A Task-Centered Approach for User Modeling in a Hypermedia Office Documentation System

Julita Vassileva

Institute for Technical Computer Science
Federal Armed Forces University Munich
85577 Neubiberg, Germany
E-mail: jiv@informatik.unibw-muenchen.de

Abstract

The development of user-adaptive systems is of increasing importance for industrial applications. User modeling emerged from the need to represent in the system knowledge about the user in order to allow informed decisions on how to adapt to match the user's needs. Most of the research in this field, however, has been theoretical, "top-down." Our approach, in contrast, was driven by the needs of the application and shows features of bottom-up, user-centered design.

We have implemented a user modeling component supporting a task-based interface to a hypermedia information system for hospitals and tested it under realistic conditions. A new architecture for user modeling has been developed which focuses on the tasks performed by users. It allows adaptive browsing support for users with different level of experience, and a level of adaptability. The requirements analysis shows that the differences in the information needs of users with different levels of experience are not only quantitative, but qualitative. Experienced users are not only able to cope with a wider browsing space, but sometimes prefer to organize their search in a different way. That is why the user model and the interface of the system are designed to support a smooth transition in the access options provided to novice users and to expert users.

Keywords: adaptation,
adaptive interfaces,
hypermedia and hypertext navigation,
intelligent information retrieval,
office / hospital documentation systems,
task-based context for information retrieval,
task-structures.

Published in User Modeling and User Adapted Interaction, vol. 6, Nos. 2-3, 1996 .

1 Introduction

Browsing is a useful technique for retrieving documents from data-bases (Thompson & Croft, 1989). It has been widely applied recently as hypertext and hypermedia systems have become increasingly popular (Begoray, 1990). The main cognitive advantage of this technique is that users in general are better able to recognize the information they want than to characterize it in advance. The disadvantages of browsing are that it is easy to get lost in a complex network of nodes representing documents and concepts and that there is no guarantee that browsing will be as effective as a more conventional search. If it offers a rich set of links, the system is responsible for helping the users understand what the links mean, how they might be used, and how to find their way in the network. Without this kind of help, browsing can take on the aspect of the user finding her way in a maze, where she can become hopelessly "lost in (hyper)space" (Conklin, 1987). User modeling can help in supporting the user's navigation and information retrieval. A user model is an explicitly represented collection of data about the user which allows the system to adapt its options to the needs of the user. The intensive development in the field of user modeling during the last decade (Kobsa & Pohl, 1994a) makes it possible to consider it as a practical approach for ensuring user-adapted information access in a hypermedia information system.

History of the Project

Our work stems from an industrial project for creating a large hypermedia documentation system for hospitals. Four years ago, such a system was developed at the University Orthopedics Clinic of Heidelberg using a relational data-base (dBase). However, it was not well accepted by the hospital staff and hardly ever used.

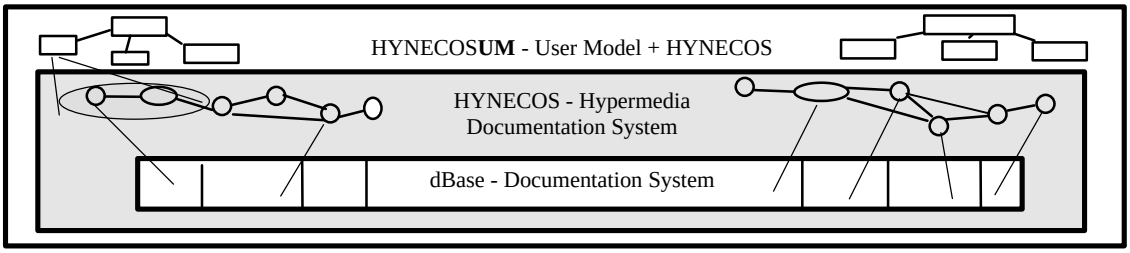


*Figure 1: The "Russian Doll" Architecture of HYNECOS**UM***

This dBase information system was later integrated into HYNECOS (HYpertext Navigation on the Electronic patient reCord on the Orthopedic ward Section). The system was developed two years later at the same clinic within the ESPRIT Project No. 6532 HIFI (Hypertext Interface For Information). The goal of the project was to demonstrate the applicability of the Hypertext Design Model HDM (Schwabe et al. 1990; Grazotto et al. 1991) for design of hypertext-based information systems from relational databases. HYNECOS contains textual and graphical data about patients (administrative data, reports, x-rays etc.), hospital staff (names, telephones and shift-information about all staff members), a medical encyclopedia (diseases, treatments, prognosis) and the location of the wards (room-plans, beds, occupancy etc.). It is implemented in ToolBook 3.0 and runs on an IBM PC 486. The third phase in the development of system was the application of user modeling

to achieve a better user interface, adaptivity and adaptability. The resulting system, HYNECOS**UM** (HYNECOS + UM) is built on top of HYNECOS (see Figure 1), therefore its architecture reminds a „Russian Doll“. In the next sections HYNECOS**UM** will be described and particular attention will be paid on the **-UM** part.

Main Idea of Our Approach

We developed a task-based interface which ensures browsing support for users with different levels of experience. The main idea is to use task-hierarchies to restrict views of information; to define experience levels and to make some hypertext links accessible or not, based on the task hierarchy and the experience level of the user. In this way the interface allows a reduced choice according to the context provided by the current user's task. Since the users population can be characterized by clearly differentiated user types with specific tasks and rights to access information, it is necessary to tailor the interface in an optimal way to the needs of the different user classes. However, the users have also individual needs and preferences, different levels of experience with the information system, which also have to be taken into account. The solution we found (Vassileva, 1994) is to create a User Model (UM) based on a representation of the typical user tasks. Normally these tasks are characteristic of a given user class, so we designed several stereotype models which can be used as a kernel, default for initialization of individual UMs. However, tools are provided for the user to extend and modify her UM (adaptability). Every individual UM is automatically adapted during the user’s work with the system to represent her level of experience. The task model, i.e. the kernel of the UM is designed to be an integral part of the interface, i.e. the user interacts with the system by means of selecting the task she wants to perform. In this way she communicates directly her intentions and the system does not have to infer them from the user’s information retrieval activities. The requirement analysis showed that the differences between the information needs of novice and expert users are not only quantitative, but qualitative. Experienced users prefer sometimes to organize their search in a different way. That is why the UM should support a smooth transition in the options for access to information depending on the user’s level of experience.

Organization of the Paper

This paper describes the whole cycle of development and testing of the system and discusses the UM and the main adaptive features of the system. It starts with analysis of the requirements (section 2). Section 3 describes the task- analysis and -modeling. Section 4 describes the individual user model: the effect of the user’s level of experience on the size of information space she is able to cope with and on her access strategy. Section 5 explains how the UM is coupled together with the information system. Section 6 discusses the adaptivity of the system to the user’s level of experience and section 7 - the tools for adaptability provided for the user. Section 8 presents and discusses the results of field tests. Section 9 compares our approach with other work.

2 Analysis of the Requirements

To analyze the requirements of the application we adopted three of the methods for acquiring cognitive models proposed by Anderson (1985): monitoring the user’s data collection techniques, interviews and questionnaires. The results of our requirement analysis can be summarized as follows:

Standard Users' Tasks

In the medical domain it is convenient to analyze prototypical task situations because many tasks arise from scheduled meetings and activities and are therefore well defined in time and have well defined information needs. After observing the work of the hospital staff at the ambulance and at one ward with HYNECOS and came to the same conclusion - that their interests in the information were comparatively short-term and strongly dependent on their current tasks. That is why we decided that it would be advantageous to use a task model as a basis for modeling the user's interaction with the information system.

Acquiring the Task Model (the Information Needs of Tasks)

The question of acquiring and eventually updating task models during the user's performance has to be decided bearing in mind several considerations. The first question to be answered is whether the users have unstable, frequently changing tasks, or tasks which are expected to evolve in some way. If the answer is yes, then the task model has to be designed so as to allow learning from the user's behavior. In this case it makes sense to perform a "deep" cognitive analysis of the user's activities in order to discover underlying goals, means and restrictions, so that the information needs can later be inferred from the user's feedback or by observation of the user's behavior. However, this is not necessary, if every task has constant information needs, i.e. when the task provides a context in which the needed information (topic) is known in advance (Tyler & Treu, 1989).

In a medical application the tasks related to work with the documentation system have stable information needs which can be acquired at a design stage with standard knowledge engineering methods (Rasmussen et. al, 1994). Of course, it can happen that the information needs of a certain task change. Such changes, however, are normally regulated and concern all the users performing the task. It would be far cheaper and more natural to provide options for a system administrator or for users to change directly the information needs of tasks when this is necessary, than to leave this out of users' control by providing the system with an incremental learning capability.

We don't claim that learning of the information needs of user tasks is in general too expensive and not necessary. As mentioned at the beginning of this section, for an application where the tasks performed by users are not well defined and standardized, i.e. there are no clear boundaries between them or not clearly definable relevance to specific information entities valid across all the users of a given class, it will be important to learn from the user's feedback and eventually change the information needs, the scope and relations between the tasks in order to reflect the individual user's cognitive model of the task. This, however, is too expensive and unnecessary for an application with relatively stable and well defined tasks. That is why in our UM the tasks are fixed in advance and they have pre-defined information needs.

Users Designate their Current Task

In order to suggest relevant information for the user, the system has to be able to recognize the user's current task from her behavior. Several interesting approaches exist for inferring the current task of the user from her information searching behavior. Most of them, however, concern query-based information retrieval from

relational databases (Croft, 1984), (Hoppe, 1992), and they can't be used in our case. The problem of finding the current task or goal of the user in browsing is more difficult than in query-based information retrieval. The reason is that the contents and structure of the query may provide some indirect evidence about the user's goal or task, while the user's browsing activities can be chaotic and non-sequential and they are not necessarily related to a specific goal or task. In principle, one could build hypotheses of possible current tasks based on the user's selection of entities (having in mind the stable information needs of tasks) and then reduce the hypothesis set with every next selection of information entity[1] provided that the user is experienced and is able to select correctly needed for her task. However, the goal of the UM in our case is to support exactly the inexperienced users, who are not likely to browse systematically. Such a user would make mistakes, browse occasionally and therefore the system will make wrong conclusions about the user's current task from her browsing.

It is far more natural to organize the interface so that the users explicitly assign the task they are currently working on. Because the task-taxonomy and the users' understanding of their tasks in our application are very well standardized, it is not obscure or unnatural for the users to name what they are doing. That is why we decided not to infer the users' current task from their behavior, but to design the interface in such a way that the users select the task they want to perform directly from a graphical representation on the screen.

Fixed Hypermedia System

One of the main problems in hypertext and hypermedia design is creating the possible links between entities. From one side, it has to reflect the logical structure of the information and from the other side, it has to be rich in links to allow a wide variety of browsing behaviors. One way of supporting the user in her browsing is to enable or disable some of these links in order to restrict the user to a specific browsing behavior (Beaumont & Brusilovsky, 1995). In this way the chance of "getting lost" in hyperspace is reduced and the browsing space is tailored according to the user's goal / task or area of interest. For example, (Biennier et al. 1990; Kaplan et al. 1993) define a maximum connected graph as an underlying structure of their hypertext systems. By learning from user feedback about the relevance of the selected entities to other entities or to her goals some of the links increase their weights, i.e. the strength of association among the entities or goals. In this way an individualized version of the Hypertext is created for every user. However, in our particular case one of the requirements was that the hypermedia system is a "given" and it can be only "masked" or "viewed" by additional indexing of the links to the entities with respect to the user's current task. This implies that the UM can't be integrated into the hypermedia, but is added as an overlay „extra-index" on top of the hypermedia, to be coupled or decoupled, if needed. We believe this is an advantage, since it allows us to apply this approach not only to hypermedia, but also to any information system without changing its own indexing scheme.

