Essay on Intelligent User Interfaces and adaptivity

Svend Andreas Horgen,
Division of Intelligent Systems, NTNU
November 2001.

1. Introduction

User interfaces have emerged from command lines to direct manipulation and now faces a new generation. From the user's point of view, intelligent interfaces should provide adaptivity, context sensitivity and more assistance with the task at hand. To achieve these goals, the system needs to understand multimodal and possibly imprecise input, generate coordinated multimodal presentations and manage the interaction. The use of models to represent and reason about the user, domain, task, discourse and situation becomes important in building intelligent user interfaces.

The superior goals of the field are to achieve more *efficient*, *effective* and *natural* interaction between user and machine [1]. Efficient interaction means the ability to complete tasks with less work. Efficient interaction means doing the right thing at the right time, i.e. by tailoring content and form according to the context. A more natural interaction includes support for natural language i.e. by use of gestures and spoken language in a possibly multimodal setting. In solving these problems, several sub-fields of research have emerged. The next section briefly summarizes the main features of the different approaches, structured according to the curriculum in the course IT379. Due to limited space of this essay, only the big picture can be presented. Some of the deeper principles of the field *adaptivity* are more closely examined in section three, however. Finally the conclusion includes some thoughts about the future of intelligent user interfaces.

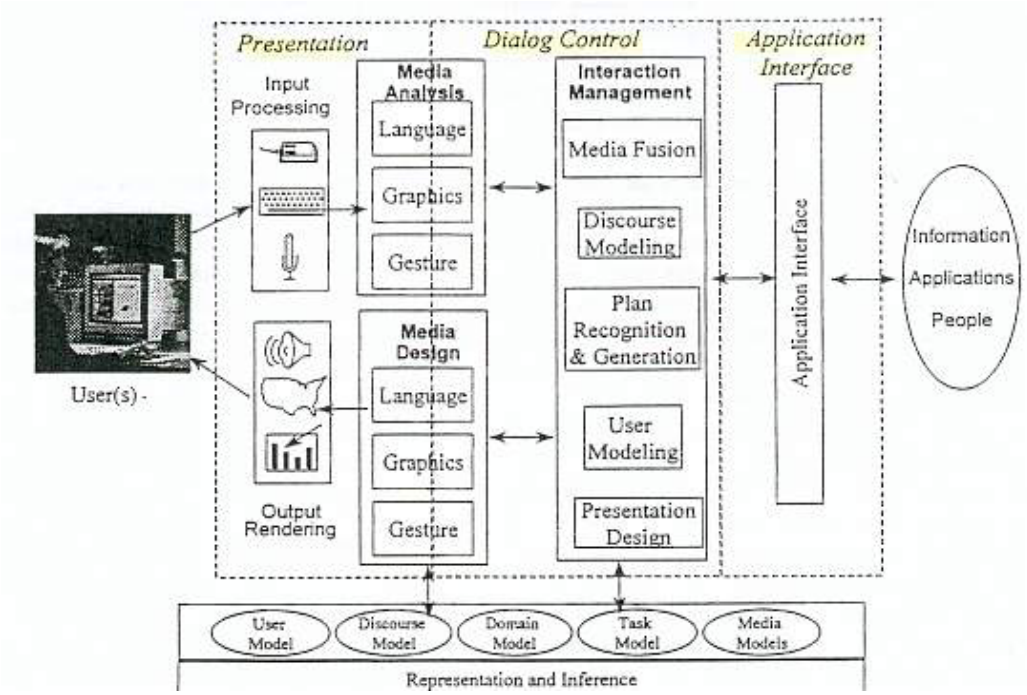

Figure 1

2. Summary of the field IUI

Traditional interface models accounted for presentation, dialogue and application. With intelligent user interfaces an extended model is needed, including **input analysis**, **management of the interaction**, and **generation of the output**, along with the use of **models** [1]. An interesting observation is that the intelligence in the input, output and interaction processes gets provided through the use of explicit models of user, task, media, domain and discourse. Figure 1 gives an overview of the model underlying most intelligent user interfaces.

