intended to make the system easier to use for users who are just beginning to use the system. These changes are as follows:

1. Animated Demonstrations are triggered when one of the Strategies or the Planning Assistant, Strategy Assistant, or Cash Flow Assistant are used for the first time. These demonstrations are meant to remind the user of the basic operations of the strategy or assistant. Each demonstration lasts for about two minutes and will not modify the family or situation that is being used. The second and third time that you access the strategy or assistant you will be asked if you would like to see the demonstration again. Demonstrations may be re-viewed at any time from the title bar of the strategy or one of the Demo buttons in the Planning Assistant or Strategy Assistant. A pull-down Demo button is also on the menu bar of the TIMS desktop.

2. Strategy Interaction Detection. An individual situation will often have a number of strategies in place. The Interaction Detection makes note of the situation's strategies and determines if the strategies could have an effect on each other. The detection between one particular strategy and all the other strategies currently in place for the situation happens immediately after the strategy is implemented and is intended to make the TIMS system clearer to the user. A full report of all strategy interactions detected for a situation is available in a report under the Report pull-down menu. This information is meant to bring the interactions to the attention of the user but interactions do not necessarily indicate a problem, just something you should be aware of.



3. Strategy Recommendations are triggered when a newer user is in the Detailed Assets dialog and inserts a Buy, Sell, Transfer or Principal Modifier record. Since the underlying system processing is identical to that available in equivalent strategies, the strategy is recommended and can be accessed from the recommendation dialog. The strategy is preferable since the information is presented at a higher level and gives the user better support.

4. Data Entry Assistance. TIMS has a logical group of data entry menus that will ensure that a complete set of data is entered for a new family and situation. Normally the Next Dialog and Previous Dialog buttons are used to navigate from one dialog to another. In order to avoid accidentally exiting from the loop the OK and Cancel buttons trigger a dialog warning you that the data entry is not complete and allowing you to return to the loop. Following data entry a logical flow from initial situation creation, to creating a new situation complete with your planning suggestions for the

client to producing the final hardcopy of the financial plan is constructed. Animated demonstrations are available to show the steps involved.

All the modifications are accessible in the **Preferences** dialog and you may turn them off at any time. To resume the adaptation just click on the option again. During normal use the Personalized TIMS system will slowly turn off the adaptations as your experience with the system increases over time.

Throughout the Quick Start manual you will notice the four adaptive system icons in the left margin of the text. This serves as an indicator to you that this section of the manual is slightly different for your system and that you are experiencing one of the special adaptations indicated by the icon.

We welcome feedback as to the helpfulness of these changes and any suggestions that you may have for future improvements. At the completion of the training program you will be asked to fill out a questionnaire where there will be ample room for comments.

Icons used in the Personalized-TIMS Quick Start Guide.

ScreenCam Demonstration

Strategy Interactions

Strategy Recommendations

Data Entry Assistance