User Classes and Rights of Access

The communication structure connecting work situations and the actors is an important aspect of the design of integrated information systems. In a typical work situation in the hospital, several actors work in a

[1] By "entity" we understand an elementary data type. Following the terminology adopted in HDM, an entity corresponds to the name of a column in a record in a relational database, i.e. a feature shared by a set of records. For example, the entity "diagnosis" is present in the records of all patients.

team whose members are identified with relevance to roles and professional backgrounds, not to persons. Due to shift work requirements, most roles in the work at a hospital are not assigned to particular individuals. Therefore, a stable configuration is linked to work situations and roles (Rasmussen et. al, 1994). We identified five different user classes characterized by different roles: doctors, nurses, administrators, students and patients. These main user classes can be refined further on the basis of their rank, location, and professional background. The different user classes perform different sets of tasks (of course, there can be partial overlapping) and have different rights of access to information. This clear identification of user classes in the hospital domain is an advantage for user modeling. It implies that the initialization of the UM can be done once for every user class (stereotype) and it can be used as a default for every individual user belonging to the class when she starts work with the system. Later on, this stereotype will be adapted, modified, extended or reduced and parametrized so as to account for the individual tasks, information needs and rights of access to the information.

Individual Models

During the knowledge acquisition phase it became clear that users performing the same tasks may have different information-access strategies and preferences. For example, experienced users who had been involved in the design of HYNECOS preferred to be able to jump to the needed entity with a minimum of clicking and often used the direct access options of the underlying dBase. Users who had some experience, but were not experts with the semantic classification based interface of HYNECOS (see Figure 2, the upper part of the screen contains four menus organized according to the semantic classification of entities), occasionally complained that it is boring to click through so many menus in order to reach the needed entity. Novice users found it extremely difficult to understand the way the HYNECOS menus were organized and to find the needed information. They often made mistakes by selecting a wrong item in the first menu and went deep into the wrong branch of the classification, which made it impossible to find the needed entity, especially if the user then tried to get to it by unsystematic browsing through the hypermedia. It became clear that a UM has to support the individual users in different ways according to their level of experience: for advanced or expert-users it has to ensure a fast, direct way of finding the information and to allow various access strategies. For novice users it is important to ensure that the user will always find the needed entity (possibly not directly and with a bit more clicking) and to teach her more advanced access options. Therefore an individual UM is needed, which must be able to recognize the transition from a novice- to an expert-level from the user's behavior and to adapt itself to more appropriate ways of supporting the user in her information access.

Adaptation at Discrete Points of Time and Understandability

Users need to have a coherent mental model of the system. A constantly changing system, even if this is supposed to happen for the benefit of the user, can bring confusion and decrease her confidence (Fischer, 1992), which is unacceptable in our application. In a medical domain sometimes it is of vital importance to access data quickly and the users want to be absolutely confident in the system. Therefore it is desirable to help the users create a metaphor of the system which helps them explain and predict its behavior. After observing users working with different versions of HYNECOS we decided that the policy usually applied in connection

with new software releases (which are announced and explained in advance) is accepted better by the users than continuous invisible adaptation - they wanted to know what could be changed, why, and to decide themselves whether they want to accept the change or not. Another reason not to strive for continuous adaptation is that the working situation in a hospital often requires that a team of users from different classes (e.g. doctor and nurse at the ambulance) shares the system together. In order to prevent the impairment of communication and the confusion caused by different interface- and access- options, there should be the possibility of defining "team models" which provide the same options for all the members of a team, if they want it and agree about it.

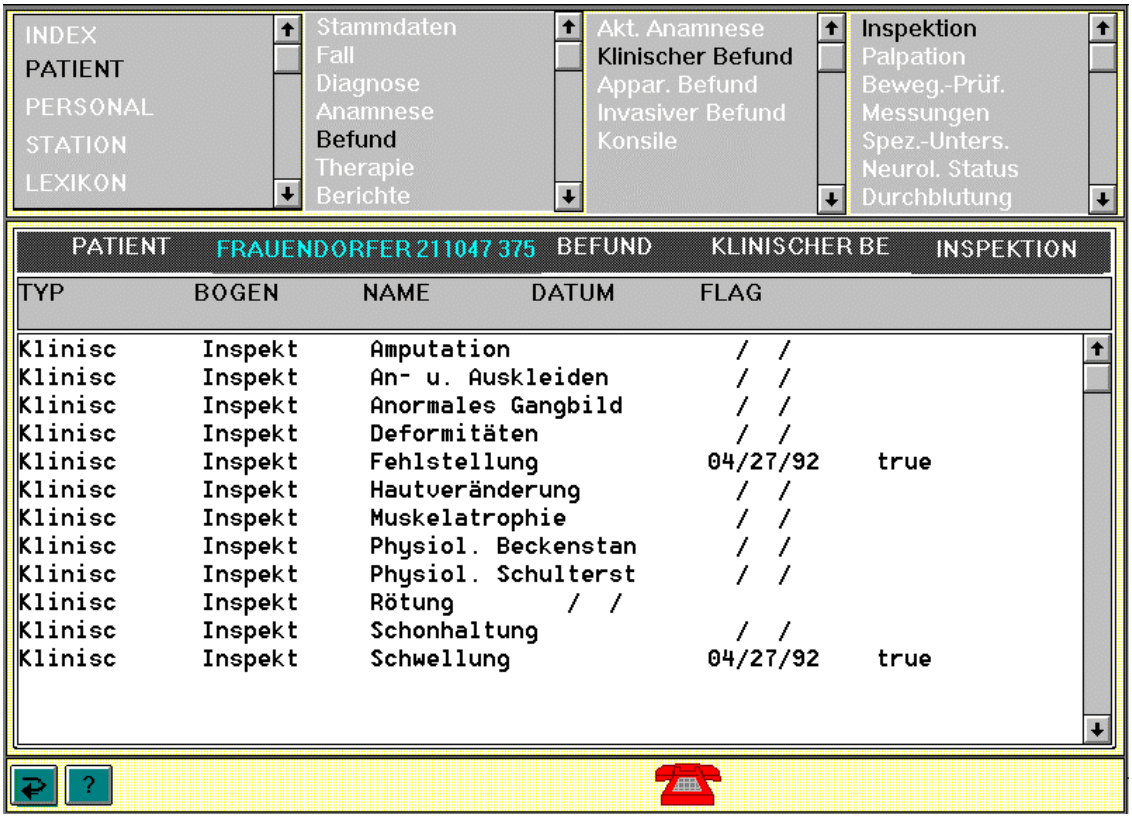


Figure 2: The Semantic Classification- Based Interface of HYNECOS

3 User Class Stereotypes: Task Modeling

Knowledge Acquisition and Task Engineering

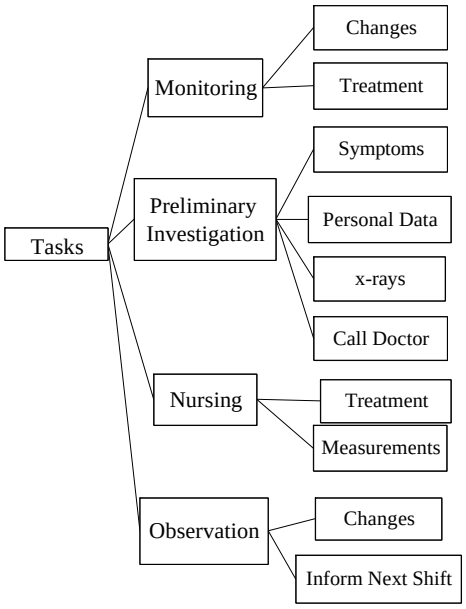
At the beginning of the knowledge acquisition phase we carried out a series of unstructured interviews with the goal to find out the main user groups; to clarify the tasks different persons perform and the understanding of the task-needs and -boundaries shared by the users. The interviewers were trying to identify typical situations and tasks. We asked the users to name all functions and tasks in which they would work with the information system and thereafter the information needs of the tasks. In order to obtain a systematic insight we asked the users to describe the functions they participate in during the course of a shift.

The results from the unstructured interviews showed that the tasks in a hospital environment in general are quite well standardized and characterized with stable information needs (forms, letters, tables etc.). This result is in agreement with (Rasmussen et. al, 1994) who shows one possible time-model of the tasks performed at an operation wing of a hospital. Traditionally, the goal of a task analysis is to produce a description of the interaction between people and their work environment in terms of sequences of actions. Work in general, can be pragmatically decomposed into segments with respect to time, location, information needs or with respect to functional content, whichever is convenient for analysis and needed for the application. However, general modeling of the time-organization and the causal relations between the activities in the hospital was not interesting for us, since the information needs of the tasks were discrete (i.e. specific for every single task) and therefore not related to the temporal- and causal-aspects. Our goal was NOT to develop an expert system that assists the users in carrying out their tasks by advising them what to do next or how to do it, but only to ensure a task-based context for information retrieval. Modeling a cognitively plausible structure of tasks with respect to aggregation and generality is enough for this purpose.

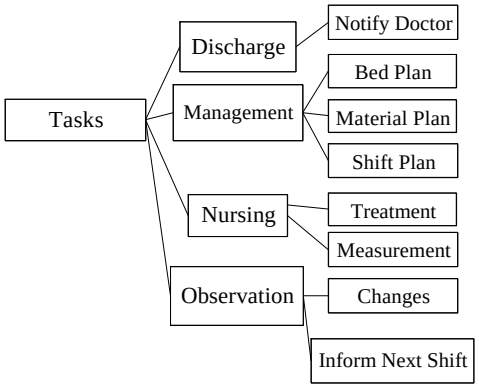
In order to identify a set of typical activity elements for every user class which implies specific information needs we carried out a series of structured interviews. The main idea was that the first decomposition of general activity of every user class should correspond to the labels used by the (Rasmussen et. al, 1994). Together with one highly motivated user (a doctor who had been involved in the design of HYNECOS) we developed a decomposition of the typical tasks he performs. Using this specific task decomposition as an example, we interviewed selected users, whom we considered typical user class representatives. They created variants of this scheme corresponding to their tasks. In this way we obtained several different task-decompositions, like those shown in Figure 3.

During the structured interviews the users spontaneously described their typical tasks (those involving work with the information system) as hierarchies. This is not surprising since the requirement to describe their activities systematically and independently from the time- and causal- relationships, resulted in isolating only the abstraction and aggregation links among the tasks. The validation of the prototype task-models showed that this hierarchical structuring is not occasional, but corresponds also to the abstract cognitive model the users have about their tasks. Once again, we have to say that this is not the only possible cognitive model which our users have. It is very probable that they have in the same time another, time-based cognitive model, which is linear (with loops). For example, if we had instructed them to show what they are doing in the course of the day including the temporal and causal relations, including those tasks for which they didn't need the information system, we would have obtained far more complex structures. However, as mentioned before, our goal was not to develop a general task model of the hospital activities, but only a cognitively adequate user-focused subset of the tasks which involve work with the information system. The users found it reasonable to represent the tasks as hierarchies with upper nodes corresponding to more general tasks or tasks representing a set (sequence) of subtasks.

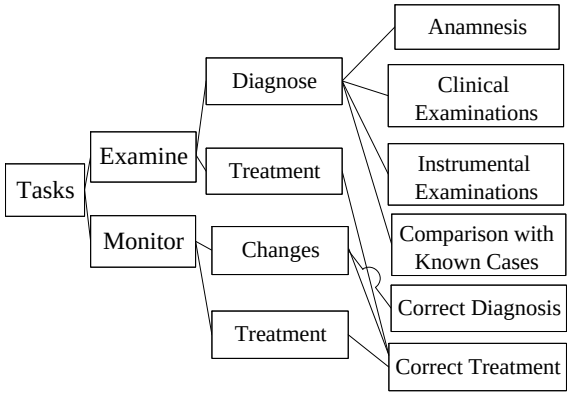
The prototypical task hierarchies obtained during the knowledge acquisition were, as expected, shared by the representatives of user classes taking different roles in the hospital work situation. These classes are defined by the work location (ward or ambulance), profession (e.g. doctor, nurse), and rank. Therefore the prototypical task hierarchies constitute the user-class models. They are the kernels (default initialization) of the individual UMs.



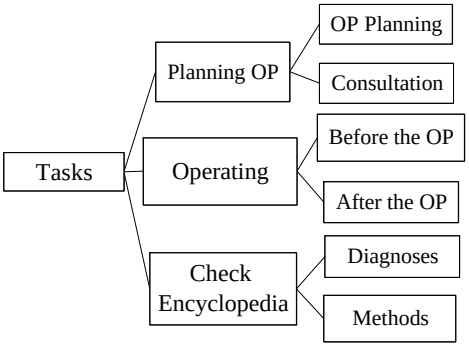
a. Nurse at the Ambulance



b. Nurse at the Ward



c. Doctor at the Ambulance



d. Doctor at the Ward

Figure 3: The Prototypical Task Hierarchies of Four User Classes

Defining the Information Needs of Tasks: Views

Every task implies specific information needs, defined during the task analysis. The hypermedia entities considered as necessary to carry out the task comprise its "view". The user ideally should not be restricted to these entities only. That is why the entities in a task view are actually the starting points for browsing in the Hypermedia system.

