TOWARDS A GENERIC APPROACH TO PROVIDING PROACTIVE TASK SUPPORT

by

Yuen Wai Leung

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Management Sciences

Waterloo, Ontario, Canada, 2001

0-612-60551-5

Abstract

An increasing amount of task support resources has been placed online in a variety of forms such as help, references, wizards, cue-cards, examples, and interactive tutorials, to reduce users' need for training and task support from human experts. However, typically, users have to leave their task context and search for task support with a query in a separate context on a trial-and-error basis. There are several problems with this approach: Users may not be able to formulate proper queries; the process is often fruitless and frustrating because users are left alone to navigate through the query result. Most importantly, when given a new system to use, users have little desire to learn if they can use methods that they already know, regardless of their efficacy.

The basic idea in this research is to provide relevant task support for a computer-based application in a proactive but non-obtrusive manner. The task support system operates as an intelligent agent, which monitors the task progress and suggests relevant online resources continuously based on the user's task context. Advice and relevant domain knowledge are then displayed continuously in separate and persistently present advice windows side-by-side to the task window, and the display is updated at short intervals, without interfering with the user's task. An artificial neural network is used to identify the current task with the user's task progress as input. The artificial neural network recognizes one or more plausible tasks to approximate a user's task so that a range of relevant advice can be offered for the user's selective use.

A prototype, called Telephone Triage Assistant (TTA), has been built to support novice nurses in identifying diseases based on a phone interview with a patient. The usability of TTA has been assessed through a field study. Results show that, on average, 41% of the subjects' task time was spent on TTA and up to 70% of their questions appeared to be influenced by TTA. Although the post-task questionnaire data shows that TTA was perceived easy to use and useful for the task, it also reveals that subjects' perception of the continuous update was barely positive.

The proposed approach is expected to significantly alleviate the problems associated with conventional task support in the following ways: Users do not need to initiate search by themselves because relevant task support is presented to them at the moment of need. In addition, because of the continuous information update, advice to the user can be refined incrementally according to the task progress. Finally, users have full control over the task support with the options of either following up on the task support or ignoring it completely. The main thrust of the proposed approach is its potential for enhancing access to online resources and literally bringing them to knowledge workers' fingertips.

Acknowledgments

It is hard for me to summarize in retrospect the important events that have led to the completion of this dissertation. When I arrived at Waterloo four years ago, I carried suitcases loaded with books and a mind full of dreams. In the embrace of a group of thoughtful, supportive, and caring friends and classmates in Waterloo, I have been able to make some of these dreams come true. This dissertation is one of them. It would not have been possible without the help and encouragement that I have received in the Department of Management Sciences at the University of Waterloo.

All members of my dissertation committee helped me develop a sharp focus on my research. Professor Ji-Ye Mao, my principal supervisor, has often supported my suggestions and encouraged my pursuit of those ideas. I am grateful for his honest criticisms, constructive advice, and continual guidance. I would also like to thank Professor Niall Fraser, my co-supervisor, who patiently assessed my work and provided invaluable feedback and recommendations.

This work would not have been possible without the help from my co-workers in the USA, Australia, and Hong Kong. Special thanks should go to Reza Samiei, Director of Magellan Health Services of USA; Dr. Mark Leaning, Director of Health Communication Network Limited of Australia; and Eric Ng, CEO of PNM Solutions of Hong Kong. There is no possible way I could thank them enough for their friendship and advice.

I would not have gotten here without what I have learned from my teachers and friends both in Canada and Hong Kong, or without the continuous support of my parents and girlfriend, whose companionship I have missed. Their love and understanding have made this work possible and worthwhile as I feel I really enjoy everything that I have done.

There is a tremendous sense of achievement in completing this work. It is the one dream that came true and will always remind me of the happy years I spent in Waterloo. I am now leaving this place with a few suitcases but with more dreams.

Table of Contents

List of Tables

List of Figures

1. Introduction

In many organizations, nearly every type of work is facilitated by computers because computer technology exerts a strong impact on how well organizations compete with their rivals in today's complex business world. However, installing a computer system alone does not necessarily lead to organizations' competitive advantages (Landauer, 1995; Rochlin, 1997). A more important factor is whether end-users are adequately trained and supported to use the computer technology to its fullest potential. A study by Anderson Consulting found that nearly 80% of management perceived the need to train and support end-users in the effective use of computer technology being one of the most critical issues on their agendas (Winslow & Caldwell, 1992).

Classroom training and field experts, which used to be the primary resorts for user training and support, are becoming more expensive but less effective. By removing end-users from their workplace and putting them in training programs, organizations need to pay expensive training costs for the trainer, transportation, classroom, meals, housing, and time away from the office (Winslow & Caldwell, 1992). Moreover, if the training is not of immediate use, the knowledge acquired may be forgotten by the time it is finally needed on the job. End-users may realize that "two weeks after the training program, I could barely remember how to get to my file directory" (Bullen & Bennett, 1996, p. 371). Meanwhile, good field experts are both hard to find and expensive to keep due to the high demand for these personnel and high turnover.

In order to reduce the cost and need for training, more and more resources have been placed online in a variety of forms such as help, references, and tutorials. For example, the size of the online resources in the Microsoft Office suite is even larger than that of its core programs (Mansfield, 1994). However, a shift in the training medium alone may not offer a satisfactory solution to better support end-users' task in today's workplace. Whereas the size and contents of online resources continue to grow, little attention has been directed to making online resources conveniently accessible. Conventional online help, for instance, is often referred to as an electronic version of a hard-copy manual referencing only the fact-oriented system functions (Elkerton, 1988). It does not help end-users to complete a task, although it might contain a great deal of information on how to work with each system function. It is not uncommon for end-users to have the feeling that "Online help is a bust. It never covers the exact problem I'm having and is written with lots of jargon" (Bullen & Bennett, 1996, p. 371). Furthermore, research has shown that end-users are active learners who "learn by doing" a computer-based task in the same job context rather than passively reading documentation or following training material (Carroll & Mazur, 1986). Consequently, most of the online resources are greatly under-utilized.

This research makes a conceptual difference between availability and accessibility of online resources, which highlights the fact that most of the online resources are available to users in theory, but not really accessible. It is aimed at providing task support in an efficient and convenient manner, i.e., narrowing the gap between availability and accessibility of online resources. In this dissertation, *task support* is considered either relevant domain knowledge tailored for the current task, or procedural advice, automation and scaffolding tools such as wizards. This research is also influenced by the notion of Electronic Performance Support Systems (EPSS), which provide integrated, on-demand access to information, advice, learning experiences, and tools to enable a high level of job performance with a minimum of support from other people (Fischer & Horn, 1997; Gery, 1991). Whereas the underlying idea of EPSS sounds appealing, there is little guidance in the literature to operationalize it. Moreover, there exists no generic methodology for implementing task support systems. This research can be considered an effort to operationalize the on-demand task support by proposing a generic approach.

The basic idea in the proposed approach is to offer relevant online resources to users in a proactive but non-obtrusive manner. The proposed approach involves a novel interface style and an architectural model for identifying and customizing relevant task support. A task support system operates as an intelligent agent, which monitors the task progress and suggests relevant online resources continuously based on the user's task context. Advice and relevant domain knowledge are displayed continuously in a separate and persistently present advice window side-by-side to the task window, and the display is updated at short intervals, without interfering with the user's task. An artificial neural network is used to identify the current task with the user's task process as input. The artificial neural network recognizes one or more plausible tasks to approximate, rather than to pinpoint, a user's task so that a range of relevant advice can be offered for the user's selective use. A prototype, called Telephone Triage Assistant (TTA), has been built to demonstrate the feasibility, style, and implementation strategy of the proposed approach. It supports the telephone triage task in a medical call center. The usability of TTA has been evaluated through a field study and the feedback was encouraging in general.

The proposed approach is expected to significantly alleviate the problems associated with conventional task support in the following ways: Users do not need to initiate search by themselves because relevant task support is presented to them continuously. In addition, because of the continuous information update, advice to the user can be refined incrementally according to the task progress. Finally, users have full control over the task support with the options of either following up on the task support or ignoring it completely.

This dissertation is structured as follows. Chapter 2 reviews prior research to illustrate the inadequacy of existing approaches and identifies relevant techniques that are instrumental to the proposed approach. Chapter 3 describes the proposed approach, which consists of a user interface based on persistently present windows on the side, and a task support engine using an artificial neural network. Chapter 4 describes details of a working prototype to illustrate the viability of the proposed approach. Chapter 5 describes a usability study of the prototype in a field setting, which involved novice nurses in a medical call center. Chapter 6 reports the results of the usability study. Finally, chapter 7 concludes this dissertation with discussion, future research directions, and conclusions.

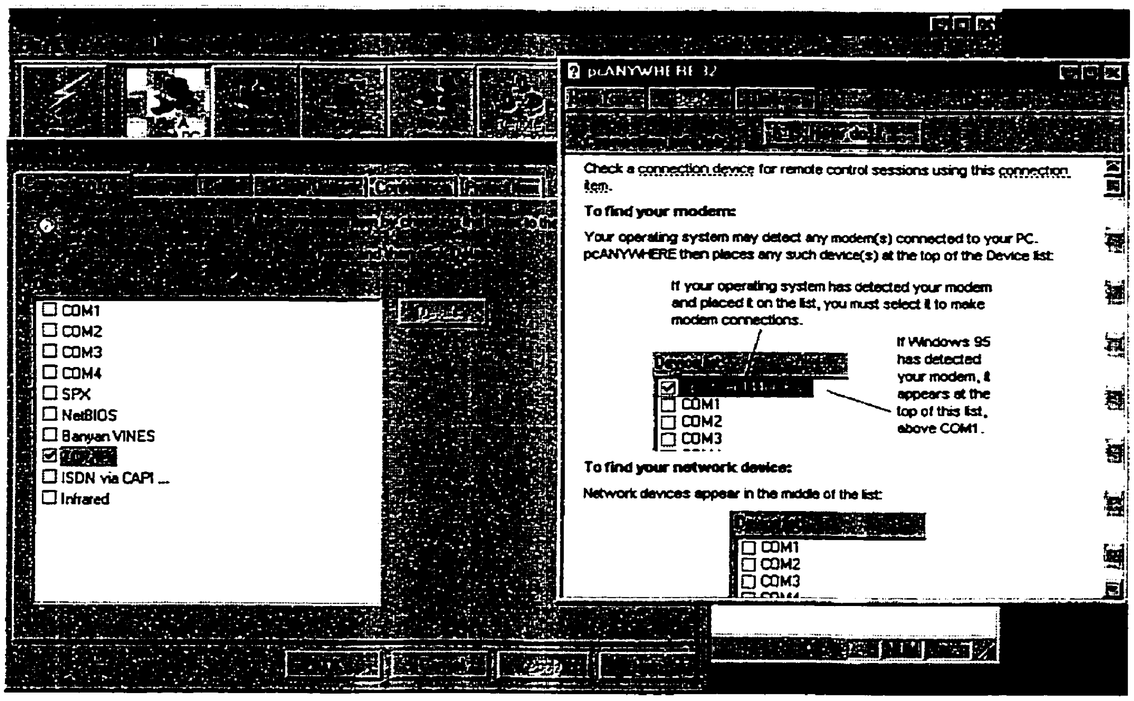
2. Literature Review

This chapter presents a comprehensive review of the task support literature drawing upon a wide range of work by both academic researchers and industrial professionals. It begins with a brief discussion of the basic ideas of task support systems, as part of the motivation for this research. Then, it examines the contributions and problems of online task support in both user-initiated and system-initiated environments. Next, it reviews some techniques that emanated from a number of relevant disciplines and can be instrumental for this research. Finally, a summary of findings from the literature review concludes this chapter.

2.1 Basic Ideas of Task Support Systems

A task support system is an independent program or a module embedded in a host application to facilitate users' task completion. In 1991, Gery first coined the term Electronic Performance Support System (EPSS) and envisioned it to be an integration of online support techniques, work-related software applications, and task automation tools (Gery, 1991). Although Gery and others (e.g., Desmarais, Leclair, Fiset, & Talbi, 1997; Dorsey, Goodrum, & Schwen, 1993; Fischer & Horn, 1997; Raybould, 1995) did not propose any specific methods to build such systems, there seems to be three underlying ideas: just-in-time support, just-enough support, and continuous performance development. These ideas appear desirable for any type of task support, and each of them is discussed briefly in the following paragraphs.

Just-in-time Support. Traditional task support such as Intelligent Tutoring Systems and Computer-Based Training often prepares users for a certain job and gives users a chance to practice the skills before actually using them. Just-in-time support occurs on-the-job. Whenever a user needs to perform a specific task, he or she is given the support to develop the competence at the moment of need, i.e., at the right time (Cole, Fisher, & Saltzman, 1997; Gery, 1995). It is believed that just-in-time support would be more effective than traditional task support approaches because users are not brought away from work. A good example of just-in-time support can be found from the online help of PCAnywhere, an application for accessing a PC from a remote site. While setting up a connection, a user can get task-specific information with a single click on the Help button, right in the context of work (Figure 2-1).



- ☐ COM1
- ☐ COM2
- ☐ COM3
- ☐ COM4
- ☐ SPX
- ☐ NetBIOS
- ☐ Banyan VINES
- ✓ [illegible]
- ☐ ISDN via CAPI ...
- ☐ Infrared

Check a connection device for remote control sessions using this connection item.

To find your modem:

Your operating system may detect any modem(s) connected to your PC. pcANYWHERE then places any such device(s) at the top of the Device list:

If your operating system has detected your modem and placed it on the list, you must select it to make modem connections.

If Windows 95 has detected your modem, it appears at the top of this list, above COM1.

- ✓ [illegible]
- ☐ COM1
- ☐ COM2
- ☐ COM3

To find your network device:

Network devices appear in the middle of the list:

- ☐ COM1
- ☐ COM2
- ☐ COM3

Figure 2-1. An Example of Just-in-time and Just-enough Support

Just-enough Support. Just-enough support emphasizes dividing online resources into task-oriented modules sufficiently small to provide support information that is just enough for a user to complete the task on hand (Bezanson, 1995; Gery, 1995). The intention of just-enough support is to maintain the job context. If too much material is provided but not of immediate use, the job context may be lost. In contrast, just-enough support presents only task-specific knowledge to retain the job context. The previous example in Figure 2-1 also exhibits the just-enough characteristic as the user is given only enough information to complete the current task.

Continuous Performance Development. Much empirical evidence has shown that most users handle their tasks on hand without necessarily optimizing the solution (e.g., Desmarais, Larochelle, & Giroux, 1987; Eberts, Villegas, Phillips, & Eberts, 1992; Fisher, Lemke, & Schwab, 1985; Furman & Spyridakis, 1992). Once at work, users stop enhancing their skills prematurely and usually do not want to invest time in learning new methods, which may be more efficient (Carroll & McKendree, 1987; Carroll & Rosson, 1987). Continuous performance development aims to make learning a less motivationally demanding task by reducing the cost of learning. Just-in-time and just-enough task support are expected to facilitate and result in continuous performance development.

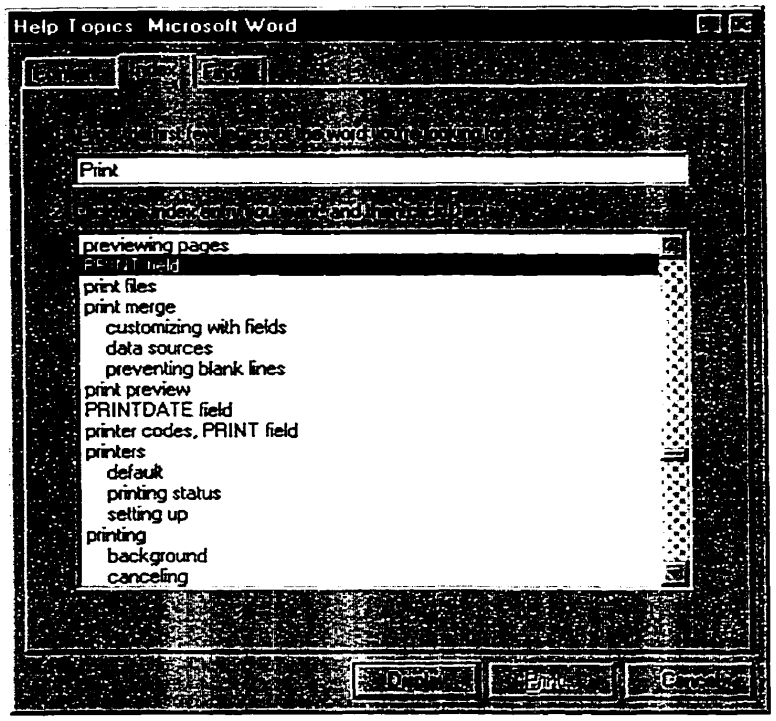
This research is influenced by the notion of EPSS and the three ideas described above. Whereas the ideas sound appealing, there is little guidance in the literature to operationalize them, e.g., when is just-in-time and how much is just-enough? Moreover, there exists no generic methodology for implementing task support systems. This research can be considered an effort to operationalize these ideas.

2.2 Access to Online Resources

Fundamentally, this research is about enhancing the accessibility of online resources. Access to online resources is typically either user-initiated or system-initiated. Two popular and reasonably representative systems, Windows Help System (WinHelp) and Office Assistant (OA), are discussed in this section together with their problems to illustrate the inadequacy of these two approaches.

2.2.1 User-Initiated Support

Windows Help System (WinHelp) is representative of user-initiated or passive support, and is the standard way to present online help in software applications running in the Windows environment (Hackos, 1997). Figure 2-2 shows a typical WinHelp window in Microsoft Word.

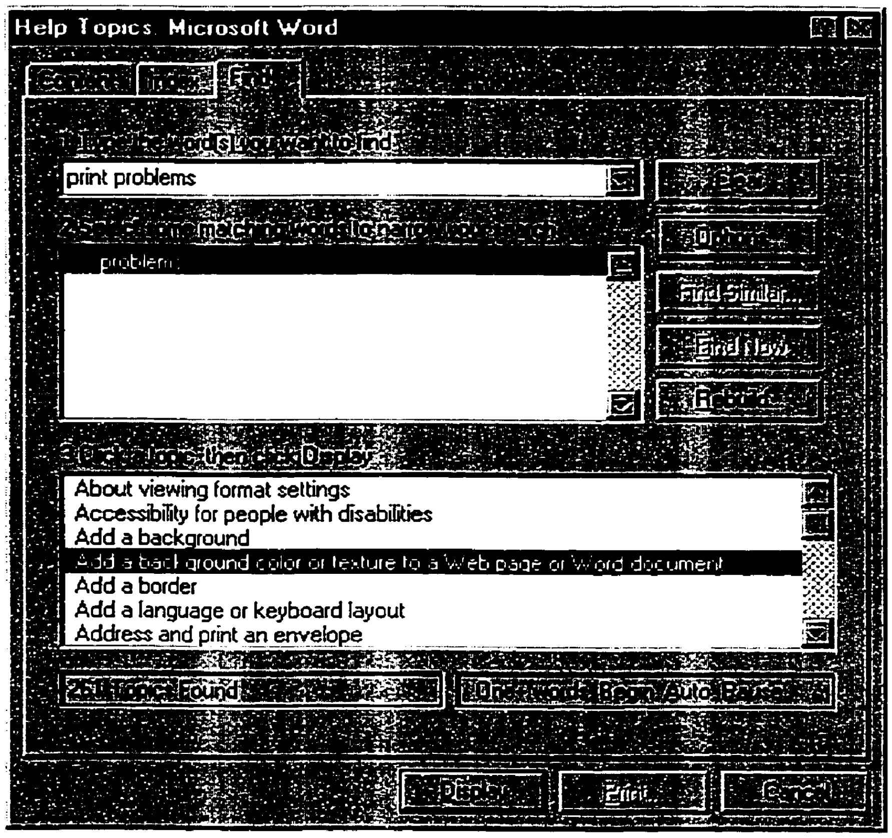

Help Topics Microsoft Word

Print

previewing pages
PRINT field
print files
print merge
  customizing with fields
  data sources
  preventing blank lines
print preview
PRINTDATE field
printer codes, PRINT field
printers
  default
  printing status
  setting up
printing
  background
  canceling

Figure 2-2. WinHelp in Microsoft Word

Users can access online resources in three ways. First, users can navigate through the online resources using the Contents tab. An analogy of the Contents tab is the table of contents of a book. On the Contents tab, all of the subjects are grouped by similar task and are organized in a hierarchical layout. Second, users can also access the online resources through the Index tab. An analogy of the Index tab is the index at the back of a book. It helps users identify a specific topic quickly without navigating through the hierarchical structure of online resources. Third, users can also use the Find tab to search for task information. The Find function is more inclusive than the Index and can be used to search the contents. For instance, if a user has trouble printing a document and uses *Print Problems* as the keywords to find help, a total of 261 "related" topics are found (Figure 2-3). Unfortunately, the result list is long and the user may need to refine the keyword and search again, or spend time looking at the topics one by one. In the above example, a "related" topic is *Add a Background Color* but it may have nothing to do with the user's current tasks. However, as the words *Print* and *Problems* are used inside the document, it is displayed as a related topic.



print problems

problems

About viewing format settings
Accessibility for people with disabilities
Add a background
Add a background color or texture to a Web page or Word document
Add a border
Add a language or keyboard layout
Address and print an envelope

Figure 2-3. Results from the Find Function

2.2.2 Problems with User-Initiated Support

There are three problems with user-initiated support. First, users may not be able to formulate queries effectively, because not all users are able to give a precise and differentiating description of something they lack knowledge about or they do not know the technology used by a given system (Bhavnani & John, 2000; Blair & Maron, 1985; Nickerson, 1999). The Index and Find functions of WinHelp require users to use keywords. If they cannot provide a precise keyword, they may not be able to use the functions. They may be forced to navigate through the Contents window topic by topic, until they find what they want. This is time-consuming and in the worst case, they may not be able to find anything useful.

Second, user-initiated support systems are inefficient. In WinHelp, users ask a question, and then wait for the result. But once the query result is returned, the job of the support system is done. Users are left alone to navigate through the result by themselves (Hertzum & Frokjaer, 1996; Horvitz, 1999). If the query result is a long list (see Figure 2-3), they need to spend time manually filtering out the topics that are not related to their current task. Sometimes, they need to re-formulate the query iteratively until a satisfactory list is obtained. Therefore, this process can be fruitless and frustrating.

Perhaps the most serious problem with user-initiated task support is one of a behavioral nature: the constant conflict between learning and working in work settings has been observed and characterized as the "production paradox:" Learning is inhibited by the overwhelming concern for throughput and working is inhibited by lack of knowledge (Carroll & McKendree, 1987; Carroll & Rosson, 1987). When given a new system to learn, novice users' typical sentiment is that "I want to do something, not learn to do everything" (Carroll & Rosson, 1987, p. 83). For more experienced users, the conflict is constant, too, between investing time in learning versus throughput. They have little desire to explore new functions or to search out new information if they can use methods they already know regardless of their efficacy. Consequently, more effective solutions are not learned and productivity suffers.

2.2.3 System-Initiated Support

There is no sophisticated system-initiated or proactive task support in the literature. Microsoft's Office Assistant (OA) offers some system-initiated task support, but it is rather limited (Noteboom, 1998; Shneiderman, 1997). For example, OA in Microsoft Word'97 automates some simple and trivial tasks such as formatting and typing, and spelling and grammar checks on the fly. It can also correct some common typographical errors such as *teh* to *the*, automatically. Moreover, OA can help simplify tasks by offering wizards. For example, as soon as "Dear John" is typed, the OA icon will appear on the screen asking "get help with writing the letter?" If the user accepts OA's offer, OA will ask a series of questions (e.g., format of the letter, recipient, and sender information) and then execute the task for the user (Figure 2-4). The entered information will be automatically inserted into a letter template, which allows the user to focus on the content of the letter. In addition, OA also asks the user whether envelope and mailing label wizards should be invoked to help prepare the envelope and mailing label. For example, the envelope wizard asks the user for the size of the envelope and using the sender's and recipient's addresses to print an envelope accordingly.

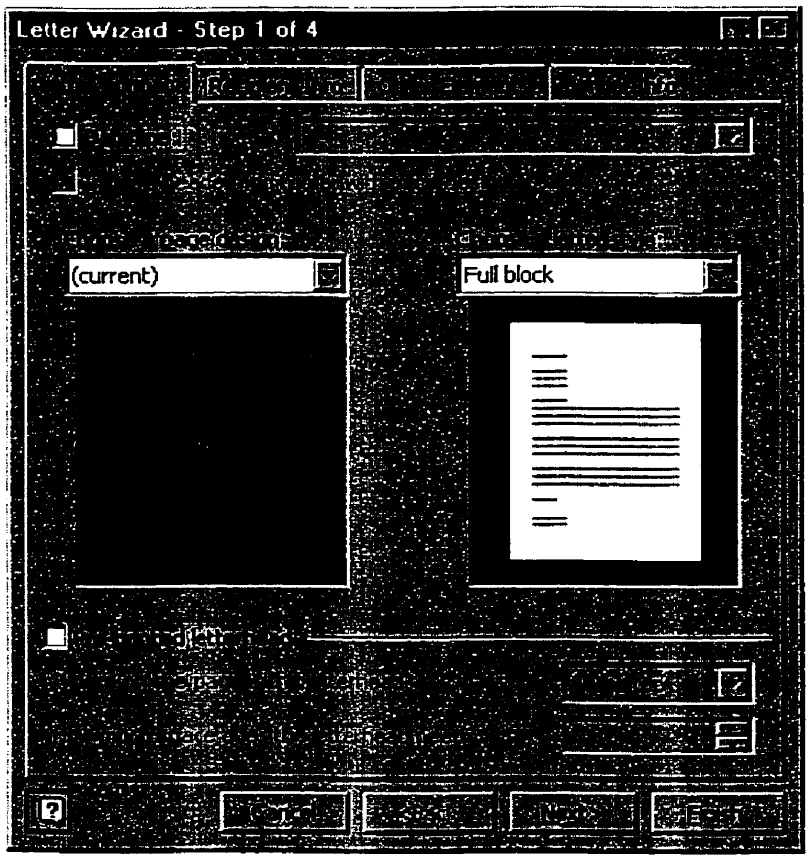


Figure 2-4. The Letter Wizard in OA

2.2.4 Problems with System-Initiated Support

There are three major problems with system-initiated support, too. First, correctly inferring a user's task is difficult without complicated algorithms (e.g., machine learning techniques) and detailed information about the user including the user's eye-movement, physical location, trace of prior activities captured in log files, and task context (Agah & Tanie, 2000; Orwant, 1991; Wolfe & Eichmann, 1997; Ye, 1997). Unfortunately, the accuracy of these algorithms remains a question and keeping all the user information seems impractical (Agah & Tanie, 2000; Beaumont, 1994). As a result, few software products provide system-initiated support, although some of them, e.g., Microsoft's OA, provide limited system-initiated support (Noteboom, 1998; Shneiderman, 1997). Whereas OA does occasionally volunteer information on applicable wizards (Figure 2-4), it is not needed most of the times. The animated OA is distracting enough to force users to take immediate action. Of course, proactive task support as such is often unwelcome, especially when OA misdiagnoses the intention of the user. For instance, automatically changing the lower case letter *i* to *I* can be obtrusive and annoying although this function can be turned off.

Second, even if the support system can successfully identify the relevant information users, software designers need to determine when the online resources should be delivered. If the information is not delivered at the right time to meet users' needs, it reduces the usefulness of the information and it may even annoy the users (Furman & Spyridakis, 1992; Shneiderman, 1997). Therefore, providing support in a system-initiated support environment introduces a new issue of timing.

Finally, users in general like predictability and to be in control (Hook, 2000; Shneiderman, 1997). They do not like surprises to result from their actions, but prefer to be the final decision maker in their choices. Therefore, proactive support may be problematic because it creates surprises for users and users may feel they are losing control over their actions.

2.3 Useful Techniques in Relevant Disciplines

This section reviews techniques that contribute to the development of task support, including multiple windows, intelligent tutoring systems, intelligent agents, artificial neural networks, and user modeling. They are used in the proposed approach for task support and the development of the prototype in this research.

2.3.1 Multiple Windows

Prior research shows that people seldom complete one task in a continuous time frame. Instead, they switch from application to application in response to events happening inside and outside the computing environment (Bannon, Cyber, Greenspan, & Monty, 1983). Multiple windows support the way that people really work by allowing them to perform multiple tasks (e.g., running multiple applications) in parallel, and to view the results of one task while performing another. In a task support environment, at least two applications run concurrently: one allows the completion of the task on hand and the other provides relevant information to help accomplish the task. Therefore, the use of multiple windows seems to be a natural choice for providing task support.

Card, Pavel, and Farrell (1984) outlined several areas in which multiple windows could be useful. First, multiple windows allow more information to be displayed on a small computer screen. For example, a handheld computer uses overlapping windows to compensate for the small screen size. Second, multiple windows allow multiple sources of information to be accessed and combined. For example, text from several electronic messages may need to be accessed and combined into a new message. This operation is simplified if the sources of information are displayed simultaneously. Third, multiple windows allow multiple programs to be controlled. For example, a programmer may use one window for the output of computer program, another for the debugger, a third one to display and edit the program code, and a fourth one to monitor the memory usage. Fourth, multiple windows could be used to help users keep track of information likely to be used in the near future without interfering with their current task. Examples include the system clock, calendar, and stock quotes. Finally, multiple windows could be used to display different representations of the same task and users could select the most appropriate representation to fit their needs. For example, in the TeSS text retrieval system, users have two windows, each with a different representation of the information retrieval task: the query in logical expressions and in Venn diagram (Hertzum & Frokjaer, 1996).

In a task support environment, the use of multiple windows can enable users to access information from multiple sources. Multiple windows also allow for proactive display of potentially relevant information without interfering with the user's task. Furthermore, displaying proactive task support in persistently present windows on the side can reduce obtrusive and unnecessary windows management (e.g., opening and closing windows).

Having persistently present windows on the side to display relevant advice is expected to be helpful to novice users. For experienced users, however, the advice should not be too distracting to them. In a different domain, research has found that experienced Web users are not distracted by animated graphics on the side (Diaper & Waelend, 2000). Therefore, using persistently present windows on the side to providing task support may suit a range of users.