2.1 Multimedia Input Analysis

In a direct manipulation environment, the mouse and keyboard are well-known interacting devices. Current user interfaces only know how to use one input-channel at the time, and these devices are regarded as tools with which the user can operate the machine. Motivated by the main principle of Gestalt psychology (the whole is more than the sum of the parts), the field of multimedia analysis supports intelligent processing of *multimodal input*. The information from all the input are interpreted and merged together into one integrated meaning, as shown in the interaction management and media analysis areas of figure 1. The goal is to let the user *communicate* with the machine instead of only using it, while accepting possibly ambiguous input. Problems with simultaneous input include the timing of events and the correct level of abstraction. Based on the fact that humans make use of both spatial and semantic knowledge and multiple modes for communicating efficiently, Koons describes a prototype operating in the blocks world which integrates and interprets simultaneous input from speech, gaze and gestures using a frame-based method [3]. The user can i.e. manipulate symbols on a map by making statements like: "that blue square below the red triangle" while simultaneously pointing on the screen and looking at a specific area. Limitations of this approach are the simplified handling of gestures.

Another approach is found in the CUBRICON system [2] where the natural human dialogue is simulated through *parsing* of mixed and asynchronous multimedia input. Using models of the domain and dialogue, different modules in the system merge and interpret the stream of input, plan appropriate actions, and coordinate an aural/visual output into the resulting presentation of multimodal output. CUBRICON is implemented by the semantic processing system SNePS.

2.2 Multimedia Presentation Design

Multimodal presentation systems are useful because they may use many media in parallel in addition to exploiting the strong sides of each medium. The general problem when using multimedia in an intelligent setting is how the computer can analyse and construct multimedia presentations on the fly. The process of generating output is related to the context, task and user expertise. *Selecting* the content, *allocating* and *realizing* the media and performing *layout* are interdependent processes. Their *underlying knowledge sources* are of great importance [5], as also depicted in figure 1.

The knowledge-based presentation system WIP described in [4] emphasizes that the generation of a multimodal presentation is an *incremental planning process*. WIP generates presentations by reasoning about the task and context. That is, the machine no longer is the

author, but rather acts as a desktop publisher allowing the user to intervene and correct the presentation if necessary. The users degree of expertise and the preferred medium of presentation are used to specify the look of the final presentation. The generation of multimodal documents in WIP is a goal-directed activity with at least one main act central to the main goal of the presentation. Important factors in this process are the role of pictures and the relations between pictures and text. WIP receives a presentation goal and a set of generation parameters from the application. A presentation planner lays a strategy for how to fulfil the goal making use of a knowledge base containing a user model, geometrical information about the objects etc. A layout manager arranges the output in the document. Multimodal interaction occurs over time. This makes it necessary to account for discourse. Tracking the history and using pipelining for incremental influence among the presentation and layout managers, makes it possible to design text and graphics before the entire presentation plan is finished. This fits the nature of real-time environments well.

The media allocation problem concerns how the designer can allocate the various pieces of information to the proper medium. Arens stresses the need for a *systematic analysis* of the factors that influences the presentation before the interface can be built [5]. By generalizing the rules flexibility and portability are achieved. In planning the presentation characteristics of the media and information should be used.

2.3 Automated Graphic Design

Designing every possible data and presentation situation is an ineffective, comprehensive task that often requires the developer to be an expert. An illustration usually has a communicative intent and gets interpreted in some way by the receiver. Hence, the goals of automated graphic design include letting the system decide how to generate the graphic presentations and from the user's point of view to remove the possible ambiguity between intended and interpreted presentations.

As seen from the architecture of intelligent user interfaces presented in figure 1, automated graphic design relies upon the use of models. The design process needs to be tailored to the context, task and user. More specifically *expressiveness* and *effectiveness* criteria of the underlying explicit representations in the reasoning processes must be considered. The graphical language should be able to express the desired information and make the most out of the output medium and human capabilities. A table from a database could be presented as a chart, a graph, plots etc. Mackinlay presents a theoretical work on how to best present relational information on the screen [6]. His main assumption is that graphical presentations are *sentences of graphical languages*.

An interesting point that may enrich the understanding of the processes underlying automated design is the possible use of well-known AI-techniques in building automated systems. Applying depth-first search and backtracking to the proposed prototype APT produces *several design alternatives*. A search is necessary because it is impossible to specify the solution directly, and a variety of designs must be generated before the presentation tool can handle a variety of input. The IBIS system makes use of a generate-and-test approach with a *goal-driven* search process. If a solution is not satisfactory, the illustration gets regenerated through the use of backtracking. Formalizing the intention of a communication reduces the ambiguity of presentations [7].