The hierarchical organization of tasks with respect to aggregation and abstraction implies that more complex or abstract tasks require more information than their subtasks. In all task hierarchies that we were able

to configure, it turned out that information needs (views) of higher level tasks include the sum of the information needs (views) of all their subtasks. The generality of this result is debatable. We justify it with the following arguments.

Since the task-hierarchies are organized with respect to abstraction and aggregation links, every parent-task (more general task) may represent

- a task of selection among alternative sub-tasks, or
- an aggregation task which is performed by (a sequence of) all its subtasks.

An aggregation task is normally an abstract task which is introduced to name a group of sub-tasks and eventually to introduce some order in executing them. That is why it can be expected that its view can include some information specific to the parent task concerning the decision of how to order or control the execution of the subtasks. Normally, the information needs of all the subtasks will be distributed along the time-axis, but still they all will be necessary for completing the parent task. In the case where the user wants to perform the parent-task without specifying which sub-task she wants to perform, she will need access to the information needed by the parent task and to the information needed by all of the subtasks. We call this definition of the information needs of a parent-task "a summary view" (see Figure 4).

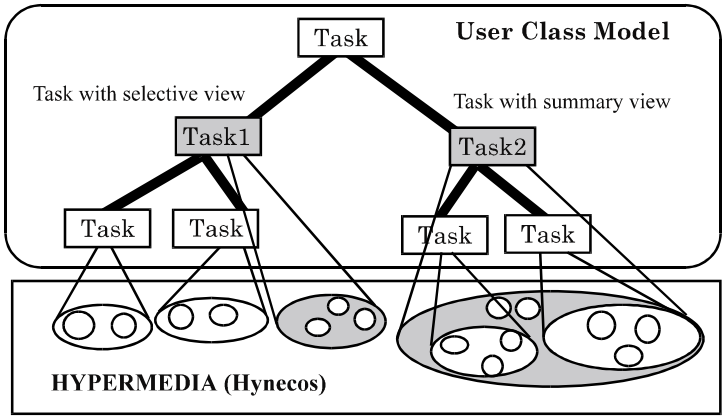


Figure 4: A Selective View and a Summary View from a Higher Level Task

A selection parent task represents a point of decision of which sub-task to choose. Therefore, the information needs of such a parent-task are defined as the information needed for making the selection.

However, the user often takes the decision for selection of which sub-task to perform next after investigating the information related to the sub-tasks. In these cases it would be convenient to have all the entities related to the alternative subtasks accessible directly from the parent-task. For example, let's say that the parent-task is "Diagnose" (see Figure 3-c) and the sub-tasks are "Anamnesis", "Clinical Investigations", "Instrumental Investigations", and „Comparison with Known Cases“. Each of the subtasks is associated with information entities including results from different types of investigations, a table of the history of the disease, data from the medical encyclopedia, etc. The specific information needs of the parent-task (selection) depend on the selection criterion. In some cases it might be possible to select a subtask without consulting the information related to the other subtasks (for example, to select „Comparison with Known Cases“). However, in most cases

the user selects a subtask after checking certain information concerning one or more sub-tasks. For example, she can decide to select „Instrumental Investigations“ only after discovering that the results of a certain clinical investigation is doubtful. So, in order to decide which subtask to choose, the user needs to have this information at the point of selection, i.e. at the parent task. That means that a „summary view“ would be appropriate for this case. Considering that a situation similar to the example might happen, we decided that a summary view should be provided for selective-parent tasks too.

The tasks define not only rights of access (by means of viewing the needed area of entities in the hypermedia), but also rights to modify the contents of these entities. One way, at least, of reducing the risk of data-corruption and loss, is to ascribe rights to modify data only to the tasks that are expected to change this data. For example (Figure 3-c), during the task "Instrumental investigations", in the context of “Diagnose” the doctor needs to know the patient's results, but she is not supposed to change them. She is only allowed to modify the patient’s diagnosis.

4 Individual User Models: Level of Experience

An individual UM is based on a task hierarchy. It can be either the task-hierarchy of the corresponding user class, or a modification of it, created manually using the adaptability tools (see section 7). Since the main characteristic which influences the individualization of information access in our application is not so much the task-hierarchy, which is more role (or user class) specific, but the level of experience of the user with the system, we shall concentrate mostly on it in this section.

The results of the requirement analysis showed that the user’s experience in work with the system influences:

- the task-context i.e. the amount of information about the task, where the user can find her way;
- the access strategy, i.e. the ability to cope and the need to use alternative access strategies.

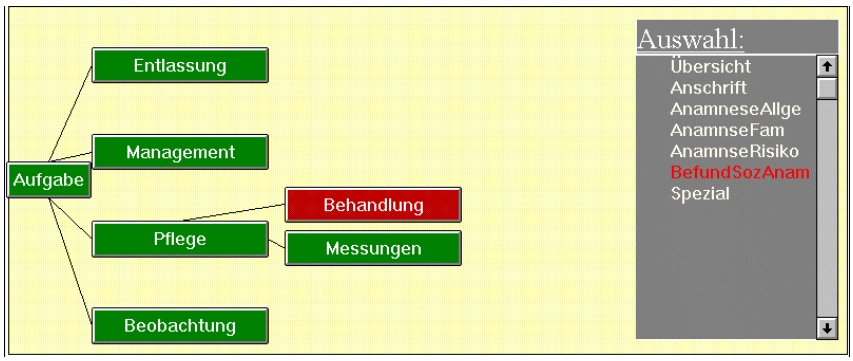


Figure 5: A Task Dependent View

The Selection menu (*Germ.* Auswahl) in the right part of the screen shows the names of the entities related to the current task „Treatment“ (*Germ.* Behandlung). All of the entities are „read only“ except one: „Domestic Situation“ (*Germ.* Befund Soz Anamnese). The same task hierarchy in English can be seen in Figure 3-b.

The user’s experience level is represented by a parameter associated with every task in the user’s task representation. It can have three values - novice, advanced and expert which define different characteristics of

the task's view of the information entities. In this way the individual UM is an overlay model over the user's task hierarchy expressing the degree to which every task is mastered with respect to ability of the user to cope with the information space related to this task. The degree of experience with a given task influences the size and the type of the view provided by the task.

Ensuring a Task-based Context for Information Access

A task-defined „view" consists of all entities related to the task. It is realized by menu containing the names of all task-related entities which are used as starting points for browsing in the hypermedia. This menu is called „Selection" (*Germ*. „Auswahl")-menu (see Figure 5). Two types of task-relevant views over the hypermedia can be defined:

- "Restricted browsing" - by restricting the user's access only within the entities directly related to the current task. In this way the normal hypermedia links outside the current task-view are disabled ("masked").
- "Free browsing with an anchor" - by allowing browsing using the view - entities as starting points (Figure 6). The user can access all other hypermedia entities that are not explicitly forbidden to her or her user class.

The user of HYNECOS**UM**'s interacts with the system by selecting the task she wants to perform from a graphical representation of the user's task decomposition on the screen (Figure 5). Access to the Hypermedia is not allowed, if the user selects a higher-level task on which she is a novice. She has to specify one of the subtasks and so on until either she selects a task on which her experience level is advanced or expert or until she reaches a task that can't be further decomposed. The corresponding view of the Hypermedia is then provided. If the user is advanced or expert on a higher level task in the task hierarchy, she will be provided with a summary view consisting of the views of all subtasks of this task. In this way she will get more starting points for browsing in the hypermedia. If the user is a beginner - she can only get a restricted browsing view, while if she has experience on the task, she can be provided with a free browsing type of view (if she wishes). In addition to the free browsing, an expert-user can use the access options offered by the underlying relational database (see Figure 6). For example, after selecting the entity "Main Data" (*Germ*. "Stammdaten"), the user can search for the same entity of another patient by different criteria, e.g., name, ID, insurance company, ward name, diagnosis, bed number etc.

The observations made of users working with different sizes of task-defined views during the testing phase showed that in our application the users' degree of experience with the system determines the size of the view with which she is able to cope. Experienced users prefer to use wider summary views from higher-level tasks which allow them to select directly the needed information, since they know exactly where to find it. In contrast, inexperienced users prefer to select a specific subtask to get the corresponding smaller view where they can't get lost.

We have now described how the local level of experience of the user on every task defines the information context of the task by means of the size of the information space and type of view which enables or disables the possibility of browsing. One can think also of a global level of experience of a user defined across the whole task-hierarchy which grows by propagating upwards through the hierarchy. The level of experience of

a parent task depends on the level of experience of its children-tasks. More specifically, it is the minimum of the values of its children tasks. The assumption behind this is that the user is not able to cope with the broader view of a general task, if she is not able to cope with the subsets of this view belonging to the single sub-tasks. After some time of work with the system, regions of the user's task hierarchy appear where she is experienced. They are initially closer to the leaf-tasks (see Figure 7). With the increasing experience of the user with HYNECOS**UM**, these regions spread up the task hierarchy towards the most general task (root-task). How this happens will be explained in section 6.

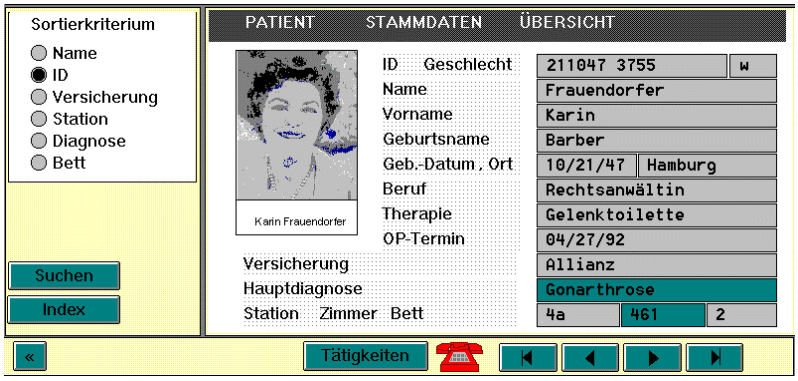


Figure 6: Free Browsing Type of Viewing

The screen-shot shows the contents of one information entity related to the user's current task (all patient data is fictitious). From here the user can browse following the Hypermedia links (the dark fields and buttons). For example, she can jump on the encyclopedia entity "Gonarthrose", click on the field "461" ("Room /Zimmer)") to get a map of the room, where she can click on every bed to see which other patients lay in the same room and browse further to get more data about them. She can also search directly in the dBase using the search options in the left side of the screen, for example, to see the same data ("Stammdaten") for other patients. All these entities are not relevant to the current task, but they are accessible by browsing. In case that a „restricted browsing" was the current type of view all these fields would have been unavailable for selection.

In summary, the task-based views depending on the user's level of experience provide an appropriate context for information retrieval. This is obtained by managing the size of the task-defined view of the information and the freedom of browsing. By gradually increasing the navigation space together with the user's global experience moving upwards toward the root of the task hierarchy, the user is always interacting with the system in an appropriate context.

The task-based organization of the interface proved successful. In the field tests the users had no problems in selecting their tasks from the task-hierarchy. All the users were using the task-based interface in the beginning, some continued preferring it even after they had reached an ultimate level of experience "expert" at their root task, though in some cases they switched to a different way of searching. Our experiments showed that especially novice users worked much more confidently with the task-based interface and found the needed information faster than when given direct access to the hypermedia (i.e. working with HYNECOS).

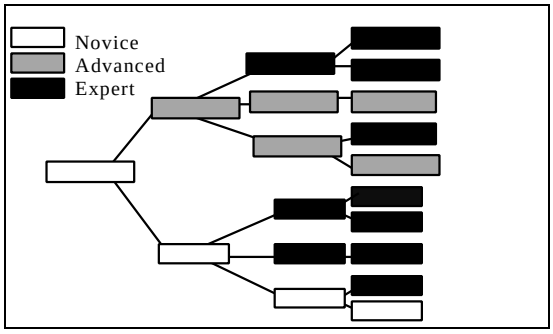


Figure 7: An Individual User Model: "Global" Level of Experience

The model represents the user's task hierarchy where the color of every task represents the level of experience of the user on this task. The global level of experience is the proportion of the tasks with advanced or expert level over the number of all tasks (79 % in this case).