2.3.2 Intelligent Tutoring Systems (ITS)[1]

An ITS is a computer program that assesses the current state of a student's knowledge and provides instruction tailored to that individual's learning needs within an overall context of courses, syllabuses, and tutorial objectives (Burns & Capps, 1988; Burton & Brown, 1982; Sleeman & Brown, 1982). Its objective is the same as that of task support systems, which is to provide customized information to address users' needs. Therefore, certain characteristics of ITS may be desirable in task support systems. Some examples of well-known ITS are GUIDON (Clancey, 1983) and Lisp Tutor (Anderson, Corbett, & Reier, 1986). These ITS typically have four components: a *user module*, an *expert module*, a *diagnosis module*, and a *user interface module.*

The *user module* stores information on how much the student knows about the concepts and relationships to be learned, and the student's level of knowledge and achievements. It often contains a history of tasks performed and the corresponding scores. The *expert module*, on the other hand, is a representation of the knowledge to be imparted. The knowledge to be taught is usually supplied by domain experts. The *diagnosis module* arranges teaching strategy. It contains methods of how to pass expert knowledge to the student. It also initiates remedial actions and adapts the difficulty level of the subject on

[1] ITS is used in both singular and plural forms in this document.

the basis of the student's previous performance. Finally, the *user interface module* comes between the system and the student, managing all interactions.

A more recent example is the Coach developed by IBM to teach Lisp programming (Selker, 1994). Coach (Cognitive Adaptive Computer Help) uses a *user module* to estimate the student's level of experience with Lisp programming. The *user module* is represented by frames of keyword. For each frame, it captures the student's experience such as how many times a keyword has been used, how long a keyword has not been used, and skill (e.g., novice, intermediate, and expert) with respect to a keyword. The *expert module* is also represented by frames of keyword. For each frame, it has three components: an example, a description, and syntax of the keyword. For each component, it has three levels of detail: novice, intermediate, and expert, which correspond to the students' skill in the user module. The *diagnosis module* is implemented in production rules. It determines the student's skill with respect to a particular keyword and decides which level of detail of information would be most helpful.

The *user*, *expert*, and *interface* modules in ITS provides an architectural framework which can be used in developing a task support system. For instance, user model is one component of *user module* (which will be explained later in this chapter). Online resources in a software application parallels to the knowledge in an *expert module*. An *interface module* can manage all interactions between the system and users.

2.3.3 Intelligent Agents

Intelligent agents are another promising technology that is useful for providing task support because one of the most important characteristics of intelligent agents is the ability to do things proactively on behalf of the user. A task support system could be analogous to an intelligent agent, which retrieves and presents information before the user requests it.

There is no commonly agreed definition of exactly what an agent is. Herein, an agent is loosely defined as a self-contained program capable of controlling its own decision-making and action, based on its perception of its environment, in pursuit of one or more objectives (Jennings & Wooldridge, 1996; Ndumu & Nwana, 1997). The other way to determine agenthood is by examining the underpinning attributes of the program. Maes (1994), and Ndumu and Nwana (1997) highlighted four key attributes: (1) autonomy (i.e., the ability to function largely independent of human intervention), (2) social ability (i.e., the ability to interact intelligently with and learn from other agents and/or users), (3) responsiveness and adaptiveness (i.e., the ability to perceive the environment and/or user and respond in a timely fashion), and (4) proactive provision (i.e., the ability to take the initiative whenever the situation demands). In a given problem domain, an agent might possess each attribute described above to a different degree and might sometimes have extra attributes, such as personality in an interface agent and mobility in a communication agent.

The proactive nature of intelligent agents can be demonstrated with Maxims (Maes, 1994), which assists users in managing their e-mail. It continuously "looks over the shoulder" of the user as the user handles his or her e-mail. As the user takes an action, Maxims memorizes the situation and actions, and it will offer to take the same action when the situation occurs again. Whereas Maxims can remind the user of important events and automate some actions, it has several problems. First, if Maxims had misdiagnosed the intention of the user, the automation would be annoying. Second, when a new situation arises, Maxims will try to find a similar situation and recommend an action for the user to confirm. Such confirmation helps prevent incorrect automation but it could be obtrusive.

Lieberman (1997) developed an intelligent agent called Letizia at the Media Lab of MIT to help web users surf the web in a continuous and cooperative manner. Letizia provides proactive advice as it is the case with Maxims but at the same time it alleviates the obtrusiveness by using multiple browsers to display its suggestions without forcing the user to comply. It records the URLs chosen by a user and reads the pages to compile a profile of the user's interests. The inference is done through a simple keyword-frequency information retrieval measure to analyze the pages accessed. Based on this information, it then suggests a list of related web sites in a separate browser and displays the content of the selected web pages in another browser, like "channel surfing" (Figure 2-5). The user may either continue browsing the current page or follow up on Letizia's suggestions, which are displayed in the browser on the right.

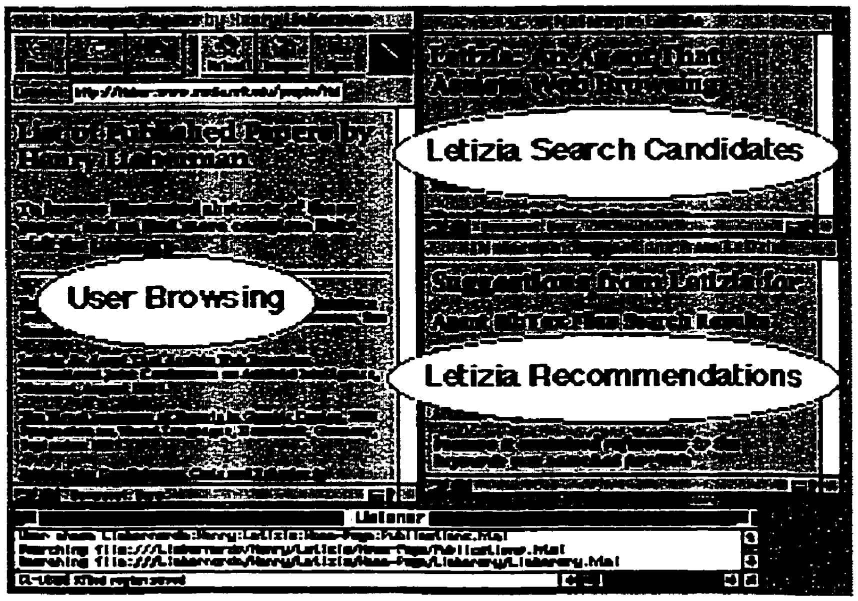


Figure 2-5. Letizia's Windows

It appears that one way of providing proactive advice is to have the task support system operate as an intelligent agent, which monitors the user's task progress and suggests relevant online resources continuously based on the user's task context. Moreover, the Letizia style of displaying task information on the side seems to be advantageous. Whereas the advice may not always be useful, its use is optional and the cost to the user is minimal as no user effort is needed. In addition, using separate browsers to display task support does not interfere with the user's task and the user still has final control of the system.

2.3.4 Artificial Neural Networks (ANN)[2]

Artificial Neural Networks (ANN) or neural networks are computer-based self-adaptive models that were first developed in the 1960s, but they became popular only in the mid-1980s after the development of the backpropagation algorithm (Rumelhart, Hinton, & Williams, 1986). Initially derived from neuroscientists' models of the human brain, ANN now encompass a wide variety of applications such as data analysis, pattern recognition, and voice recognition.

ANN are a useful tool to infer users' tasks because of their modeling power, robustness, ability to recognize patterns and to work with incomplete or noisy data in particular (Rumelhart, Widrow, & Lehr, 1994). As a modeling tool, ANN do not require strict assumptions about the data as other methods do, e.g., statistical models (Sarle, 1994). ANN's robustness arises from its capability of learning from examples and representing knowledge implicitly in weights between nodes (Simpson, 1990). When an ANN is required to learn from examples of a new environment, re-training is all that is needed. Explicit knowledge acquisition and knowledge representation, which are the bottlenecks of most knowledge-based systems, are not required (Ye, 1997). Moreover, ANN's ability to recognize patterns also makes this technique attractive for task support. Searching for and filtering information can be considered a pattern recognition task (Eberts & Habibi, 1995). ANN can help identify common patterns based on data collected from users. For instance, ANN have been successfully applied to categorizing messages from electronic bulletin boards based on words from posted messages, classifying meeting transcripts based on words from meeting transcripts in a group decision support environment, and

[2] ANN is used in both singular and plural forms in this document.

sorting e-mail to different categories based on words from e-mail (Eberts & Habibi, 1995). Finally, the ability of ANN to work with incomplete and noisy data is particularly useful and advantageous for providing task support because task support should occur before task completion, and novice users may not always be able to chart the correct course of action. In other words, input to a task support system is always incomplete and compromised by noise.

ANN in Medical Diagnosis. The application of ANN in medical diagnosis is reviewed in detail for two reasons: (1) the task may be representative of diagnostic tasks in general, and (2) the task domain of the prototype built in this research is medical diagnosis.

A typical medical diagnostic process involves the following steps: (1) observation of an abnormal situation, (2) collection of symptoms, (3) identification of likely diseases, (4) observation and collection of further information, and (5) final determination of likely diseases (Patel & Groen, 1991; Patil, 1988). Prior research has shown that task support would be helpful in the collection of symptoms and identification of possible diseases (Lindgaard, 1995; Patil, 1988). Physicians usually generate hypotheses (likely diseases) based on the observed symptoms. Therefore, in order to generate plausible hypotheses, they should know the symptoms well, have access to the knowledge of symptoms, and possess effective retrieval mechanisms. Furthermore, physicians tend to perform a relatively superficial search for a possible disease and ignore the possibility that the same symptoms are likely to occur in other diseases (Lindgaard & Triggs, 1990). Research has even found that physicians may not be able to generate a complete set of hypotheses when presented with certain symptoms (Schaafstal, 1993). If a single disease being diagnosed happens to be wrong, this may not be realized until compelling evidence convinces the physicians to pursue other diseases, which may have wasted much time. In the worst case, wrong diagnosis and treatment may further deteriorate the situation (Lindgaard, 1995). Therefore, task support such as presenting a range of likely diseases in a ranked order would be useful.

A search in MEDLINE[3] for articles about ANN applications between the years 1990 and 2000 resulted in more than 800 citations (e.g., Ashutosh et al., 1992; Baxt, 1990; Maclin & Dempsey, 1993; Rogers, Ruck, & Kabrisky, 1994; Silva & Roque Da Silva, 1998). For instance, Baxt (1990) used the backpropagation algorithm with a three-layer ANN to help diagnose whether patients had coronary occlusion. A total of twenty nodes were used in the input layer. The input nodes were the patients' details (e.g., age and gender), symptoms (e.g., nausea and syncope), medical history (e.g., hypertension and cigarettes), and examination results such as pulse and edema. They were coded in a binary manner such that 1 indicated the presence of a fact and 0 the absence of it. Patients' age, blood pressure, and pain intensity were normalized and coded as values between 0.0 and 1.0. There was only one output node. The value of the output was coded as 0 for the absence of coronary occlusion and 1 for the presence of it. Ten hidden nodes were used in the hidden layer. Three hundred and fifty six patients' cases were collected. Half of them were used for training and the rest was used for testing. The results were encouraging. In the testing set, the ANN correctly diagnosed 56 of the 60 patients with coronary

[3] MEDLINE is the U.S. National Library of Medicine's database of references, which has more than 11 million articles published in 4300 biomedical journals. It can be accessed at http://medline.cos.com.

occlusion (92%) and 113 of the 118 patients without coronary occlusion (96%). A similar application of ANN was used to classify hepatic masses in liver cancer into metastatic carcinoma, hepatoma, cavernous hemangioma, abscess, and cirrhosis (Maclin & Dempsey, 1993).

Silva and Roque Da Silva (1998) used an ANN in medical diagnosis using a partial list of symptoms, as is the case in the early stages of the evolution of a disease. There were twenty diseases (output) and one hundred and fifty two symptoms (input). For each disease, a partial list of their symptoms was prepared, each containing 25%, 50%, and 75% of the total set of symptoms. The symptoms in each partial list were chosen randomly. The partial lists of symptoms were input into the ANN. The average error was approximately 42% for the 25% partial lists, 15% for the 50% partial lists, and 11% for the 75% partial lists respectively. The results are important, as ANN seems to alleviate one of the difficulties exhibited by conventional medical expert systems, which is dealing with incomplete data. In general, when a patient seeks medical help, not all of the symptoms characterizing the disease have already manifested themselves. During the course of the disease, certain symptoms emerge earlier while others appear later. The earlier the detection of the disease, the higher the chances for the patient to receive appropriate treatment. Therefore, it is desirable to be able to diagnose the disease or likely diseases from a partial list of symptoms.

In the development of a task support system in medical diagnosis, the same problem arises: Input to the task support system is always incomplete, and may be compromised by noise because a patient may not be able to clearly articulate symptoms or test results may be inaccurate. ANN seems to fit this situation because it can handle this challenge better than other techniques (Rogers et al., 1994; Silva & Roque Da Silva, 1998).

ANN in Human-Computer Interaction (HCI). ANN in HCI is described to show how ANN could help infer users' tasks based on their interaction with the computer. ANN have been used in HCI since the 1980s (Ye, 1997) and there are conferences on using ANN in HCI (e.g., Workshop on Pattern Recognition and Neural Networks in HCI, 1991). In addition, a special issue was devoted to ANN by the International Journal of Human-Computer Interaction in 1997 (cf. Ye, 1997).

Villegas and Eberts (1994) studied the use of an ANN in identifying text-editing tasks based on users' activities using a text editor. More specifically, the ANN had six output nodes representing different text-editing tasks such as addressing a memo and organizing lines in a memo, and 195 input nodes representing activities such as entering a date and copying a line. A three-layer network was used with five hidden nodes. The coding scheme was similar to the one used by Baxt discussed earlier except that "1" was used to indicate no activation and "10" high activation. In total, 4182 cases were used in training and 5505 cases were used in testing. The ANN correctly recognized 96% of the cases in the testing sample. Villegas and Eberts' study is interesting because it showed that using an ANN to infer users' tasks based on users' activities is fruitful. However, in Villegas and Eberts' work, task identification occurred after task completion. This can be extended to using an ANN during a task progress to infer users' tasks and provide online support to help a user accomplish the task.

Relationship between Input and Output. In addition to previously discussed strengths of ANN, a less commonly used one is the possibility to determine the relationship between the input nodes and output nodes based on the structural property of a trained ANN. In this subsection, a method for inferring the relationship is briefly discussed and a mathematical model for assessing the relationship between the input and output nodes of feedforward ANN is outlined.

ANN have no explicit representation of declarative knowledge as in rule-based systems, from which explanations can be generated to justify the reasoning process and outcome (Charniak & McDermott, 1985). However, in ANN, knowledge is implicitly encoded in weights between nodes and distributed all over the network. Much research has been carried out to extract knowledge from ANN (e.g., Diederich, 1992; Feldman & Ballard, 1982; Mozer & Smolensky, 1989). For instance, Diederich (1992) proposed a model based on the structural property of an ANN to generate reasoning paths and explanation traces. The activation energy of node x is defined as S_x and is calculated as follows:

$$S_x = \sum_{j=1}^{m} w_j x_j$$

where x_j's are the input values, w_j's are the weights, and m is the total number of input links to node x_j. If S_x is greater than a predefined threshold in an inference, node x_j will be marked as activated and recorded. If a user requires an explanation after an inference, a replay of activation patterns of activated nodes is shown to illustrate how the inference was made. Thus, an explanation is generated in this way.

Another model for assessing the relationship between input nodes and output nodes of feedforward ANN, based on a similar idea, was developed (Chiu & Leung, 1996) and is briefly reviewed as below. Consider the simplest case of a two-layer feedforward ANN with m nodes in the input layer and one node in the output layer. The output value is denoted as

$$z = f\left(\sum_{j=1}^{m} w_j x_j\right)$$

where f is defined as a non-linear differentiable monotonically non-decreasing function, w_j 's are the weights, and x_j 's are the inputs. Since z is distributed among the inputs, a dependency relationship between input x_j (> 0) and output z can be calculated as follows:

$$I_{A(w)}(x_j, z) = \frac{|w_j x_j|}{\sum_{k=1}^{m} |w_k x_k|} z \qquad (1)$$

where $A(w)$ denotes the given feedforward ANN with the set of weights $w = (w_1, w_2, ..., w_m)$, $\sum_{k=1}^{m} |w_k x_k| \neq 0$, and $|\ |$ denotes the absolute value. For convenience, $I_{A(w)}(x_j, z)$ is called the **I-value** between x_j and z. The higher the I-value, the stronger the dependency of z on x_j (or input x_j is strongly relevant to z).

For a fully connected feedforward ANN of $(n\text{-}1)$ hidden layers (h) with intermediate nodes y and outputs z, a measure of dependence between the output z_i on x_j is then defined, based on the summation of the absolute relative weights of all the paths connecting the two nodes, as:

$$I_{A(w)}(x_j, z_i) = \sum_{\forall h_{n-1}} \sum_{\forall h_{n-2}} \cdots \sum_{\forall h_1} \left\{ \left| z_i \left(\frac{\left| w_{h_{n-1}(n-1)}^{i(n)} y_{h_{n-1}(n-1)} \right|}{\sum_k \left| w_{k(n-1)}^{i(n)} y_{k(n-1)} \right|} \right) \left(\frac{\left| w_{h_{n-2}(n-2)}^{h_{n-1}(n-1)} y_{h_{n-2}(n-2)} \right|}{\sum_k \left| w_{k(n-2)}^{h_{n-1}(n-1)} y_{k(n-2)} \right|} \right) \cdots \right.\right.$$
$$\left.\left. \cdots \left(\frac{\left| w_{h_1(1)}^{h_2(2)} y_{h_1(1)} \right|}{\sum_k \left| w_{k(1)}^{h_2(2)} y_{k(1)} \right|} \right) \left(\frac{\left| w_{j(0)}^{h_1(1)} x_j \right|}{\sum_k \left| w_{k(0)}^{h_1(1)} x_k \right|} \right) \right| \right\} \qquad (2)$$

The numerators inside the curly brackets correspond to the weighted path between an input node x_j and an output node z_i. Note, $\sum_k \left| w_{k(n-1)}^{i(n)} y_{k(n-1)} \right|$, $\sum_k \left| w_{k(n-2)}^{h_{n-1}(n-1)} y_{k(n-2)} \right|$,... $\sum_k \left| w_{k(1)}^{h_2(2)} y_{k(1)} \right|$, and $\sum_k \left| w_{k(0)}^{h_1(1)} x_k \right|$ are always larger than zero. The rationale of the above equation is that, first, an output value z_i is distributed back to an input x_j according to the number of paths between the input node and the output node. Second, the I-value is proportional to the ratio of the absolute value of the weighted term to the weighted sum of the inputs in each node that makes up the steps of the path.

Evaluating the dependency relationship between an input node and an output node of an ANN using I-values has been proven to be useful and reliable in identifying the relevant input with artificial images and biomolecular images (Chiu & Leung, 1996; Chiu & Leung, 1998). This method can be useful for task support if an ANN is used to infer the task based on already completed steps as input, it could help justify the inference or suggested actions.

2.3.5 User Modeling

A user model is an explicit representation of the properties of a particular user (Jameson, Paris, & Tasso, 1997). The goal of user modeling is to create systems that are adaptive to an individual's needs, abilities, and preferences. User models may improve the effectiveness of task support systems by providing customized information according to the user's level of understanding. Three examples are reviewed herein, and their strengths and weaknesses are discussed.

UNIX Consultant (UC) is a natural-language online help system based on user modeling to advise users on using the UNIX operating system (Chin, 1986; Chin 1989). It uses stereotyping to categorize users' experience from novice to expert, and task complexity from simple to complex. For instance, if a user is familiar with the use of the *rwho*, an advanced command in UNIX to list all users on the network, the user is most likely at the intermediate or expert level. Therefore, UC assumes that the user knows all of the simple commands in UNIX (e.g., *ls* and *lpr*). Once the user's skill level is determined, the help information will be customized accordingly. The stereotyping of a user's experience and complexity of UNIX system concepts reduces the overhead to "ask" what the user knows and does not know. However, the user model in UC is for single session use without a long-term memory.

A more recent example, the Expert Finder (Vivacqua & Lieberman, 2000), classifies users' expertise by analyzing the usage of classes in users' Java programs based on a simple keyword-frequency algorithm. The user model has a long-term memory and is updated by the Expert Finder periodically by reading through a user's Java programs. Table 2-1 shows an example of the user model in Expert Finder. The information in the user model is used to find an expert whose expertise level is higher but close to the user's level to consult when problems arise. Vivacqua and Lieberman's model assumed that users would not forget anything that they had learned. This might not always be true.

Area	**Usage**	**Expertise Level**
java.io	10	Novice
java.util	15	Novice
System	20	Novice
elementAt	5	Novice
println	46	Intermediate

Table 2-1. An Example of the User Model in Expert Finder

Strachan, Anderson, Sneesby, and Evans (2000) proposed an algorithm to update the user model in a commercial software system for financial management. A simplified version is outlined in Figure 2-6. This algorithm is useful in a real-world environment, which the time that information was last accessed and the number of times the information has been accessed can both be used to determine users' skill level. Strachan et al. also assessed the effectiveness of the user model by comparing users' experience with versus without the user model. Forty-five subjects were involved in the evaluation, and using the software with user models had a positive effect on user satisfaction, ease of use, power and flexibility of the application.

```
If the user has been away for 30 days, downgrade user's skill level by one level.

If the user has used the system long enough since the last update (e.g., 10 sessions),
upgrade user's skill level by one level.

Examine all functions used recently (e.g., within five sessions) by the user:

    If the user has used two functions more advanced than the user's skill level,
    upgrade user's skill level by one level.

    If the user has used two functions less advanced than the user's skill level,
    downgrade user's skill level by one level.
```

Figure 2-6. An Algorithm to Update a User Model

2.4 Findings from the Literature Review

This chapter has reviewed the basic ideas of task support in terms of just-in-time, just-enough, and continuous performance development principles. It has identified the inadequacy and difficulties associated with the existing task support. In addition, useful techniques to be used in the proposed approach for proactive task support are also reviewed. In summary, four major conclusions can be drawn from this chapter.

First, as mentioned earlier, user-initiated support is ineffective, inefficient, and leads to behavioral problems due to the "production paradox." On the other hand, system-initiated support is prone to error and lacks user control. To overcome the deficiencies of user-initiated support, some degree of proactive task support is necessary because it is effective for raising users' awareness of the availability and applicability of relevant online resources and for facilitating the accessibility of online resources at the point of need. Therefore, this research explores a middle-ground approach between the above two established approaches to rectify the deficiencies of user-initiated task support and to overcome the difficulties associated with system-initiated task support. One possible middle ground between user-initiated and system-initiated task support is to use separate and persistently present windows to display task support to avoid interfering with users' task. For instance, task support is displayed in separate windows side-by-side to users' primary task window. In this way, users can follow up on the task support or ignore it completely without interfering with users' primary task window.

Second, to provide proactive task support, a system should operate as an intelligent agent, which constantly observes its master's work, to figure out how it can help, and whisper quietly whenever it finds something potentially useful. Therefore, certain general principles and techniques for developing intelligent agents can be used for task support.

Third, a system must be able to identify a user's task in order to provide task support timely and proactively in keeping with the just-in-time principle. The literature reviewed in this chapter shows that ANN seems to be an attractive technique for inferring a user's task based on incomplete and noisy data.

Finally, a key dimension for providing just-enough task support is to use users' task experience to customize supporting information to be presented. User modeling, often used in Intelligent Tutoring Systems (ITS), can be applied here by customizing task support to address an individual's needs, abilities, and preferences, based on the user's history of tasks performed and level of knowledge.

3. Proposed Approach

This chapter proposes a novel approach for providing proactive task support. It begins with a description of the scope and assumptions of the proposed approach. Next, it describes the interaction style and an architectural model. Then, it illustrates different components of the model and shows how they work together to achieve the objective of reducing the accessibility gap between users and online resources. Finally, a hypothetical example is constructed to illustrate the function of the key components of the proposed model.

3.1 Scope and Assumptions

Whereas this research aims at providing a generic approach to operationalizing just-in-time and just-enough task support, it begins with a particular type of tasks, namely diagnostic tasks. As described in Section 2.3, these tasks typically involve a process of collecting a set of symptoms, followed by inferring one or more likely problems, and finally determining the problem(s) based on known relationships between the symptoms and problems. In this context, task support means providing access to contextualized task domain knowledge, providing advice to influence the direction of exploration to collect symptoms, and suggesting likely underlying problems based on confirmed symptoms.

There are several reasons for choosing diagnostic tasks as the focus of this research. First, the problem-solving process type of tasks involves several discrete parallel steps. Therefore, the task process can be described in terms of completed parallel steps, and different problems in the same domain have a common set of basic steps. In other words, the task domain can be modeled with a system that has a finite set of input variables and a finite set of output variables. Not all tasks have this characteristic, e.g., composing a computer program is a task that cannot be modeled by a system with a finite output, each of which corresponds to one program.

Second, and less importantly, the order of the steps is not critical for task completion, e.g., symptoms can be identified in several different orders in the process of identifying the same problem in a trouble-shooting task, although some orders may be more natural or meaningful than others in practice. This characteristic makes the modeling simpler, but it is not an essential requirement for the proposed approach. Once again, computer programming does not have this characteristic as the sequence of statements can completely determine the function of a program.

Formally put, the assumptions about the task domain are: (1) a task should be composed of a series of discrete parallel steps, and (2) the task domain involves a finite set of basic steps (input) and a finite set of solutions (outcome). These assumptions make it easier for using ANN to identify users' task, and influence the type of support to be provided in this research.

3.2 Interaction Style

Figure 3-1 shows a conceptual schema of the proposed task support approach. The task support system behaves as if it is a human assistant watching over the shoulder of the user to identify and suggest relevant online resources continuously based on the user's task progress. The task support is in the form of a list of relevant online resources such as help, wizards, job aids, and best practices.

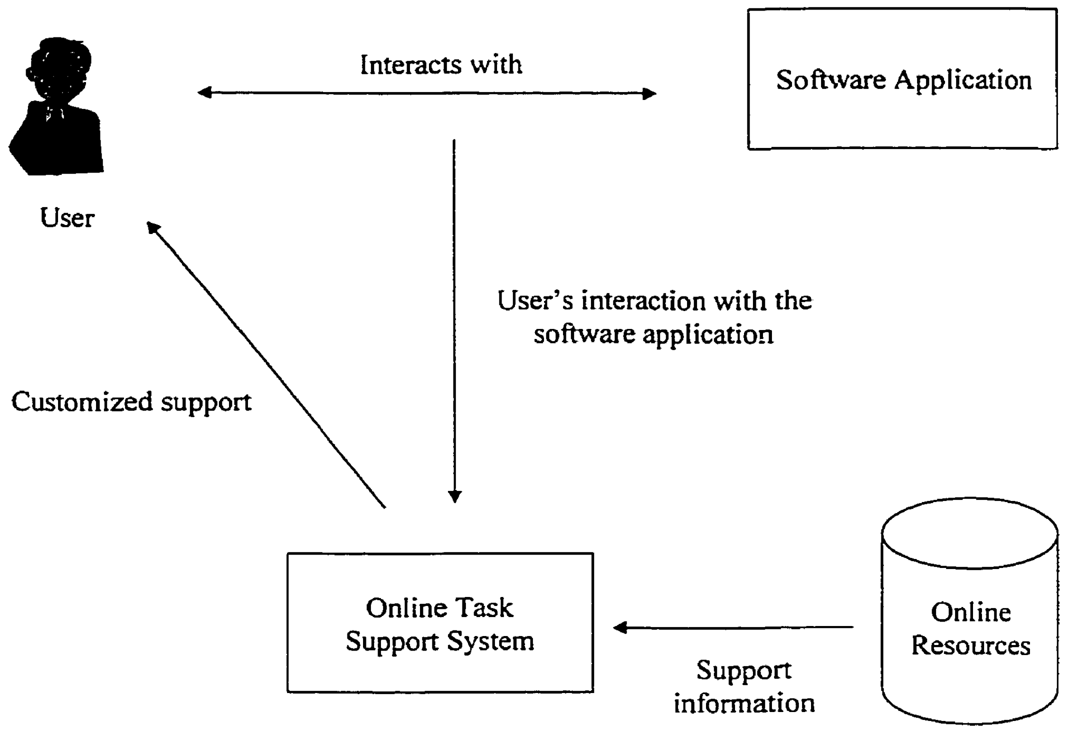


Figure 3-1. Conceptual Schema of the Proposed Task Support System

The user interface of the proposed approach makes use of persistently present windows to display task support. More specifically, task support is displayed in two windows, Subject Window and Detail Window, which are side-by-side to the Task Window where users perform their tasks (Figure 3-2). While a user works on his or her current task, relevant topics are displayed proactively in the Subject Window, without interfering with the user's primary Task Window. The topics are revised continuously according to the user's progress and, more importantly, multiple items of advice for any given situation are displayed in the Subject Window simultaneously. The Detail Window shows the content of each advice, one at a time in the decreasing order of estimated relevance to the task, and the display is updated at short intervals.