2.4 Adaptivity

Adaptive systems adjust the interface according to the user's skills, knowledge and preferences. Adaptable systems allow the user to control these adjustments. In order to achieve adaptivity, underlying models of the users and tasks are essential [28]. Adaptivity is further explored in section three.

2.5 User and discourse modelling

Typical tasks of intelligent systems like planning explanations, answering questions based on prior discourse and supporting interruption, rely on the use of underlying models. A user model contains explicit assumptions about users. A discourse model has descriptions of the history, syntax, semantics and pragmatics of the dialogue between the user and the system. Why do the models exist explicitly? The system KN-AHS does not integrate the user-modelling component BGP-MS into the application [13]. This leads to adaptivity in the following senses: many types of information about the user can be represented simultaneously, the user-modelling component can receive and answer questions, and accumulation of knowledge takes place more naturally.

UCEgo is a consultation system that corrects the misconceptions of a user or provides needed information that is not explicitly asked for. Gaps in knowledge are discovered through reasoning about the user model i.e. whenever changes in environment or internal state occur. The UCEgo agent needs to be both *autonomous* and do *rational planning* to take intelligent initiatives. Goals depend on context, and the central problem for UCEgo concerns how to detect new goals when they are not given by a human planner [11].

A careful *analysis of the discourse context* of a gesture is necessary to avoid reference failure when using ambiguous pointing. A problem is how to convert the pointing behaviour into formal semantics. Wahlster addresses *how* the user and discourse models influence the comprehension of multimodal communication (and vice versa). The XTRA system accepts multimodal input/output and assists the user in filling out tax forms. *Functions* to be performed by user and discourse modelling components are building the models incrementally, maintaining consistency and supplementing other components of the system with information about the user and the dialogue [12].

Acquisition of information in a model may be either explicit or implicit. Explicit models gather information by prompting the user (user driven acquisition), leaving the system to play a passive role. Implicit models are acquired by the system (system driven acquisition) during the course of dialogue without explicitly consulting the user. Hence implicit information is more dynamic and requires rules of inference and a way of handling conflicting information [29]. Stereotypes may be used when information about the user is limited, or as a supplement to other methods used with the user model. Rich presents the system GRUNDY that acts as a librarian recommending books based on a dialogue with the user [10]. Stereotypes get activated or deactivated by triggers. *Learning* happens through the modification of the stereotypes. In general, updating user models is important when interacting with a user over time.

User models play an important role in adaptive systems and hence are further explored in section 3.2

2.6. Model based user interfaces

User interface design environments are characterized by expressiveness and ease of use. If the designer chooses pure programming as a tool, the expressiveness is at best but at the cost of demanding programming skills. Interface builders are easy to use but have limited possibilities for the design, and typically only support the preliminary stages of the design cycle.[1] Model based user interfaces constitute a different approach to intelligent user interfaces by allowing the designer to *describe a model* consisting of facts rather than using large procedural programs. The goals comprise *decreasing the time* and expertise required to create user interfaces, *identifying reusable components,* and building *extensible* models in an easy, comprehensible way while maintaining as much of the expressiveness as possible.

Model based development has advantages over traditional user interface toolkits and UIMS systems. Dialog control is separated from the application code and the designer is provided more powerful design tools. Modifying the behaviour of an interface only requires changing the model instead of reprogramming a certain section, the latter being more challenging. These facilities meet the goals of model based user interfaces listed above. One objection to the approach is that models limit the possible outcomes of interfaces [25].

Merging the best aspects from the UIDE and HUMANOID systems, the Mastermind project constitutes a step towards a complete model supporting the entire design-cycle [17]. This is possible because of great similarities between the models underlying the two systems. The main design stages of Mastermind reflect the advantages with a model based approach:

1. Use of *explicit* models: The GOMS model is currently used for task analysis, which is described explicitly. This ensures consistency between task and design. Legal operations are described in a command-model and layout is described in a presentation model.
2. Conceptualization is regarded as a search in a space of *alternative* designs. The design space is orthogonalized which ensures modularity in the design.
3. The prototyping stage gets easier when the above is done.
4. *Accumulated knowledge* about the design decisions from the previous design-stages may be used throughout the cycle and in particular in the maintenance stage. [2]

The overall goal of Mastermind is to generate automated and animated help facilities, as in UIDE [14], [16], and use the design models to "map low-level user gestures onto high-level semantics." [26]

2.7 Agents

Among the arguments in favour of an agent-based approach to intelligent user interfaces, are the need for distributed computing and the limitations of direct manipulation. As debated in [20], direct manipulation and software agents are complementary rather than mutually exclusive. Still each methodology clearly has its strong sides. A conclusion to be drawn from the debate is that agents are more convenient in settings with complex environments, difficult tasks, a dynamic network of people and information, or relatively naïve users.