Accommodating Different Access Strategies

So far, our main assumption has been that the current task performed by the user always provides the needed context for browsing and the only difference between experienced and novice users concerns their ability to cope with wider task-dependent information context, i.e. to work on more general tasks (i.e. on several sub-tasks simultaneously). The requirement analysis, however, showed that the differences between the information needs of a novice and an expert concern not only the size of the browsing space, but also the access strategy.

Though the information needs of the users in our application domain are triggered in most of the cases by the standard tasks they are performing, there are cases when users approach the system with a different goal or attitude, not related to a specific task. For example, while performing the task "Nursing" (see Figure 3-b), the nurse may want to know suddenly something not directly related with the current task - the amount of a certain type of drug in the storage of the ward. Now she has to remember for which task this data is relevant and to select this task - in this case "Management". However, is it really worth changing the context of the current task (the selected patient, entity etc.) just for a small "jump" aside? In this case the task-based interface provides no support, just the other way around! When such a situation arises, users typically try to browse starting from the current entity. However, free browsing in the hypermedia in most of the cases is the most inefficient way to find the needed information (a result valid across a wide range of information retrieval applications - see Rasmussen et al., 1994). Other authors also point out the necessity of combining browsing with other search strategies (Thompson & Croft, 1989), (Dumais et al, 1992). There is a need to provide a mechanism for access corresponding to an alternative strategy, which is task-neutral. Such an alternative provides the original interface of HYNECOS which is based on a semantic classification of the information, a way orthogonal to the task-based way of indexing of the hypermedia entities (see Figure 8).

It is likely that experienced users would be more brave than novices and try to take "shortcuts" or to use alternative search strategies. Their confidence in their ability to cope with the system supports their wish to explore other strategies of obtaining information in order to minimize the amount of clicks or cognitive load. In the example, the problem is that using the task-based strategy requires a bigger cognitive load (remembering the name of the task that provides access to the needed entity) and possibly, bigger ergonomic effort (more clicking). Results of empirical studies in other domains (e.g. design) show that expert users are more inclined to take

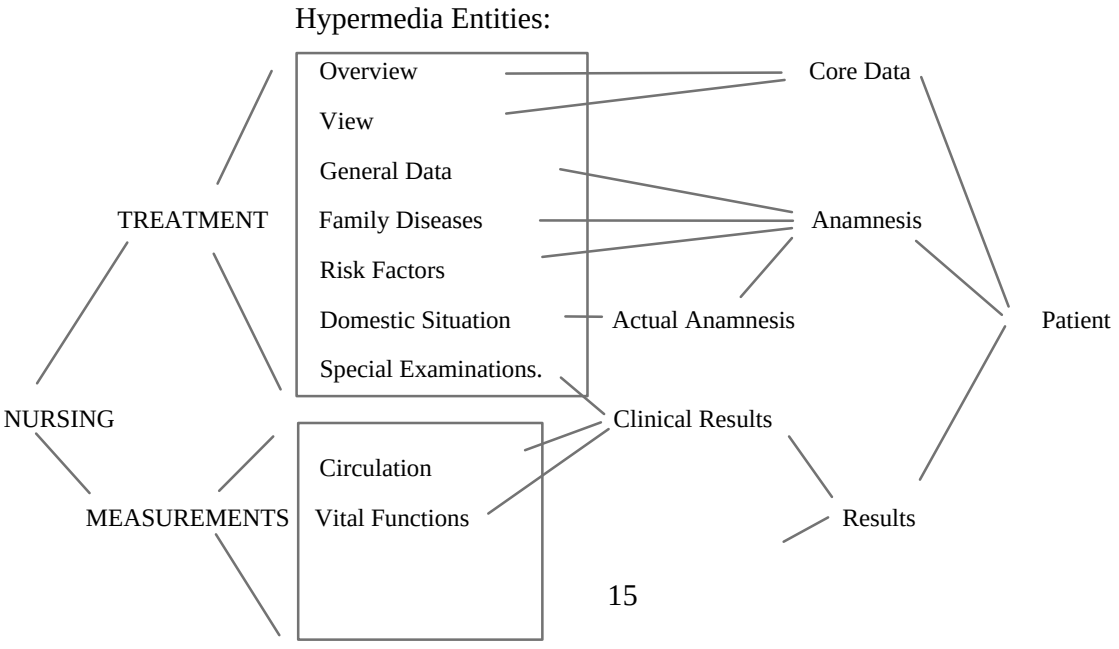
"shortcuts" than inexperienced users. Instead of following a plan, they tend to act opportunistically, in order to minimize their cognitive load (Visser, 1994).

If the user is experienced in working with the system and knows how the entities are semantically classified, she can organize her search according to the semantic classification of the information space and the current task remains (temporarily or completely) implicit. Knowledge of the alternative access strategy, i.e. of the semantic classification of information is not necessarily related to the experience of the user in working with the system. However, the need for alternative access strategies becomes bigger with the growing experience (practice and confidence) of the user. Our experiments (see Section 8) also showed that users with higher global level of experience (see Figure 7) employ alternative access strategies.

In order to help in learning how to use an alternative access the UM supports a simple teaching facility for novices. It consists of presenting in parallel two alternative interfaces corresponding to the two different access strategies: a graphical representation of the task hierarchy where the user can select the task she wants to perform and four hierarchically organized menus corresponding to the semantic classification of entities (compare with the original HYNECOS interface in Figure 2). For a novice user only the task-based interface is available for selection. However, when an entity is selected from the "Choice" (*Germ.* "Auswahl") -window (see Figure 9), the HYNECOS menus in the top part of the screen show how this entity is semantically classified by highlighting the corresponding menu items. Gradually, (in parallel with their increasing level of experience), the users are allowed to use the semantic classification-based interface of HYNECOS. This is explained in more detail in Section 6.

5 Coupling the UM with the Hypermedia System

The UM doesn't modify the underlying structure of the hypermedia links, but just provides views on it. The viewing mechanism is based on additional links from the tasks in the UM to the hypermedia entities. At the same time it ensures task-dependent data-protection. It is based on weighted links between the tasks and the hypermedia entities.



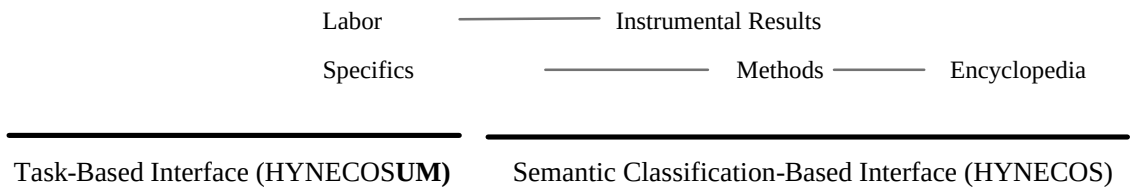


Figure 8: The Two Types of Interfaces to Hypermedia Entities

Unlike the normalized weights of Hyperflex (Kaplan et al., 1993) which denote the strength of association between a goal and a node, we use discrete values:

-2 - forbidden for the user entity, display blocked from any task;

-1 - "forbidden" for the task entity, display blocked for current task;

0 - neutral entity, not related to the task, but accessible through free browsing;

1 - connected to the task with "read only" rights of access;

2 - connected to the task with "read and write" rights of access.

The two types of viewing are implemented in the following way:

- free browsing with an anchor — all the entities irrelevant for the task are considered by default as connected with links of value 0. They are not included in the task view, but the user can reach them by free browsing following the standard hypermedia links.
- restricted browsing — all entities irrelevant for the task are considered by default as connected with links of value -1. Entities connected with a link of value -1 to the current task are not available for browsing and they are “masked”, i.e. their display is blocked.

The display of every entity is "filtered" through the current task of the user. If the link between the task and the entity has the value -1, the display is blocked and the user has no way to get this information. If the link has value 0 or 1 and the user has selected the entity it is displayed (but can't be changed by the user). If the value is 2, the user can change the contents of the entity.

If a certain entity is forbidden for a particular user or user class, the link between this entity and the root-task of this user's (user class) task-hierarchy has the value -2. Even if the user is an expert and gets a view over the entire hypermedia from her root-task, or she uses the HYNECOS interface, she will not be able to access this entity, since she is always by default performing the root task.

Straightforward procedures change the read/write rights of a given user-class or individual user. The mechanism for defining a task-view includes two parts:

- assigning values to the links between the Hypermedia entities and the lowest tasks in the hierarchy which need them;
- propagating the values of the links upwards in the hierarchy so that the parent-tasks get access to all entities accessible from their sub-tasks.

The first part of the mechanism is carried out during the creation of tasks. When a task is created, the entities necessary for completing the task are selected from a list of all hypermedia entities. For each selected entity a link with the assigned value is created between the task and the current entity. Only specially authorized

users are allowed to connect tasks to entities with read and write links (i.e. links with value 2), because otherwise the task-dependent write-protection can be violated. Normal users are allowed to create tasks from which they can only read information, i.e. a user is provided only with the "create access link" procedure (links with value "1"). As mentioned before, it is impossible for a non-authorized user to invoke this procedure to change a link from -1 to 1, since she can only operate on entities visible to her (with link 0 to the tasks of her former task-hierarchy) that can be reached through browsing. So, users can't violate their access rights via the adaptability tool.

The second part consists of propagating the view of the sub-tasks up to their parent-tasks the task-hierarchy. The "free browsing" type of view-definition is done by creating links from every parent-task to the entities linked to its sub-tasks with values equal to the maximum value - link to any entity from any of its subtasks. The "restricted browsing" way of viewing is obtained by the same procedure, followed by disabling (assigning a weight of -1) to all entities which are not connected with positive weights to the task.

The procedures for taking away write- or access-rights from a task to a given entity and limiting the access to information for a given user class or for a single user consist of decreasing the values of all links from this entity correspondingly to 1, 0, -1, and -2. The propagation of the weight (-2) goes in the opposite way, from the root to the leaves in the hierarchy. In order to prohibit access to a given entity for a given user, it is not necessary to find out all tasks which potentially require the information, it is enough to set the link from the root-task to this entity to -2.

6 Adaptation

Adaptation is a notion referring to the ability of the system to change dynamically according to the changing user's needs in order continually to maintain the appropriate context for interaction (Kuehme, 1993). In our case, as in the majority of cases (Norcio & Stanley, 1989), the most important time-dependent factor is the user's experience, therefore the system must have a means for finding out and reacting to the changes in the user's level of experience. One can think of other factors that are time-dependent: for example, the information needs of tasks, or the task-model itself. In our application, however, these factors are not dynamic. The changes of task-structures and information needs normally are regulated and the necessary changes can be conveniently made by the system administrator, if she is provided with the necessary tools. Ensuring adaptivity with respect to these factors would undoubtedly be an interesting research problem, however, it would have been too expensive in the context of our project. Therefore, we provide tools to support adaptability.

Inferring the User's Experience Level on a Given Task

The user's browsing behavior, ability to select correctly and the type of mistakes she makes are taken as evidence of the user's changing level of experience.

A user who works with HYNECOS**UM** for the first time is considered as a novice at all tasks of her task-hierarchy. If there is evidence that she is able to cope with the browsing space of a certain task, the level of experience on this task is increased. Parent-tasks can only increase their level of experience, if the user has

demonstrated the higher level of experience on all subtasks of this task. In this way, the level of experience gradually climbs up through the branches of the task-hierarchy. The increase in the experience level starts from the leaf-tasks (i.e. the most specific tasks which can't be decomposed further) since their views are smaller and the user learns faster how to select correctly (see Figures 4 and 7).

Technically the user's experience on a given task is represented as a number, called a task's score, which is calculated as a sum of points. Depending on the task score, the user can fall into one of three possible categories: a Novice, Advanced or Expert on the task.

The user's navigation actions are recorded and if patterns are found which imply that the user's proficiency has increased, points are added to the task's score until a threshold (defined empirically) is reached. Then a flag is set indicating that it seems appropriate to increase the user's level of experience. The patterns of behavior concern the user navigation through the task hierarchy, the time spent on an information item (it is important to be more than a certain threshold to believe that the data was used and not just browsed and considered to be inappropriate), and the correct attempts to select from the semantic classification menus of HYNECOS.