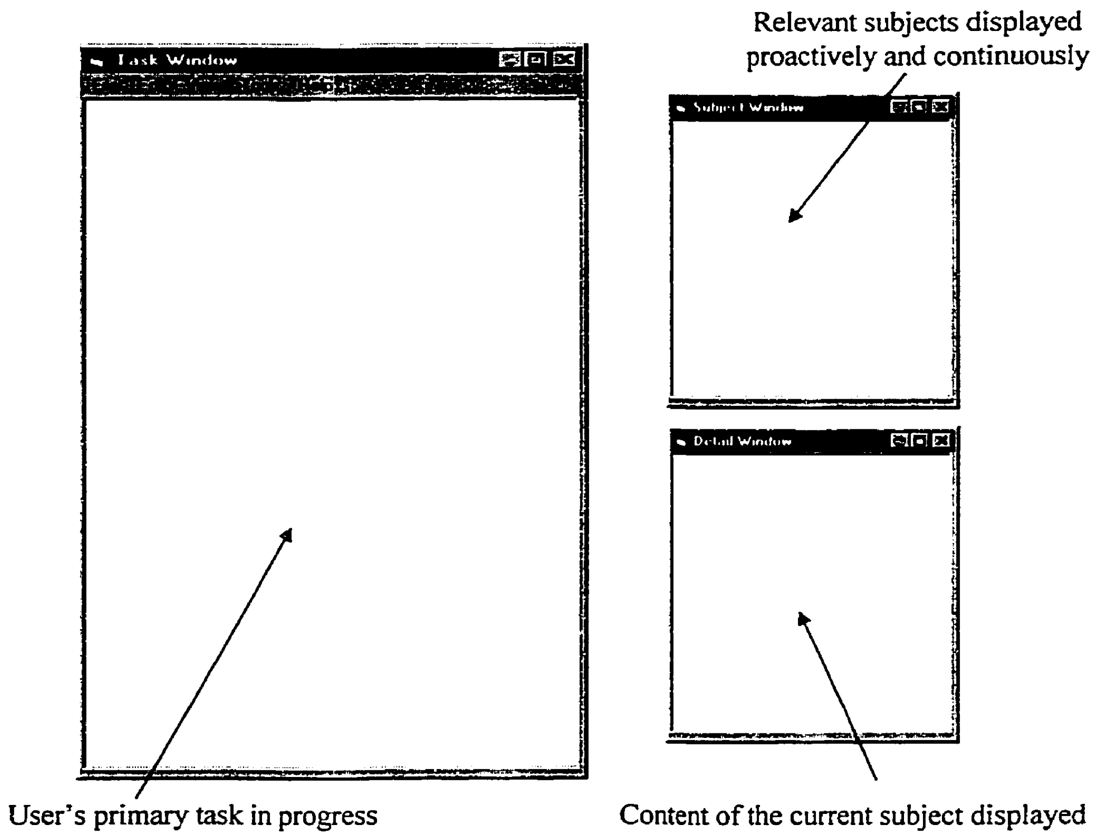

Relevant subjects displayed proactively and continuously
Task Window
Subject Window
Detail Window
User's primary task in progress
Content of the current subject displayed

Figure 3-2. User's View with Task, Subject, and Detail Windows

In order to provide a flexible interaction style, users are given the possibility of closing the task support windows. This option can be disabled by an organization or a user depending on the purpose of the application or the circumstance. However, even if the task support windows are present persistently, users may not be distracted by the advice on the side over the long term, as the literature suggested in Section 2.3.

The above style of user interface seeks to draw synergy between the user-initiated and system-initiated support. It may be considered a middle-ground approach, which is expected to significantly alleviate the inadequacy associated with passive task support discussed in Section 2.2.

The proposed interface style based on the use of multiple and persistently present windows also addresses the three major difficulties associated with conventional proactive task support identified in Section 2.2. First, while it is still desirable to be able to infer a user's intended task as accurately as possible, this approach allows for some margin for error in the inference because nothing is imposed upon the user. Irrelevant and erroneous help can be simply ignored. Therefore, it may be acceptable to identify several likely possibilities and provide task support sequentially, instead of just the correct user intention. Users may be more tolerant because not only is the help unobtrusive but it also requires no search effort, e.g., burdening the user with numerous questions and then providing not necessarily useful information. The implication is that, by relaxing the objective function, the proposed approach bypasses the need for complicated algorithms aimed at accurate detection of users' intention, which is a bottleneck for proactive task support. Second, the challenge of providing support at the right time is less of an issue since the support is given continuously and updated at short intervals according to the task progress. In other words, the idea of just-in-time task support is essentially operationalized by the continuous display and update. Finally, users have full control over the task support, with the options of either following up on the task support or ignoring it completely. In short, this approach has the advantage of providing proactive and yet unobtrusive support to users.

3.3 An Architectural Model

Following the description of the interaction style of the proposed approach presented in the previous section, this section focuses on how to implement the task support engine of the proposed approach, and proposes an implementation strategy that can be applied to diagnostic tasks. The proposed architectural model has four major components. They are the *Advisory Module*, *Knowledge Network (KNet)*, *Performance-Tracking Module*, and *User Model* (Figure 3-3).

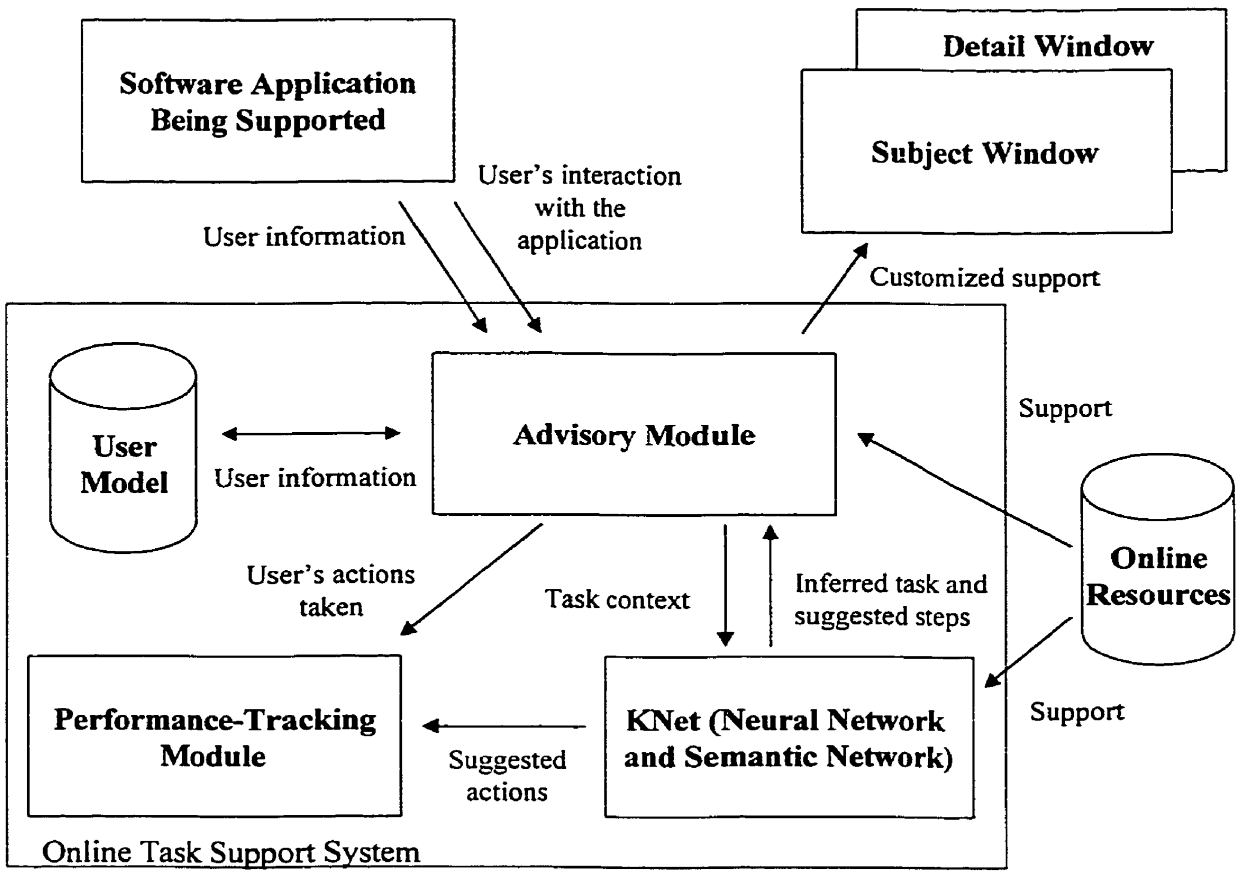

Software Application Being Supported
Detail Window
Subject Window
User's interaction with the application
User information
Customized support
User Model
User information
Advisory Module
Support
Online Resources
User's actions taken
Task context
Inferred task and suggested steps
Performance-Tracking Module
Suggested actions
KNet (Neural Network and Semantic Network)
Support
Online Task Support System

Figure 3-3. The Proposed Architectural Model

The online task support system works in the following manner: The *Advisory Module* captures the user's interaction with the host application. It maps the user's current step onto the online resources. If the *Advisory Module* recognizes a step, it will look up the user's skill level with regard to the step in the *User Model* and customize the advice to be displayed. The *Advisory Module* also sends the recognized steps to the *Knowledge Network* (*KNet*) which consists of a neural network and semantic network. The neural network infers the user's task (i.e., the likely problem in diagnostic tasks) based on the recognized completed steps as inputs, and suggests next steps. The semantic network identifies other steps which are closely associated with the current step. Then, the *Advisory Module* retrieves relevant task support based on the inferred task and suggested steps. The *Advisory Module* and *KNet* also pass the user's actions taken and the suggested actions respectively to the *Performance-Tracking Module*. This information is analyzed to provide an ongoing performance evaluation of the task support system.

Each of the four components is described in detail herein. The central component of the proposed approach is the *Knowledge Network (KNet)* as shown in Figure 3-4[4]. It consists of a neural network and semantic network. The *KNet* can be useful in supporting a diagnostic task in three ways: (1) suggesting likely problems based on already confirmed symptoms, and additional symptoms to be confirmed, (2) identifying symptoms that are commonly associated with the ones that are already confirmed, and (3) identifying potentially relevant symptoms in subsequent steps. It will be shown later that the first type of support is through the neural network and I-value, the second type through the semantic network, and the third type through the integration of the neural network and semantic network by the I-value.

[4] For simplicity, the hidden layer of the neural network is not shown. Furthermore, the link "Contribute to" can also mean leading to the identification of a problem, not necessarily causing the problem.

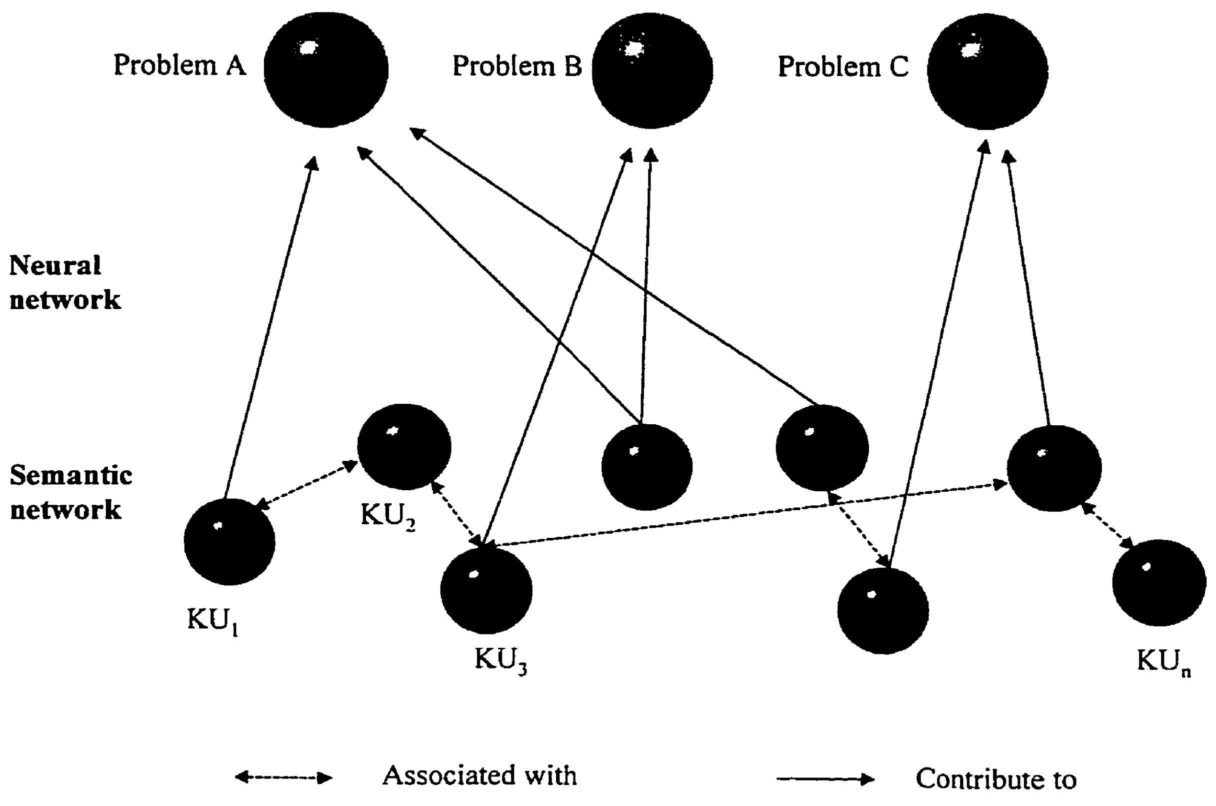


Figure 3-4. The Knowledge Network (*KNet*)

Online resources can be decomposed into basic elements labeled knowledge units (KUs). For example, in medical diagnosis, KUs can start with symptoms such as fever, pain, and vomiting; the contents of a KU in this case include the definition, medical textbook type of domain knowledge, and questions that a nurse should ask a patient.

As mentioned earlier, the neural network is used to infer the user's task and suggest potential subsequent steps. Neural networks are well suited for inferring users' tasks because of their modeling power, robustness, ability to recognize patterns and to work with incomplete or noisy data, as discussed in Section 2.3. The stream of completed steps (i.e., recognized KUs) by a user is the input to the neural network, whereas an inferred task is the output form the network. The mapping between the input and output can be obtained through task analysis or learned by the neural network from past cases. Once a task is inferred, the neural network can suggest potential subsequent steps in decreasing order of estimated relevance to the task. The order is determined by the relevance measure using I-value based on Equation 2 in Section 2.3. The detail of how to construct a neural network from past cases is given in the following chapter.

KUs are connected to form a semantic network, as part of the *KNet*. Semantic networks have been used for a long time as a knowledge representation method. They usually consist of nodes, which represent objects, actions, or events, and links, which represent the relationship between nodes (Barr & Feigenbaum, 1981; Charniak & McDermott, 1985). In this research, the semantic network is used to model the association among the KUs only. The connection link (or weight) between two nodes reflects their relevance with each other. However, it does not limit the generalizability of the semantic network to represent more comprehensive and contextualized semantic information in a task domain.

The semantic network is used to identify steps that are closely associated with the current steps. In order to suggest alternative paths or subsequent steps for a given path, it can be useful to identify commonly associated steps. For instance, in medical diagnosis, nausea and vomiting are closely associated with each other although they may be symptoms of different diseases. Once one of them is identified, the identification or elimination of the other one through the semantic network can greatly help the diagnosis.

The semantic network works in the following manner: There are two ways to activate the semantic network. First, a user's most recent step is mapped onto KUs. If there is a corresponding one, activation is spread with an activation energy of 1 through the network from this mapped KU. The KUs connected nearby may be activated if the product of the activation energy and weight is larger than a predefined threshold. The activation mechanism is based on Jennings and Higuchi's (1993) approach, and is given in Figure 3-5. Thus, through the semantic network, task support relevant to the current step can be identified. With the semantic network, control over the amount of relevant support can be adjusted as the initial activation energy diminishes and attenuates the further it is propagated from its starting point. If the initial activation energy is high, more relevant KUs will be activated. If the initial activation energy is low, fewer relevant KUs will be activated.

1. Map the KU to *KNet*
2. Fire the mapped KU with an activation energy (e.g., 1 for recognized users' step or normalized I-value) to all its connected KUs
3. Calculate and accumulate the energy level (i.e., activation energy × weight) for each connected KUs
4. Fire and spread the accumulated activation energy to all connected KUs if the accumulated activation energy level is greater than or equal to the predefined threshold
5. Repeat steps 3 to 5 until no KUs in the network have accumulated activation energy levels greater than or equal to the predefined threshold
6. Collect the set of fired (or activated) KUs

Figure 3-5. The Procedure to Control Spreading Activation

Second, the semantic network can also be activated by the I-value from the neural network, based on Equation 2 in Section 2.3, which acts as the activation energy from the neural network "feeding" to the semantic network. The reason for feeding the I-value to the semantic network is to identify subsequent steps and their closely associated steps. The neural network and the semantic network are integrated through the relevance measure using I-value. The activation energy is normalized and coded as a value between 0 and 1 depending on the output from the neural network. The highest output value from the neural network has an activation energy of 1 feeding to the semantic network whereas the lowest output value is 0. The activation energy is distributed and normalized among the subsequent KUs based on their I-values. Similarly, the second highest output value from the neural network has an activation energy of less than 1 depending on its output value. The same activation mechanism outlined in Figure 3-5 is used. The higher the relevance between the KU and the task, the higher the I-value and the higher the chance the KU is activated. A hypothetical example based on the *KNet* is given in the following section.

The *Advisory Module* captures the user's task progress (interaction with a host application) in the Task Window. It maps users' steps to KUs in *KNet*, which identifies the user's current task and subsequent steps. It also compiles the suggested information into a list for display in the Subject Window, and this list is revised continuously according to the user's progress. The content of each online resource is displayed in the

Detail Window for a predefined period of time, followed by the next one in decreasing order of ranked relevance.

The *Performance-Tracking Module* provides an ongoing tracking and evaluation of the performance of the task support system. It is particularly important for any proactive task support, as it monitors whether advice suggested by the system is followed by users. If a piece of advice is repeatedly ignored, it should not be offered proactively in the future. If users always respond to a given situation in a certain way, which is different from the system's advice, the performance-tracking module should help the system adjust and learn. For example, when advice is compiled and displayed in the Subject and Detail Windows, but it has no effect for several times, this scenario should be logged and a new rule based on the user's behavior may be generated to overrule the previous advice. When a similar situation occurs in the future, the new rule will be used. Furthermore, the performance-tracking module can use this information to evaluate the usefulness of the task support system by accumulating evidence of reliability, i.e., the percentage of advice taken and the number of new rules generated. If the reliability is low or the number of rules is high, it may indicate that the performance of the system has degraded and remedial actions should be taken, e.g., re-training of the neural network with new training samples. Finally, if the task support windows are closed by the user, the performance-tracking module can also be used to determine when to re-open them proactively. Although the task support windows are meant to be present persistently, users may be given the option to close them as discussed in Section 3.2. However, even if the advice windows are closed, the inference continues behind the scene. Once a user's action deviates from the recommended action by the system, the task support windows can be re-opened and stay on the side. Suppose a user chooses to close them again for several times, they should not be re-opened proactively in the future. This information, i.e., a user's consistent rejection of certain task advice, should also be stored in the user model.

The *User Model* is used to customize task support for addressing each user's unique needs. It is part of the mechanism to operationalize the idea of just-enough task support. It is composed of records of the history of each user's experience with the application being supported. A user's skill level with regard to each KU is updated continuously based on his or her usage of the KU based on Vivacqua and Lieberman's approach (2000). For each user, information such as the time a KU was last accessed and the total number of times that the KU has been accessed is captured. Users' preferences (e.g., show multimedia demonstrations and show examples) are also stored. Re-categorization of users' skill level is done by a simple algorithm (see Figure 2-3) similar to Strachan et al.'s approach (2000).

In summary, the proposed middle-ground approach is expected to alleviate the difficulties associated with user-initiated and system-initiated task support. This approach prescribes that a proactive task support system should behave like an intelligent assistant, constantly observing its master's work, trying to figure out how it can help, and making suggestions whenever it finds something potentially useful. The suggested information, which is revised continuously according to the user's task progress, is displayed in separate windows, without interfering with the user's task window. In short, the proposed task support approach provides task support proactively, unobtrusively, and continuously.

3.4 A Hypothetical Example

Whereas a prototype is presented in Chapter 4 to illustrate the proposed approach in detail, this section uses a hypothetical example to demonstrate the *KNet*, a central component in the architectural model. The example deals with trouble-shooting personal computer (PC) problems, a typical diagnostic task. The scope of this example is limited to five common PC problems associated with *Power supply*, *Video card*, *Memory*, *Hard drive*, and *Monitor*. For instance, a *Power supply* problem may exhibit symptoms such as *no display*, *system lock-up occasionally*, *no fan noise*, and *error code occasionally.*

The *KNet* of this example is shown in Figure 3-6[5] with the neural network on top, semantic network at the bottom. They are described in detail in the following paragraphs. The output of the neural network corresponds to the five PC problems, whereas the input consists of all of the known symptoms. If data is available, e.g., from archived files of a corporate helpdesk, the neural network can be trained. I-values between the symptoms (input) and PC problems (output) can be calculated based on Equation 2 in Section 2.3 (Tables 3-1).

[5] For simplicity, the hidden layer of the neural network is not shown and the detail of how to construct a neural network is illustrated in the following chapter. Furthermore, the link "Contribute to" can also mean leading to the identification of a problem, not necessarily causing the problem.

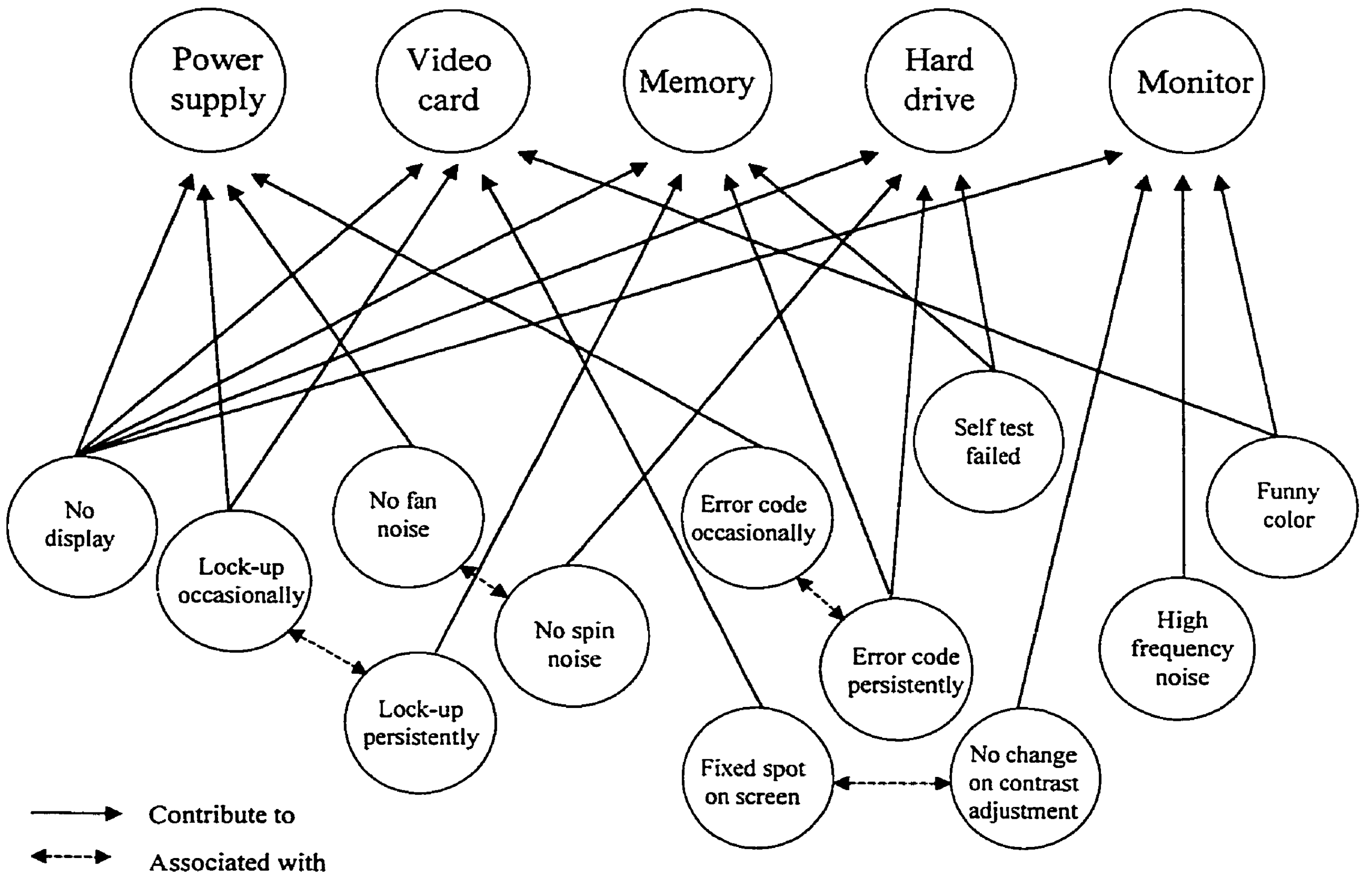


Figure 3-6. The *KNet* for PC Diagnosis

PC Problems	Symptoms	I-values
Power supply	No display	0.10
	Lock-up occasionally	0.25
	No fan noise	0.25
	Error code occasionally	0.40
Video card	No display	0.10
	Lock-up occasionally	0.25
	Funny color	0.25
	Fixed spot on screen	0.40

Table 3-1. Sample I-values between Symptoms and Problems (hypothetical data)

The neural network can suggest the likely problems based on already confirmed symptoms and additional symptoms to be confirmed. Suppose a PC exhibits a blank display and locks up occasionally, symptoms *no display* and *lock-up occasionally* are both turned on with 1 (cf. Baxt, 1990). Assume the neural network with this given input infers the most likely problem to be *Power supply* with a probability of 0.5 and the second most likely problem to be *Video card* with a probability of 0.4, further questions can be formulated to determine whether the problem is caused by the *Power supply* or *Video card.* From Table 3-1, symptom *error code occasionally* has the highest I-value among the symptoms of *Power supply.* This indicates that *error code occasionally* is strongly related to the problem of *Power supply.* Therefore, the next question can be formulated based on this symptom. The bold part in Figure 3-7 shows the activated portion of the *KNet* after the likely problems are inferred.

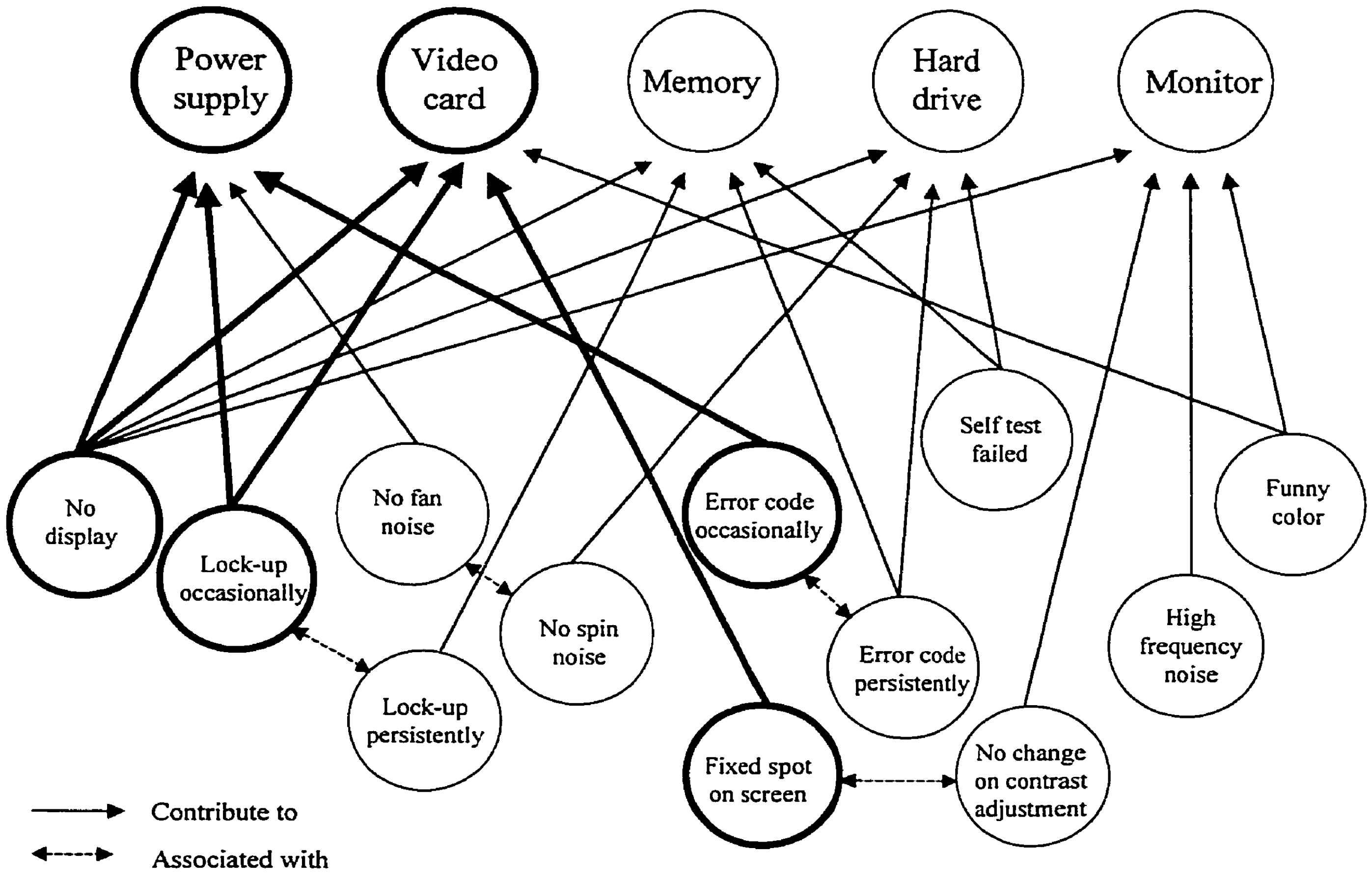


Figure 3-7. Activated Portion of the *KNet* based on Initial Input

For the semantic network part in this example, only an *associated-with* relationship is used to represent certain symptoms are correlated. The connection strength (i.e., weight) between related symptoms in the semantic network can be determined through either statistical methods, e.g., correlation matrix, or heuristics (Table 3-2).

Symptoms	**Related symptoms**	**Weight**
No fan noise	No spin noise	0.50
Fixed spot on screen	No change on contrast adjustment	0.50
Error code occasionally	Error code persistently	0.80
Lock-up occasionally	Lock-up persistently	0.80

Table 3-2. Sample Weights between Related Symptoms (hypothetical data)

The semantic network can identify other symptoms closely associated with the ones already confirmed. For the same scenario discussed in the preceding paragraphs, since the symptom *lock-up occasionally* is turned on with 1, this value (or activation energy) is spread to its nearby symptoms through the semantic network. Its connected symptoms will be activated if the product of the activation energy and weight is greater than or equal to its threshold, say 0.5. In this case, the product is 0.8 given that the activation energy is 1 and the weight between symptoms *lock-up occasionally* and *lock-up persistently* is 0.8. As a result, the symptom *lock-up persistently* is also activated. This information may be helpful for helpdesk personnel to collect, thus should be brought to their attention by a task support system. The bold part in Figure 3-8 shows the activated portion of the *KNet*.