A singular definition for the term *agent* does not exist. According to Bradshaw, two main approaches attempt to define an agent [19]. One may either look at the agent as an ascription

[1] The design cycle comprises analysis of the user and tasks, designing the system, evaluating the design with respect to well-known HCI-principles, implementing and testing the prototype.
[2] The system may generate animated help, provide context sensitive presentations at run-time etc

made by a person in terms of *what they are*, or as a description of its attributes (i.e. a list of *what they do*). Applying this second view, an agent should be:

- Reactive, that is able to sense and act on its environment
- Proactive which means the ability to start something itself autonomously
- Collaborative and able to work in concert with others if requested to
- Communicating with humans in non-symbolic natural language-like ways
- Using models to infer new knowledge
- Persistent over time
- Adapting to its environment, and able to learn from experience
- Mobile and move itself from one place to another when necessary

The nature of agents makes different settings easy and economically to obtain. Agents may have different roles for different people and situations, i.e. as personal assistants, intelligent user interface managers, agent behind the scenes, performing agent-to-agent communication etc. Robust and scalable distributed software systems require the use of agents, because an agent-based system has several advantages that fit the requirements of complex, dynamic, flexible and uncertain environments. Jennings+ argue that the current methods come short in for example the development of social intelligent agents [18].

Acting as a personal assistant, agents become more effective as they learn the preferences, habits and interests of a user. How does the agent acquire sufficient competence of its user, and does the user actually trust the agent? A knowledge-based approach makes use of domain-specific background knowledge about both application and the user. An alternative approach relying on machine learning techniques is presented in [34] and may be more convenient if different users use different strategies and habits in the interaction with the application. Given only a minimum of background knowledge, the agent has to search actively for information about the user and his tasks. This is done either by monitoring the user i.e. by searching for repetitive actions, through user feedback, training examples provided explicitly from the user or through the interaction with other agents. As examples of the machine-learning strategies, Riecken demonstrates several systems like e-mail agents, calendar agents, news filtering agents etc.

To summarize, agents fit tomorrow's way of computing well by radically changing the style of interaction in a dynamic environment with relatively naïve users. Trust and competence are important factors to consider. Many areas of research challenge the agent-based approach, like the question of metaphors, privacy, collaboration on the Internet, use of facial expressions etc.

3. Adaptivity – a survey

Developing an adaptive user interface (AUI) requires an interface that can be adapted, a user model and a strategy for how the adaptation should to take place. This section explores adaptive systems in the context of user interfaces as far as the scope and length of this text allows. The *importance of the models* underlying adaptive systems cannot be underestimated, and are exemplified in section 3.2. With this, a foundation for understanding the strategies of adaptive hypermedia is made. This field is particularly interesting as the Internet increasingly gets integrated into our daily lives. Some concepts are introduced in section 3.3 [1]

3.1 Adaptive systems

Presenting easy, efficient and effective interfaces is the main goal of adaptation. [9] It is difficult to write software that will fit millions of users perfectly. Nobody learns a system completely, but use different parts of the system and share some common basic knowledge about its functionality. Adaptive systems change this paradigm by turning use time into a different kind of design time [27].

Users need to adjust the interface to their own needs and preferences, goals, tasks and contexts. For the system to say the right thing at the right time in the right way implies reducing the information overload and adapting the presentation to the relevant task, knowledge and experience of the user. An AUI supports this process in more or less sophisticated ways, while a static interface doesn't. Among unsuccessful examples are the help provided from the Office Assistant in the Microsoft Office software. Malinowski+ describes a taxonomy that places adaptive systems in the context of intelligent user interfaces [9]. The roles of an intelligent interface[2] are fulfilled by the integration of an AUI, an intelligent help system and an intelligent tutoring system. The taxonomy used is based on four stages of the adaptation process, namely *initiation* of the adaptation, *proposal* of possible changes, *decision* of actions to be taken and *execution* of the selections. The degree of adaptation depends on whether the user or the system performs each of the stages. As an example, a system is self-adaptive if it performs all of the above stages itself.