A point is added to the score of a "novice"-task in every case when a sequence of actions defined as a "good" task-navigation behavior is recognized: e.g. selecting a sub-task; then selecting one or more entities from a view provided in the "Selection" window; using the provided data (spending more time than a certain threshold); not trying to browse in the hypermedia and being able to come back to the super-task without using the "Help" or "Reset" buttons. At a task with "advanced" level of experience, the increase in the task score happens as a result of correct use of alternative access strategies, for example, using correctly the semantic classification menus or the direct access option to shrink the view provided from the current task.

The same mechanism works in the opposite direction for decreasing the user's level of experience. The user is "punished" for incorrect navigation actions throughout the task hierarchy, in the HYNECOS menus, for asking for "Help", "Info", or using the emergency buttons. For example, when the user selects sibling-tasks one after the other, without selecting any entity from the view provided for the first selected task, this is considered as a sign that the user has selected a wrong task the first time and that she is not familiar enough with the principle of the task-based interface. Therefore, points are subtracted from the user's score on this task. Another significant pattern of behavior is chaotic browsing in the hypermedia from the provided starting points which is terminated with emergency button ("Reset"). A too short time spent on an information entity once it has been selected shows that the entity was wrongly selected. Trying to click on items of the semantic classification menu which are unavailable at the moment, as well as trying to select already selected items; trying to get back to a familiar place by back-tracking (clicking the "<<"- button) - these are all patterns of behavior which lead to "punishing" the user by reducing points from the current task's score. If the threshold to the next lower level of experience is reached, a suggestion of reducing the level of experience for this task is generated by the system.

Strongly adaptive systems, however, threaten the user with a feeling of a loss of control. That is why we decided that the system's adaptation to the user's needs has to be carried out not continuously, but at discrete points in time, and only after the user has given specific permission.

When there is evidence that the user's recorded level of experience can be changed, the system stores this in a list of suggestions together with a pre-stored explanation of how this will effect the user. The list of suggestions can be read by the user if and when she wishes. For example, the system can be configured to

present the list of suggestions at the beginning of every week, or with every second use, or to present it only when requested. If the user decides to accept the suggestion, the system updates the level of experience which comes into effect from this moment. In this way our system follows the strategy of adaptively supported adaptability (Oppermann, 1994a): supporting the user's adaptation of the system by initial adaptive suggestions showing the rationale for the adaptations and the way of performing them, and adapting only after receiving users' consent.

Effects of Adaptation

The system's adaptation to the user's level of experience concerns the size and type of the task-defined views (the task-based context) and the access strategies which are available for the user.

Adapting the information context to the current task

As explained in section 4, in our approach appropriate context for information retrieval is obtained by selecting starting points for browsing appropriate for the current task and reducing the size of the browsing space for inexperienced users. By gradually increasing the number of starting points and the freedom of browsing together with the user's growing global experience, the user is always interacting with the system in an appropriate context.

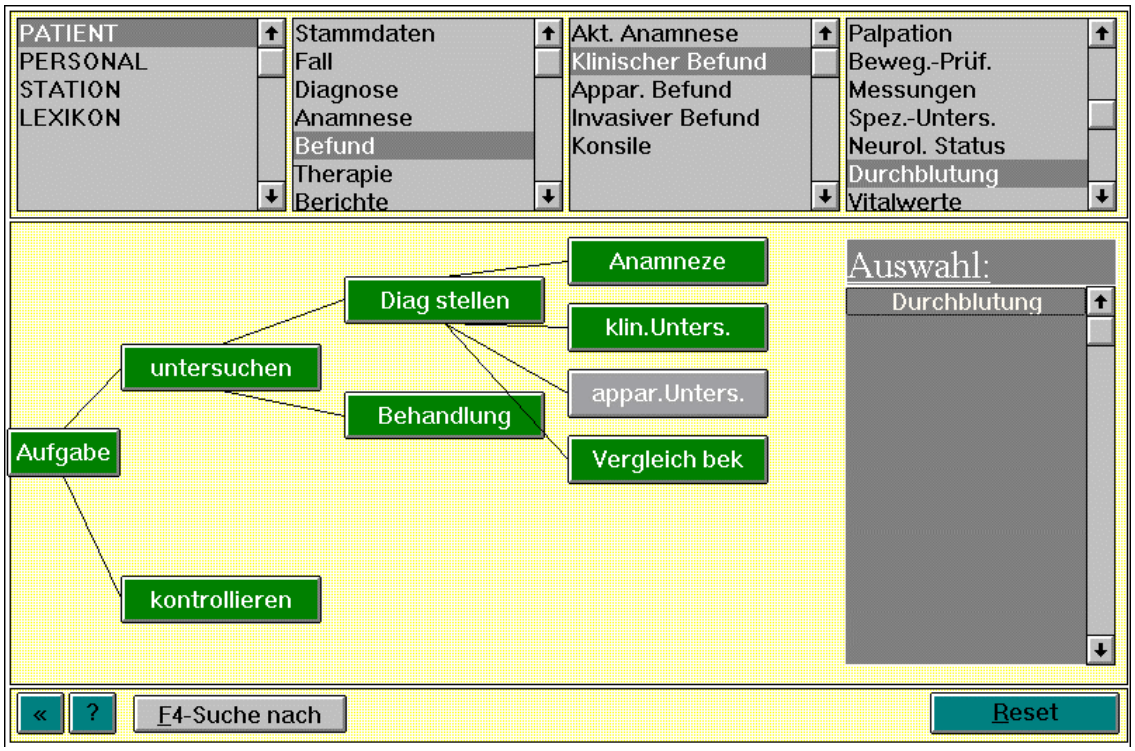


*Figure 9: Alternative Search Strategies Offered by the HYNECOS**UM** Interface*

The user is a doctor at the ambulance (see in Figure 3-c this task hierarchy in English) and a novice at the current task " Instrumental examinations" (*Germ*. „appar.Unters"). The semantic classification menus in the upper part of the screen are not available for selection, but they show how the selected entity from the task-view is classified. Direct search options (button F4 in the lower part of the screen) are currently unavailable (dimmed).

When a user is a novice on a given task she is allowed to browse only within the information entities provided for the task (restricted browsing). If the task is not a leaf-task (i.e. if it has sub-tasks), the user has to select these sub-tasks first before being allowed to access the hypermedia entities. When the task score of the current task exceeds the task-threshold for level "Novice" and the user accepts the suggestion of being classified as "Advanced", the system checks whether the level of experience of the user on all sister-tasks is already "Advanced" in order to see whether to propagate the higher level up the task hierarchy. The increased experience level at the task means, that the user can (if she wants) get access to more starting points in the hypermedia (wider view) and to a free browsing type of viewing. In case she gets lost, she can get back to the starting points by clicking on the task-button again.

If all of the sister-tasks have level "Advanced", the system suggests increasing the level of the parent-task. If the user agrees, the system provides the possibility to get a summary view from the parent task and a free style of browsing (both of the features the user can accept or reject). In this way, by propagating the higher level of experience up to the root task, the user is allowed to browse freely in the Hypermedia and to get wider views (more starting points for navigation) to the Hypermedia without having to specify precisely the task she wants to perform.

The transition from "Advanced" to "Expert" level at a given task doesn't concern the amount of information the user gets for the task. It concerns only the freedom of the user to use alternative access strategies. This will be discussed in the next section.

Adaptation to the user's access strategy

Two interfaces are incorporated in the system corresponding to the different access strategies (the task-based one and the semantic-classification one). There is also an option for direct access which is available at any time for the expert user by pressing a function key (button F4 in Figure 9 below) - a window appears where she can write a query. The task-based interface is orthogonal to the semantic classification interface (see Figure 8): it reflects a different cognitive access strategy. The user can use either of them or a combination or both. For a user who is searching for certain information outside the context of the specified task, it is more appropriate to use direct access or the original HYNECOS interface. For a user who approaches with a specific task, it is more efficient to use the task-based interface.

In order to create an image of the system consistent for all users and to support the learning of the more difficult semantic classification interface, the HYNECOS-menus (in the upper part of the screen), the task buttons (in the central part), as well as the button for direct access (below), are all displayed (see Figure 9). If the user is a novice on the currently selected task, the HYNECOS menus and the direct access button are not yet available for selection — she can only use the „Selection"-menu where the relevant entities for the selected task are presented. When the user chooses an item from this menu, the items which correspond to the semantic classes containing this entity in the inactive HYNECOS menus on the top of the screen are highlighted to show the user how this entity is semantically classified. In this way she sees what selections she would had to make, if the menus were active, in order to find this entity.

If the user becomes "Advanced" at a given task, the HYNECOS menus are enabled for selection, but the user is still restricted to select only among the items from the view provided from the current user's task. In

this way the user is stimulated to use the HYNECOS menus to find the information related to the task, in case she gets lost in her free browsing. Other ways to get back are to select again the current task from the task hierarchy or to use the "backtrack" option of the hypermedia system. However, they lead to reduction of points from the task's score, while using the HYNECOS menus is rewarded by increasing the task score.

If the user becomes "Advanced" on a task that is located higher in the task-hierarchy, she will get a wider view composed of the views of the subtasks, i.e. a bigger choice of entities in the "Selection"-menu. This makes the access to the data easier since all the entities are directly accessible from the higher order task. However, in some cases this can be inconvenient since the "Selection"-window has a fixed size and if the task-view it too wide, one has to scroll to select the needed entity. Thus the user is stimulated to use the semantic classification menus since she can additionally reduce the choice by selecting the logical type of the entity from the standard HYNECOS menus on the top of the screen which have now been enabled (this is awarded by adding points to the current task-score). Of course, the user can shrink the "Selection" menu also by selecting one of the sub-tasks, but for this is she will be punished by subtracting points from the task score.

So, at every time the user can select either as before directly from the „Selection" menu, or by using the menus corresponding to the logical classification of the entities, or by selecting a sub-task as before. During the field-test interviews, this redundancy was commented on by most of the users as an advantage, since they were confident that there are several ways to achieve the same effect, if they have difficulties with either of the options.

If the user becomes an expert on a given task, she is provided with a summary view of the views of all sub-tasks, with a free style of browsing and the possibility of using the HYNECOS interface freely to search through all the hypermedia. In addition, the user is allowed to search directly by name or keyword for the needed data.

In summary, with the growing experience of the user, the system provides her with a wider spectrum of search strategies.

7 Adaptability

User-adaptable systems support users in modifying systems according to their own needs (Fischer, 1992). Our architecture for user modeling allows the user to adapt her individual model. A set of tools are provided for this purpose: a task-editor, allowing the user to graphically create task-structures and define the information needs of tasks (views of the hypermedia); a user-class editor which allows her to define new user class-stereotypes (i.e. to name task hierarchies, to grant or restrict rights of access to information), and an individual model editor (allowing the user to change the level of experience and the type of viewing of every task and to enable or disable alternative access strategies: direct keyword search, task-based search, or the HYNECOS menus).

Definition and Modification of the Task-Hierarchies

We have developed a graphical task editor and a library of task-aggregates which allow the user to define a task hierarchy of her own, i.e. to create herself her individual model. This graphical tool supports the creation of new task-hierarchies and the modification of existing ones. The user can define her own tasks and add them to the library of task aggregates. The task-editor allows her to select information entities and to link them ("read-only") to the task, which she wants to create. During the modification of the task-hierarchy and creation of new tasks, the rights of access to information of the user can't be changed, since the "forbidden" entities are specified in her user class and are therefore invisible for the user. However, she can extend the number of starting points for browsing and modify her task hierarchy to make it more convenient for search according to her own preferences.

Creating New User Classes

The user class editor supports creating stereotype models for groups of users. It takes as input a task-decomposition created with the graphical task editor which is going to be shared by all representatives of the class/group. It allows assigning different read/write rights for every task, banning access to certain information entities or providing access from a task to additional entities. This editor can only be used by specially authorized users.

Changing the Parameters of the Individual User Model

The user can modify her task hierarchy with the task-editor tool. In addition she is provided with an "individual UM editor", which allows:

- Changing the recorded level of experience on a task. This can be done directly by the user without waiting for the adaptation mechanism to suggest a change in the level.
- Selecting the type of viewing. The user can select the "free browsing with an anchor" type or the task-restricted browsing. In this way if she is a novice on a given task, she can still change to a browsing style of viewing, if she feels uncomfortable with the restrictive view.
- Enabling or disabling alternative access strategies. The user can "switch-off" the HYNECOS menus or the option for direct access, if she doesn't want to use them; or, alternatively, she can "switch-off" the task-based interface and work only with the HYNECOS interface and the direct search option.
- Enabling/disabling access to specific entities which are forbidden /allowed to other members of the user class. This is a privileged option which can be only performed by authorized users.