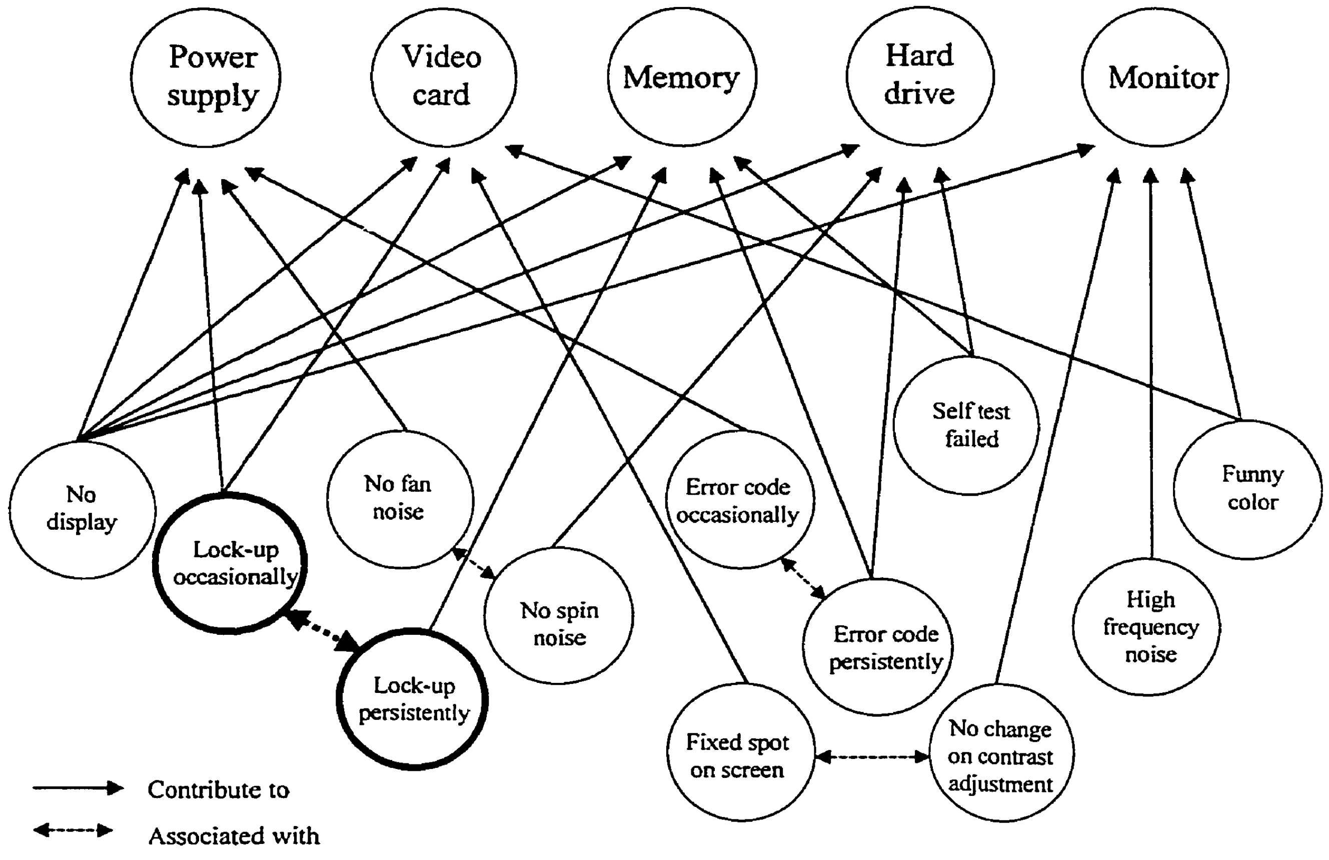


Figure 3-8. Activated Portion of the *KNet* based on Closely Associated Symptoms

The I-value also acts as the activation energy to feed from the neural network to the semantic network to identify potential subsequent questions to ask. Assume the output from the neural network is 0.5 for *Power supply* (i.e., the probability of having *Power supply* problem is 0.5) and 0.4 for *Video card*, then the normalized activation energy feeding from *Power supply* will be 1 and from *Video card* will be 0.8 (1 × 0.4 / 0.5). The activation energy from *Power supply* is distributed to *error code occasionally* and *no fan noise* based on their I-values. In this case, the activation energy for *error code occasionally* is 0.62 (1 × 0.4 / (0.4 + 0.25)) and for *no fan noise* is 0.38 (1 × 0.25 / (0.4 +

0.25)). Similarly, the activation energy of *fixed spot on screen* is 0.5 (0.8 × 0.4 / (0.4 + 0.25)) and *funny color* is 0.3 (0.8 × 0.4 / (0.4 + 0.25)). For these activation energy levels, and the corresponding weights from Table 3-2, the only activated symptoms are *error code persistently* and *no change on contrast adjustment*. In this way, subsequent questions can be formulated to influence the identification of other potentially related symptoms. The bold part in Figure 3-9 shows the activated portion after the I-values are fed to the semantic network.

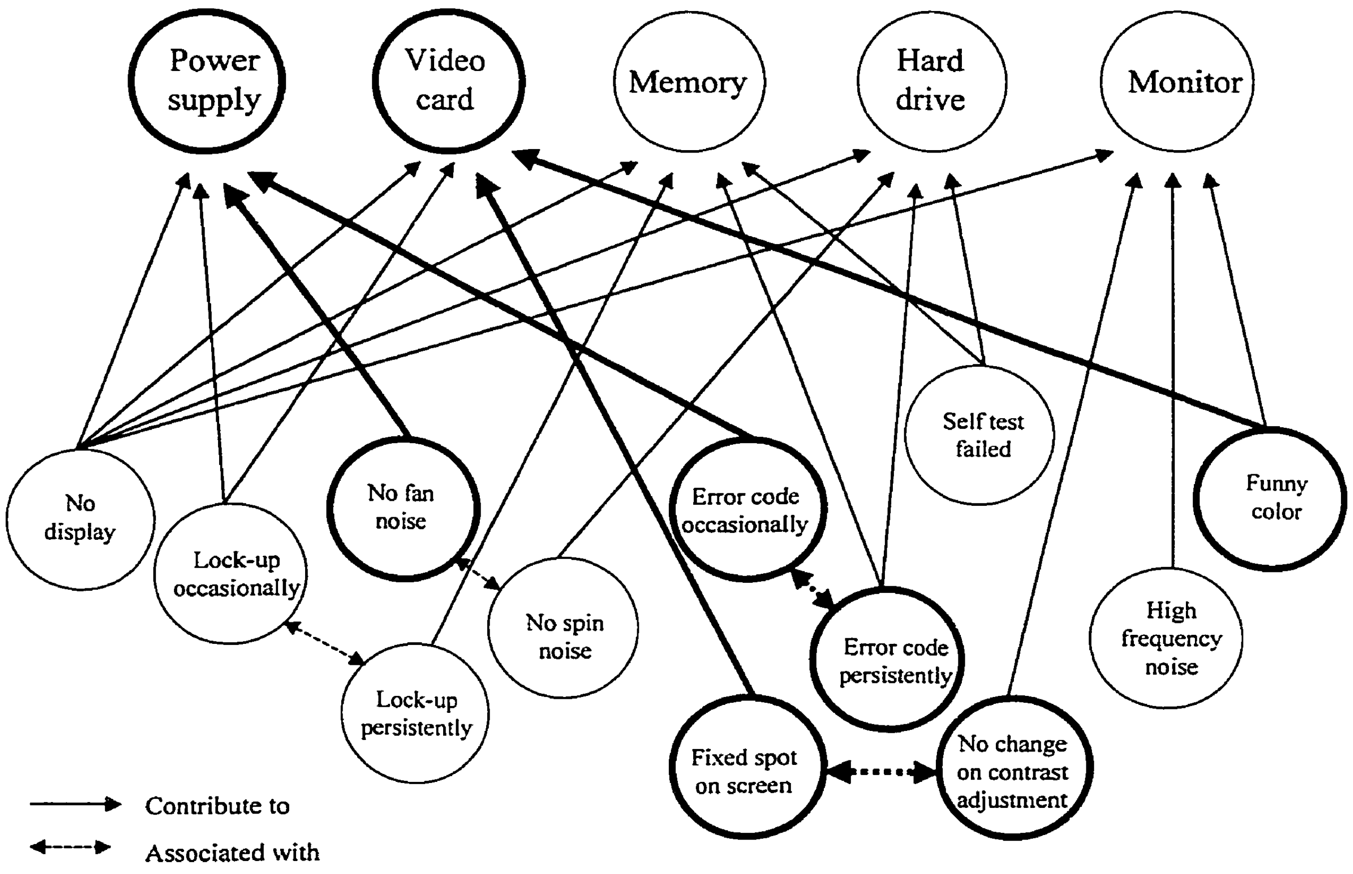


Figure 3-9. Activated Portion of the *KNet* using I-values as Activation Energies

In conclusion, this chapter proposes an approach to developing a task support system. It uses persistently present windows to display task support side-by-side to users' task window as the user interface and a *KNet*, i.e., a neural network and semantic network, to retrieve relevant task support in the task support engine. The latter is illustrated with a hypothetical example. To demonstrate the feasibility of the proposed approach, a prototype is described in the following chapter.

4. Prototyping

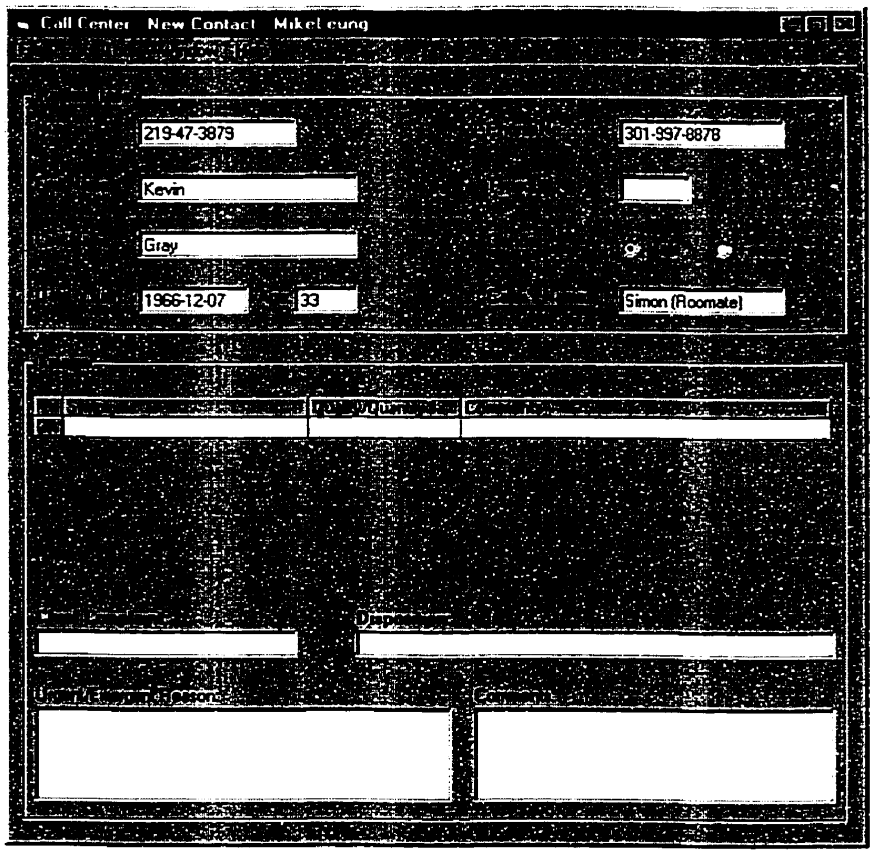
Based on the architectural model and development methodology proposed in the previous chapter, a prototype, called Telephone Triage Assistant (TTA), was developed to assist novice nurses in identifying the nature of patients' diseases and the appropriate treatment. This chapter documents the prototyping work by describing the background, conceptual design, system architecture, and functionality.

4.1 Background

Telephone triage is both challenging and critical. Nurses in the call center need to listen to, record, and interpret patients' symptoms, then make an assessment about the nature of the diseases, and recommend an appropriate treatment accordingly. The quicker the nurses can identify the disease by asking the right questions, the quicker the proper care can be suggested to the patient. Sometimes, in a life-threatening situation where time is limited, it may help save a life (Wheeler & Windt, 1993). It is, therefore, important for the nurses to (1) have access to relevant and up-to-date domain knowledge at the point of need, and (2) provide consistent and accurate answers and responses (Grossman, 1999; Handysides, 1995).

ABC Medical Call Center (a pseudonym) in Maryland is part of one of the largest health care organizations in the United States with over 70 million insured individuals nationwide. The call center receives more than 500 calls a day. Nurses use a Disease Management System (DMS) to document task information whilst interviewing patients.

Since the researcher was not given the source code of the DMS, a prototype similar to the DMS was developed. The major function of the prototyped DMS was to record the patient's personal details, symptoms, chief complaint, and disposition (Figure 4-1).



219-47-3879

301-997-8878

Kevin

Gray

1966-12-07

33

Simon (Roomate)

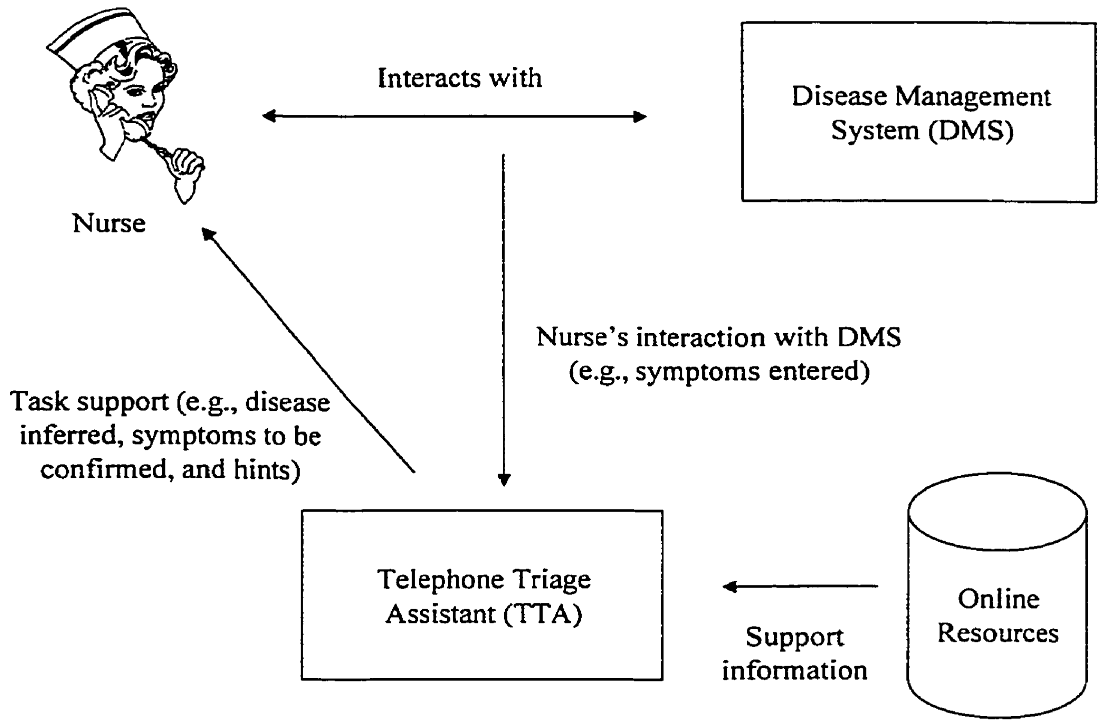
Figure 4-1. The Prototyped Diseases Management System (DMS)

The decision to involve ABC as the sponsoring organization was based on several factors. First, it was a convenient choice because the researcher used to work in the Software Development Department of ABC, and access to data and medical knowledge was secured. Second, ABC had been expanding lately, and many new nurses joined the organization. The management of ABC was concerned about the lack of experience of the new nurses and was studying various ways to ramp up their performance. One possible way identified by the management was to re-develop their DMS by adding more "intelligence" and functions. ABC expressed interest in this research and intent to support it. Third, the new nurses at ABC were seen as a pool of potential subjects to participate in a usability study because the target users of TTA were novice nurses.

A prototyped Telephone Triage Assistant (TTA) was developed to support the triage task, i.e., identifying the nature of diseases and the need for appropriate care by confirming the presence of one or more symptoms with a patient. TTA influences a nurse's line of questioning through three types of task support: (1) showing diseases inferred by TTA, based on already confirmed symptoms, and additional symptoms to be confirmed to complete the identification of the disease, (2) showing symptoms that are commonly associated with the ones already confirmed, along with background knowledge relevant to the symptoms already confirmed by the nurse, and (3) showing potentially relevant symptoms in subsequent steps. In the telephone triage domain, KUs typically are symptoms such as *fever*, *pain*, and *vomiting*. The scope of TTA is modest and limited to recognizing and supporting nine presentations of Abdominal Problem: *Aortic Aneurysm*, *Appendicitis*, *Bowel Obstruction*, *Cholecystitis*, *Cholelithiasis*, *Constipation*, *Pancreatitis*, *Peritonitis*, and *Renal Calculi*. Abdominal Problem was chosen because it is one of the most common complaints which patients have in ABC.

4.2 System Architecture

Figure 4-2 shows a conceptual schema of TTA. TTA behaves like an intelligent assistant, which constantly observes the nurse's interaction with DMS, and tries to figure out how it can help by finding and displaying potentially useful information.



Nurse
Interacts with
Disease Management System (DMS)
Nurse's interaction with DMS (e.g., symptoms entered)
Task support (e.g., disease inferred, symptoms to be confirmed, and hints)
Telephone Triage Assistant (TTA)
Support information
Online Resources

Figure 4-2. Conceptual Schema of TTA

TTA consists of three major components (Figure 4-3): (1) the advisory module, which monitors symptoms entered into DMS, and then compiles and displays task support information, (2) the *KNet* consisting of a neural network for inferring the likely diseases and suggesting additional symptoms to be confirmed, and a semantic network for suggesting other commonly associated symptoms and showing relevant domain knowledge based on the desired level of relevance, and (3) the user model which records each user's task experience. TTA's architecture is based on the proposed architectural model described in Figure 3-3 except the absence of the performance-tracking module. The reason is that TTA is not expected for routine use or longitudinal usability study. TTA has two windows named Diseases/Symptoms Window (DSW) and Detailed Description Window (DDW) which are side-by-side to the task window of DMS.

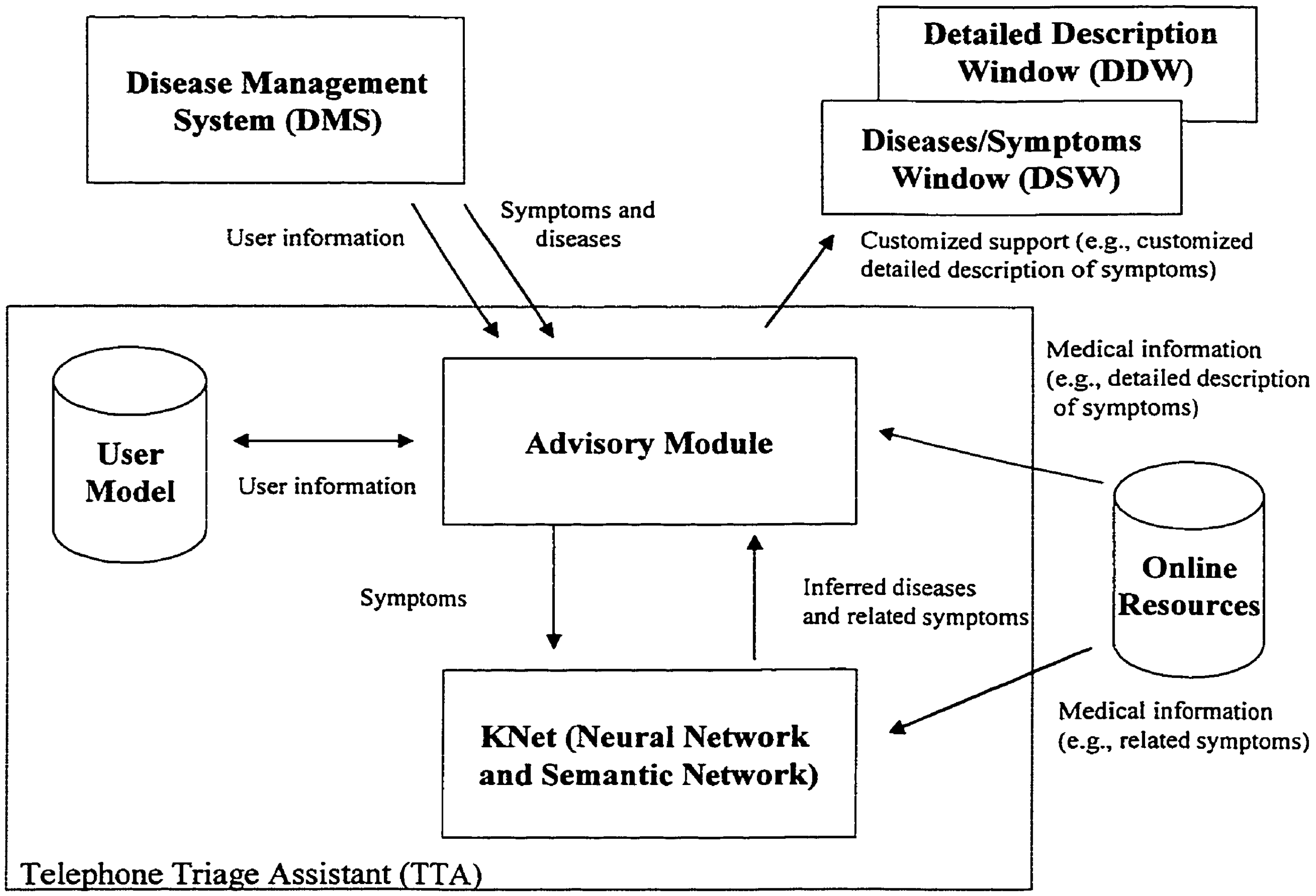

Disease Management System (DMS)
Detailed Description Window (DDW)
Diseases/Symptoms Window (DSW)
User information
Symptoms and diseases
Customized support (e.g., customized detailed description of symptoms)
Medical information (e.g., detailed description of symptoms)
User Model
User information
Advisory Module
Online Resources
Symptoms
Inferred diseases and related symptoms
KNet (Neural Network and Semantic Network)
Medical information (e.g., related symptoms)
Telephone Triage Assistant (TTA)

Figure 4-3. The Architecture of TTA

4.2.1 The Advisory Module

The advisory module observes the task in progress but does not interfere with the DMS. Symptoms entered by the nurse are captured in an input queue for the neural network and the semantic network. It is also responsible for determining the appropriate level of detail at which information will be displayed based on user information (e.g., skill level). Ideally, less detailed information for novice nurses includes some routine questions such as region and intensity of pain whereas highly detailed information for experienced nurses includes more specialized information in medicine and surgery such as indications and contraindications of medicine. However, in this prototype, the less detailed information is limited to definition and formation of pain whereas the highly detailed information is limited to follow-up questions. For instance, if a nurse has entered *Abdominal Pain* and she is a novice recognized by the system through her user ID, the low level of detail of *Abdominal Pain* (e.g., definition and formation of pain) will be displayed. In a similar situation, if the nurse is at the expert level, a high level of detail (e.g., follow-up questions) will be displayed only.

4.2.2 The Neural Network

The development of the neural network in the *KNet* has gone through the following typical steps: (1) identifying the inputs and outputs of the network, (2) determining an architecture for the network, (3) training the network, and (4) testing the generalization and validity of the network.

Inputs and outputs. Input to the neural network consists of the symptoms confirmed through the ongoing telephone conversation. There are 33 possible symptoms (input) associated with the nine presentations (output) of Abdominal Problem. A node is turned on with a value of "1" or off with a value of "0" (cf. Baxt, 1990). For example, *Cholelithiasis* is associated with at least *abdominal pain*, *back pain*, *nausea*, *vomiting*, *right pain*, *dyspepsia*, and *jaundice* as symptoms. Thus, if these input nodes are on, the output of *Cholelithiasis* will be on.

Architecture. For simplicity, a feedforward, fully-connected, backpropagation network is used (Simpson, 1990). The architecture of the neural network is determined for the most part by the given 33 inputs and nine outputs. The next step is to determine the number of hidden nodes in the hidden layer between the input layer and the output layer. Few guidelines exist for how many nodes to include in this hidden layer (Simpson, 1990; Smith, 1993). A general practice is to train the neural network with different numbers of hidden nodes and measure their performance, and finally choose the number of hidden nodes that yields a relatively good performance with the smallest number of hidden nodes (Smith, 1993). It will be shown in the following section that seven hidden nodes were used. Figure 4-4 shows the architecture.

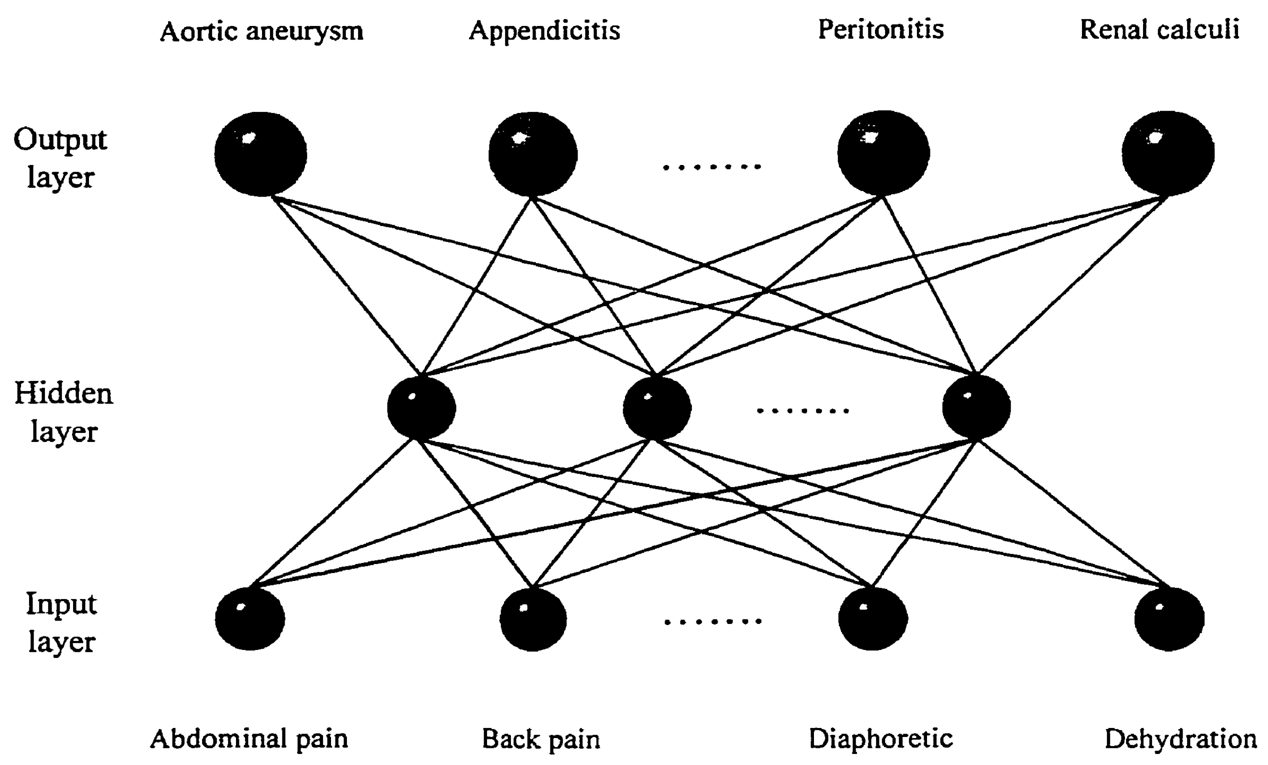


Figure 4-4. The Neural Network Architecture Used in TTA

Training. In ABC Medical Call Center, senior nurses are required to follow up with most of the cases by calling the patients, doctors, or hospitals to make sure the diagnosis was accurate and the recommended treatment was appropriate. Once the case has been audited, it is archived by the DMS. In total, 1080 cases of past audited triage tasks involving Abdominal Problem were available. They were collected by the ABC Medical Call Center over a six month period. The data set was broken down to three sets following Masters' (1993) recommendation[6]: Eight hundred and eighty two cases were used for training the neural network, 100 cases were used for validation, and the remaining 98 cases were held back for testing. An example of the training set is shown in Table 4-1.

	Case	1	2	...	882
Input	Abdominal pain	1	1	...	1
	Back pain	1	1	...	0
	Nausea	1	1	...	1
	Vomiting	1	1	...	:
	Fever	1	0	...	:
	Rigidity	1	0	...	:
	Jaundice	0	1	...	:
	Right pain	0	1	...	:
	Dyspepsia	0	1	...	:
	Tenderness	1	0	...	:
	...	0	0	...	:
Output	Aortic aneurysm	0	0	...	:
	Appendicitis	0	0	...	:
	Bowel obstruction	0	0	...	:
	Cholecystitis	0	0	...	:
	Cholelithiasis	0	1	...	:
	Constipation	0	0	...	:
	Pancreatitis	0	0	...	:
	Peritonitis	1	0	...	:
	Renal calculi	0	0	...	:

Table 4-1. The Training Data Set for the Neural Network

Table 4-2 shows the training result with regard to varying numbers of hidden nodes. Seven hidden nodes were considered optimal because of the relatively good performance of the network in the training set with the smallest possible number of hidden nodes. In

[6] Masters recommends that the number of training cases should be at least two times the number of connections in the network. In this prototype, there were 294 connections with 33 input, seven hidden, and nine output nodes. Consequently, 882 (3 x 294) training cases were used.

training the neural network, the learning rate and the momentum values were both arbitrarily set at 0.3. Every 1,000 iterations, the performance of the network was checked using the validation set to avoid overfitting (i.e., the neural network model fits too well with the training data without generalization to new data). It took 10,000 iterations for the network to achieve best performance on the validation set (Figure 4-5).