Furthermore two groups of adaptation are distinguished: adaptation of *communication* and adaptation of *functionality*. The first group includes systems that provide context sensitive help, like UIDE [16]. Adapting functionality covers the automation of tasks and generation of new complex functions. This is more complicated but of great importance as it fits one goal of intelligent systems: let the computer carry out the routine tasks and allow the user to perform the creative ones. At the syntactic level, adaptation may yield counting the number of interaction steps, while higher-level adaptation accounts for goals and tasks of the user as a basis for achieving functional adaptation.

It is important to decide *when* interaction should occur, *what* information to present and *how* to present this information on the screen. Before the system is built, the needs of future individual users or groups of users must be considered. During use of the system, adaptation

[1] The topics of section 3.3 are further explored in my coming main work as a master student, planned finished the summer 2002

[2] The roles of an intelligent interface comprise adapting to the needs of the user, provide context sensitive help, and supporting the user of the system.

may happen continuously by comparing the situational changes to the user's needs, but also on junctures (predefined critical situations), on special occasions or on user requests. Adaptation implies a certain risk. Situations may occur where the user and the system are trying to adapt to each other and thus never reach upon an agreed interface. The adaptation process may also confuse the user if not done carefully. Fischer stresses that in an adaptive setting little or no effort is required from the user, and loss of control may be the result [27]. In adaptable systems, however, the user is regarded to know best what his tasks are and should therefore make the changes to the functionality by setting preferences. This requires the user to learn about the existence of, and how to use the adaptation component.

The environment is important in designing intelligent user interfaces. Most systems behave intelligently only in their original environments - with environmental changes the performance degrades. The adaptive system, however, gains its power by reacting to a changing environment. One way to defer the design is by building different variants into the system and let triggers activate the set of design choices. Hence metrics for evaluating the benefits of design are important for the adaptation process. Rautenbach uses a simple game for classifying adaptive systems and proposes a two level architecture for adaptation. A higher level identifies major changes and chooses the best design-variant. A lower level concentrates on adapting the interface according to the user's needs [8]. This separation of modeller and introspector, is convenient.

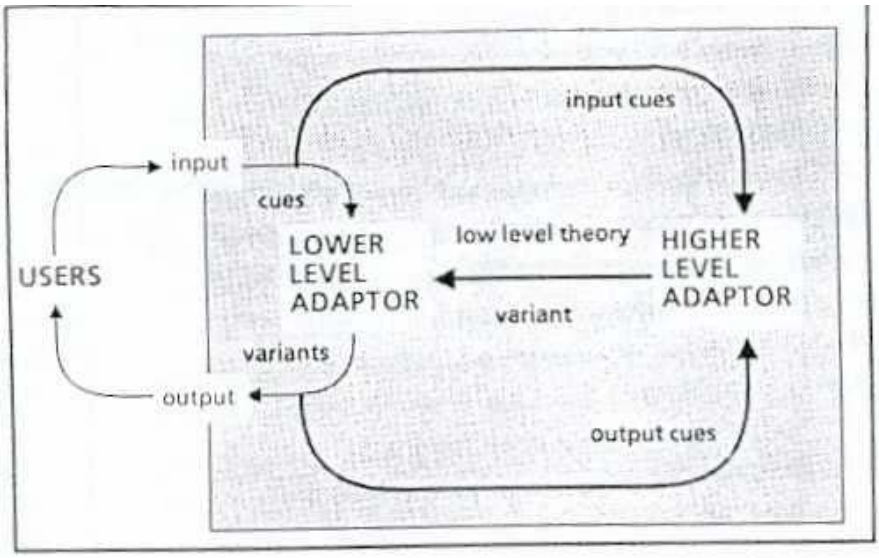

Fig. 3.9 Two level architecture for adaptation.

3.2 The need for models

An AUI is generated at run-time meeting the demand that interfaces to complex systems should be able to adapt to different users. User modelling is a key term in providing adapted services and covers the process of gathering information about the users. This knowledge source is essential for the dialogue behaviour of the system and for reasoning about the user [29].

Goals of intelligent tutoring systems (ITS) are curriculum sequencing and interactive problem solving support. These reflect the need for models in adaptive systems:

1. *Curriculum sequencing* concerns the order in which new knowledge should be learned. In textbooks the author has predefined the curriculum of the learning path in advance of the interaction. This static organisation assumes an average learner and does not take into account individual preferences. Electronic textbooks allow for random surfing, but the adaptive system should also "give hints as to what pages will be most suitable for visiting next" [32]. If the user already possesses some knowledge of a domain, he should not have to work his way through new pages that offer concepts with which he is already familiar. Today browsers may annotate visited links, but come short in comparing the content with a user model. The user model is essential for adapting the presentation distinctively for each user.
2. *Interactive problem solving support* - Model tracing is an advanced technique where the system i.e. monitors the user during problem solving and gives advice when it discovers that the path followed will lead to an error.

Individual student models that represent the student's knowledge of a domain are subject to frequent changes and hence have to be updated continuously. One way to achieve adaptivity in this setting is by inferring which concepts are learned, and compare the student-models with an ideal model made in advance of the interaction. The result is that new information may be customized according to what is most useful and understandable for each student. Used along with the hypertext and/or the Internet this yields a new way of teaching, i.e. by replacing the traditional learning situation with a non-linear course *tailored* to each student, including distance learning.

Modelling the user's knowledge and beliefs are important for an adaptive system. The KNOME-system infers what the user knows about UNIX [29], providing different answers to users with different levels of expertise. Even though the information contained in a user model varies according to the application, situation and the sort of modelling used, a typical user model needs maintenance on the following attributes [28]:

- User preferences, interests, attitudes and goals
- Skills of the user (concerning both domain and system)
- Interaction history
- User stereotypes if present

The values for the above attributes may be captured by the system implicitly or they may be explicitly told by the user. The *analysis engine* is essential to the system as a means for deriving new facts about the user. Next potential steps can be suggested and techniques for constructing the user model include Bayesian or logic based methods, neural networks, stereotypes, inference rules etc. Kules further suggests some guidelines that should be considered when constructing user models for adaptive systems, focusing on the importance of embedding the philosophy "know thy user" into the system [28]. The user should also understand his own user model and be able to adjust the attributes himself, and a prototype of the model and algorithms should be tested with the users before implementing the model into the system.

3.3 Adaptive hypermedia

The research field of adaptive hypertext can be traced back to the early 1990s and is a result of the two somewhat older parent fields of Hypertext and User Modelling. The year 1996 is a milestone in the "adaptive world" because of the explosive growth of the World Wide Web and the accumulation of experience in the field. Before 1996 the systems mostly were laboratory systems built to demonstrate ideas. After 1996 the systems built demonstrate real world settings. Brusilovski presents the current state of the art in [31], naming adaptive

recommendation systems as a challenging future research. These systems monitor the browsing activity of the user and try to deduce his goals and interests. If successful the system can present a set of relevant links. It is important to distinguish recommenders working in a closed information space from those working with the whole Internet. Today search engines make use of techniques from information retrieval research. A similar (yet far more powerful) adaptive hypermedia system needs to learn about the structure and content of nodes by analysing the documents and turn them into a corresponding closed hyperspace. Learning about the structure of documents requires working within a closed space. According to Brusilovski, there are two ways to close an open hyperspace:

1. Select the most relevant links by analysing a few steps ahead of the current browsing point of a single user
2. Learn about the documents by collecting browsing data from a community of users

The goal of this process is to understand hypertext-documents and links without use of a human indexer, and obtain documents indexed corresponding to the user's goals, knowledge and background.

Even though simple user models are able to represent all the necessary knowledge to achieve curriculum sequencing and adaptive guidance, the knowledge state of a user in www-based learning systems is complicated to maintain. Weber+ present a solution using a combination of an overlay model and an episodic user model (ELM) [32]. The episodic learner model stores knowledge about the user in terms of a collection of episodes and has several advantages: it provides selection of examples that best fit the current learning situation, it is suited for diagnosing solutions to problems and gives individualized help. Building the adaptive, knowledge based tutoring system ELM-ART II included the following steps, and illustrates some interesting details related to the development of adaptive hypertext systems in general:

1. Translating text to small sections of units/HTML-code associated with *concepts* to be learned.
2. Building a *conceptual network* with links among related concepts. When a page gets visited, the corresponding node in the network is updated. Dynamic slots are stored with the learner model for each user and make it possible for the system to guide the user optimally through the domain. By marking concepts of a unit as known an inference process (possible recursive) that marks all prerequisites to this unit as inferred is started. This corresponds to the curriculum sequencing and adaptive guidance noted above.
3. Recording all interactions of the learner (the student) in an individual *learner model*
4. Using *traffic lights* visible to the user during surfing as a metaphor for annotating links should reflect the information in the user model.
5. Dealing with *inconsistent knowledge* by means of tests.
6. Incorporating means for the user to re-use the code of previously analysed examples and to easily navigate an *optimal learning path* by clicking a next button. This feature also helps the user from getting lost in the hyperspace.

As outlined in the systems presented above, hypertext documents tend to overwhelm the reader with to much information, or an inappropriate level of detail is presented. Several other studies and systems propose solutions to these problems: METADOC uses a technique called "stretchtext" where classifications of users and concepts are used to vary the amount of detail presented. The HYPERFLEX system supports navigation by recommending topics based on preferences and goals [29]. The system KN-AHS achieves adaptivity by using the shell system BGP-MS. The user may ask about more information related to hotwords, and the data

presented reflects what the system believes the user knows by then [30]. Three systems are mentioned by Kules: AVANTI is a system that customizes web pages about a metropolitan area for different users (tourists, handicapped, elderly, residents). INTERBOOK is an advanced WWW application and supports incremental learning. In a web-environment the HTML model and the HTTP protocols limit the details about user actions. Therefore in order to infer what the user know, the system keeps track of what the user has seen. Finally, ORIMUHS is a context-sensitive help system and has a sophisticated user model [28].

3.4 Future challenges

Adaptive systems provide dynamic adaptation through possibly implicit acquisition and hence little or no effort is required from the user who may not even know about the existence of user models. With the shift from expert users to relatively naïve inexperienced users, complex systems providing adaptivity at satisfactory levels while preserving the need of the user feeling in control are required [20], [24]. Fischer emphasizes the need of separating user modelling from task modelling. Privacy should be maintained and misuse of the models avoided. These aspects challenge many commercial strategies found on the World Wide Web today [27].

4. Conclusions

The approach taken in the field of intelligent user interfaces heavily rely on underlying models and techniques as a main step towards efficient, effective and more natural ways of interaction, as outlined in this essay. A model based approach focus on models of facts as an easy way to build and maintain user interfaces. Traditional interfaces implicitly assume that the user thinks in terms of documents and applications, instead of *thinking in terms of problems.* The widespread use of the web leads to demands on adaptive Internet sites that present the information according to the user context. These days Microsoft concentrates on the .net technology [33] and claims to have given birth to a new generation of web services. Consisting of small dumb units gathered in a central database, the goal is to improve machine-to-machine communication for an easy exchange of programs embedded in web sites. This contrasts the agent-based approach focusing on small intelligent units interacting with each other.

5. References

1. “Intelligent User Interfaces: An Introduction”, Maybury, M. T. and Wahlster, W. (eds.): Readings in Intelligent User Interfaces, 1-13, Morgan Kaufmann Publisher, 1998

2. “Natural Language with Integrated Deictic and Graphic Gestures”, Neal, J. G., Thielman, C. Y., Dobes, Z., Haller, S. M., and Shapiro, S. C. (1989) Proceedings of the 1989 DARPA Workshop on Speech and Natural Language. 410-423

3. “Integrating Simultaneous Input from Speech, Gaze, and Hand Gestures”, Koons, D. B., Sparrell, C. J., and Thórisson, K. R. (1993). Maybury, M. T. (ed) Intelligent Multimedia Interfaces. 257-276. AAAI/MIT Press, 1993

4. “Plan-Based Integration of Natural Language and Graphics Generation”, Wahlster, W., André, E., Finkler, W., Profitlich, H.-J., and Rist, T. (1993) Artificial Intelligence. 63(1-2), 387-427. Elsevier Science-NL"

5. “On the Knowledge Underlying Multimedia Presentations”, Arens, Y., Hovy, E., Vossers, M. (1993). Maybury, M. (ed.). Intelligent Multimedia Interfaces. 280-306. AAAI/MIT Press. 1993.

6. “Automating the Design of Graphical Presentation of Relational Information”, Mackinglay, J. D. (1987), Transactions on Graphics, 5(2), 110-141.

7. “Automated Generation of Intent-Based 3D illustrations”, Seligman, D. D., and Feiner, S. (1991) Computer Graphics, 25(4), 123-132.

8. “Adaptation as a problem of design”, Totterdell, P., and Rautenbach, P. In D. Browne, P. Totterdell, and M. Norman (eds.) Adaptive User Interfaces. London: Academic Press, 1990, pp. 59-84.

9. “A taxonomi of adaptive user interfaces”, Malinowski, U. ,Kuhme, T., Dietrich, H., Schneider-Hufschmidt, M., In Monk, Diaper, Harrison, editors, People and Computers VII, Cambridge University Press 1992, pages 391-414.

10. “User Modeling via Stereotypes”, Rich, E. (1979) Cognitive Science. 3, 329-354.

11. “Intelligent interfaces as agents”, Chin, D. (1991). Sullivan, J. and Tyler, S. (eds.) Intelligent User Interfaces. 177-206.

12. “User and Discourse Models for Multimodal Communication”, Wahlster, W. (1991) Sullivan, J. and Tyler, S.,(eds). Intelligent User Interfaces. 45-67. ACM Press.

13. “KN-AHS: An Adaptive Hyphertext Client of The User Modeling System BGP-MS”, Kobsa, A., Müller, D., and Nill, A. (1994) Proceedings of the Fourth International Conference on User Modeling, Hynnais, MA. 99-105. Association of Computing Machinery

14. “A Knowledge-Based User Interface Management System”, Foley, J., Gibbs, C., Kim, W. C., and Kovacevic, S. In Proceedings of the 1988 Conference on Human Factors in Computer Systems (CHI'88), 1988, pp. 67-72. ACM, Inc.

15. “Beyond Interface Builders: Model Based interface Tools”, Szekely, P., Lou, P., and Neches, R. In Proceedings of Human Factors in Computing Systems, INTERCHI'93., 1993, pp. 383-390.

16. “Supporting Adaptive Interfaces in a Knowledge-based User Interface Environment”, Sukaviriya, P., and Foley, J. Proceedings of Intelligent User Interfaces IUI93 1993, pp. 107-113

17. “Knowledgeable Development Environments using Shared Design Models”, Nechest R., Foley J. D., Szekely Pedro, Sukaviriya P., Luo P., Kovacevic S., Hudson S. Proceedings of Intelligent User Interfaces IUI93 1993, pp. 63-93.

18. “Agent-Based Computing: Promise and Perils”, Jennings, Nicholas R. Proceedings of Sixteenth International Joint Conference on Artifical Inelligence, IJCAI-99, Vol. 2, pp 1429-1436, 1999.

19. “Software Agents”, Bradshaw, J., AAAI Press/MIT Press, 1997, 4-27.

20. “Direct Manipulation vs Interface Agents”, Schneiderman, B, Maes, P. Interactions, nov + dec, 1997, pp. 42-61.

21. “Multiagent Model of Dynamic Design. Visualization as an Emergent Behavior of Active Design Agents”, Ishizaki, S. In Proceedings of Human Factors in Computing Systems, CHI'96, 1996, pp. 347-354.

22. “Integrating User Interface Agents with Conventional Applications”, Lieberman, H. In Proceedings of Intelligent User Interfaces IUI93 1998.

23. “ANIMATED CONVERSATION: Rule-based Generation of Facial Expression, Gesture & Spoken Intonation for Multiple Conversation Agents”, Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Douville, B., Prevost, S., and Stone, M. (1994) SIG-GRAPH '94 413-420. © 1994 ACM

24. “User Modeling in Human Computer Interaction”, Gerhard Fischer, 2000

25. “Model-based user interfaces - what are they and why should we care? ”, Sukaviriya, Kovacevic

26. “User Interface Management Systems: The CLIM Perspective”, Ralf Möller University of Hamburg, http://kogs-www.informatik.uni-hamburg.de/~moeller/uims-clim/clim-intro.html#Li13

27. “User Modeling in Human Computer Interaction”, Gerhard Fischer, 2000

28. “User Modeling For Adaptive And Adaptable Software Systems”, Bill Kules, 2000

29. “User modelling for adaptive computer systems: a survey of recent developments”, Michael F. McTear, 1993

30. “KN-AHS An adaptive hypertext client of the user modeling system BGP-MS”, Kobsa+

31. “Adaptive Hypermedia”, Peter Brusilovsky, 2001

32. “User Modeling and Adaptive Navigation Support in www-based Tutoring Systems”, Weber and Specht, 1997

33. http://msdn.microsoft.com/

34. “Agents that reduce work and information overload”, Doug Riecken, 1994