One consequence of having adaptability tools is that it becomes possible for the users to build highly individualized personal versions of the information system. This is, however, not always desirable. Once control is given to the user, one has to take care that she will use it for her own good and will not, intentionally or not, cause damage or violation of access restrictions. This question has not been addressed in our application. In fact,

as the field tests showed, our users didn't strive to a high degree of individualization. However, we believe that it is important to provide these tools for eventual changes that might occur in the task models.

Another possible application of the adaptability tools is the creation of "team models". A too high individualization could become counter-productive to the team nature of the work in the hospital. In this sense, the information system sometimes serves communicative functions. For example, it can be used on the same computer at the ward by two or three doctors and several nurses. In order not to confuse and impair the communication among them, the system can be adjusted to appear in the same way to users performing the same tasks. That is why the design of the UM should provide for creation and adaptation by a group of users, usually from different user-classes, who are going to work together in a team. In other words, the system should be able to support collaborative work by means of a "team model". The original semantic classification interface of HYNECOS is well suited for this purpose, since it is user-neutral. However, it alone doesn't provide any help on how to find the needed information entity. If a group UM is going to be created, the group has to reach an agreement about the team task hierarchy, the style of viewing, the level of experience and the type of presentation and then create a group UM using the adaptation tools.

8 Field Tests

The field testing of the system involved 19 users from different classes and with different levels of experience with the system. We posed three main questions which the tests had to answer:

1) Does the task-based interface (i.e. the UM-supported task-based context for information retrieval) provide better support for novice users than the semantic classification interface? Does the system support relevant tasks?

2) Does HYNECOS**UM** support the user's learning of the semantic classification interface?

3) Does the UM-supported shift among different access strategies really correspond to the changing needs of users with growing level of experience?

4) Is the adaptive capability of the system comprehensible?

The following methods were used to gather data about the user's performance during the experiments performed with HYNECOS**UM**:

- questionnaires;
- on-line logging of all dialogue events (mouse clicks, selections, typing);
- direct observation and protocols;
- interviews.

In this way some redundancy was achieved which offered the possibility of checking what users did against what they said they did.

Testing the Task-Based Interface

In order to answer the first question we performed the following experiment.

Experiment 1. The goal was to test whether the task-based interface of HYNECOS**UM** provides a better learning environment for the novice user than the semantic classification-based interface of HYNECOS. For this purpose we assigned a certain task to two groups each including five novice participants of the same class. All the participants have had some experience with computers in general, and they were familiar with some applications, e.g. text processors. The first group used HYNECOS**UM** with the task-based interface and the other group used HYNECOS with the semantic classification interface. We selected the user class "nurse at the ward" (see Figure 3-b) who had to perform the task "Nursing". This task is performed relatively frequently and follows a well established routine. It contains two subtasks "Treatment" and "Measurements. The first subtask requires access to seven entities and the second one - to four entities. Figure 8 shows how the access to the needed entities can be organized via the two orthogonal interfaces.

The participants of each group had to perform the task four times (the results are shown in the columns in Table 1 framed with thicker line), with data prepared in advance (e.g. measurements have been made in advance and written on paper for typing into the computer), so that they could perform the task entirely on the computer. Time for accomplishing the task was limited to 15 minutes. There was no possibility of using external help or manuals.

An observer sat with every participant during the completion of the task taking detailed notes, including timing of actions and the outcomes. Overall time and success data (see Table I) showed that the task-based interface of HYNECOS**UM** provided a better environment for novices. Four out of the five HYNECOS**UM** participants managed to complete the task in the assigned time in the second, third and fourth sessions. Only one participant failed to complete the task in the first session and the reason was that she took too long working on the contents of the entities once she had accessed them! In contrast, no one from the five participants working with the semantic classification interface was able to perform the task in the first two attempts and only two participants were able to complete the task in the third and fourth session. These two participants turned out to be extraordinarily good learners since they were able to achieve an average task completion time similar to the one of the first group. However, they spent far longer finding the task-relevant entities than the first group and comparatively less time in actually working with the entities, i.e. reading and writing data. The third row of the table shows that the time to access the task-relevant entities (across all participants) differs dramatically in the two groups: while the participants of the task-based interface needed seconds (from 16 down to 6), the participants of the semantic classification-based interface needed minutes (from 13 down to 7)! This can be explained by the more complex cognitive task which the participants of the semantic classification interface have to cope with: they have not only to find their way through the different classes of entities, but also to remember which entities are relevant for completing the task.

The participants of the HYNECOS**UM** group worked confidently with the task-based interface from the very beginning, without any complaints about missing or badly named tasks or entities. They didn't need any explanation in advance of how the tasks are represented - the hierarchical organization was obvious for them. This result confirmed that the knowledge acquisition methods used were suitable and achieved an adequate representation of the task-repertoire performed by the users.

In summary, the experiments showed that the task-based interface ensures a higher effectiveness (speed of finding the needed information) than the semantic classification interface.

Table I: Comparison of the work of two participant-groups with the task-based (TB) and the semantic-classification based (SCB) interfaces.

Times are given in min.(') and sec.(").

Attempts / *Results*	*TB*	*SCB*	*TB*	*SCB*	*TB*	*SCB*	*TB*	*SCB*
Average total time of task completion	13,3' (n=4)	> max.	11,2' (n=5)	15,0' (n =1)	9,8'	12,2' (n=2)	8,0'	9,3' (n=2)
Numb. of subjects who succeeded on task	4	0	5	1	5	2	5	2
Av. time for finding a task-related entity	16''	12,8'	11''	10,4' (n=5)	8''	7,8' (n=5)	6"	7,2' (n=5)

Testing HYNECOSUM Training Capability

Experiment 2. In order to evaluate the teaching capability of HYNECOS**UM** we decided to compare the performance with the semantic classification (HYNECOS) interface of a group of participants who have been working during a certain period of time with HYNECOS**UM** (and, therefore, have been trained to use the HYNECOS menus by seeing how the selected items are classified semantically - see Figure 9) with a group of participants who have been working the same period of time only with the semantic classification interface of HYNECOS and with a third group of participants who have not worked with the system at all before.

The three groups were as follows: one group of six participants who have been working over a course of a week with HYNECOS**UM** and had increased their global level of experience from novice to advanced or expert on more than 70 % of their tasks (we call it the HYNECOS**UM** group); another group of six participants who have been working approximately one week (the same number of sessions) with HYNECOS (HYNECOS-group) and a third group of six participants who have not been working with HYNECOS or with HYNECOS**UM** at all (the Ø-group). During the experiment all groups had to work only with the semantic-classification (HYNECOS) interface. In a course of four sessions (half a day) we observed the work of the groups and measured the change in the average times needed for entity-retrieval for the same tasks.

The results can be seen in Figure 10. One can see that in the first session the average time for the HYNECOS**UM** group is higher than the average time of the untrained HYNECOS group (an effect of the change of interface), but lower than that of the Ø-group. However, the speed of performance increases faster in the HYNECOS**UM**-group than that of the Ø-group and from the second session on it is close to the HYNECOS group.

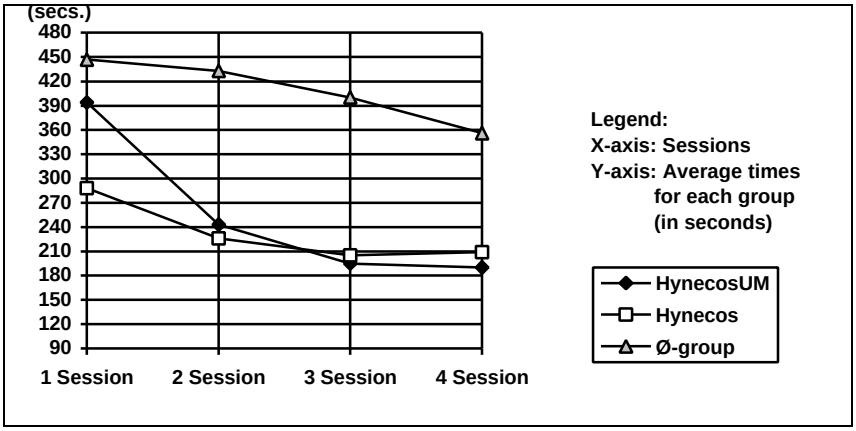


Figure 10: Average Times for Task-Related Entity-Access with the HYNECOS Interface

Testing the alternative access strategies

In order to test whether the different access strategies were really used and whether they satisfied the anticipated user needs, we performed an experiment with a group of participants who could select freely among different strategies. Their UMs indicated that they were already experts at some tasks, advanced in various regions of their task hierarchies and still beginners in other regions.

Experiment 3. The experiment took place in the course of one day and involved 17 experienced participants (with global level of experience more than 70 % advanced or expert) from different user classes. Their actions were monitored and recorded. At every information access, the user's current task was either automatically recorded (when the participant was working with the task-based interface) or the participant had to say aloud what she was doing. At the end the participants were shown their protocols and asked to explain why they used an alternative access strategy. The average (across tasks) usage of different access strategies and their combinations with respect to the user's level of experience on the task are shown in Table II.

The results about tasks where the user is "advanced" showed, as expected, that most popular was still the task-based access strategy. The reason is that the semantic classification interface and direct query strategies are only available in the context of the current task, so the user has first to select a task, in whose context to search using the HYNECOS menus. The results show that advanced participants preferred (with 62% against 25%) to keep to the restrictive-browsing. The results in the second row of the table show that the users still prefer to perform task-based access (29%) even from tasks where they are experts. However, nearly as often the participants were combining the task-based access with the semantic classification-based (23%) access and direct queries (18%). A significant number of participants were applying the semantic classification and the direct query strategies in their "pure" form.

Table II: The usage of different access strategies (in % of all usage) at tasks with a certain level of experience.

(Abbreviations used: RB - Restricted Browsing view type; FB - Free Browsing.)

Interface used: / Experience at task:	Task - based	Semantic Classif.	Direct Query	Task-based + Semantic Classif.	Semantic Classif. + Direct Query	Task-based + Direct Query	Task based + Semantic Classif. + Direct Query
	RB | FB			RB | FB		RB | FB	RB | FB
Advanced	87 %	_	_	12 %	_	_	1 %
	62 | 25%			8% | 4%			-- | 1%
Expert	29 %	10 %	5 %	23 %	8 %	7 %	18 %
	10 | 19%			1% | 22%		-- | 7%	-- | 18%

We wanted to let the participants explain why they used a given strategy. First we interviewed the participants to get an idea of the possible motives and then asked them to fill in a questionnaire allowing the participants to evaluate in a scale form (Pearlman, 1989) their preferences to different answers. The results are presented in Table III.

The results show that in most (72%) of the cases the participants select the task-based strategy since they were used to it and felt confident that they will find the relevant information, while the use of a different strategy was in most of the cases motivated by the wish to save time or mouse clicks. Most of the participants (63%) who motivated their choice "since I know how to do it" had a lower global level of experience in the task-hierarchy, while the participants motivating their choices with the wish to save time / clicks and to see what happens had a higher level of experience (77%). These results confirmed our assumptions about the influence of the individual level of experience of the user on the preference for different access strategies (see Section 4). The conclusion is that experienced users do not distinguish formally among strategies, but perceive them as alternative options or routes for any type of information need. Each strategy option was designed to be an optimal way of satisfying different needs. Expert users were inventive and used the flexibility of the system by creating individual shortcuts to satisfy their needs.

Table III: Reasons for selecting a given strategy.

(presented in % of all the cases when a certain strategy was selected, abbreviations used: TB-task-based, SC-semantic classification based, DQ-direct query).