Number of hidden nodes	**Iterations**	**Errors for training set**
2	failed to convergent	--
3	62825	0.00152
4	26680	0.00146
5	15821	0.00141
6	13800	0.00133
7	10430	0.00106
8	11552	0.00104
9	8620	0.00102
10	7260	0.00101
12	3354	0.00092
15	2073	0.00091
20	1090	0.00091
25	877	0.00091

Table 4-2. Performance with Different Number of Hidden Nodes

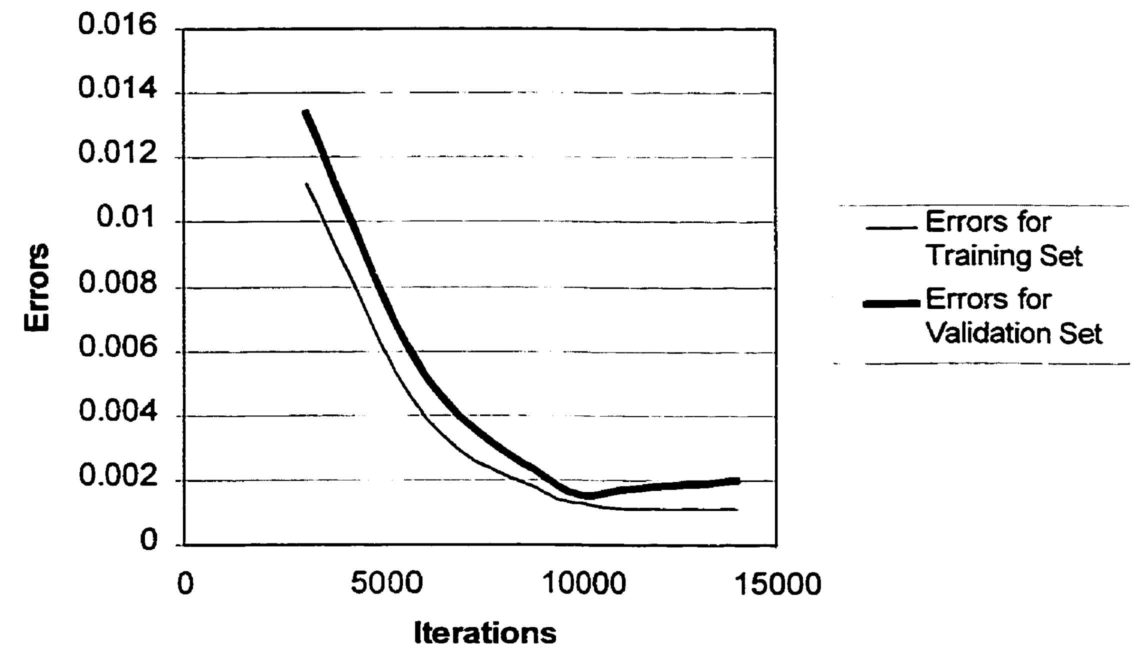


Figure 4-5. Number of Training Iterations Required without Overfitting

Testing. The final step is to test the generalization of the neural network, i.e., its ability to infer the proper output when incomplete input is given. This is especially important for proactive task support because the idea is to influence the nurse's line of questioning. Once the input is complete, i.e., a nurse has probably independently confirmed the presence of all symptoms to conclude about a particular disease, thus, there is little or no need for support. On the other hand, if there is too little input, e.g., if only one input is known for each test case, it is not sufficient to make a meaningful inference. Therefore, the inference starts when two or more inputs are known (i.e., two or more input nodes are turned on with a value of "1") while the others are unknown. Table 4-3 shows some results using the testing sample (98 cases).

Number of known input	**Percentage of correct identification of disease**
2	61.2
3	69.4
4	82.7
5	87.8
6	90.8

Table 4-3. Accuracy of inference for Incomplete Input

4.2.3 The Relationship between Diseases and Symptoms

Most diseases have many different symptoms. To assist novice nurses in identifying a disease quickly, it is important to suggest the right questions to ask about the critical or indicative symptoms, as is the case with most diagnostic tasks. This task can be facilitated by a model outlined in Section 2.3. The model is based on the structural property of the trained neural network of *KNet* for determining the relationship between a disease (output) and symptoms (input). For instance, *abdominal pain*, *vomiting*, *fever*, *constipation*, *hypotension*, and *hyperactive bowel sound* are all symptoms of *Bowel Obstruction.* Applying Equation 2 in Section 2.3 on the training sample, the average I-value of each symptom was calculated and shown in Table 4-4.

Symptoms of Bowel Obstruction	**I-value**
Constipation	0.21
Hyperactive bowel sound	0.20
Abdominal pain	0.17
Vomiting	0.17
Hypotension	0.16
Fever	0.09

Table 4-4. I-value of Symptoms of Bowel Obstruction

Suppose in a situation that *Bowel Obstruction* is inferred as a likely disease and *abdominal pain* and *fever* have been confirmed as symptoms. Then, *constipation* is the next symptom with the highest I-value that should be confirmed with the patient. This information may help nurses formulate relevant questions and identify the disease in a timely manner. The I-values of symptoms on the studied diseases are shown in parentheses in Table 4-5.

Diseases	**Symptoms**
Aortic aneurysm	Pulsatile abdominal mass (0.21), Shock (0.20), Mottling (0.20), Back pain (0.16), Flank pain (0.12), Abdominal pain (0.11)
Appendicitis	Tachycardia (0.23), Anorexia (0.17), Peritoneal (0.13), Abdominal pain (0.12), Right low pain (0.10), Pallor (0.10), Fever (0.07), Nausea (0.06)
Bowel obstruction	Constipation (0.21), Hyperactive bowel sounds (0.20), Abdominal pain (0.17), Vomiting (0.17), Hypotension (0.16), Fever (0.09)
Cholecystitis	Tachypnea (0.18), Right upper pain (0.14), Tachycardia (0.13), Back pain (0.12), Jaundice (0.11), Fever (0.10), Abdominal pain (0.08), Vomiting (0.07), Nausea (0.06)
Cholelithiasis	Dyspepsia (0.23), Right pain (0.19), Vomiting (0.12), Abdominal pain (0.11), Back pain (0.11), Jaundice (0.08), Nausea (0.06)
Constipation	Headache (0.25), Restlessness (0.25), Discomfort (0.16), Fatigue (0.14), Back pain (0.11), Abdominal pain (0.04)
Pancreatitis	Tachypnea (0.21), Upper pain (0.17), Tachycardia (0.15), Vomiting (0.11), Fever (0.11), Abdominal pain (0.10), Hypotension (0.09), Nausea (0.05)
Peritonitis	Tenderness (0.26), Vomiting (0.17), Fever (0.15), Rigidity (0.15), Back pain (0.12), Abdominal pain (0.10), Nausea (0.05)
Renal calculi	Diaphoretic (0.25), Lower pain (0.17), Dehydration (0.17), Pale (0.17), Vomiting (0.09), Nausea (0.07), Abdominal pain (0.06)

Table 4-5. I-values Between Symptoms and Diseases

4.2.4 The Semantic Network

The semantic network of *KNet* used in TTA consists of a set of KU nodes with certain assigned threshold energy levels. Each link connecting two adjacent KU nodes has a weight or connecting strength. The higher the value of the weight, the greater the relevance between the two KUs. For example, *nausea* is closely related to *vomiting* as they represent different stages in ejecting the contents of the stomach through the mouth. The individual weight is assigned based on the International Classification of Disease (ICD-9-CM) code[7] together with the help from a senior nurse in ABC. For instance, both *nausea* and *vomiting* are grouped with the ICD-9-CM code of 787.0 indicating that they are commonly associated with each other. In this case, 0.8 was assigned as the weight between *nausea* and *vomiting*. In some cases, according to the senior nurse, patients are confused about describing their pain such as *back pain* and *flank pain*, which they are not clinically related to each other. In this case, 0.5 was assigned to the weight between *back pain* and *flank pain*. Similarly, all the weights among related symptoms in the semantic network can be assigned in this manner.

The semantic network works in the following manner: The initial activation energy is assigned a value of either "1" or "0" to indicate whether the user wants to see commonly associated symptoms or not. This is done by adjusting the slider on the DSW and will be illustrated in a later section. Whether a node is active or not depends on the activation energy and the weight. The node will be active if the result of multiplying the activation energy by the weight is greater than or equal to the threshold (0.5). Similarly, the node will be inactive if the result of multiplying the activation energy by the weight is less than the threshold (0.5). If the weight of the connection between two nodes is high (i.e., they are commonly associated symptoms) and one of them is activated, the chance is high that the other node is also active. On the other hand, if the activation energy is high (i.e., the

[7] International Classification of Disease, 9th revision, was published by the World Health Organization (WHO). ICD-9-CM (Clinical Modification) is a classification system that groups related disease and symptom entities for the reporting of statistical information. It can be subscribed and accessed at http://www.icd-9-cm.org.

nurse has selected to see more commonly associated symptoms), the chance is high that the other connected node is also active.

There are two ways to activate the semantic network. First, when a symptom such as *nausea* is detected from a nurse's input to the DMS, the activation energy is spread through all of its connected nodes such as *vomiting* by multiplying the activation energy with their respective weights. If the result for any adjacent node is greater than its threshold, that node will also be "fired." This firing process continues till the network settles down to an equilibrium state, i.e., there is no other node which multiplying the activation energy and weight yields a result greater than its threshold. At the completion of the process, the set of active (or fired) nodes will be considered most relevant to a nurse's diagnosis, thus presented to the nurse. For example, suppose that the thresholds of all the nodes are set at 0.5 and the weight between *nausea* and *vomiting* is 0.8. If the activation energy level set up by the nurse is 1 (by changing the slider to "More" in the DSW) and *nausea* is detected, the only activated node would be *vomiting*; decreasing the activation energy level to 0 (by changing the slider to "Few" in the DSW) would lead to no activated node.

Second, the I-value from the neural network discussed in the previous section is used as the activation energy from the neural network "feeding" to the semantic network. For instance, if *Bowel Obstruction* is inferred as a likely disease and confirmed symptoms are *constipation*, *hyperactive bowel sound*, and *abdominal pain* (Table 4-4), then based on the remaining highest I-value, *vomiting* should be the next symptom to be confirmed with the patient. This would automatically trigger an activation energy of "1" to the semantic network as if *vomiting* were detected from the nurse's input. In this case, *nausea* will be activated and displayed because *vomiting* and *nausea* are closely associated symptoms and multiplying the activation energy and weight yields a result greater than the threshold. The basic idea is to remind the nurse that she needs to clarify whether the patient has *nausea* or *vomiting* in her next question because they are closely associated symptoms, which may lead to the identification of different diseases.

In summary, whenever a nurse enters a symptom in the DMS, the advisory module searches the database to decide whether any task support can be found from online resources. If the symptom matches a KU, the propagation will spread to the connected KUs based on the activation energy and corresponding weights. In this manner, the semantic network identifies task-specific knowledge not only for the current KU, but also for other closely associated and subsequent KUs.

4.2.5 The User Model

The user model is composed of records of users' details. A user's skill level is updated continuously as the user gains more experience. The last reference to a symptom and the number of references are captured. Such information is used, in a way similar to Vivacqua and Lieberman's approach (2000), by the advisory module to provide customized support suitable to a user's unique needs and preferences. Re-categorization of users' skill level is done by a simple algorithm (Figure 4-6) similar to Strachan et al.'s approach (2000). Although this implementation of user modeling may not be suitable for all domains, this is deemed appropriate for the prototype, as user modeling is not a main focus of this research.

If the user has been away for 30 days, downgrade user's skill level by one level.

Reset number of times used accordingly.

If the user has used a KU for less than 20 times, set user's skill level to *Novice*.

If the user has used a KU for less than 50 but more than 20 times, set user's skill level to *Intermediate*.

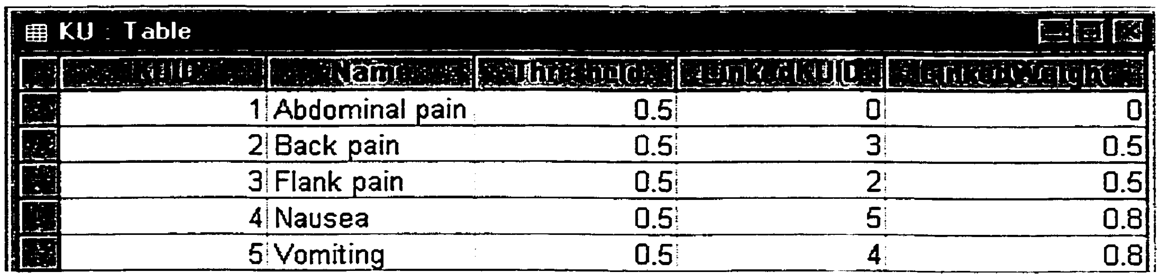
If the user has used a KU for more than 50 times, set user's skill level to *Expert*.

Figure 4-6. The Algorithm to Update Users' Skill Level

4.3 Implementation

The prototyped DMS and the advisory module were written in Visual Basic. Since the advisory module observes the task in progress but does not interfere with the DMS, it is constructed as an out-of-process ActiveX server.

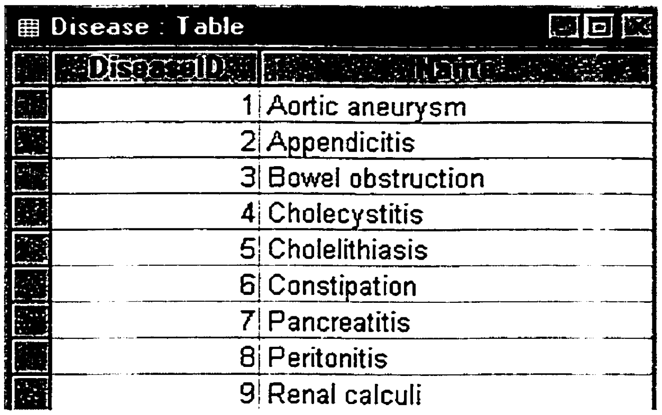
The neural network is written in Visual Basic and the semantic network is implemented using a database table in Microsoft Access. After the neural network is trained, all the weights are captured in a text file. When TTA is invoked for the first time, this text file (weights) will be read together with tables KU (input nodes) and Disease (output nodes) as shown in Figures 4-7 and 4-8 respectively. The core algorithm of how diseases are inferred by the neural network is shown in Appendix A. Figure 4-7 also shows an example of the semantic network. For instance, *back pain* (i.e., KUID = 2) is linked with *flank pain* (i.e., LinkedKUID = 3) with a weight of 0.5 (i.e., LinkedWeight = 0.5). Similarly, *nausea* is linked with *vomiting* with a weight of 0.8 as discussed earlier. The core algorithm of how activation is spread across the semantic network is shown in Appendix B.



KU : Table

KUID	Name	Threshold	LinkedKUID	LinkedWeight
1	Abdominal pain	0.5	0	0
2	Back pain	0.5	3	0.5
3	Flank pain	0.5	2	0.5
4	Nausea	0.5	5	0.8
5	Vomiting	0.5	4	0.8

Figure 4-7. Sample KUs

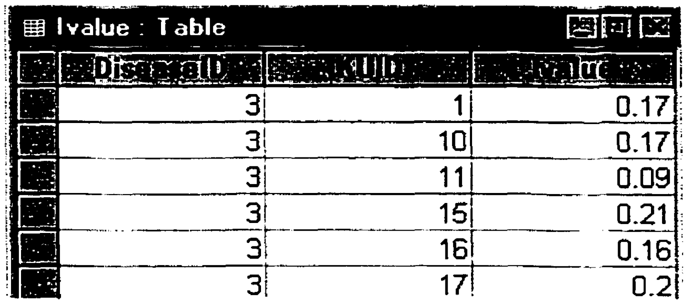


Disease : Table

DiseaseID	Name
1	Aortic aneurysm
2	Appendicitis
3	Bowel obstruction
4	Cholecystitis
5	Cholelithiasis
6	Constipation
7	Pancreatitis
8	Peritonitis
9	Renal calculi

Figure 4-8. Sample Diseases

The I-values between diseases and symptoms are stored in a database table in Microsoft Access. Figure 4-9 shows an example to represent I-values between *Bowel Obstruction* and its symptoms. For instance, the I-value between *Bowel Obstruction* (i.e., DiseaseID = 3) and *abdominal pain* (i.e., KUID = 1) is 0.17. Each time when a likely disease is inferred, its corresponding symptoms will be displayed in descending order according to the I-value. If a symptom has been confirmed, a check mark is displayed next to the symptom. If a symptom has the highest I-value and has not been confirmed, a question mark is displayed next to the symptom to remind the nurse to formulate questions based on this symptom as it is highly relevant to the inferred disease.



DiseaseID	KUID	Ivalue
3	1	0.17
3	10	0.17
3	11	0.09
3	15	0.21
3	16	0.16
3	17	0.2

Figure 4-9. Sample I-values between Bowel Obstruction and its Symptoms

The online resources contain definitions, examples, advice, references and follow-up questions, and are prepared using Microsoft Word document in rich text format. Figure 4-10 shows examples of *Abdominal Pain* and *Anorexia*.

ABDOMINAL PAIN

This is usually ill defined but can be very unpleasant and is termed visceral pain. Pain is initially felt near the mid-line of the abdomen.

Questions that can be asked:

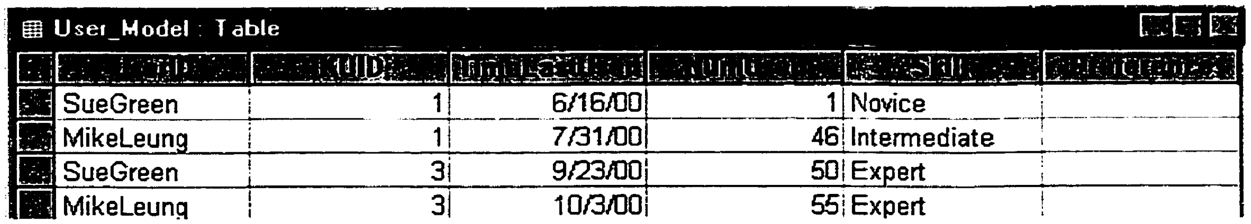
- What makes it better or worse?
- Is the pain severe/moderate/mild?
- What is the region/radiation of the pain?
- How long does it last?
- Have you tried any treatment?
- Any past medical history (diabetes, hypertension, etc.)?

ANOREXIA

It is the most common sign of dyspepsia due to gastritis and of cancer of the stomach. In some cases it is a manifestation of stress or stain such as domestic worry or difficulties at work.

Figure 4-10. Examples of Online Resources

The user model, following Vivacqua and Lieberman's approach (2000), is implemented using a Microsoft Access table as shown in Figure 4-11. Users' skill is categorized into three levels: novice, intermediate, and expert. When a user is working on a task for the first time, the user's skill level is assigned as novice. A user's skill level with respect to a KU is updated continuously as the user gains more experience. For each KU and each user, information such as the time of last access to the KU and the number of times the KU has been accessed are captured.



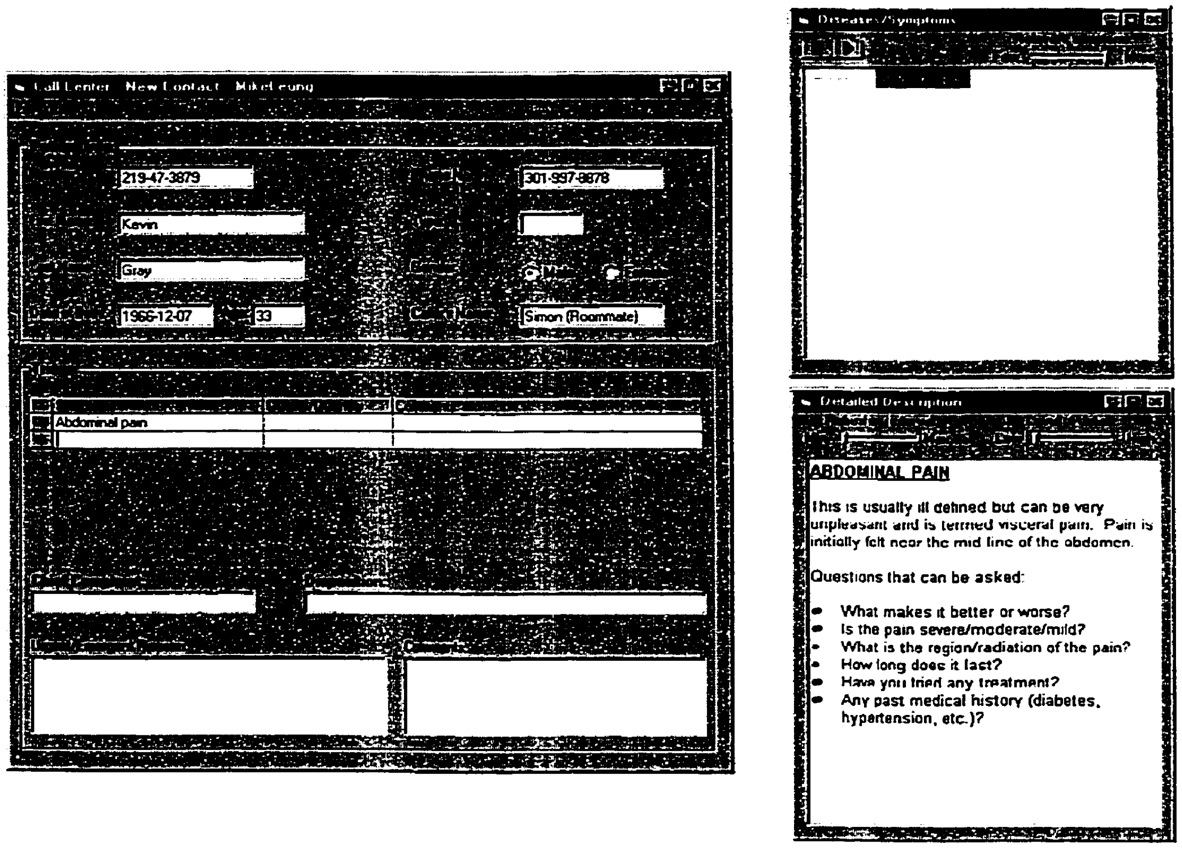
User_Model : Table

[illegible]	KUID	[illegible]	[illegible]	[illegible]	[illegible]
SueGreen	1	6/16/00	1	Novice	
MikeLeung	1	7/31/00	46	Intermediate	
SueGreen	3	9/23/00	50	Expert	
MikeLeung	3	10/3/00	55	Expert	

Figure 4-11. An Example of the User Model

4.4 Illustration

In this section, the triage process in which a novice nurse accomplishes her task with the help of TTA is illustrated step by step. First, the nurse logs in to the DMS and enters details of the caller. For example, Figure 4-12 shows a triage process that begins when the patient mentions that he has *Abdominal Pain*. The nurse enters *Abdominal Pain* in the symptoms box and TTA is loaded automatically displaying related information proactively in the DSW and DDW on the right hand side.



219-47-3879

301-997-8878

Kevin

Gray

1966-12-07

33

Simon (Roommate)

Abdominal pain

ABDOMINAL PAIN

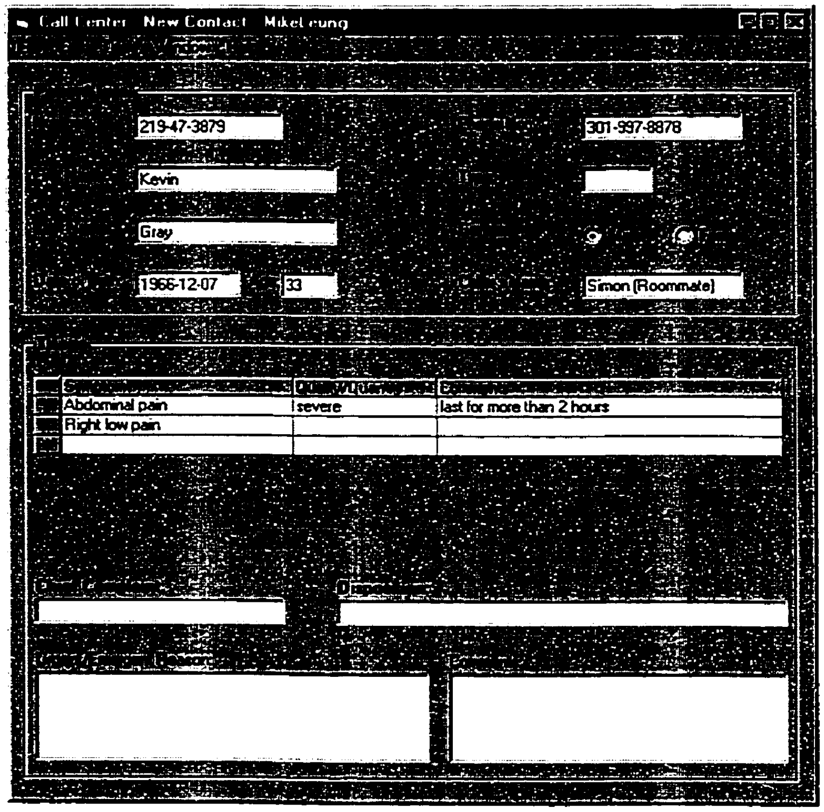
This is usually ill defined but can be very unpleasant and is termed visceral pain. Pain is initially felt near the mid line of the abdomen.

Questions that can be asked:

- What makes it better or worse?
- Is the pain severe/moderate/mild?
- What is the region/radiation of the pain?
- How long does it lact?
- Have you tried any treatment?
- Any past medical history (diabetes, hypertension, etc.)?

Figure 4-12. Proactive Task Information based on Confirmed Symptom

Assuming the nurse is a novice with respect to *Abdominal Pain*, the level of detail of medical information displayed in the DDW is automatically chosen at low according to the user model. However, the nurse can always change the level of details by clicking on the slider of "Level of details" in the DDW. For instance, in the above case, TTA suggests that the nurse ask further questions such as the severity, region, and duration of the pain. The answers from these questions may quantify the pain and lead to further input to the DMS. In Figure 4-13, the nurse enters *severe* in the Quality/Quantity box and *last for more than 2 hours* in the Comments box. She also enters *Right Low Pain* as the patient mentions that the pain is localized in the lower right region.



219-47-3879

Kevin

Gray

1966-12-07 33

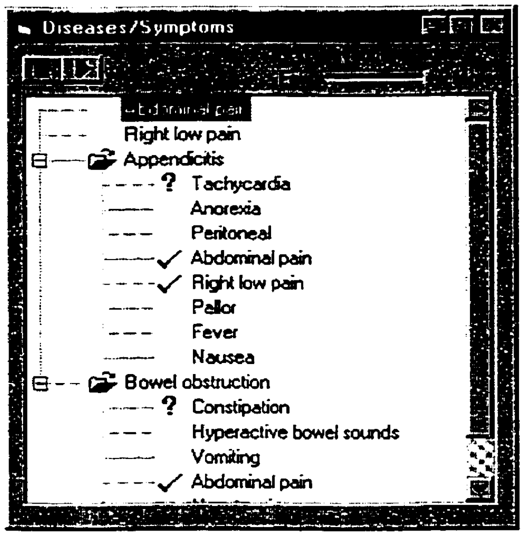
301-997-8878

Simon (Roommate)

Abdominal pain	severe	last for more than 2 hours
Right low pain		

Figure 4-13. Further Input based on TTA's Suggestions

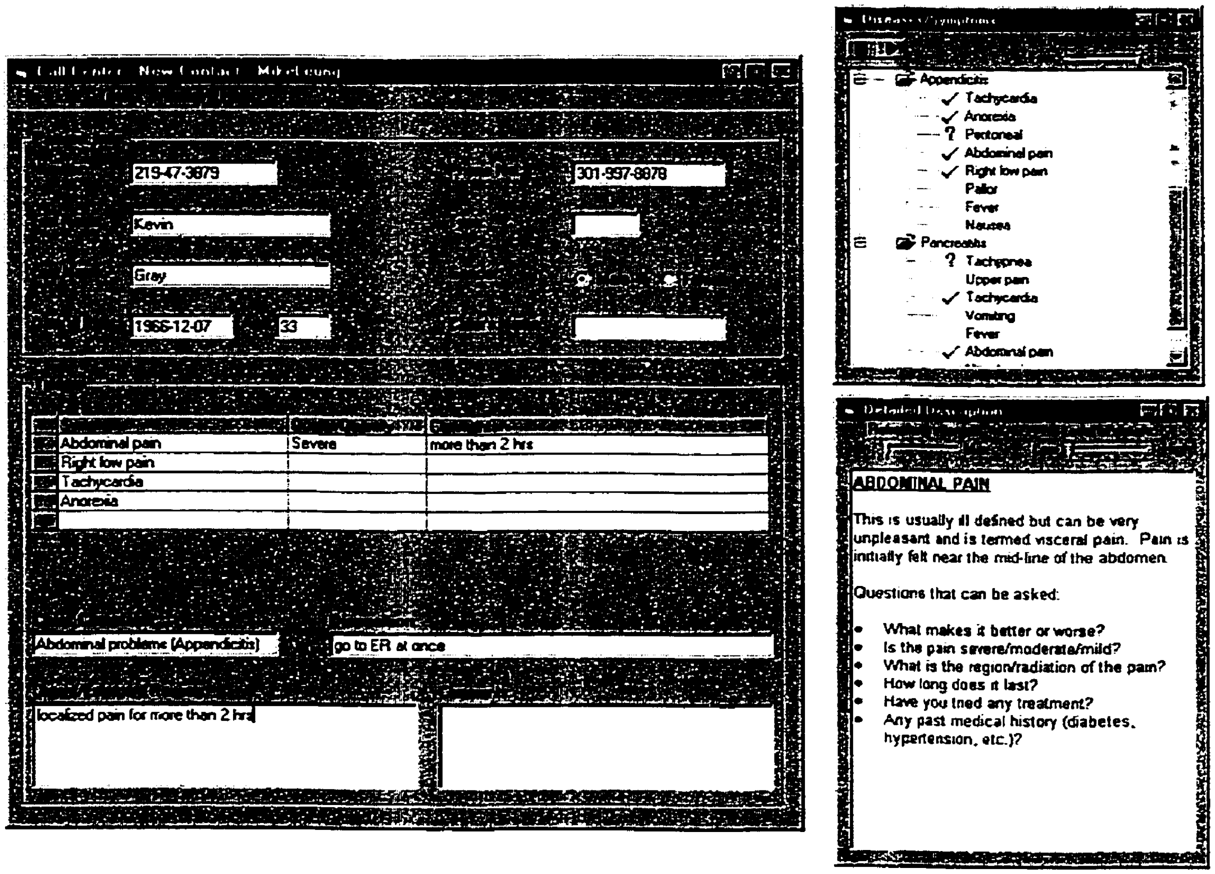
Based on the two symptoms entered, TTA starts its inference through the neural network and a list of likely diseases and their corresponding symptoms are displayed in the DSW. In addition, a check mark is shown next to the confirmed symptoms and a question mark is shown next to the symptoms with the strongest link to the disease. These are the symptoms that TTA suggests that the nurse confirm the presence with the patient (Figure 4-14).



```
Diseases/Symptoms
Abdominal pain
Right low pain
Appendicitis
    ? Tachycardia
      Anorexia
      Peritoneal
    ✓ Abdominal pain
    ✓ Right low pain
      Pallor
      Fever
      Nausea
Bowel obstruction
    ? Constipation
      Hyperactive bowel sounds
      Vomiting
    ✓ Abdominal pain
```

Figure 4-14. Likely Diseases and their Corresponding Symptoms Inferred by TTA

In this case, TTA considers *Appendicitis* the most likely disease and *Bowel Obstruction* the second. Now the nurse is advised to confirm whether the patient has *Tachycardia* or *Constipation.* Assuming she follows TTA's suggestion and confirms the patient has *Tachycardia*, she enters a third symptom of *Tachycardia* as a confirmed symptom. A check mark is shown next to *Tachycardia* and a question mark is shown next to *Anorexia.* The nurse continues to follow TTA's suggestion and confirms that the patient has *Anorexia.* The line of questioning and confirmation are formulated in this manner. When new symptoms are entered, the list in the DSW is updated accordingly to reflect the latest inference of TTA. The information in the DDW is also updated continuously. Suppose at this stage, the nurse has identified the likely disease is *Appendicitis.* She advises the patient to take appropriate treatment and the triage process is finished (Figure 4-15). Her task experience is recorded in the user model.



219-47-3879
301-997-8879
Kevin
Gray
1966-12-07
33

Abdominal pain	Severe	more than 2 hrs
Right low pain		
Tachycardia		
Anorexia		

Abdominal problems (Appendicitis)
go to ER at once
localized pain for more than 2 hrs

- Appendicitis
  - ✓ Tachycardia
  - ✓ Anorexia
  - ? Peritoneal
  - ✓ Abdominal pain
  - ✓ Right low pain
  - Pallor
  - Fever
  - Nausea
- Pancreatitis
  - ? Tachypnea
  - Upper pain
  - ✓ Tachycardia
  - Vomiting
  - Fever
  - ✓ Abdominal pain

ABDOMINAL PAIN

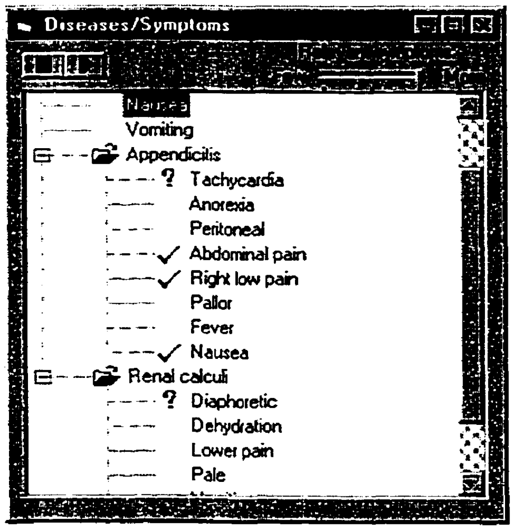
This is usually ill defined but can be very unpleasant and is termed visceral pain. Pain is initially felt near the mid-line of the abdomen.

Questions that can be asked:

- What makes it better or worse?
- Is the pain severe/moderate/mild?
- What is the region/radiation of the pain?
- How long does it last?
- Have you tried any treatment?
- Any past medical history (diabetes, hypertension, etc.)?

Figure 4-15. TTA's Advice based on Confirmed Symptoms

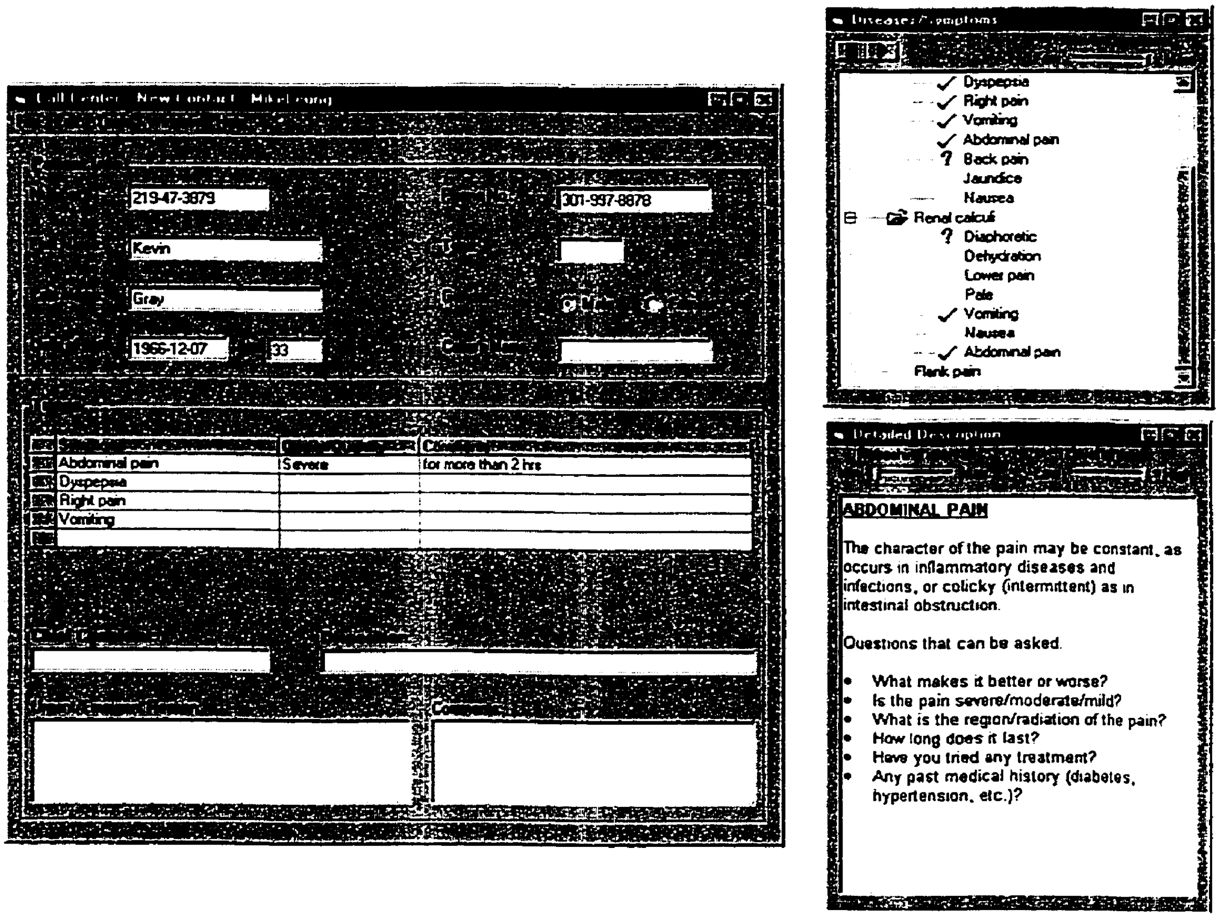
Suppose in another case where the patient has *Abdominal Pain* and *Right Low Pain*, TTA considers *Appendicitis* the most likely disease and *Bowel Obstruction* the second. Assuming the nurse does not follow TTA's suggestion of confirming the presence of *Tachycardia* or *Constipation* with the patient (Figure 4-14), as soon as the nurse enters a third symptom of *Nausea* as a confirmed symptom, *Vomiting* is also displayed in the DSW. It is because she has set "Relevant symptoms" to "More" using the slider in the DSW (Figure 4-16). This sets the initial activation energy to 1 in the semantic network. Since *Vomiting* and *Nausea* are commonly associated symptoms and they are connected in the semantic network, they are both displayed. In contrast, if "Relevant symptoms" is set to "Few" (i.e., initial activation energy has been set to 0), no propagation occurs. Therefore, only information on *Nausea* is displayed in the DDW. Suppose at this stage, the nurse has also identified the likely disease is *Appendicitis*. The triage process is finished when appropriate advice is given to the patient.



```
    Nausea
    Vomiting
⊟ Appendicitis
      ? Tachycardia
        Anorexia
        Peritoneal
      ✓ Abdominal pain
      ✓ Right low pain
        Pallor
        Fever
      ✓ Nausea
⊟ Renal calculi
      ? Diaphoretic
        Dehydration
        Lower pain
        Pale
```

Figure 4-16. Commonly Associated Symptoms Displayed by TTA

Suppose in a third case where the patient has *Abdominal Pain*, *Dyspepsia*, *Right Pain*, and *Vomiting*, TTA considers *Cholelithiasis* the most likely disease and *Renal Calculi* the second. A question mark is shown next to *Back Pain* suggesting that the nurse should confirm the presence of this symptom with the patient. At the same time, *Flank Pain* is also shown at the bottom of the list in the DSW (Figure 4-17). It is because the I-value, which is used to identify *Back Pain* as the next symptom to be confirmed, is fed to the semantic network that connects *Back Pain* and *Flank Pain* as commonly associated symptoms. Therefore, they both are activated and displayed in the DSW. In this way, subsequent questions about *Back Pain* and *Flank Pain* can be formulated. In this manner, the semantic network identifies the closely associated symptoms not only for the current symptom, but also for the suggested symptoms to be confirmed subsequently.



219-47-3879

301-997-8878

Kevin

Gray

1966-12-07 33

Abdominal pain	Severe	for more than 2 hrs
Dyspepsia		
Right pain		
Vomiting		

✓ Dyspepsia
✓ Right pain
✓ Vomiting
✓ Abdominal pain
? Back pain
Jaundice
Nausea
Renal calculi
? Diaphoretic
Dehydration
Lower pain
Pale
✓ Vomiting
Nausea
✓ Abdominal pain
Flank pain

ABDOMINAL PAIN

The character of the pain may be constant, as occurs in inflammatory diseases and infections, or colicky (intermittent) as in intestinal obstruction.

Questions that can be asked.

- What makes it better or worse?
- Is the pain severe/moderate/mild?
- What is the region/radiation of the pain?
- How long does it last?
- Have you tried any treatment?
- Any past medical history (diabetes, hypertension, etc.)?

Figure 4-17. Commonly Associated Symptoms of Subsequent Symptoms

4.5 Expected Benefits

TTA is expected to help the telephone triage process in the following ways: First, displaying likely diseases inferred by TTA may help nurses generate a more complete set of likely diseases, which is a recognized limitation of the diagnostic process (Lindgaard, 1995; Schaafstal, 1993). Having a list of likely diseases may help avoid the pre-mature determination of a single disease and remind nurses the possibility that the same symptoms are likely occur in other diseases.

Second, showing confirmed symptoms linked to a disease may reduce bias in a diagnosis. Bias occurs in medical diagnosis in two ways: Nurses may tend to select data favoring a disease while ignore data which might be likely to contradict that disease. In addition, nurses may fail to change their opinion about a disease in the face of non-supporting or even contradictory data (Lindgaard, 1995). Displaying a disease with its corresponding confirmed symptoms may help de-bias the process.

Third, to assist nurses in identifying a disease quickly, it is important to suggest the right questions to ask about the critical or indicative symptoms. Displaying diseases with their corresponding symptoms may help nurses save time in finding information and remind them of questions they need to ask to confirm a disease in a timely manner. This is especially helpful in a life-threatening situation where time is limited.

Finally, some symptoms are closely associated with each other although they may be symptoms of different diseases. TTA identifies the closely associated symptoms for the current symptoms and the suggested symptoms to be confirmed subsequently. Once one of them is identified, the identification or elimination of the other can help the diagnostic process.

5. Evaluation Method

A usability study in a field setting, involving novice nurses in the call center performing their tasks using TTA, was chosen to evaluate the task support features of the proposed approach (i.e., proactive advice and continuous update of information in a non-obtrusive manner). More specifically, the researcher wanted to answer the following questions:

1. How can TTA help nurses identify the disease?
2. To what extent is TTA used?
3. What is the impact of TTA on nurses' decision-making process?
4. Are TTA's task support features useful in practice?
5. Are TTA's task support features easy to use in practice?
6. What are nurses' attitudes towards TTA's user interface such as proactive advice and continuous update of information?

This chapter provides details of the evaluation method that was used.

5.1 Subjects

A pre-test in a lab setting was conducted prior to the field study to determine the procedure and materials for the study, and the general acceptability of TTA. Three subjects, who were university students, took part in the pilot study. Based on the pre-test, some minor programming changes were made to TTA.

The researcher contacted the management of ABC about the usability study, and received a positive response. Due to their heavy workload, the management was willing to assign only five nurses, representing one third of the practicing nurses at the call center, to participate in the study. Since TTA, as a prototype, covers only Abdominal Problem related calls, it could not be used on a longitudinal basis. Therefore, the study was set up like a field experiment. The study was designed to last for two days.

The clinical experience of the five subjects ranged from one to three years, with a mean of two years. Three of the subjects were female and two were male. All of them had a relatively high level of education (e.g., college degrees in nursing) and they were all registered nurses of the State of Maryland. They had computer experience at a self-identified intermediate level. Their background information is summarized in Table 5-1.

Subject	**Highest academic qualification achieved**	**Gender**	**Years of clinical experience**	**Years of experience in ER**	**Years of experience in call center**	**Years of experience with computer-based medical systems**
1	B.S.	F	3	1	1	3
2	M.S.	M	2	0	1	1
3	B.S.	F	1	0	0	1
4	B.S.	M	3	1	2	3
5	B.S.	F	1	0	0	1
Average			***2***	***0.4***	***0.8***	***1.8***

Table 5-1. Subjects' Background Information

The management of ABC helped the researcher send out a brief description of the study to all subjects five days before the study (Appendixes C to E). It explained the nature of the study. Emphasis was placed on the fact that the study was intended to measure the effectiveness of TTA, and it was by no means a measure of subjects' medical knowledge, computer knowledge, or telephone interviewing skill. A screen shot of TTA and instructions of thinking aloud were included, to help subjects get familiar with the usability study in advance. In addition, the Chief Clinical Officer (CCO) and a senior nurse (SN) were also invited to participate in an informal evaluation of TTA.

5.2 Measurement

As stated earlier, the primary objective of the field study was to investigate to what extent TTA was used and the impact of TTA on subjects' decision making process. The secondary objective was to investigate the usefulness and ease of use of TTA in a field setting.

To address the primary objective, subjects were asked to carry out some tasks as close to their normal practice as possible using TTA. From the task process (e.g., recording patient's personal details, asking questions, confirming symptoms or disease, and making decisions), the extent of TTA's use and its impact were captured. Thinking aloud was chosen to be an instrument to investigate the cognitive process (Ericsson & Simon, 1993). Subjects' thinking aloud can be described in three different levels: The first level of verbalization is simply the vocalization of oral encoding in which there is no intermediate process. The second level of verbalization involves description of the thought content. The third level of verbalization requires subjects to explain their thought process, which may require linking the information to earlier thoughts. Subjects were asked to provide a running commentary on what they were attempting to do and what was going through their mind. The think-aloud instruction was: "Say out loud everything that passes through your mind for each step." There was also a supplementary instruction: " It does not matter if your sentences are not complete. Just act as if you are alone in the room speaking to yourself loudly." Subjects were also told that if they were silent for more than 15 seconds, they would be reminded to keep talking.

The next question was how to record the comments made by the subjects while thinking aloud data when they were performing the tasks. The researcher had planned to use a video camera to capture the computer screen as well as the subjects' verbalization throughout the task process. However, the quality of the recording and the obtrusiveness of having a video camera behind the subjects were major concerns. Finally, a software package called Lotus ScreenCam was used instead. It could run as a background program and capture everything shown on the computer screen, keyboard input, mouse movement, and subjects' thinking aloud. From the data, the researcher could also study how TTA helped the subject in each step throughout the triage process. For instance, the screen capture allowed the researcher to count the number of times the subject clicked on TTA's windows for retrieving information. The only shortcoming was that the overall system response was slightly slower than a computer without running the ScreenCam.

To address the secondary objective of the study, a questionnaire based on Davis' (1989) perceived usefulness (six questions) and ease of use (six questions) was distributed to the subjects after the tasks. Four questions were added to check subjects' attitude towards TTA's interface (Table 5-2).

TTA's features	Questions
Display related information	The information provided by TTA reminds me other information that is relevant to the task.
Update information continuously	I have no trouble focusing on my work even though the contents of TTA's windows are updated continuously.
Update information continuously	I did not find the continuous information update by TTA distracting.
Display a list of likely diseases	TTA is useful despite that sometimes it displays irrelevant information.

Table 5-2. Questions that Addressing TTA's interface

The subjects were also asked to comment on the perceived benefits and shortcomings of TTA (Appendix I). Finally, the subjects' background information such as educational background, clinical experience, and computer knowledge was collected.

5.3 Tasks

The subjects were requested to perform two tasks. The first task was a typical and relatively simple task. The disease could be identified based on two or three of the symptoms outlined in Table 5-3. The second task was more challenging as it could take up to all of the symptoms in Table 5-3 to conclude the diagnosis. In both tasks, the subjects were required to identify a specific presentation of Abdominal Problem and to determine the appropriate disposition. Both tasks were provided by a senior nurse at the ABC Call Center. The symptoms and diseases in each task are shown in Table 5-3.

The diseases in Tasks 1 and 2 could occur in an adult of either gender according to the senior nurse. The patient/caller was instructed to respond with "I do not know" or "I am not sure" when asked whether he or she had any symptoms other than the ones listed in Table 5-3.

Task	**1**	**2**
Likely disease	**Appendicitis**	**Cholelithiasis**
Known symptoms for the patient	• Severe abdominal pain for more than 2 hours • Pain is localized at the right low region • Rapid heart rate • Loss of appetite • Fever of 101F • No past medical history of any kind	• Severe abdominal pain for more than 2 hours • Pain occurs also in back region • Discomfort after eating • Vomiting • Past medical history of abdominal trauma

Table 5-3. Disease and Symptoms for Tasks 1 and 2

In order to simulate the tasks as close to reality as possible, a patient/caller was required. At first, a medical student in Canada was selected to play the part of the caller. But due to difficult coordination between long distance (e.g., the caller had to call when the subject was ready to perform the task), this arrangement seemed to be inappropriate. Next, the background of the patient/caller was also considered. The conclusion was that the background should not be a factor especially the diseases in Tasks 1 and 2 could happen to anyone. Therefore, a reasonable scenario would be for the caller to play the role of the patient's roommate. In this way, the responses of "I do not know" and "I am not sure" were more convincing. A staff member at the call center played the role of a patient/caller calling the subjects.

5.4 Data Collection Procedures

The prototyped Disease Management System (DMS) and TTA ran on an IBM ThinkPad 380D. It was connected to an external monitor, keyboard, and mouse so that the subjects' experience would be similar to the normal use of their DMS. The computer was also connected to an Iomega Zip Drive so that data could be backed up immediately after every session. The usability study was conducted in a conference room in ABC Call Center. Figure 5-1 shows the major parts in the study and detail can be found in Appendixes C to J.

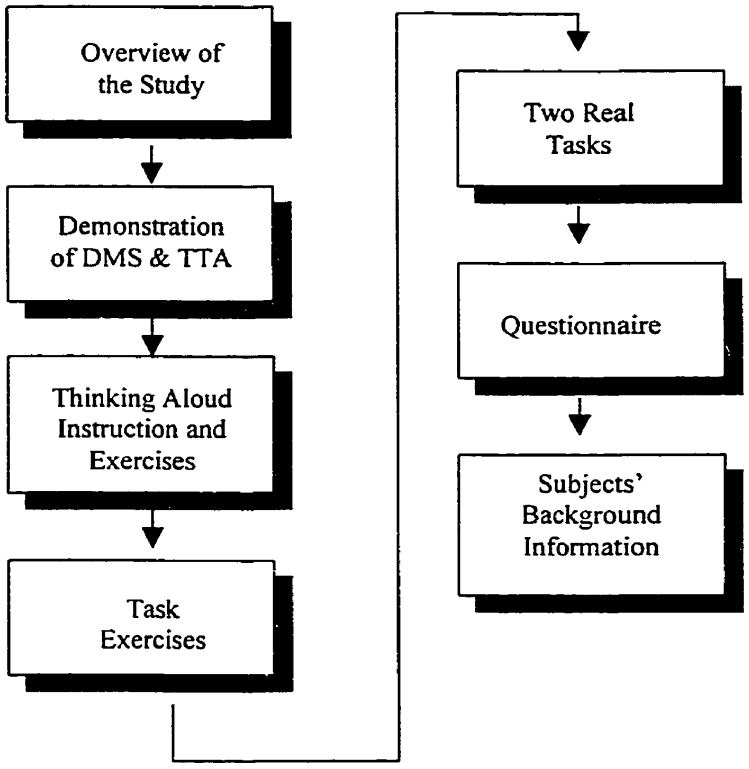

Overview of the Study
Demonstration of DMS & TTA
Thinking Aloud Instruction and Exercises
Task Exercises
Two Real Tasks
Questionnaire
Subjects' Background Information

Figure 5-1. Major Parts in the Study

The study began with a brief introduction by the researcher. The researcher established a rapport and credibility with the subjects by talking about his background and connection with ABC Call Center. The nature of the research and the activities to be completed throughout the study were explained to the subjects. It was highlighted that three major types of support information would be introduced:

1. likely diseases inferred by TTA based on already confirmed symptoms, and additional symptoms to be confirmed,
2. symptoms that were commonly associated with the ones already confirmed, along with background information about the confirmed symptoms, and
3. symptoms that were potentially relevant in subsequent steps.

In the Demonstration part, a twelve-minute automated multimedia demo recorded in Lotus ScreenCam was played. It discussed the use of DMS and TTA. In some cases, the researcher also did a live demo afterwards to clarify some questions from the subjects.

Next, the researcher explained the importance of thinking aloud during the experiments. In order to make sure the subjects understood the process of thinking aloud, a short question was included. The subjects were requested to think and talk aloud while performing the following addition: 476 + 688. This part took about 3 minutes.

The next part was the Task Exercises. Subjects were given a worksheet containing an exercise, which required them to enter symptoms, click on the list in the DSW, and find information in the DDW. The exercise involved a scenario with a patient experiencing *Abdominal Pain*, *Back Pain*, and *Shock*, because of a likely disease *Peritonitis*. It was provided by a senior nurse at the ABC Call Center. The main goal of this exercise was for the subjects to get familiar with using the different functions of TTA. Table 5-4 outlines the major steps in the exercise.

Functions of DMS and TTA	**Actions performed by subjects**	**Expected Outcome**
Input of symptoms in DMS	Enter *Abdominal Pain* as the first symptom	*Abdominal Pain* is displayed in the DSW of TTA.
Display of detailed description of a symptom	Click on *Abdominal Pain* in the hierarchical list in TTA's DSW	Medical information of *Abdominal Pain* is displayed in the DDW of TTA.
Inference of likely disease based on two or more symptoms Display of a confirmed symptom by a check (√) and a symptom that should be confirmed by a question mark (?)	Enter *Back Pain* as the second symptom	TTA starts to display its inference by a list of likely diseases and their corresponding symptoms. Checks (√) are displayed next to the symptoms. Question marks (?) are displayed next to the symptoms.
Change of the refreshing rate	Move the slider of Refreshing Rate to Max	The contents are updated continuously in TTA's DDW.
Change of the level of details	Move the slider of Level of Details to High	The contents are changed in TTA's DDW.
Input of the disposition in DMS	Enter *Peritonitis* or drag and drop *Peritonitis* from the hierarchical list in TTA's DSW	Disposition is entered and the case is closed.

Table 5-4. Some Functions, Actions, and Expected Outcome in the Exercise

Subjects were encouraged to think aloud in this part. They were also reminded that they should become fully familiar and confident with the features of TTA before proceeding to the next part. Subjects were advised to take as much time as they needed and that they could ask whatever questions they had in their mind. It took about fifteen minutes to twenty minutes to go through the exercise.

Prior to the real task, the prototyped DMS and Lotus ScreenCam were loaded. The researcher also reminded the subjects to perform the task as close to their normal practice as possible, apart from thinking aloud and using TTA. Then the researcher called the caller to signal the start of the triage process. The caller called in within fifteen seconds. The telephone conversation began with the subject asking for the details of the patient and caller. The call ended when the subject entered the suspected disease as the Chief Complaint, their advice to the caller as Disposition, and their justification for their advice. There were three instances when the caller failed to mention that he was the patient's roommate. The subjects spent about ten minutes on average on each task.

Finally, subjects were instructed to take a few minutes to fill out the questionnaire and background information sheet. The time taken to complete the study was about an hour for each subject. The CCO and the SN were invited to participate in an informal evaluation of TTA. This evaluation resembled the study mentioned above except that it did not involve performing the tasks. Instead, the main focus was on performing the task exercises. During the exercises, they thought aloud and gave comments from their perspectives.

6. Results of the Usability Study

The results of the study were analyzed based on subjects' thinking aloud, computer screen captured by ScreenCam throughout the task process, and the completed questionnaires. This chapter describes the results and details of the data analysis.

6.1 Thinking Aloud Data

The thinking aloud data was categorized for analysis based on the context such as viewing the display of medical advice, identifying symptoms, and formulating the line of questioning.

This section presents some representative examples of verbal protocols, along with their contexts in terms of the interaction among the patient/caller, nurse, and TTA, regarding the display of medical information (e.g., likely diseases and follow-up questions) during the task process. These examples were grouped together to show that subjects considered information provided by TTA useful and informative. Table 6-1 shows that a subject asked follow-up questions based on those suggested by TTA in the DDW. For example, she asked about the nature and region of pain. The patient's response led to the identification of another symptom of *Right Low Pain*. She followed up on TTA's line of questioning, then commented, "The follow-up questions are informative." Similarly, having gone through the same process, another subject commented, "The follow-up questions are comprehensive and we could add our best-practice guidelines and clinical pathway into the Detailed Description Window."

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1	... *Abdominal Pain* ...				
t_2		... Symptom is *Abdominal Pain* ...	Entered *Abdominal Pain* as the symptom.		
t_3				*Abdominal pain* was displayed.	Detailed description of *Abdominal pain* was displayed.
t_4			Clicked on the DDW and scrolled down to view further details and questions.		A list of questions was displayed.
t_5			Mouse pointed to each question and asked the questions one by one.		
t_6		... What makes the pain better or worse?			
t_7	... not sure ...				
t_8		... Is the pain severe, moderate, or mild?			
t_9	... severe ...				
t_{10}		... What is the region of the pain?			
t_{11}	... lower right region ...				
t_{12}		... Symptom is *Right Low Pain* ...	Enter *Right Low Pain* as the symptom.		
t_{13}				*Right Low Pain* was displayed.	Detailed description of *Right Low Pain* was displayed.
t_{14}		... This is informative ... it reminds me what questions I should ask ...			

Table 6-1. Verbal Protocols on Usefulness of TTA's Follow-up Questions

Table 6-2 shows that the subject reviewed the information of *Dyspepsia* and *Vomiting*. She also clicked on the slider to view a more detailed description about *Dyspepsia* and

Vomiting. Finally, she commented that the detailed description of symptoms and diseases in the DDW was useful and new information was learned.

Time	**Patient/ Caller**	**Subject's Thinking Aloud**	**Subject's Action**	**TTA's Diseases/ Symptoms Window (DSW)**	**TTA's Detailed Description Window (DDW)**
t_1	... discomfort after eating... vomited ...				
t_2			Entered *Dyspepsia* as the symptom. Entered *Vomiting* as the symptom.	*Dyspepsia* and *Vomiting* were displayed.	
t_3		... Symptom may be *Dyspepsia* or *Vomiting* ...			
t_4			Clicked on *Dyspepsia* in the DSW and viewed the detailed description in the DDW.		Detailed description of *Dyspepsia* was displayed.
t_5			Clicked on the slider to view a more detailed information about *Dyspepsia*.		More detailed description of *Dyspepsia* was displayed.
t_6			Clicked on *Vomiting* in the DSW and viewed the detailed description in the DDW.		Detailed description of *Vomiting* was displayed.
t_7			Clicked on the slider to view a more detailed information about *Vomiting*.		More detailed description of *Vomiting* was displayed.
t_8		... This is interesting and useful. ... I have learned some new things from the detailed description ...			

Table 6-2. Verbal Protocols on Usefulness of TTA's Detailed Description of Symptoms

Tables 6-3 to 6-5 present some examples that subjects could either adopt TTA's advice or ignore it. Table 6-3 shows that TTA identified *Appendicitis* and *Bowel Obstruction* as likely diseases based on the patient's symptoms. As soon as TTA displayed the inferred diseases, she moved the mouse to the DSW and clicked on *Appendicitis* and *Bowel Obstruction*. Next, she moved the mouse to the DDW to view a more detailed description about *Appendicitis* and *Bowel Obstruction*. Finally, she appeared to agree with TTA's inference by telling the patient that the disease was likely *Appendicitis* or *Bowel Obstruction*. She also commented that TTA could help determine the disease.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1	... pain is at my right low region ...				
t_2		... Symptom is *Right Low Pain* ...	Entered *Right Low Pain* as the symptom.		
t_3				*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Bowel Obstruction* was displayed as the second most likely disease below *Appendicitis*, along with its corresponding symptoms.	
t_4			Clicked on *Appendicitis* and its corresponding symptoms.		Detailed description of *Appendicitis* was displayed.
t_5			Moved mouse to the DDW. Clicked on slider to view a more detailed description of *Appendicitis*.		More detailed description of *Appendicitis* was displayed.
t_6			Clicked on *Bowel Obstruction* and its corresponding symptoms.		Detailed description of *Bowel Obstruction* was displayed.
t_7			Moved mouse to the DDW. Clicked on slider to view a more detailed description of *Bowel Obstruction*.		More detailed description of *Bowel Obstruction* was displayed.
t_8		... The list is very good and can help me determine the disease ...			
t_9		... So you might have *Appendicitis* or *Bowel Obstruction*...			

Table 6-3. Verbal Protocols on Usefulness of TTA for Identifying Diseases on Task 1 (Subject 1)

Table 6-4 shows that the subject clicked on *Peritoritis* and *Aortic Aneurysm* immediately after TTA identified *Peritoritis* and *Aortic Aneurysm* as likely diseases. Later, he moved the mouse to the DDW, but commented that the disease inferred by TTA did not match his own diagnosis, and then continued the task based on his own medical knowledge. He did not move the mouse to any of TTA's windows after making his comment.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1	... vomited ...				
t_2		... Symptom is *Vomiting* ...	Entered *Vomiting* as the symptom.		
t_3				*Peritoritis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Aortic Aneurysm* was displayed as the second most likely disease below *Peritoritis*, and its corresponding symptoms.	
t_4			Clicked on *Peritoritis* and its corresponding symptoms.		Detailed description of *Peritoritis* was displayed.
t_5			Clicked on *Aortic Aneurysm* and its corresponding symptoms.		Detailed description of *Aortic Aneurysm* was displayed.
t_6		... It did not match my own diagnosis ...			
t_7			Moved the mouse to the DMS.		

Table 6-4. Verbal Protocols on Usefulness of TTA for Identifying Diseases on Task 2 (Subject 2)

Table 6-5 shows that the subject moved the mouse on *Appendicitis* and then on its corresponding symptoms in the DSW. Under *Appendicitis*, most of the symptoms were checked. She appeared to agree with TTA's inference and entered *Appendicitis* as the Chief Complaint. She also commented that TTA could help save time in finding the necessary medical information.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		... Have you experienced loss of appetite? ...			
t_2	... yes ...		Entered *Anorexia* (loss of appetite) as the symptom.		
t_3				*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. Under *Appendicitis*, most of the symptoms were checked.	
t_4		... you might have *Appendicitis* ... Chief complaint ... *Appendicitis* ...	Enter *Appendicitis* as the Chief Complaint.		
t_5		... It saves me time in finding the information myself ...			

Table 6-5. Verbal Protocols on Usefulness of TTA for Identifying Diseases on Task 1 (Subject 3)

Table 6-6 shows that the subject clicked on *Cholelithiasis* in the DSW and then moved the mouse to the DDW. He appeared to agree with TTA's inference by entering *Cholelithiasis* as the Chief Complaint.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		... Have you noticed anything special about the color of the skin? ...		*Cholelithiasis* was displayed as the most likely disease proactively, along with its corresponding symptoms.	
t_2	... not sure...	... your friend has past medical history, so the chance of *Cholelithiasis* is high ...	Enter *Cholelithiasis* as the Chief Complaint.		

Table 6-6. Verbal Protocols on Usefulness of TTA for Identifying Diseases on Task 2 (Subject 4)

Tables 6-7 and 6-8 present some examples that subjects appeared to use TTA to formulate their line of questioning. In Table 6-7, it shows that as soon as the subject entered *Vomiting* as a symptom, both *Vomiting* and *Nausea* were displayed in the DSW. After she had clicked on both *Vomiting* and *Nausea*, detailed descriptions of *Vomiting* and *Nausea* were displayed in the DDW. She used the information in the DDW that "*Nausea* does not always lead to *Vomiting*" to ask further questions for clarification. Finally, she confirmed that the patient did have *Vomiting*.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1	... vomited ...				
t_2		... Symptom is *Vomiting* ...	Entered *Vomiting* as the symptom.		
t_3				Both *Vomiting* and *Nausea* were displayed.	
t_4		... Hey, it also shows *Nausea*. Let me click on each of them ...	Clicked on *Vomiting* and moved mouse to the DDW.		Detailed description of *Vomiting* was displayed.
t_5			Clicked on *Nausea* and moved mouse to the DDW.		Detailed description of *Nausea* was displayed.
t_6		... Did you really vomit? ...			
t_7	... yes ...				
t_8		... So it is not *Nausea* ...			

Table 6-7. Verbal Protocols on Usefulness of TTA for Influencing the Line of Question (Subject 1)

Table 6-8 shows that following TTA's suggestion to confirm the existence of *Tachycardia* (rapid heart rate), which was indicated by a question mark next to *Tachycardia* in the DSW, the subject asked whether the patient had a rapid heart rate. Later, TTA suggested confirming the existence of *Anorexia* (loss of appetite), which prompted him to ask whether the patient had any loss of appetite. Finally, he commented that the checks and question marks were helpful. In a similar situation, another subject also followed TTA's suggestions to formulate her line of questioning.

Time	Patient/ Caller	Subject's Thinking Aloud	Subject's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1				*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. There were checks next to *Abdominal Pain* and *Right Low Pain.* There was a question mark next to *Tachycardia* (rapid heart rate).	
t_2		... Have you experienced rapid heart rate? ...			
t_3	... yes ...				
t_4		... Symptom is *Tachycardia* ...	Entered *Tachycardia* as the symptom.		
t_5				*Appendicitis* was displayed as the most likely disease, along with its corresponding symptoms. There were checks next to *Abdominal Pain*, *Right Low Pain*, and *Tachycardia.* There was a question mark next to *Anorexia* (loss of appetite).	
t_6		... Have you experienced loss of appetite? ...			
t_7	... yes ...				
t_8		... Symptom is *Anorexia* ...	Entered *Anorexia* as the symptom.		
t_9				*Appendicitis* was displayed as the most likely disease, along with its corresponding symptoms. There were checks next to *Abdominal Pain*, *Right Low Pain*, *Tachycardia*, and *Anorexia.*	
t_{10}		... The checks and question marks are really good ...			

Table 6-8. Verbal Protocols on Usefulness of TTA for Influencing the Line of Question (Subject 2)

In summary, the verbal protocols show that the subjects appeared to respond positively to the way TTA displayed the information proactively. They recognized that the proactive display of information could help save their time in finding information and remind them of questions they needed to ask (see Tables 6-1 and 6-5). Furthermore, subjects also appeared to be positive about TTA's determining likely diseases based on patients' symptoms and formulating the line of questioning (see Tables 6-3 and 6-8). Finally, when TTA's advice was not deemed useful, the subjects continued the task execution based on their own medical knowledge as if TTA was not there (see Table 6-4).

6.2 ScreenCam Data

Data captured by ScreenCam provided some quantitative measure for analyzing: (1) to what extent TTA was used, and (2) what impact TTA had on the subjects' work, in addition to helping interpret the thinking aloud protocols. To answer the first question, two measures were taken: the total task time for the triage process (i.e., from the time when the first symptom was entered to the time when the chief complaint was entered) and the amount of time that the subject's mouse pointer was on either the DSW or the DDW. It is assumed that when the mouse pointer was moved to the TTA windows, the focus of the subject was also on TTA, or the thought of the subject was related to TTA. It is understood that this is only a rough proxy of TTA usage. The results are summarized in Table 6-9.

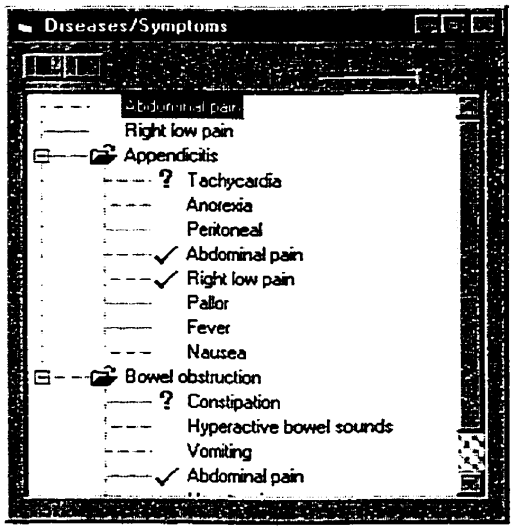
Subject	Task	Total task time (s)	Mouse-on-TTA time (s)	Percentage of mouse-on-TTA time (%)
1	1	257	146	57
	2	285	104	37
2	1	250	69	28
	2	129	41	32
3	1	232	134	58
	2	243	130	54
4	1	210	87	42
	2	139	47	34
5	1	255	88	35
	2	186	50	27
Average		*219*	*90*	***41***

Table 6-9. Total Task Time and Mouse-on-TTA Time

As shown in Table 6-9, on average, 41% of the subjects' time was spent using TTA in the diagnostic process. Therefore, it appears that TTA was indeed used to a significant extent. Note that all subjects spent less time on TTA during Task 2 than they did in Task 1. It turns out that the average was 105s (44%) for Task 1 versus 74s (37%) for Task 2. About 30% more time was spent on TTA in Task 1 than in Task 2. A Wilcoxon test suggests that this difference is statistically significant at 95% ($n = 5$; $\alpha = 0.05$). One explanation may be that TTA was still relatively new to the subjects when they performed Task 1, despite a generous allowance for training and tutorial time. Therefore, they spent more time exploring the information provided by TTA in Task 1 than in Task 2. Another reason may be that Task 2 was more challenging (see Section 5.3). Results from ScreenCam showed that *Choleithiasis* was displayed in the DSW as one of the likely diseases only after four or more symptoms had been entered in the DMS in Task 2.

Therefore, in Task 2 when the subjects found that the suggestions, based on two or three symptoms, provided by TTA did not match with what they were thinking, they might have just ignored the information provided by TTA and continued the task based on their own medical knowledge. This was confirmed by the thinking aloud data that subjects did not spend any time checking the information in the DDW when the inferred disease displayed in the DSW did not match with what they were thinking (see Table 6-4). However, if the novelty effect is taken out, 37% of task time on TTA could be more realistic and is still appeared to be quite sugnificant.

The impact of TTA on the subjects' work was also examined in terms of the overlap between TTA's advice and the actual line of questioning. The total number of questions asked by the subjects and the number of questions that appeared to be prompted by TTA (only if the current question matched the currently recommended question by TTA) were counted. For instance, when a question mark was shown next to *Tachycardia* and the subject did ask questions related to *Tachycardia*, these questions were considered to be influenced by TTA (Figure 6-1).



```
Diseases/Symptoms
    Abdominal pain
    Right low pain
⊟ Appendicitis
        ? Tachycardia
          Anorexia
          Peritoneal
        ✓ Abdominal pain
        ✓ Right low pain
          Pallor
          Fever
          Nausea
⊟ Bowel obstruction
        ? Constipation
          Hyperactive bowel sounds
          Vomiting
        ✓ Abdominal pain
```

Figure 6-1. TTA's Suggested Symptoms to be Confirmed

On the other hand, if the subjects' questions deviated from the TTA's advice, it is assumed that they were based on their own medical experience and knowledge, rather than influenced by TTA. The results are summarized in Table 6-10. It is understood that this measure tends to over-estimate the influence of TTA for two reasons. First, without TTA, the subjects could have independently come up with the same line of questioning. Second, subjects might have used TTA for curiosity and exploration to see what would happen, which is likely to happen in the initial use of any system.

Subject	Task	Total number of questions asked	Number of questions overlapped with TTA	Percentage of overlap (%)
1	1	11	9	81
	2	10	7	70
2	1	15	10	67
	2	8	5	63
3	1	10	7	70
	2	12	6	50
4	1	9	8	89
	2	10	7	70
5	1	9	7	78
	2	12	7	59
Average		*11*	*7*	***69***

Table 6-10. Total Number of Questions and Overlaps with TTA

Table 6-10 indicates that on average 69% of the subjects' questions were consistent with TTA's prompting. It could mean that TTA appeared to be influential in the diagnostic process. A closer examination of Table 6-10 reveals that the degree of overlap in Task 1 was always greater than or equal to that in Task 2. It turns out that the average number of questions which overlapped with TTA was 8 (76%) in Task 1 versus 6 (62%) in Task 2. The percentage was about 25% higher in Task 1 than in Task 2. A Wilcoxon test confirms that the difference is statistically significant at 99.5% ($n = 5$; $\alpha = 0.005$). The reason for the difference could be that in Task 1, the suggestions provided by TTA closely matched with what the subjects were thinking, and therefore, the suggestions were accepted. These numbers are consistent with Table 6-9, and support the interpretation for them.

6.3 Questionnaire Data

The results of the questionnaire are summarized in Tables 6-11, 6-12, and 6-13. The answers to the questions were arranged on a seven-point scale (Appendix I). Overall, the subjects perceived the ease of use of TTA to be high (Table 6-11). The average of the scores for all subjects ranged from 5.0 to 6.2 and the overall average is 5.7[8], on a 7-point scale.

Question	**4**	**6**	**8**	**10**	**11**	**13**
Subject 1	6	7	6	5	7	7
Subject 2	6	7	7	7	7	7
Subject 3	5	5	4	5	6	6
Subject 4	6	5	4	5	6	5
Subject 5	6	5	4	5	5	5
Average	***5.8***	***5.8***	***5***	***5.4***	***6.2***	***6***

Table 6-11. Results on Ease of Use of TTA

The subjects' perception of the usefulness of TTA was also high (Table 6-12). The average of the scores for all subjects ranged from 5.6 to 6.2 and the overall average is 5.8, on a 7-point scale.

[8] During the usability study, subjects also pointed out some usability problems in the interface design of TTA such as the lack of drop-down list and spell-check function. The high ease-of-use score could have been higher without these problems.

Question	**1**	**3**	**5**	**9**	**12**	**15**
Subject 1	6	6	7	6	6	6
Subject 2	7	7	7	7	7	7
Subject 3	6	5	5	5	6	5
Subject 4	6	5	6	5	6	5
Subject 5	5	5	6	5	5	5
Average	***6***	***5.6***	***6.2***	***5.6***	***6***	***5.6***

Table 6-12. Results on Usefulness of TTA

The subjects perceived the features (i.e., proactively displaying related information, continuously updating information, and displaying a list of likely diseases) of TTA to be satisfactory (Table 6-13). The average of the scores for all subjects ranged from 4.2 to 5.4 and the overall average is 4.8, on a 7-point scale.

Question	**2 ("The information provided by TTA reminds me other information that is relevant to the task")**	**7 ("I have no trouble focusing on my work even though the contents of TTA's windows are updating continuously")**	**14 ("I did not find the continuous information update by TTA distracting")**	**16 ("TTA is useful despite that sometimes it displays irrelevant information")**
Subject 1	6	4	4	6
Subject 2	7	5	6	7
Subject 3	5	4	5	5
Subject 4	5	4	3	4
Subject 5	4	4	4	4
Average	***5.4***	***4.2***	***4.4***	***5***

Table 6-13. Results based on TTA's Features

On average, questions 7 ("I have no trouble focusing on my work even though the contents of TTA's windows are updating continuously") and 14 ("I did not find the continuous information update by TTA distracting") had a relatively lower score than the other questions (4.2 and 4.4 respectively); they were close to neutral. The results from questions 7 and 14 indicate that the subjects had ambivalent feelings towards the continuous information update.

6.4 Anecdotes

The Chief Clinical Officer (CCO) and Senior Nurse (SN) were invited to participate in an informal evaluation of TTA. They thought aloud and gave comments from their perspectives while going through the exercises.

Tables 6-14 and 6-15 show some examples that TTA helped them to formulate the line of questioning during the task process. Table 6-14 shows that as soon as the SN entered *Abdominal Pain* as a symptom, *Abdominal Pain* was displayed in the DSW and its detailed description was displayed in the DDW. She read the follow-up questions in the DDW and commented, "I like the questions here. It reminds me of questions I should ask." She continued, and entered *Back Pain* and *Flank Pain* as the next symptoms. *Peritonitis* was displayed in the DSW as the most likely disease, along with its corresponding symptoms. Two of the symptoms were *Fever* and *Vomiting. Aortic Aneurysm* was also displayed in the DSW as the second most likely disease, along with its corresponding symptoms. She clicked on *Fever* and then *Vomiting* under *Peritonitis* in the DSW and moved the mouse to the DDW. Then she said, "I should confirm whether the patient has *Fever* and *Vomiting.*"

Time	Participant's Thinking Aloud	Participant's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		Entered *Abdominal Pain* as the symptom.		
t_2			*Abdominal Pain* was displayed.	Detailed description of *Abdominal Pain* was displayed.
t_3		Clicked on the DDW and scrolled down to view further details and questions.		A list of questions was displayed.
t_4		Mouse pointed to each question and asked the questions one by one.		
t_5	... What makes the pain better or worse?			
t_6	... I like the questions here. It reminds me of question I should ask ...			
t_7		Entered *Back Pain* and *Flank Pain* as symptoms.		
t_8			*Peritoritis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Aortic Aneurysm* was displayed as the second most likely disease below *Peritoritis*, and its corresponding symptoms.	
t_9		Clicked on *Peritoritis* and its corresponding symptoms.		Detailed description of *Peritoritis* was displayed.
t_{10}		Clicked on *Aortic Aneurysm* and its corresponding symptoms.		Detailed description of *Aortic Aneurysm* was displayed.
t_{11}	I should confirm whether the patient has *Fever* and *Vomiting*			

Table 6-14. Verbal Protocols on Usefulness of TTA for Formulating the Line of Question (SN)

In a similar situation, shown in Table 6-15, the CCO moved the mouse to the DDW and read the follow-up questions for *Abdominal Pain*. Then she commented, "Suppose I've a patient on the line, I can just ask the questions here." She continued with the exercise and entered *Right Low Pain* as the second symptom. *Appendicitis* was displayed in the DSW as the most likely disease, along with its corresponding symptoms such as *Tachycardia*, *Fever*, and *Nausea*. *Bowel Obstruction* was displayed in the DSW as the second most likely disease, along with its corresponding symptoms such as *Constipation*. She clicked on *Tachycardia*, *Fever*, *Nausea*, and *Constipation*. Then she said, "If I believe the patient has *Appendicitis*, I can ask whether the patient has *Tachycardia*, *Fever*, and *Nausea*. But if I believe the patient has *Bowel Obstruction*, I can ask whether the patient has *Constipation*."

Time	Participant's Thinking Aloud	Participant's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		Entered *Abdominal Pain* as the symptom.		
t_2			*Abdominal Pain* was displayed.	Detailed description of *Abdominal Pain* was displayed.
t_3		Clicked on the DDW and scrolled down to view further details and questions.		A list of questions was displayed.
t_4		Mouse pointed to each question and asked the questions one by one.		
t_5	... What makes the pain better or worse?			
t_6	... Suppose I've a patient on the line, I can just ask the questions here ...			
t_7		Entered *Right Low Pain* as the next symptom.		
t_8			*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Bowel Obstruction* was displayed as the second most likely disease below *Appendicitis*, and its corresponding symptoms.	
t_9		Clicked on *Appendicitis* and its corresponding symptoms.		Detailed description of *Appendicitis* was displayed.
t_{10}		Clicked on *Bowel Obstruction* and its corresponding symptoms.		Detailed description of *Bowel Obstruction* was displayed.
t_{11}	... If I believe the patient has *Appendicitis*, I can ask whether the patient has *Tachycardia*, *Fever*, and *Nausea*. But if I believe the patient has *Bowel Obstruction*, I can ask whether the patient has *Constipation*.			

Table 6-15. Verbal Protocols on Usefulness of TTA for Formulating the Line of Question (CCO)

Table 6-16 shows an example that the CCO used TTA to identify the disease during the task process. As soon as *Abdominal Pain* and *Right Low Pain* were entered as symptoms, *Appendicitis* was displayed in the DSW as the most likely disease, along with its corresponding symptoms. *Bowel Obstruction* was displayed in the DSW as the second most likely disease, along with its corresponding symptoms. There were checks next to *Abdominal Pain* and *Right Low Pain* indicating that they were confirmed symptoms. Then she commented, "I can continue to ask whether the patient has the other symptoms. By that time, I'd already know what disease the patient has."

TTA can also be used as an online training tool. For instance, according to the verbal protocol shown in Tables 6-2 and 6-18, two participants of the usability study highlighted the potential benefits of using TTA as an online training tool. However, in general there is a difference between task support systems and online training tools (e.g., intelligent tutoring systems). For instance, online training tools typically focus on education while task support systems focus on task completion. Therefore, a task support system like TTA can be used as a training tool like an intelligent tutoring system, but not vice versa.

Time	Participant's Thinking Aloud	Participant's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		Entered *Abdominal Pain* and *Right Low Pain* as symptoms.		
t_2			*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Bowel Obstruction* was displayed as the second most likely disease below *Appendicitis*, and its corresponding symptoms. There were checks next to *Abdominal Pain* and *Right Low Pain.*	
t_3		Clicked on *Appendicitis* and its corresponding symptoms.		Detailed description of *Appendicitis* was displayed.
t_4		Clicked on *Bowel Obstruction* and its corresponding symptoms.		Detailed description of *Bowel Obstruction* was displayed.
t_5	... I can continue to ask whether the patient has the other symptoms. By that time, I'd already know what disease the patient has. ...			

Table 6-16. Verbal Protocols on Usefulness of TTA for Identifying of Disease (CCO)

Tables 6-17 and 6-18 show the comments made by the COO and SN on the TTA's interface. Table 6-17 shows a situation that *Peritonitis* was suggested by TTA to be the most likely disease. Under *Peritonitis*, the confirmed symptoms, which had checks next to them, were *Abdominal Pain*, *Back Pain*, and *Flank Pain.* The SN said, "I like the checks. They are very clear and like a log of all the symptoms the patient has."

Time	Participant's Thinking Aloud	Participant's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		Entered *Abdominal Pain*, *Back Pain*, and *Flank Pain* as symptoms.		
t_2			*Peritoritis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Aortic Aneurysm* was displayed as the second most likely disease below *Peritoritis*, and its corresponding symptoms. There were checks next to *Abdominal Pain*, *Back Pain*, and *Flank Pain*.	
t_3		Clicked on *Peritoritis* and its corresponding symptoms.		Detailed description of *Peritoritis* was displayed.
t_4		Clicked on *Aortic Aneurysm* and its corresponding symptoms.		Detailed description of *Aortic Aneurysm* was displayed.
t_5	... I like the checks. They are very clear and like a log of all the symptoms the patient has. ...			

Table 6-17. Verbal Protocols on Usefulness of TTA's Interface (SN)

Table 6-18 shows that when the CCO was reading the follow-up questions aloud in the DDW, she commented, "I like the windows (DSW and DDW) being up-front. The nurses do not need to go back and forth to find the information. They are also good for training. We could put our online training information here."

Time	Participant's Thinking Aloud	Participant's Action	TTA's Diseases/ Symptoms Window (DSW)	TTA's Detailed Description Window (DDW)
t_1		Entered *Abdominal Pain* and *Right Low Pain* as symptoms.		
t_2			*Appendicitis* was displayed as the most likely disease proactively, along with its corresponding symptoms. *Bowel Obstruction* was displayed as the second most likely disease below *Appendicitis*, and its corresponding symptoms. There were checks next to *Abdominal Pain* and *Right Low Pain*.	
t_3		Clicked on *Appendicitis* and its corresponding symptoms. Read the contents in the DDW.		Detailed description of *Appendicitis* was displayed.
t_4		Clicked on *Bowel Obstruction* and its corresponding symptoms. Read the contents in the DDW.		Detailed description of *Bowel Obstruction* was displayed.
t_5	... I like the windows being up-front. The nurses do not need to go back and forth to find the information. They are also good for training. We could put our online training information here			

Table 6-18. Verbal Protocols on Usefulness of TTA's Interface (CCO)

In the informal evaluation of TTA, both the CCO and SN appeared to find TTA useful in formulating the line of questioning (see Tables 6-14 and 6-15) and identifying diseases (see Table 6-16). They also appeared to be positive about the features of the interface, such as proactive display of information in the DSW and DDW side-by-side to the user's task window and the use of checks to indicate the existence of confirmed symptoms (see Tables 6-17 and 6-18).

In summary, the subjects appeared to be positive about TTA's proactive display of advice and its continuous update of information in a non-obtrusive manner. This is supported by the thinking aloud data. In addition, on average, 41% of the task time was spent on TTA and up to 70% of subjects' questions appeared to be influenced by TTA. The post-task questionnaire data shows that TTA was perceived easy to use and useful for the task.

7. Discussion and Conclusions

This concluding chapter discusses the contribution, scalability, generalizability, and limitation of this research, along with some directions for further work.

7.1 Contribution

This research contributes to the field of online task support by proposing a novel approach to providing proactive task support. The approach consists of a user interface based on persistently present windows on the side to display proactive advice, and a task support engine utilizing neural networks to approximate a user's intention. An architectural model, which integrates the user interface and task support engine, is also proposed.

The proposed approach is significant for two reasons. First, it sounds appealing to provide task support in a just-in-time and just-enough manner in order to narrow the gap between the availability and accessibility of online resources. However, there is no generic methodology for implementing these ideas, although they frequently appear in the literature. In this research, the just-in-time requirement is reasonably satisfied by the use of continuous display of relevant task support in persistently present advice windows. It is generalizable and suitable for a broad range of tasks beyond diagnostic tasks. The just-enough requirement is satisfied by identifying the most relevant advice using a neural network based on a user's current task progress, and customizing the advice based on the user's profile and experience. The proposed approach seems to be generic, at least well suited for diagnostic tasks, and can be considered an effort to operationalize the just-in-time and just-enough requirements of task support.

Second, to some extent, the proposed approach explores the middle ground between the two existing approaches of user-initiated and system-initiated task support: It offers relevant online resources to users in a proactive but non-obtrusive manner, to rectify the deficiencies of user-initiated support and to overcome the difficulties associated with system-initiated support. As discussed earlier, user-initiated support is ineffective, inefficient, and prone to behavioral problems due to the "production paradox ." Likewise, there are also problems associated with system-initiated support such as the difficulty with correctly inferring users' intention, obtrusiveness, and interference with users' task. The proposed approach identifies one or more plausible tasks to approximate a user's task so that a range of relevant advice can be continuously displayed and updated in a separate advice window side-by-side to the task window for the user's selective use. Therefore, the proposed approach is definitely proactive, but less obtrusive and interfering than the conventional system-initiated task support.

The working prototype, TTA, has illustrated the viability of the proposed approach, implementation strategies, and the function of the key components of the architectural model. The usability study in a field setting has shown that, on average, 41% of the subjects' task time was spent on TTA and up to 70% of their questions appeared to be influenced by TTA. Results from a post-task questionnaire also show that TTA was perceived easy to use and useful for the task. It seems that the proposed approach is feasible and reasonably effective for diagnostic tasks.

7.2 Scalability and Generalizability

The scalability of the proposed approach can be examined by looking into the use of persistently present windows at the user interface, and the neural network, which is central to the task support engine. At the user interface, two extra advice windows are used persistently to display proactive advice. The idea seems generalizable to any application rather than diagnostic tasks only, and the only factor that may limit the scalability is the size of the computer screen, which may limit the number of windows concurrently opened by users. However, the main issue of interest is user preference and attitude to the interface style, which is an empirical question.

In the task support engine, the scalability of the proposed approach is determined by the modeling power and robustness of neural networks in general. Each time a new task is to be supported (e.g., a new disease is added), a new output node needs to be added to the neural network, and the neural network needs to be re-trained. Increasing the size of a neural network in this way has at least two problems. First, if the number of output nodes is large and only one of them is turned on and the rest off to represent any given task (cf. Baxt, 1990), the output patterns would be similar to each other. For example, suppose that there are 20 output nodes. The corresponding output pattern for the third and fifth tasks would be (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), and (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), respectively. The values of 18 output nodes are exactly the same for both cases. This will produce a relatively small mean square error in training. Thus, an optimal solution may not be found (Masters, 1993). Second, in general, larger neural networks are more difficult to train[9] (Ballard, 1990; Jacobs & Jordan, 1992).

Some researchers suggest that modularization is a possible way to solve the scalability problem in neural networks (e.g., Ballard, 1990; Battiti & Colla, 1994; Happel & Murre, 1994). A module is a self-contained neural network with its own input, hidden, and output nodes, although two different modules may share some common input nodes. For instance, a modularization technique is to break up the data samples, and different modules are trained on different subsets of the data samples (Baxt, 1992). Different modules can be used to represent different tasks in this research, where the first module may be *abdominal problem*, the second *headache*, and so on. In this way, not only can the neural network be scaled up fairly easily, it also alleviates the problems mentioned in the previous paragraph, as the network consists of modules of relatively simple networks instead of one huge complex network. However, having multiple networks designed to operate in parallel to produce multiple output merits further investigation to avoid outputting too many results. Although it may be an issue at the level of system design to control the number of output or which modules of network are activated at when, this has to be addressed before modular neural networks can be used in practice.

[9] As a simple example, a standard backpropagation network of 30 input nodes, 20 hidden nodes, and 30 output nodes has 1200 connections, i.e., a total of 1200 weights need to be determined in training. If the size of the network is doubled (i.e., 60 input nodes, 40 hidden nodes, and 60 output nodes), the network will have 4800 connections (four times larger than the original network).

The generalizability of the proposed approach can be examined by looking into the nature of the task domain within and beyond diagnostic tasks. The proposed model should be generalizable to any diagnostic or trouble-shooting tasks. This has been shown in the hypothetical example of trouble-shooting for PC in Section 3.4, and medical diagnosis in Chapter 4. As specified in Section 3.1, this research assumes diagnostic tasks typically exhibit the following characteristics: (1) a task can be decomposed into a series of discrete parallel steps, and (2) the task domain involves a finite set of basic steps (input) and a finite set of solutions (outcome). The proposed approach may not be suitable for other task domains, which do not have the above characteristics. For example, a computer programming domain has an unlimited number of tasks based on limited statements and control structures. Therefore, in this case, the proposed approach, which can only support a limited number of pre-specified task scenarios, may not be useful.

7.3 Limitations

The limitations of this research are discussed from three perspectives: the user interface, task support engine, and usability study. With respect to the user interface, proactive task support is displayed in persistently present windows side-by-side to the user's task window, and revised continuously according to the user's task progress. Although TTA users perceived the proactive display useful, the post-task questionnaire data also revealed that their perception of the continuous update was barely positive. In other words, the content of proactive advice was welcome, but there was some ambivalent feeling towards the continuous update. Therefore, a major design challenge is to retain the proactive nature of TTA's advice, which appeared to be effective and influential, but to reduce the potentially distracting effect of the continuous information update.

The task support engine's weaknesses originate from the use of neural networks. First, the neural network does not recognize certain patterns of symptoms-diseases if they are missing in the training samples. Second, premature advice swings among different diseases in response to newly confirmed symptoms before it eventually stabilizes on one or two diseases with enough confirmed symptoms. It confuses the user and discredits the task support system. This scenario is more likely to occur for more challenging diagnostic tasks such as the Task 2 in the usability study, where TTA could not identify the likely disease until four or more symptoms were confirmed. The proposed performance-tracking module can play a key role in any remedial measures.

Finally, the usability study has three limitations. First, the data from the subjects' thinking aloud did not give much insight to their cognitive during the task execution processes (cf. Ericsson & Simon, 1993). This could be due to the way that nurses interact with patients over the telephone: Nurses are already busy talking to patients. They may have other valid reasons to refrain from revealing what exactly they have in their mind, e.g., the fact that they receive advice from the computer, or the severity of disease in order not to intensify patients' stress level. Therefore, thinking aloud turned out to be incompatible with the task and the nurses' normal work habits, resulting in little insight to the subjects' cognitive processes as to the extent to which they were influenced by TTA. Second, the small number of available subjects did not afford controlled experiments or conclusions with statistical significance. Ideally, it would have been more interesting to compare the performance of subjects with TTA versus without TTA, and the performance of subjects with passive advice versus proactive advice. Third, the study was based on initial use of TTA, which inevitably involved some novelty effect.

7.4 Future Work

Based on the results of this research, there are at least three avenues of exploration for future research. First, with regard to the user interface, different display strategies could be adopted to find a balance between retaining the proactive nature of task advice and keeping the distracting effect of the continuous update to a minimum. One possibility is to display advice only when its likelihood is greater than a threshold. The threshold could be set at a high level to reduce the chance of displaying irrelevant information, which may affect the credibility of the task support, and to reduce the frequency of update. Another way to achieve the same goal is to display task support more conservatively, e.g., at a later stage, when the number of confirmed steps is greater than three or four. Hopefully, the increase of the number of confirmed steps may improve the accuracy of the inference. The ideal balance is reached if task support is provided soon enough to help formulate the line of questioning, and late enough to minimize the distracting effect and maintain the credibility of the task support. Furthermore, it may be useful to show the probabilities next to the diseases and symptoms to indicate their likelihood. With this information, nurses may stop asking unnecessary questions if one of the diseases is above a predefined probability (e.g., 0.8), which may help shorten the diagnostic process.

Second, in the task support engine, the performance-tracking module can be implemented to evaluate the effectiveness of the task support system continuously (see Section 3.3). For example, if users' actions do not seem to be influenced by task support for a given scenario consistently, this information should be captured and used for enhancing the performance of the system. This function can be extremely important for proactive task support as a distinct feature.

Finally, a baseline measure can be performed to measure how nurses are asking questions without TTA in real life. By comparing the overlap between the actual line of questioning without TTA and with TTA, the impact of TTA on nurses' decision making process can be measured. Moreover, the proposed approach can be applied to a different domain to test its scalability and generalizability. Although it is expected that the approach is well suited for diagnostic tasks, replicating TTA in a different domain would help confirm the belief. Furthermore, it would be better if the replication allows a longitudinal usability study, which tracked the users' behavior over a longer period of time. A longitudinal usability study in a field setting could offer insight to how users make use of an online task support system in an organization, and whether there are enough long-term benefits to justify the effort of developing such a system. A detailed cost benefit analysis is required to assess the potential of online task support as a credible alternative to conventional end-user training and support.

7.5 Concluding Remarks

This dissertation has identified the weaknesses in user-initiated and system-initiated task support, and proposed a novel approach that can alleviate the difficulties with these two approaches. The proposed approach has also operationalized the ideas of just-in-time and just-enough task support. Although the task domain of the prototype is medical diagnosis, the proposed approach is aimed at a broad range of applications. The main thrust of the proposed approach is its potential for enhancing access to online resources and literally bringing them to the users' fingertips.

References

Agah, A., & Tanie, K. (2000).
Intelligent graphical user interface design utilizing multiple fuzzy agents. *Interacting with Computers*, 12, 529-542.

Anderson, J., Corbett, P., & Reier, B. (1986).
Essential Lisp. Reading, MA: Addison-Wesley.

Ashutosh, K., Lee, H., Mohan, C., Ranka, S., Mehrotra, K., & Alexander, C. (1992).
Prediction criteria for successful weaning from respiratory support: Statistical and connectionist analyses. *Critical Care Medicine*, 20(9). 1295-1301.

Ballard, D.(1990).
Modular learning in hierarchical neural networks. In E. Schwartz (Ed.), *Computational Neuroscience*. London: MIT Press.

Bannon, L., Cyber, A., Greenspan, S., & Monty, M. (1983).
Evaluation and analysis of users activity organization. *Proceedings of the CHI '83 Conference on Human Factors in Computer Systems*, 54-57.

Barr, A., & Feigenbaum, E. (1981).
The Handbook of Artificial Intelligence. Vol. 1. Kaufmann Press, Los Altos, CA.

Battiti, R., & Colla, A. (1994).
Democracy in neural nets: Voting schemes for classification. *Neural Networks*, 7, 691-707.

Baxt, W. (1990).
Use of an artificial neural network for data analysis in clinical decision making: The diagnosis of acute coronary occlusion. *Neural Computation*, 2, 480-489.

Baxt, W. (1992).
Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation*, 4, 772-780.

Beaumont, I. (1994).
User modelling in the interactive anatomy tutoring system ANATOM-TUTOR. *User Modeling and User-Adapted Interaction*, 4, 21-45.

Bezanson, W. (1995).
Performance support: Online, integrated documentation and training. *SIGDOC 95*. Savannah, Georgia, 1-10.

Bhavnani, S., & John, B. (2000).

The strategic use of complex computer systems. *Human Computer Interaction*, 15(2&3), 107-138.

Blair, D., & Maron, M. (1985).
An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Communications of ACM*, 28, 289-299.

Bullen, C., & Bennett, J. (1996).
Groupware in practice: An interpretation of work experiences. In R. Kling (Ed.), *Computerization and Controversy: Value Conflicts and Social Choices*. 2nd Ed. San Diego, CA: Academic Press.

Burns, H., & Capps, C. (1988).
Foundations of intelligent tutoring systems: An introduction. In M. Polson & J. Richardson (Eds.), *Foundations of Intelligent Tutoring Systems*. NJ: Lawrence Erlbaum.

Burton, R., & Brown, J. (1982).
An investigation of computer coaching for informal learning activities. In D. Sleeman & J. Brown (Eds.), *Intelligent Tutoring Systems*. UK: Academic Press.

Card, S., Pavel, M., & Farrell, J. (1984).
Window-based computer dialogues, *INTERACT '84, First IFIP Conference on Human-Computer Interaction*. London, 355-359.

Carroll, J., & Mazur, A. (1986).
Lisa learning. *Computer*, 19(11), 35-49.

Carroll, J., & McKendree, J. (1987).
Interface design issues for advice-giving expert systems, *Communications of ACM*, 30(1).

Carroll, J., & Rosson, M. (1987).
Paradox of the active user. In J. Carroll (Ed.), *Interfacing Thought*. (pp. 81-111).

Charniak, E., & McDermott, D. (1985).
Introduction to Artificial Intelligence. MA: Addison-Wesley.

Chin, D. (1986).
User modeling in UC: The UNIX Consultant. *Proceedings of the CHI '86 Conference on Human Factors in Computing Systems*. 24-28.

Chin, D. (1989).
KNOME: Modeling what the user knows in UC. In A. Kobsa & W. Wahlster (Eds.), *User Models in Dialog Systems*. NY: Springer-Verlag.

Chiu, D., & Leung, M. (1996).
A measure of dependency in feedforward networks. *Proceedings Vision Interface 1996.* 57-63.

Chiu, D., & Leung, M. (1998).
Evaluating input dependency in a feedforward neural network. *Proceedings 5th International Conference on Neural Information Processing (ICONIP 98).* IOS Press, 801-804.

Clancey, W. (1983).
GUIDON. *Journal of Computer-based Instruction*, 10, 1, 8-14.

Cole, K., Fisher, O., & Saltzman, P. (1997).
Just-in-time knowledge delivery. *Communications of ACM*, 40(7), 49-52.

Davis, F. (1989).
Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3), 319-340.

Desmarais, M., Larochelle, S., & Giroux, L. (1987).
The diagnosis of user strategies. In H. Bullinger & B. Shackel (Eds.), *Human-Computer Interaction – INTERACT '87.* Amsterdam: Elsevier.

Desmarais, M., Leclair, R., Fiset, J., & Talbi, H. (1997).
Cost-justifying electronic performance support systems. *Communications of ACM*, 40(7), 39-48.

Diaper, D., & Waelend, P. (2000).
World Wide Web working whilst ignoring graphics: good news for web page designers. *Interacting with Computers*, 13, 163-181.

Diederich, J. (1992).
Explanation and artificial neural networks. *International Journal of Man-Machine Studies*, 37, 335-355.

Dorsey, L., Goodrum, D., & Schwen, T. (1993).
Defining and building an enriched learning and information environment. *Educational Technology*, 33(11), 10-13.

Eberts, R., & Habibi S. (1995).
Neural network-based agents for integrating information for production systems. *International Journal of Production Economics*, 38, 73-84.

Eberts, R., Villegas, L., Phillips, C., & Eberts, C. (1992).
Using neural net modeling for user assistance in HCI tasks. *International Journal of Human-Computer Interaction*, 4(1), 59-77.

Elkerton, J. (1988).
Online aiding for human-computer interfaces. In M. Helander (Ed.), *Handbook of Human-Computer Interaction*. North-Holland: Elsevier.

Ericsson, K., & Simon, H. (1993).
Protocol Analysis: Verbal Reports as Data. Cambridge, MA: MIT Press.

Feldman, J., & Ballard, D. (1982).
Connectionist models and their properties. *Cognitive Science*, 6, 205-254.

Fischer, O., & Horn, R. (1997).
Electronic performance support systems. *Communications of ACM*, 40(7), 31-32.

Fisher, G., Lemke, A., & Schwab, T. (1985).
Knowledge-based help systems. *Proceedings of the CHI '85 Human Factors in Computing Systems*, NY: ACM.

Furman, A., & Spyridakis, J. (1992).
The effect of system-initiated advice on the use of menu navigation shortcuts. *IEEE Transactions on Professional Communication*, 35(2).

Gery, G. (1991).
Electronic performance support systems: How and why to remake the workplace through the strategic application of technology. MA: Weingarten Publications.

Gery, G. (1995).
Attributes and behaviors of performance-centered systems. *Performance Improvement Quarterly*, 8(1), 47-92.

Grossman, V. (1999).
Quick Reference to Triage. NY: Lippincott.

Hackos, J. (1997).
Online documentation: The next generation. *Proceedings of the 15th Annual International Conference on Systems Documentation SIGDOC*. NY: ACM.

Handysides, G. (1995).
Triage in Emergency Practice. NY: Mosby.

Happel, B., & Murre, J. (1994).
Design and evolution of modular neural network architectures using a genetic algorithm. In I. Alexsander & J. Taylor (Eds.), *Artificial Neural Networks*. Holland: Elsevier Science.

Hertzum, M., & Frokjaer, E. (1996).

Browsing and querying in online documentation: A study of user interface and the interaction process. *ACM Transactions on Computer-Human Interaction*, 3(2), 136-161.

Hook, K. (2000).
Steps to take before intelligent user interfaces become real. *Interacting with Computers*, 12, 409-426.

Horvitz, E. (1999).
Principles of mixed-initiative user interfaces. *CHI '99 Conference Proceedings: The CHI is the Limit*. NY: ACM.

Jacobs, R., & Jordan, M. (1992).
Computational consequences of a bias toward short connections. *Cognitive Neuroscience*, 4, 323-336.

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

Jennings, A., & Higuchi, H. (1993).
A user model neural network for a personal news services. *User Modeling and User-Adapted Interaction*, 3, 1-25.

Jennings, N., & Wooldridge, M. (1996).
Software agents. *Knowledge Engineering Review*. 11(3).

Landauer, T. (1995).
The Trouble with Computers: Usefulness, Usability, and Productivity. MA: MIT Press.

Lieberman, H. (1997).
Autonomous interface agents. *CHI '97 Conference Proceedings: Looking into the Future*. NY: ACM.

Lindgaard, G. (1995).
Human performance in fault diagnosis: Can expert systems help? *Interacting with Computers*, 7(3), 254-272.

Lindgaard, G., & Triggs, T. (1990).
Can artificial intelligence outperform real people? The potential of computerized decision aids in medical diagnosis. In W. Karwowski, A. Genaidy, & S. Asfour (Eds.), *Computer-Aided Ergonomics: A Researcher's Guide*. UK: Taylor & Francis Press.

Maclin, P., & Dempsey, J. (1993).

A neural network to diagnose liver cancer. *International Conference on Neural Networks. Vol. III,* 1492.

Maes, P. (1994).
Agents that reduce work and information overload. *Communications of ACM,* 37(7).

Mansfield, R. (1994).
The Compact Guide to Microsoft Office Professional. CA: Sybex.

Masters, T. (1993).
Practical Neural Network Recipes in C++. CA: Academic Press.

Mozer, M., & Smolensky, P. (1989).
Using relevance to reduce network size automatically. *Connection Science,* 1, 3-16.

Ndumu, D., & Nwana, H. (1997).
Research and development challenges for agent-based systems. *IEE Proceedings of Software Engineering,* 144(1), February.

Nickerson, R. (1999).
Why interactive computer systems are sometimes not used by people who might benefit from them. *International Journal of Human-Computer Studies,* 51, 307-321.

Noteboom, N. (1998).
Die clippy, die. <http://www.zdnet.com/anchordesk/story/story_2589.html> (accessed on 4/4/1999).

Orwant, J. (1991).
The doppelganger user modeling system. *Proceedings of the IJCAI Workshop W4: Agent Modeling for Intelligent Interaction.* 164-168.

Patel, V., & Groen, G. (1991).
The general and specific nature of medical expertise: A critical look. In K. Ericsson & J. Smith (Eds.), *Towards General Theory of Expertise: Prospects and Limits.* UK: Cambridge University Press.

Patil, R. (1988).
Artificial intelligence techniques for diagnostic reasoning in medicine. In H. Shrobe (Ed.), *Exploring Artificial Intelligence: Survey Talks from the National Conferences on Artificial Intelligence.* CA: AAAI.

Raybould, B. (1995).

Performance support engineering: An emerging development methodology for enabling organizational learning. *Performance Improvement Quarterly*, 8(1), 7-22.

Rochlin, G. (1997).
Trapped in the Net: The Unanticipated Consequences of Computerization. NJ: Princeton University Press.

Rogers, S., Ruck, D., & Kabrisky, M. (1994).
Artificial neural networks for early detection and diagnosis of cancer. *Cancer Letters*, 77(2-3), pp. 79-83.

Rumelhart, D., Hinton, G., & Williams, R. (1986).
Learning internal representations by error propagation. In D. Rumelhart & J. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1. Foundations*. MA: MIT Press.

Rumelhart, D., Widrow, B., & Lehr, M. (1994).
The basic ideas in neural networks. *Communications of ACM*, 37(3).

Sarle, W. (1994).
Neural networks and statistical models. *Proceedings of the Nineteenth Annual SAS Users Group International Conference.*

Schaafstal, A. (1993).
Knowledge and strategies in diagnostic skill. *Ergonomics*, 36(11), 1305-1316.

Selker, T. (1994).
Coach: A teaching agent that learns. *Communications of ACM*, 37(7).

Shneiderman, B. (1997)
Direct manipulation for comprehensible, predictable, and controllable user interfaces. *Proceedings of IUI97, 1997 International Conference on Intelligent User Interfaces*, Orlando, FL, 33-39.

Silva, R., & Roque Da Silva, A. (1998).
Medical diagnosis as a neural networks pattern classification problem. In E. Ifeachor, A. Sperduti, & A. Starita (Eds.), *Neural Networks and Expert System in Medicine and Healthcare*. Singapore: World Scientific Publishing.

Simpson, P. (1990).
Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations. NY: Pergamon Press.

Sleeman, D., & Brown, J. (1982).

Introduction: Intelligent tutoring systems. In D. Sleeman & J. Brown (Eds.), *Intelligent Tutoring Systems*. UK: Academic Press.

Smith, M. (1993).
Neural Networks for Statistical Modeling. NY: Van Nostrand.

Strachan, S., Anderson, J., Sneesby, M., & Evans, M. (2000).
Minimalist user modelling in a complex commercial software system. *User Modeling and User-Adapted Interaction*, 10(2), 109-146.

Villegas, L., & Eberts, R. (1994).
A neural network tool for identifying text-editing goals. *International Journal of Human-Computer Studies*, 40, 813-833.

Vivacqua, A., & Lieberman, H. (2000).
Agents to assist in finding help. *CHI 2000 Conference Proceedings: The Future is Here*. NY: ACM.

Wheeler, S., & Windt, J. (1993).
Telephone Triage: Theory, Practice, and Protocol Development. NY: Delmar.

Winslow, C., & Caldwell, J. (1992).
Integrated performance support: A new educational paradigm. *Information Systems Journal*, Spring, Boston, MA: Auerbuch.

Wolfe, B., & Eichmann, D. (1997).
A neural network approach to tracking eye position. *International Journal of Human-Computer Interaction*, 9(1), 59-79.

Ye, N. (1997).
Neural networks approach to user modeling and intelligent interface: A review and reappraisal. *International Journal of Human-Computer Interaction*, 9(1), 3-23.

Appendix A: Algorithm of the Inference of Diseases by the Neural Network

```
If
    there is new input (symptom) to the DMS
then
    check the online resources to see whether the symptom matches with any KU.
    If
        matched
    then
        display the KU in the DSW,
        display the detailed description of the KU in the DDW,
        take the activation energy from the slider on the DSW,
        activate the propagation network with the KU and activation energy (see
        Appendix B).
        If
            there are two or more matched KU to the DMS
        then
            reset all input values of the neural network to "0",
            for all matched KU:
                set the corresponding input value of the neural network to
                "1",
            compute the output values of the neural network for all diseases,
            sort the output values,
            select the largest output value (i.e., the most likely disease
            inferred),
            select the second largest output value (i.e., the second most likely
            disease inferred),
            display the most likely and second most likely diseases in the
            DSW,
            display their corresponding symptoms ordered by I-values in the
            DSW,
            for each symptom displayed in the DSW:
                If
                    the symptom has been confirmed  (i.e., input to the
                    DMS)
                then
                    display a check mark next to the symptom.
                If
                    the symptom has not been confirmed and has had
                    the largest I-value among other unconfirmed
                    symptoms
                then
                    display a question mark next to the symptom
```

Appendix B: Algorithm of the Activation of the Semantic Network

```
For all connected KUs:
    If
        the KU is not the same as the activated KU (i.e., forward firing only)
    then
        calculate: computed-energy = activation energy $\times$ weight between the KU
        and the activated KU,
        If
            computed-energy $\geq$ threshold of the KU
        then
            display the KU in the DSW,
            set activation energy = computed-energy,
            repeat the process with the KU and the activation energy
```

Appendix C: An Evaluation of the Telephone Triage Assistant

An Evaluation of the Telephone Triage Assistant

The objective of this research is to evaluate the effectiveness of some emerging computer-based techniques that can be applied to telephone triage. To this end, a prototyped Telephone Triage Assistant (TTA) has been developed to assist novice nurses in identifying the nature of diseases and the need for appropriate care based on the symptoms identified via a telephone interview. TTA supports the task by showing three types of relevant information for forming the path of questioning during the interview:

1. likely diseases inferred by TTA based on already confirmed symptoms, and additional symptoms to be confirmed,

2. symptoms that are commonly associated with the ones that are already confirmed, along with background information about the confirmed symptoms, and

3. symptoms that are potentially relevant in subsequent steps.

The scope of the current version of TTA is modest and limited to nine presentations of Abdominal Pain such as Abdominal Aortic Aneurysm and Appendicitis, as the starting point.

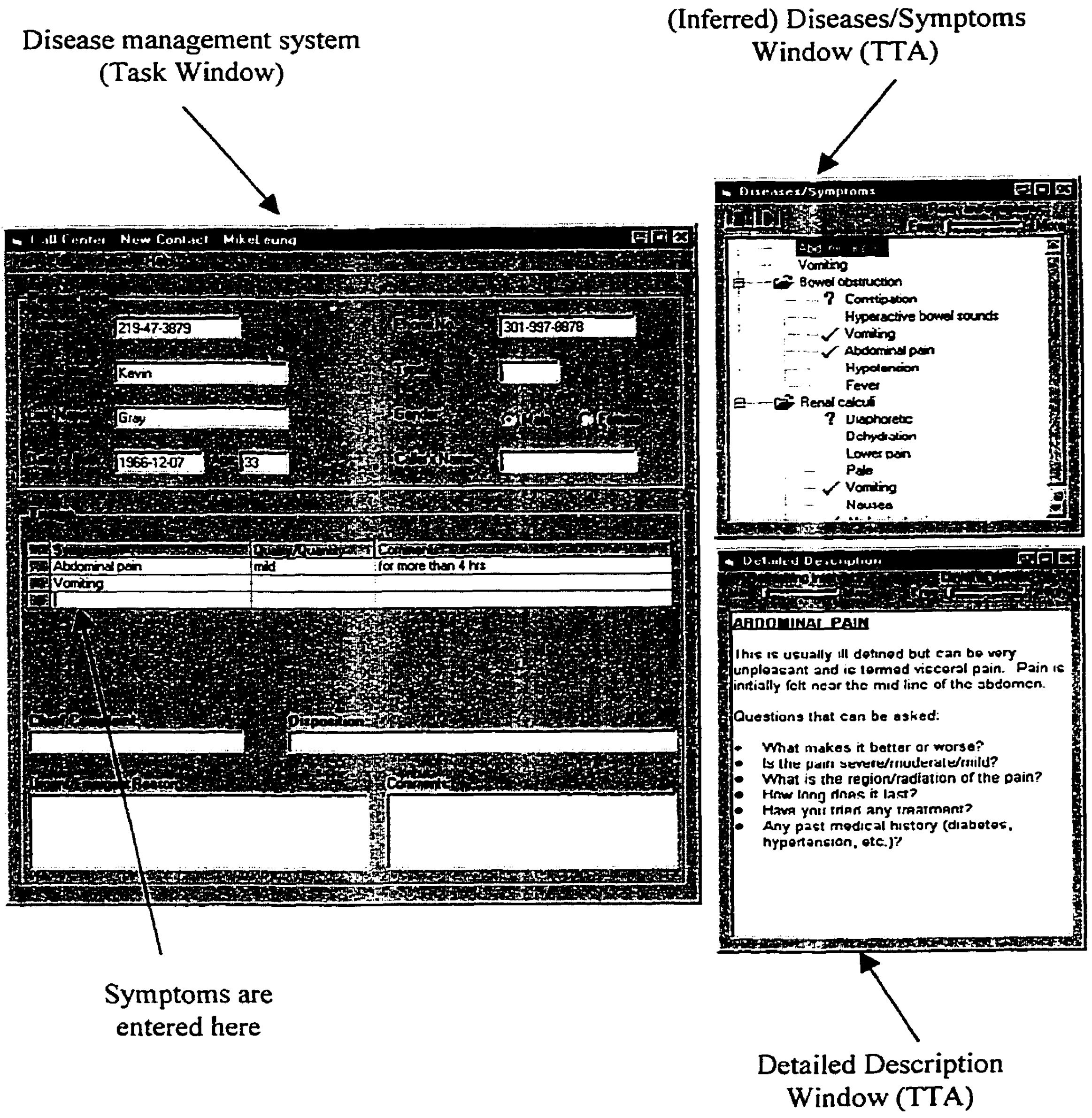
In the next half an hour, you will be trained to use TTA first, and given a couple of exercises to get familiar with TTA. Then, you will perform two telephone triage tasks using TTA. This process will be captured by the computer, including everything shown on the computer screen, your keyboard input and mouse movement, and your thinking aloud, to be analyzed for further improvement of TTA. At last, you will fill out a questionnaire about TTA.

All we are interested is the effectiveness of TTA. This study is by no means a measure of your medical knowledge, computer knowledge, or telephone interviewing skill.

Appendix D: A Demo of TTA

A Demo of TTA

The researcher will give you a short demo of how to use TTA now. Please do not hesitate to ask should you have any questions. The following figure illustrates different components of TTA for your reference.


Disease management system (Task Window)
(Inferred) Diseases/Symptoms Window (TTA)
Symptoms are entered here
Detailed Description Window (TTA)
Call Center New Contact MikeLeung
Diseases/Symptoms
Detailed Description

219-47-3879
301-997-8878
Kevin
Gray
1966-12-07 33

Abdominal pain	mild	for more than 4 hrs
Vomiting		

Vomiting
Bowel obstruction
? Constipation
Hyperactive bowel sounds
✓ Vomiting
✓ Abdominal pain
Hypotension
Fever
Renal calculi
? Diaphoretic
Dehydration
Lower pain
Pale
✓ Vomiting
Nausea

ABDOMINAL PAIN

This is usually ill defined but can be very unpleasant and is termed visceral pain. Pain is initially felt near the mid line of the abdomen.

Questions that can be asked:

- What makes it better or worse?
- Is the pain severe/moderate/mild?
- What is the region/radiation of the pain?
- How long does it last?
- Have you tried any treatment?
- Any past medical history (diabetes, hypertension, etc.)?

Appendix E: Instructions for Thinking Aloud

Instructions for Thinking Aloud

We are interested in your running commentary on what you are attempting to do and what is going through your mind while you interact with the TTA. Therefore, we ask you to think aloud and talk aloud constantly. That is, say out aloud everything that passes through your mind for each step.

It does not matter if your sentences are not complete. Just act as if you are alone in the room speaking to yourself loudly.

It is most important that you keep talking. If you are silent for more than 15 seconds, the researcher will remind you to keep talking aloud.

Before turning to the real task, we will start with an example. Please think aloud while you work on the following problem.

Now think and talk aloud while you calculate:

$$476 + 688 = ?$$

Appendix F: Getting Familiar with TTA

Getting Familiar with TTA

You may now try out different functions of TTA. Take as much time as you need and feel free to ask any questions. To make sure you are fully familiar and confident with the features of TTA, please go through the following scenario. Please also keep thinking aloud and talking aloud while working on this exercise.

- Enter *Abdominal pain* as the first symptom
- Click on *Abdominal pain* in the hierarchical list in TTA's Diseases/Symptoms Window
- Enter *Back pain* as the next symptom (notice TTA starts to display its inference when two or more symptoms have entered)
- Click on the other symptoms to view the detailed information in TTA's Detailed Description Window
- Notice a check (√) indicates a symptom that has been confirmed and a question mark (?) indicates a symptom that should be confirmed with the patient
- Move the slider of Refreshing rate to **Max** (notice the contents of TTA's Detailed Description window will be updated continuously)
- Move the slider of Level of details to **High** (notice the contents will be changed in TTA's Detailed Description window)
- Enter *Shock* as the third symptom
- Enter *Peritonitis* to the Disposition in the Task Window (or Drag and drop *Peritonitis* from the hierarchical list in TTA's Diseases/Symptoms Window to Disposition, by holding the **Ctrl** button on your keyboard).

Appendix G: Telephone Triage Task 1

Telephone Triage Task 1

In the next few minutes, you will perform two telephone triage tasks. We would like you to make the process as close to your normal practice as possible, other than thinking aloud and using TTA. For each step, take as much time as you need, as task completion time is not a factor of evaluation.

1. Pick up the phone and carry out your telephone triage procedure as normal, i.e.,
   - ask questions to identify a symptom, then enter it in the Task Window, and
   - ask further questions to identify and enter more symptoms until the likely disease is diagnosed

2. Enter the Disposition in the Task Window according to your assessment

3. Enter the Urgent/Emergent Reason, if any, in the Task Window

4. Alert the researcher that you have completed the above steps.

Appendix H: Telephone Triage Task 2

Telephone Triage Task 2

1. Pick up the phone and carry out your telephone triage procedure as normal, and in addition,

    1.1 ask questions to identify a symptom, then enter it in the Task Window,

    1.2 check out the information in the hierarchical list of TTA's Diseases/Symptoms Window,

    1.3 check out the information in TTA's Detailed Description Window,

    1.4 If you have reached a conclusion about the disease, move on to the next step. Otherwise, go back to 1.1.

2. Enter the Disposition in the Task Window according to your assessment

3. Enter the Urgent/Emergent Reason, if any, in the Task Window

4. Alert the researcher that you have completed the task.

Appendix I: Questionnaire about TTA

Questionnaire about TTA

In the following questionnaire, TTA refers to the information displayed in the Disease/Symptoms Window and the Detailed Description Window. Please answer each question by circling the appropriate number.

For example, if you strongly agree that "there is too much violence on TV programs," you might answer as follows:

There is too much violence on TV programs.

Strongly disagree 1 2 3 4 5 6 [✓ 7] Strongly agree

---

1. Using TTA in my job would enable me to accomplish tasks more quickly.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

2. The information provided by TTA reminds me other information that is relevant to the task.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

3. Using TTA would enhance my effectiveness on the job.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

4. My interaction with TTA would be clear and understandable.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

5. I would find TTA useful in my job.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

6. Learning to use TTA would be easy for me.

   Strongly disagree 1 2 3 4 5 6 7 Strongly agree

7. I have no trouble focusing on my work even though the contents of TTA's windows are updated continuously.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

8. I would find it easy to find the information I want from TTA.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

9. Using TTA would improve my job performance.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

10. I would find TTA to be flexible to interact with.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

11. It would be easy for me to become skillful at using TTA.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

12. Using TTA in my job would increase my productivity.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

13. I would find TTA easy to use.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

14. I did not find the continuous information update by TTA distracting.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

15. Using TTA would make it easier to do my job.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

16. TTA is useful despite that sometimes it displays irrelevant information.

Strongly disagree 1 2 3 4 5 6 7 Strongly agree

17. In your opinion, what are the benefits of TTA?

____

____

____

18. In your opinion, what are the shortcomings of TTA?

____

____

____

Appendix J: Background Information

Background Information

1. What is your highest academic qualification achieved (e.g., Cert., B.S., BSN, or M.S.)? ____________

2. Are you a Registered Nurse (RN) in the State of Maryland? (Please circle)

   Yes No

3. How many years of clinical experience do you have? ____________

4. Have you worked in an ER before? If yes, for how long? ____________

5. Have you worked in a telephone triage call center before? If yes, for how long? ____________

6. Have you used any computer-based disease management system before? If yes, for how long? ____________

7. How would you rank your skill with using Microsoft Windows? (Please circle)

Novice	1	2	3	4	5	6	7	Expert

8. How would you rank your skill with using Microsoft Windows-based applications such as Microsoft Word or Web browser? (Please circle)

Novice	1	2	3	4	5	6	7	Expert