Explanation / use interface	TB	SC	DQ	TB + SC	SC + DQ	TB + DQ	TB + SC + DQ
"since I know how to do it"	72%	39%	8%	26%	22%	27%	8%
"to save time/ mouse clicks"	23%	43%	89%	37%	70%	52%	66%
"to see what happens"	2%	13%	--	43%	12%	19%	22%
other	3%	5%	3%	4%	6%	2%	4%

Testing the Understandability of the System

In order to evaluate the understandability of the system and especially with respect to adaptivity we interviewed all the nineteen subjects participating in the field tests (to get as much qualitative data as possible) and afterwards asked them to answer questionnaires. The interviews with participants were targeted at finding out if the users recognize the difference between the opportunities provided by the system at different stages of their experience progress and if they feel disturbed by this. 95% (eighteen) of the users who participated in the field tests liked working with the system and preferred it to any other information system they had used before. An additional effect, which was pointed out by participants was that the task-based interface allowed them to concentrate more on their tasks instead of information searching and that the visual decomposition of tasks into subtasks helps them organize their work better and reminds them about filling in or checking documents that were needed and which they might have otherwise forgotten.

It turned out that the metaphor which we used to explain the adaptation when we first introduced the system to the users, was accepted and used by most of them as a basis on which they had built their cognitive model of the system. When explaining why the views from higher level tasks are wider, we compared the task hierarchies with peaks in a mountain: the more one climbs up, the more one sees around, therefore, one also sees more possible ways (access strategies) to get to a given place in the valley below. By “climbing up in the mountain” we meant increasing the global level of experience. During the interviews, as well as while giving explanations of their protocols, the users spontaneously used this metaphor. Several users explained that they were using a “different route to that spot” meaning that they were using the HYNECOS interface, and many used the words “from here I can already jump/ glide” to the entity in question. Some of the users even developed further the metaphor in order to explain themselves the difference between the access options available for novice and advance users on one task.[1] Once during the interviews, a user got stuck explaining the difference, and another user offered help, saying: “But you haven’t been often enough on this peak to take a new route alone - you should study the map first!” The differences between the access options available for advanced and expert users were better understood: “The experts are allowed to jump and flounder all around; they won’t get lost even if they get into new territory”.

So far, the users have been quite reluctant to use the adaptability tools. During the last experiment we provided them with access to the adaptation tools and explained to them how to make adaptations themselves. Only one user tried to use the adaptation tool - “to see how it works”. Two other users expressed a wish to change their task hierarchy, but instead of trying to do this themselves they consulted one of their colleagues who participated in the design of the system and was for them like a “guru”. Their wish for the change was not so much driven by the need to include some particular personal preferences, as by the worry that something is wrong with their system, because it is different - "Why does Dr. X (working on the same computer) in the ambulance get a different task-hierarchy?". We interpreted this as evidence that probably team UMs would be

[1] This was the only point which was not self-evident to the users - i.e. why a novice can only see how a task-related entity is semantically classified, but can’t use the classification menus for selection and the direct access options.

appropriate after the initial learning stage is over. It also supports Norcio & Stanley's (1989) and Oppermann's (1994b) suggestion that individualized systems are of biggest importance for supporting a beginner in the learning phase and their importance fades when the user is able to apply the full range of options provided by the system.

9 Comparison with Other Work

Tasks are often used for representing context in office systems. Several systems providing task support for office systems exist. Croft (1984) describes two adaptive systems: an adaptive office assistant (POISE) and an adaptive document retrieval system. POISE is able to identify the user's current task and to provide assistance in selecting the necessary tool (word processor, e-mailer etc.) and assigning the appropriate parameters for the tool invocation. The Adaptive Document Retrieval System is able to represent the user's needs, query, and the history of the current search in an Associative Search Net[1]. Croft proposes also combining these two approaches in a system for task-based document retrieval in the office. However, we are not aware of the existence of an implementation of this idea so far. In addition the original proposal doesn't concern hypermedia documentation systems, but standard information retrieval systems using Boolean- and full text-search, where the problems are essentially different.

Different approaches exist for making hypermedia adaptive. The most common way is to adapt the presentation of the hypermedia nodes to the needs of the user by, for example, increasing the amount of details or changing the level of explanation (Boyle & Encarnacion 1994; Beaumont 1994; Hohl et al. 1996). Educational hypertexts *restrict the user's browsing* within small portions of the hyperspace which are focused on a certain topic (lesson) and in this way reduce the risk that the user gets lost (Brusilovsky, 1992). In the context of educational hypertext it has been proposed (Böcker, Hohl & Schwab 1990; Hohl et al. 1996; Brusilovsky, Pesin & Zyryanov 1993) to *provide appropriate starting points* for navigation, and local orientation by suggesting links from the current node which will be appropriate to follow (having in mind a specific goal node). More general approaches record a user's typical patterns of navigation and adapt to them by changing the links between the entities to reflect paths that are likely to be traversed having in mind the user's current goal (Kibby & Maes 1989; Monk 1989).

Our approach falls in this educational category of Hypermedia by providing task-relevant starting points in the hypermedia and by restricting browsing to smaller subspaces for inexperienced users. This is not surprising, since the possibility of getting lost in the hypermedia system is small for experienced users of office documentation systems and far bigger for novice users. In this sense our system has an educational function.

The problem of finding out which are the relevant starting points and information subspaces for a given goal (task) is very important, but not well investigated in the field of adaptive hypermedia. More relevant work on this problem exists in the field of adaptive information retrieval. The semantic indexing of information with respect to the individual user's interests, goals or tasks has been extensively investigated (Belew 1986;

[1] A similar structure for indexing hypermedia entities with respect to the user's goals and preferences appears later in (Kaplan et al., 1993) and in (Mathé & Chen, 1996).

Belkin et al. 1987; Kok 1991). Though using different approaches and representation schemes: blackboard architecture with cooperating experts (Belkin et al., 1987), rule based expert systems (Brajnik et al., 1990), first order logic (Kok, 1991), connectionist schemes (Belew 1986; Biennier et al. 1990; Crouch et al. 1994), relevance networks (Mathé and Chen, 1996), or agent-based architectures (Thomas & Fischer, 1996), these approaches share one feature: users' feedback is used as an information source for updating the semantic indexing scheme. Like any machine learning system, a UM utilizing such an individualized semantic indexing scheme works in two regimes:

- learning - modifying / extending the task model on the basis of any kind of feedback from the users about the information suggested by the system, and
- applying - inferring the user's current task and recommending information according to the task model.

In most approaches (Thompson & Croft 1989; Kok 1991; Brajnik et al. 1990; Bennier et al. 1990; Crouch et al. 1994; Mathé & Chen 1996;) the user is asked directly to estimate the relevance of every unit that is retrieved. Some approaches, however, employ other sources of evidence about the user in the learning phase to index the information to the current task /goal of the user. For example, (Kaplan et al., 1993) assume that the more time the user spends on a unit, the more interesting it is, i.e. more relevant to the current goal. Maes (1994) proposes apart from user feedback, three types of evidence about the user that could be used for learning about her goals (tasks or interests): observing her behavior, e.g. what she does with the retrieved documents, direct training of the system by the user by giving examples, communication and sharing knowledge about the user with other UMs ("agents" in Maes' terminology).

The most common approach - direct user feedback - has inherent disadvantages. First, the user is not always a reliable source of evaluation about the relevance of the information to her goal or task (especially when she is inexperienced), and second, it poses an additional burden on the user and might distract her from her goals. For example, in our application it is absolutely impossible to expect direct user feedback on the relevance of information to tasks in everyday work. It would have been possible to apply this approach in the knowledge acquisition phase, as an exception, only for a short time. However, we would have needed users who are confident working with the system and who give reliable feedback. Novice users are not good teachers for a machine learning system.

Other ways of getting feedback, for example, using the assumption that the more time is spent on an unit, the more interesting it is (Kaplan et al., 1993), can also be unreliable, since it is not clear what the user is actually doing in this time. Of course, if she spends too little time, not enough even to read the information, this is a sort of evidence that the user was not interested in it. However, still the question remains whether the reason is that she considers this irrelevant to the goal (indexing has to be changed) or that she has changed her goal in the meantime (a new goal has to be recognized). Still, user feedback is the only possibility for learning approaches aiming at adaptive semantic indexing when the information space is very large, as in library applications (Crouch et al. 1994; Mathé & Chen 1996) and the set of possible user goals is virtually endless. In our case, the information space is comparatively smaller and the number of user tasks is not too big. Also the relevance of information to tasks is stable and doesn't depend on the individual user. That is why a learning approach in our case is not justified. It is possible to index the information explicitly to the tasks in advance. This has two advantages: first, it is more reliable than automatic learning "which is nothing more than a guess"

(Kay, 1994), and second, it doesn't require any feedback from the users at runtime. We suspect that this is the case with most of the office documentation system applications.

In most approaches (Thompson & Croft 1989; Brajnik et al. 1990; Kok 1991; Hoppe 1992; Kaplan et al. 1993; Mathé & Chen 1996) the information obtained from the user (or the observation of her behavior) is used also to change the structure of the goals, tasks, interests contained in the UM. Our approach doesn't learn about the user's task-model. In principle this is possible. For example, Mathé & Chen build a hierarchical relevance network which starts from goals and keywords and builds up combinations called "compound nodes" whose information needs are derived from the needs of their sub-nodes. One could think of the relevance network as a dynamic task hierarchy where the nodes represent combinations of keywords, information descriptors, tasks /goals, and interests, instead of task-names. The weights of relevance among the nodes in the network are updated based on the user's feedback and in this way a dynamic model of user's interests with respect to tasks can be built. In a similar way, the systems described in (Biennier et al. 1990; Kaplan et al. 1993) learns about the weights of the links among different goals from the user's feedback.

However, this was not necessary in our case. The comparatively smaller number and dynamics of the tasks performed by users distinguishes the domain of our application from other information retrieval applications. In office documentation systems or "institutional hypermedia" (Brusilovsky, 1996), typically the organization of work assumes well defined tasks associated with actors performing them.

Another aspect according to which we can compare our approach to other approaches concerns the representation of the user model. The approaches aiming at a dynamic representation of the user's goals, beliefs, and interests employ rule-based systems (Brajnik et al., 1990), logic-based representations (Kok, 1989; Kobsa & Pohl, 1994b), connectionist schemes (Belew, 1986), or relevance networks (Kaplan et al., 1993; Mathé & Chen, 1996). As already mentioned, our user model's representation is a combination of stereotype (ensuring the appropriate context of information access) and an overlay model (ensuring the adaptation to the user's level of experience). Stereotype approaches are in practice the most popular user modeling approaches (Kay, 1994) and overlay models are the most practical student modeling approaches for intelligent tutoring systems (Wenger, 1987). Such a combination has been considered by many authors (Kobsa & Pohl, 1994a) as a good basis for user modeling.

Once a UM has acquired and represented some knowledge about the user, it has to apply it in order to diagnose the current situation (task, goal etc.) and to suggest relevant information. In our approach, unlike (Hoppe, 1992) and (Oppermann, 1994b) there is no inferring of the current user's task - the user assigns it explicitly herself. With respect to this feature our approach is similar to (Kaplan et al., 1993), (Mathé & Chen, 1996) and (Höök et al., 1996), where the user interacts with the system by designating her current task. The parallel with the Glass-box model (Höök et al., 1996), can be drawn further since the user in fact sees her model all the time on the screen and manipulates it directly during her interaction with the system. The advantages of this approach are summarized by (Höök et al, 1996) and we don't need to repeat them here. The disadvantages are that the if the tasks are too many, it might become difficult for the user to select or to predict what an alteration of the task will result in. However, unlike Höök, who solves this problem by combining a small set of pre-defined tasks with user's task/plan-inference, we do it by providing the user with alternative, task-independent search strategies and with a simple teaching facility showing how to use them.

With respect to the ability of our system to support the user's learning of the system, the idea of our approach is somewhat similar to Carroll's idea of "training wheels" (Carroll & Carrithers, 1984) which consists of disabling from the interface for inexperienced users the more complex options of the system and then gradually increasing the amount of available options along with the increasing level of experience of the users. Experimental studies show that this approach provides a very good support in the stage of user learning to work with complex applications, e.g. word-processing programs. However, the idea that the user's level of experience influences not only the amount of information (options) which she is able to cope with but also her search strategy seems not to be investigated elsewhere in the field of information retrieval.

Finally, task-based interfaces to software systems have proven to be very effective (Rasmussen et al. 1994; Fischer 1995). Our filed tests also confirmed this result not only for novice users but for experienced users, who turned out to prefer the task-based interface even when they had the freedom to choose among different interfaces. There is a notable tendency in the development of man-machine interfaces, summarized by Fischer: *"... the emphasis in human-computer interaction should be concentrated on the humans and their tasks - not on computers, interfaces and tools*"(Fischer, 1995).

10 Conclusions

This paper describes how user modeling has been applied to ensure task-specific context and adaptation to the individual user's level of experience in a hypermedia-based hospital documentation system. Stereotype models of user classes characterized with specific tasks have been defined. These task-models serve as a kernel of every individual model. This model is transparent to the user and she interacts with the system by means of selecting the task she wants to perform. Every task provides a specific "view" of the hypermedia consisting of the documents which are needed for completing the task. They serve as starting points for browsing in the hypermedia. The individual UM is an overlay on the user's task hierarchy (it can be the user-class task hierarchy or a modification of it) where the user's level of experience is recorded at each task. The level of experience is diagnosed by the system by observing the behavior of the user: the way she selects entities, the mistakes she makes and the way she browses through the hypermedia. The user's level of experience influences the number of starting points, the size of the browsing space and the access strategy to the information. Three different strategies have been provided: direct search (by keywords), and two strategies corresponding to different ways of indexing the information: a semantic classification and a task-based classification. The task-based access is provided for users with lower levels of experience together with a teaching option showing how the selected information is semantically classified. For advanced and expert users, the other search strategies are gradually enabled, together with the possibility of free browsing in the whole hypermedia. The adaptation takes place at discrete points of time and only after obtaining the user's consent. Adaptability tools are available, allowing the users to change nearly all parameters of their UMs and the organization of the interface.

The testing of the system showed that the tasks models represented in the system correspond to the real tasks performed by the users, and that the task-based interface provides a better support for novice users than the semantic classification interface. HYNECOS**UM** proved to support a faster learning of the semantic classification interface. The UM-supported shift among different access strategies really corresponds to the

changing needs of users with growing level of experience. The users seemed to understand well the adaptive capability of the system and to be able to explain its behavior.

In summary, the UM in the hospital documentation system ensures the following features:

- increases the operational speed and efficiency of task performance;
- provides a task-based context for information access;
- supports novice users by ensuring access to all of the documents needed for a task, while providing experienced users with appropriate starting points for browsing and alternative strategies for information access;
- provides a simple teaching facility to help novice users learn alternative strategies;
- supports adaptivity to the user's level of experience at discrete points in time and after asking the user;
- provides a transparent, directly manipulable model, an integral part of the user's interface and an understandable metaphor for the adaptation;
- provides adaptability tools allowing the creation of group UMs supporting team work.

We believe this approach to user modeling can be applied to achieve adaptivity and novice-user support in many office information and documentation systems. The requirements are to have comparatively stable, well-defined task structures associated with specific user roles in the working situation which can help in defining user classes. Another requirement is that the tasks themselves should have relatively stable information needs. If any of these conditions is not present, this approach needs to be extended with learning capabilities, like those described in section 9, so that the UM could acquire automatically from observing the user's behavior the indexing of relevant information to the tasks and /or the task-structures itself.

Acknowledgments

I would like to thank my student, Thomas Stoyke who implemented entirely the **-UM** part of HYNECOS**UM**; to Karin Hertwig, Dirk Langkafel and Jack Shiff from SIEMENS ZFE who developed HYNECOS and helped with their experience and advice in the requirement analysis and knowledge acquisition; to the staff of the University Orthopedics Clinic of Heidelberg and especially to Dr. Krämer; to Ralph Deters, Peter Brusilovsky, to three anonymous reviewers who gave valuable comments on previous drafts of this paper, and especially to one of them who helped a lot with the English.

This research has been partly supported by Projects 644 "Documentation" with Siemens ZFE and I-406 with the Bulgarian Ministry of Science and Higher Education.

References

Anderson, J.: 1985, 'Cognitive Psychology and its Implications'. In G. Atkinson, R. Lindzey, and R.Thompson (eds.): *A Series of Books in Psychology*. New York: W.H. Freeman & Co.

Beaumont, I. and P. Brusilovsky: 1995, 'Adaptive Educational Hypermedia: Form Ideas to Real Systems'. *Third International Conference on Educational Multimedia and Hypermedia ED-MEDIA'95*, Graz, Charlottesville: AACE, pp. 93-98.

Beaumont, I.: 1994, 'User Modeling in the Interactive Anatomy Tutoring System ANATOM-TUTOR'. *User Modeling and User Adapted Interaction* **4** (1), 21-45.

Begoray, J.: 1990, 'An Introduction to Hypermedia Issues, Systems, and Application Areas'. *Int. J. Man-Machine Studies* **33**: 121-147.

Belew, R.K.: 1986, 'Adaptive Information Retrieval: Machine Learning in Associative Networks'. Ph.D. Thesis, Dept. of Computer & Communication Sciences, University of Michigan: Ann Arbour.

Belkin N.J. et al.: 1987, 'Distributed Expert Based Information Systems: An Interdisciplinary Approach', *Informat. Process. Management* **23** (5).

Biennier, F., M. Guivarch and J.-M. Pinon: 1990, 'Browsing in Hyperdocuments with the Assistance of a Neural Network'. In: Rizk, A., N. Streitz, J. André (eds.): *Hypertext: Concepts, Systems and Applications.* Cambridge: Cambridge University Press.

Böcker H.-D., H. Hohl and T. Schwab: 1990, 'HYPADAPTER - Individualizing Hypertext'. In: Diaper D. et al. (Eds.) *INTERACT'90 Proceedings of the 3rd International Conference of Human Computer Interaction*, North-Holland: Amsterdam, 931-936.

Boyle, C. and A. Encarnacion: 1994, 'MetaDoc: An Adaptive Hypertext Reading System' *User Modeling and User Adapted Interaction* **4** (1), 1-19.

Brajnik G., G. Guida, C. Tasso: 1990, 'User Modeling in Expert Man-Machine Interfaces: A Case Study in Intelligent Information Retrieval', *IEEE Transactions on Systems, Man, and Cybernetics* **20** (1), 166-185.

Brusilovsky, P., L. Pesin, and M. Zyryanov: 1993, 'Towards An Adaptive Hypermedia Component for an Intelligent Learning Environment'. In Bass L., J. Gornostaev and C. Unger (eds.): *Human Computer Interaction*. Lecture Notes in Computer Science No. 753, Berlin: Springer Verlag, pp. 348-358.

Brusilovsky, P.: 1992, 'Intelligent Tutor, Environment and Manual for Introductory Programming', *Educational and Training Technology International.* **29** (1), 26-34.

Brusilovsky, P.: 1996, 'Methods and Techniques of Adaptive Hypermedia', This issue.

Carroll J.M. and C. Carrithers: 1984, 'Training Wheels in a User Interface'. *Communications of the ACM* **27** (8), 800-806.

Conklin, J.: 1987, 'Hypertext: An Introduction and Survey'. *IEEE Computer* **20**: 17-41.

Croft, B.: 1984, 'The Role of Content and Adaptation in User Interfaces'. *International Journal of Man-Machine Studies* **21**: 283-292.

Crouch, C., D. Crouch and K. Nareddy: 1994, 'Associative and Adaptive Retrieval in a Connectionist System'. *International Journal of Expert Systems* **7** (2), 193-202.

Dumais, S., G. Furnas, T. Landauer, S. Deerwester, and R.Harshman: 1988, 'Using Latent Semantic Analysis to Improve Access to Textual Information'. *Proceedings CHI'88*, New York: ACM, pp. 281-285.

Fischer, G.: 1992, 'Shared Knowledge in Cooperative Problem Solving Systems: Integrating Adaptive and Adaptable Systems'. *Proceedings UM'92: Third Int. Conference on User Modeling*, Dagstuhl, pp. 148-161.

Fischer, G.: 1995, 'New Perspectives on Working, Learning and Collaborating and Computational Artifacts in Their Support'. In H.-D. Böcker (ed.): *Proceedings Software-Ergonomie'95*, Stuttgart: Teuber Verlag.

Grazotto, F., P. Paolini, and D. Schwabe: 1991, 'HDM - a Model for the Design of Hypertext Applications'. *Proceedings Hypertext'91*, ACM Press, pp. 313-328.

Hohl, H., H.-D. Böcker, and R. Gunzenhäuser: 1996, 'HYPADAPTER: Hypertext System for Exploratory Learning and Programming', This issue.

Höök, K., J. Karlgren, A. Waern, N. Dahlbäck, C. Jannsson, K. Karlgren, and B. Lemaire: 1996, 'A Glass Box Approach to Adaptive Hypermedia', This issue.

Hoppe, H.: 1992, 'Towards Task Models for Embedded Information Retrieval', *in Proceedings CHI'92*, 173-180, New York: ACM.

Kaplan, C., J. Fenwick, and J. Chen: 1993, 'Adaptive Hypertext Navigation Based on User Goals and Context', *User Modeling and User-Adapted Interaction* **3** (3), 193-220.

Kay J.: 1994, 'Lies, Damned Lies and Stereotypes: Pragmatic Approximations Of Users', *Proceedings of UM'94, 4th Int. Conference on User Modeling*, Hyannis, MA, pp. 175-184.

Kibby M. and P. Maes: 1989, 'Towards Intelligent Hypertext'. In: McAleese (ed.*): Hypertext Theory into Practice*, Ablex, pp. 164-172.

Kobsa, A. and W. Pohl: 1994a, 'Workshop on Adaptivity and User Modeling in Interactive Software Systems'. *User Modeling and User-Adapted Interaction* **3** (4), 359-367.

Kobsa, A. and W. Pohl: 1994b, 'The User Modeling Shell System BGP-MS'. *User Modeling and User-Adapted Interaction* **4**(2), 59-106.

Kok, A.: 1991, 'A Formal Approach to User Modeling in Data-Retrieval'. *Int. J. Man-Machine Studies* **35**: 675-693.

Kuehme T.: 1993, 'User-Centered Approach To Adaptive Interfaces'. *Knowledge-Based Systems* **6** (4).

Maes, P.: 1994, 'Agents that Reduce Work and Information Overload'. *Communications of the ACM* **37**(7): 31-40.

Mathé, N. and J. Chen: 1996, 'User-Driven and Context-Based Adaptive Information Access'. This issue.

Monk A.: 1989, 'The Personal Browser: A Tool For Directed Navigation In Hypertext Systems'. *Interacting with Computers* **1** (2), 190-196.

Norcio, A. and J. Stanley: 1989, 'Adaptive HCI: A Literature Survey and Perspectives'. *IEEE Trans. on Systems, Man and Cybernetics* **19** (2), 399-408.

Oppermann, R.: 1994a, 'Adaptively Supported Adaptability'. *Int. J. Human-Computer Studies*. **40**: 455-472.

Oppermann, R.: 1994b, 'Adaptive User Support'. Hillsdale, NJ: Lawrence Erlbaum Assoc.

Pearlman G.: 1989, 'Evaluating How Your User Interfaces Are Used'. *IEEE Software*, January 1989, 112-113.

Rasmussen, J., A. Pejtersen, and L. Goodstein: 1994, 'Cognitive Systems Engineering'. New York: John Wiley & Sons.

Schwabe, D., B. Feijó, and W. Krause: 1990, 'Intelligent Hypertext for Normative Knowledge in Engineering'. In: Rizk, A., N. Streitz, J. André (eds.): *Hypertext: Concepts, Systems and Applications*. Cambridge: Cambridge University Press.

Thomas Ch. and G. Fischer: 1996, 'Using Agents to Improve the Usability and Usefulness of the World-Wide Web'. *Proceedings UM'96, 5th Int. Conference on User Modeling*, Hawaii, pp. 5-13.

Thompson R.H. and W.B.Croft: 1989, 'Support for Browsing in an Intelligent Text Retrieval System'. *Int. J. Man-Machine Studies* **30**: 639-668.

Tyler, S.W. and S. Treu: 1989, 'An Interface Architecture to Provide Adaptive Task-Specific Context for the User', *Int. J. Man-Machine Studies* **30**: 303-327.

Vassileva, J.: 1994, 'A Practical Architecture for User Modeling in a Hypermedia-Based Information System'. *Proceedings UM'94, 4th Int. Conference on User Modeling,* Hyannis, pp. 115-120.

Visser, W.: 1994, 'Organisation of Design Activities: Opportunistic, with Hierarchical Episodes'. *Interacting with Computers* **6** (3), 239-274.

Wenger, E.: 1987, 'Artificial Intelligence and Tutoring Systems', Los Altos: Morgan Kaufmann.

List of Figures:

List of Tables:

Contents: