4 ADAPTIVITY

Developing an adaptive user interface (AUI) requires an interface that can be adapted, a user model and a strategy for how the adaptation should take place. An overview of the field intelligent user interfaces (IUI) is briefly introduced in section 4.1. Within this context, adaptive systems are more deeply examined in section 4.2. For adaptive systems, the importance of the underlying models can not be underestimated, and are discussed in section 4.3. With this, a foundation for understanding adaptive hypermedia, whose strategies are brought to light in section 4.4, is made. Finally, section 4.5 outlines an architecture for an adaptive hypertext system (AHS) to help illustrate the main points of this thesis.

4.1 A model for IUI

Traditionally, 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 [*Maybury+98*]. 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 4–1 gives an overview of a generic model underlying most intelligent user interfaces. Important sub-fields of IUI are exemplified below.

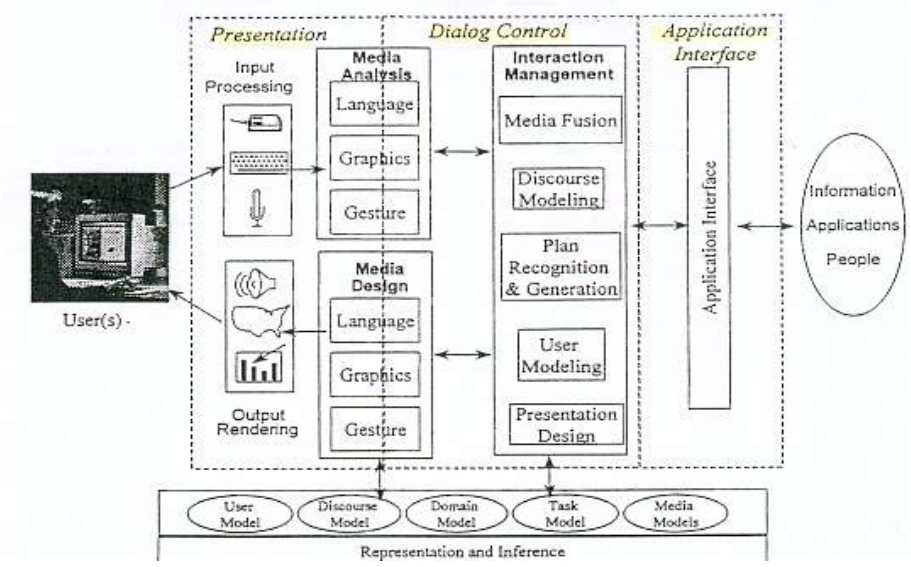

Figure 4–1: Overview of the field of IUI

4.1.1 Multimedia analysis

The field of multimedia analysis supports intelligent processing of multimodal input. The information from all the input sources are interpreted and merged together into one integrated meaning, as shown in the “interaction management” and “media analysis” areas of Figure 4–1. The goal is to let the user *communicate* with the machine instead of only using it, while accepting possibly ambiguous input. 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 [*Koons+93*].

4.1.2 Multimedia presentation

Multimodal presentation systems use many media in parallel in addition to exploiting the strong sides of each medium. The generic problem when using multimedia in an intelligent setting concerns 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 [*Arens+93*], as also depicted in Figure 4–1. The knowledge-based presentation system WIP generates multimodal presentations by means of an incremental planning process while reasoning about the task and context [*Wahlster+93*].

4.1.3 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 graphical presentations and from the user’s point of view to remove the possible ambiguity between intended and interpreted presentations. Since the design process needs to be tailored to the context, task and user, it relies upon the use of models. The IBIS system makes use of a generate-and-test approach with a goal-driven search process. If a solution is not satisfactory, the system backtracks so that the illustration can be regenerated. Formalizing the intention of a communication reduces the ambiguity of presentations [*Seligman+91*].

4.1.4 Modelling and plan detection

Typical tasks of intelligent systems like planning explanations, answering questions based on prior discourse and supporting interruption, rely upon the use of underlying models. A user model contains information about users. A discourse model has descriptions of the history, syntax, semantics and pragmatics of the dialogue between the user and the system. 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 e.g. whenever changes in environment or internal state occur. The UCEgo agent needs to be both autonomous and do rational planning to make intelligent initiatives. Goals depend on context, and the central problem for UCEgo concerns how to detect new goals when they are not stated by a human planner [*Chin91*].

4.1.5 Model based approaches

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 the reduction of the time and expertise required to create user interfaces, the identification of reusable components, and the construction of 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 blessed with more powerful design tools. Modifying the behaviour of an interface only requires to change the model instead of reprogramming a certain section. 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 [*Nechest+93*]. The overall goal of Mastermind is to generate automated and animated help facilities, as in UIDE [*Foley+88*], [*Sukaviriya+93*], and use the design models to “map low-level user gestures onto high-level semantics” [*Möller*]. The prototyping stage is easier because conceptualization is regarded as a search in a space of alternative designs and explicit models ensure consistency between task and design.

4.1.6 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 [*Schneiderman+97*]. Direct manipulation and software agents are complementary rather than mutually exclusive. Agents are more convenient in settings with complex environments, difficult tasks, a dynamic network of people and information, or relatively naïve users.

Tveit observes that the most common classification scheme distinguish weak from strong agencies [*Tveit01*]. In a weak notion, agent attributes are:

- Autonomy - agents decide for themselves what to do and when to do it
- Interactivity - they are willing to work in concert with other agents when requested to
- Reactivity - they are able to sense and act on the environment
- Proactivity - they take initiative

In a strong notion, the following attributes add to the previous list:

- Veracity - their actions can be trusted by a person and they do what they are told

- Mobility - they move around from one place to another whenever necessary
- Rationality - they perform their actions in an optimal manner

According to Bradshaw [*Bradshaw97*], one may either look at the agent as an ascription 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). He points out that communication with humans should take place in a non-symbolic, natural language-like way and that the agent should be using models to infer new knowledge. In short, its overall task is to adapt to its environment and learn from experience over time.

Agents fit complex software systems well, since they can be viewed as organised sub-systems that facilitate decomposition and interaction [*Jennings99*]. Furthermore, they may play various roles for different people and situations, i.e. as personal assistants, intelligent user interface managers, agents behind the scenes, performing agent-to-agent communication etc. Acting as personal assistants, agents become more effective as they learn the preferences, habits and interests of a user. How do they acquire sufficient competence of their users, and do users actually trust the help offered? A knowledge-based approach to the problem makes use of domain specific background knowledge about both application and the user [*Maes94*]. Maes argues that an alternative approach that relies on machine learning techniques 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.

4.2 Adaptive systems

Presenting easy, efficient and effective interfaces is the main goal of adaptation. [*Malinowski+92*]. It is difficult to write software that will fit millions of users perfectly. Nobody learns a system completely but uses different parts of it and shares some common basic knowledge about its functionality. Adaptive systems change this paradigm by turning use time into a different kind of design time by adjusting the interface according to the user's skills, knowledge and preferences. In order to achieve adaptivity, underlying models of both user and task are essential, as well as the separating the user interface from the application [*Fischer00*].

4.2.1 Classification of adaptive systems

When working with computers, users need to adjust the interface according 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. Malinowski et. al describe a taxonomy that

places adaptive systems in the context of intelligent user interfaces: Roles of an intelligent interface are fulfilled by the integration of an AUI, an intelligent help system and an intelligent tutoring system. These roles comprise adapting to the needs of the user, provide context sensitive help, and supporting the user of the system [*Malinowski+92*]. 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 called 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 [*Sukaviriya+93*]. The second covers the automation of tasks and generation of new complex functions, offering a solution to an important goal of intelligent systems, namely to 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 a higher-level adaptation accounts for goals and tasks of the user as a basis for achieving functional adaptation.

4.2.2 Factors and roles

It is important to decide when interaction should occur, what information to use and how to present it on the screen. The needs of users or groups of users must be considered before the system is built. At use time, adaptation can 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, e.g. situations where the user and the system are trying to adapt to each other and thus never reach upon an agreed interface [*Malinowski+92*]. The adaptation process might also confuse the user if not done carefully. Fischer stresses that in an adaptive setting little or no effort is required from the user, possibly resulting in loss of control [*Fischer00*]. In adaptable systems, however, the user is regarded to know its tasks best and should therefore make 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 an important factor when designing intelligent user interfaces. Most systems behave intelligently only in their original surroundings - with changes, the performance degrades. An adaptive system, however, gains its power by reacting to a changing environment. One way to defer the design is by incorporating different variants into the system and let triggers activate the set of design choices. Hence measurements for evaluating the benefits of the design are important for the adaptation process, whose requirements are:

- a theory which relate user behaviour to user interface needs
- access to what the user does
- models of e.g. task and user
- a flexibility in the user interface to accommodate new design variants
- an agent to make this design choice

Rautenbach uses a simple game for classifying adaptive systems and proposes a two level architecture for adaptation.

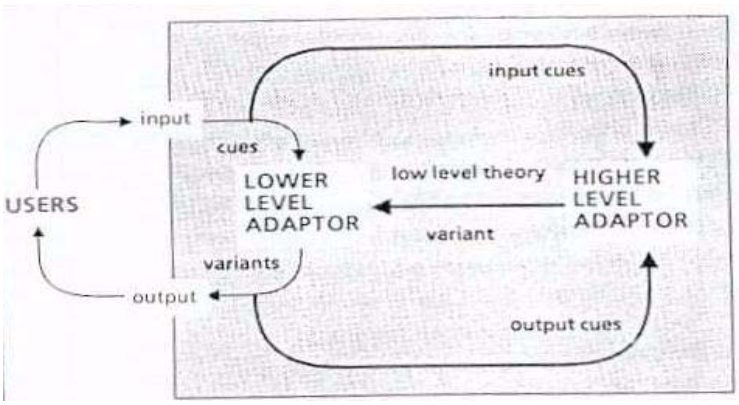

Figure 4–2: Two-level architecture for adaptation

In the model in Figure 4–2 a higher level adaptor identifies major changes and chooses the best design-variant, while at the lower level, the focus is on adapting the interface according to the user's needs. This separation of modeller and introspector is convenient [*Rautenbach+90*].

4.3 Models add power to adaptive systems

An AUI is generated at run-time, meeting the demand that interfaces to complex systems should be able to adapt to different users. In particular, user modelling is a key term for the provision of adapted services, and covers the process of gathering relevant information about each user. This knowledge source is essential for the dialogue behaviour of the system and for reasoning about the user. In this section models, and especially user models, are explored in the context of adaptive systems.

4.3.1 The importance of external models

Making models explicit to the system extends flexibility, but these sources may in some systems be implicitly contained in the code, and furthermore distributed or centralised [*Malinowski+92*]. Modelling components should maintain the models, e.g. by building the model incrementally, maintaining its content, providing consistency, and supplementing other components of the system with information about the user or the dialogue. The XTRA system uses external components in order to assist the user in filling out tax-forms [*Wahlster91*].

4.3.2 Representation

According to Davis et. al, a knowledge representation is a fragmentary theory of intelligent reasoning [*Davis+93*]. Despite its incompleteness, adaptable behaviour does require intelligent selection of content, and in order to achieve it, the choice of representation becomes important.

A user model can represent individual users or classes of users. The dimension ranges from individual user models, through stereotypes, to canonical models. Individual models may be cost-expensive with respect to maintenance, but they provide more flexibility to the system. Canonical models do, on the other hand, characterise abstract, typical users. In between, stereotypes are clusters of characteristics and have proven useful for building powerful models of users when no, or only a small amount of information, is available [*McTear93*]. Rich presents the system GRUNDY that acts as a librarian recommending books based on a dialogue with the user [*Rich79*]. Two kinds of information is required for enabling an effective use of stereotypes in GRUNDY. First, a set of *facets* associated with values will in sum characterise the user. Secondly, *triggers* signal when a stereotype is appropriate and should be activated. The information in the stereotypes is probabilistic, and therefore constitute <attribute, value, rating> triples. The user model, or user synopsis (USS), is built from information of actions, stereotypes and the user. Since the system must justify its own information, the USS consists of a set of <attribute, value, rating, justification> quadruples and is used for guiding the rest of the system. Inheritance plays an important role for stereotypes in order to deduce information.

While the stereotypes in GRUNDY rely on linear parameters with simple numeric scales, other techniques for constructing user models exist. If the system should represent the user's knowledge, goals, plans etc., a more expressive scheme, like concept-based representations, is needed. [*McTear93*]. In FRONTMIND, a bayesian network is used [*Kobsa01*]. Other schemes can be based on logic, inference rules, frames, production rules and connectionist (neural) networks. In the SiteIF project the user models are represented by semantic networks where every node and relation have weights in order to represent different levels of interest for the user [*Stefani+99*]. Kules 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 [*Kules00*]. The user should also be aware of the existence of, and understand the user model, possibly being allowed to adjust its attributes. Fink et. al stresses privacy issues and an open dialogue with the user in their AVANTI system. In additional to technical solutions regarding security, users can choose the sort of modelling to be used [*Fink+97*].

4.3.3 Acquisition, maintenance and reasoning

In GRUNDY, learning happens through the modification of the stereotypes. In general, updating user models is important when interacting with a user over time. The values of the above attributes may be either explicitly captured by prompting the user for information (user driven acquisition), or implicitly during the course of

dialogue (system driven acquisition). In the first the system plays a passive role. Stereotypes might be used when information about the user is limited, or as a supplement to other methods. Implicit acquisition, on the other hand, is more dynamic and requires rules of inference and a way of handling conflicting information [*McTear93*]. The analysis engine is essential to the system as a means for deriving new facts about the user, and next potential steps can be suggested.

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 [*Kules00*]:

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

Connectionist networks can more easily handle inexact information and incremental acquisition by assigning each node with energy levels (which are spread to connected nodes in the network) and allow the weights of the nodes to gradually evolve over time.

How to model the user's knowledge and beliefs is important for an adaptive system. Plan recognition helps inferring new tasks and is a source for further information about the adaptation. In order to provide context-sensitive feedback, plans can be recognised by monitoring or reasoning about the user *[Malinowski+92].* The KNOME-system infers what the user knows about UNIX, providing different answers to users with different levels of expertise. McTear emphasise that models tend to be incomplete and inconsistent. The information provided by the user is more likely to be true than information based on the user's stereotype. The more specific information should therefore override the more generic information if the properties are inherited. Methods for combining or adjusting numerical values are also part of the maintenance process [*McTear93*].

Dynamic models are closely related to adaptive systems, but require methods for resolving conflicts. According to Kobsa, the shift from traditional shell systems towards less demanding domains like user-tailored web sites, made complex user modelling redundant, i.e. other aspects yield significance:

- quick adaptation should be based on short initial interaction
- companies can integrate their own methods or third-party tools
- heavy work should be distributed
- mechanisms to recover in case of system breakdown
- inconsistencies and faults in the models must be avoided

New services like predicting the future actions of a user based on trends among similar users, demand support for privacy policies and explicit representations of the patterns inferred. Systems like PERSONALIZATION SERVER and GROP LENS, provide many benefits: easy access among applications on user information, methods that can be applied for model protection, easy integration of complementary information from different sources, and centralising the user model servers in order to relieve the clients from the user modelling tasks. Serving many

applications at the time is solved by allowing the user modelling system to communicate with the application through inter process communication [*Kobsa01*].

4.3.4 ITS, an illustrating example

Hypertext documents tend to overwhelm the reader with information or present an inappropriate level of detail. User modelling helps an adaptive system to avoid presenting information that is already known to the user. Goals of intelligent tutoring systems (ITS) are curriculum sequencing and interactive problem solving support. These goals reflect the need for models in adaptive systems well. The first 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. Such a static organisation assumes an average learner and does not take into account individual preferences. Electronic textbooks take user freedom a step further since they allow for a more random-like surfing through the text by choosing links. An adaptive system should, in addition to this freedom, "give hints as to what pages will be most suitable for visiting next" [*Weber+97*]. In other words, the user should not have to work way through new pages that offer knowledge with which he or she is already familiar. Browsers that annotate visited links come short in comparing the content of a document with the user knowledge. Representing the individual user knowledge in corresponding user models is essential for adapting the presentation distinctively for each user. The second goal concerns interactive problem solving support and can be facilitated through model tracing, using techniques where the system e.g. monitors the user during problem solving and gives advice when it discovers that the path followed will lead to an error.

In ITS, individual student models representing the student's knowledge of a domain, are subject to frequent changes and therefore have to be updated continuously. One way to achieve adaptivity in this setting is by inferring which concepts are learned, and compare each student model with an ideal model constructed in advance of the interaction. The result is that new information can be customised according to what is most useful and understandable for each student. Combined with hypertext and a Web environment, this yields new opportunities for net-based teaching, i.e. replacing the traditional learning situation with a non-linear course tailored to each student.

4.4 Adaptive hypermedia

Conventional hypermedia applications offer navigational freedom, but rely on a strong authoring model which assumes that the author "knows best" [*Bodner+00*]. Adaptive hypermedia systems attempt to overcome these problems of orientation and comprehension, particularly necessary in a Web environment. As noted before, an adaptive system uses explicit models to achieve its goals often founded on intelligent technologies for user modelling and adaptation.

4.4.1 A brief history of adaptivity

Brusilovsky presents the state of the art in one of his surveys [*Brusilovsky01*]. The research field of adaptive hypermedia 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, mostly laboratory systems were built to demonstrate ideas, whereas systems born after 1996 demonstrate real world settings. Currently, three major technologies exist, namely adaptive content selection, adaptive navigation support and adaptive presentation:

1. **Adaptive content selection:** Such systems select and prioritise the information resulting from a query according to what is assumed to be most relevant for the user.
2. **Adaptive navigation support:** Studies have shown that manipulating link anchors can increase the navigational speed and prevent the user from feeling lost in hyperspace, but also that only certain strategies work [*Höök+99*]. Approaches include directed guidance, link hiding, link sorting, link removal, link annotation and map adaptation.
3. **Adaptive presentation:** Systems that have means to present the content dynamically or adaptively belong to this category. A widespread strategy is to conditionally show, hide, highlight or dim fragments, whereas other systems tailor new, adaptive documents by comparing a user model to an overlay domain model. Presentations according to the needs of each individual visitor is often referred to as *customization*, whereas *transformation* improves the presentations by identifying interaction patterns from all visitors. Finally, *content based adaptation* organises material based on content and apply for ITS and the like.

Whereas the first generation of adaptive hypermedia concentrated on modelling user knowledge and goals serving adaptive navigation support and adaptive presentation, the second focused on adaptive recommendation systems based on modelling user interests. These systems monitor the browsing activity and try to deduce the goals and interests of the user. If successful they can present a set of relevant links. Nowadays, a third, mobile generation adds to the system models of context like location, time, bandwidth and platform, hence improving functionality so as to adapt to the user situation as well [*Brusilovsky+02*].

4.4.2 Recommenders – an example of adaptive hypermedia challenges

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 Brusilovsky, there are two ways to close an open hyperspace [*Brusilovsky01*]:

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

The goal of the process is for the system 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.

4.4.3 Other approaches to individualised presentations

Next door to adaptive hypermedia systems we find the field of dynamic hypertext which uses natural language techniques in order to fuse information and move away from the strong authoring model offered by traditional hypertext. Dynamic hypertext unifies querying and browsing, thereby avoiding the need of switching between search mode and browse mode. Another benefit is that dynamic links don't get broken. Bodner et. al claim that systems implementing the idea work best for collections with coherent vocabulary and well written text [*Bodner+00*]. Both adaptive hypertext and dynamic hypertext tailor documents to the user. The latter contrasts adaptive approaches since there are no existing hypertext documents before the user requests them [*Milosavljevic+98*]. In other words, dynamic hypertext systems move further than adaptive hypertext systems, thereby overcoming the problem of committing to some pre-written segments.

4.4.4 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 in the Web environment, complex systems providing adaptivity at satisfactory levels while preserving the need of the user feeling in control are required [*Schneiderman+97*]. Fischer emphasises 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 [*Fischer00*].

4.5 Planning an adaptive hypertext system

According to Wu et. al, most adaptive hypertext system architectures depend on a domain model (DM), a user model (UM) and an adaptation model (AM) [*Wu+01*]. In this thesis we focus on ensuring quality to the construction of the domain model of an AHS, whose overall strategy is introduced in this section. Some components are based on ideas from ITS.

In order for adaptation to take place in a sophisticated way, the AHS needs to build a model of the domain, maintain individual user models, and generate adaptive documents by reasoning on what the user should be presented next.

4.5.1 Building a domain model

A domain model must represent important information in order to express the domain knowledge. As shown in Figure 4–3, we plan to first parse each HTML-document in order to extract as much information as possible, aiming at automatically extracting proper concepts and relations of high quality. Rules should assist the selection of concepts that describe the knowledge of the documents and their sections, and help identify relations among the concepts. DM is incrementally built as documents are exposed to analysis, but since it is regarded a difficult task to determine the meaning of a Web page reliably [*Perkowitz+00*], one might assume the results to be far from perfect. Somewhat similar to ITS, whose modelling concern how a domain expert would represent the knowledge to be taught to the learner [*Beck+96*], manual evaluation and adjustment from someone skilled in the domain is probably needed in order to perfect the DM. For our approach the interesting question is whether we can realise a domain model without asking too much of the author.

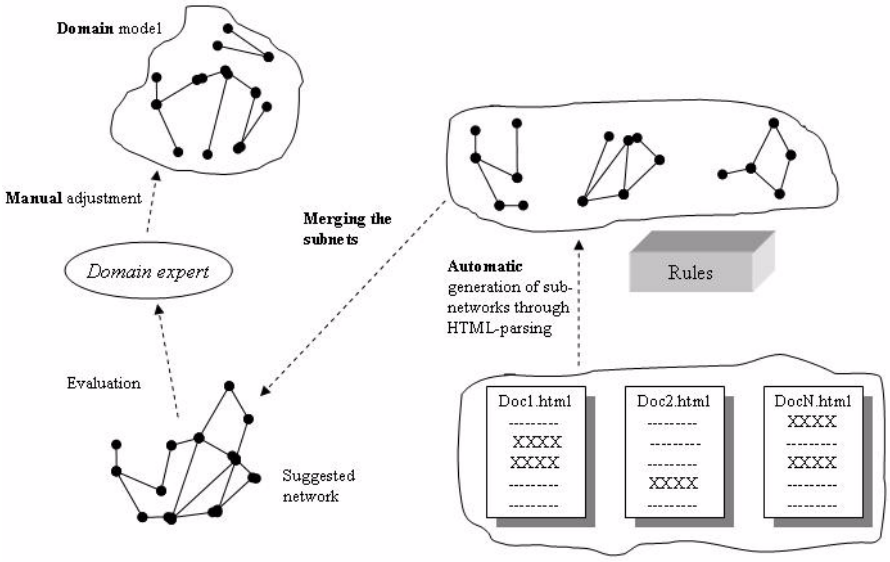

Figure 4–3: The process of making a domain model is semi-automatical, which means an expert should evaluate the proposed model and adjust it manually.

4.5.2 User modelling

By monitoring the user during interaction a conceptual user model that represents the user knowledge in terms of concepts is incrementally built. This information reflects which concepts the system believes the user presently knows. In ITS, one way to represent student models is through an overlay model [*Beck+96*]. The same assumption is made in our system so that the knowledge of the student is regarded

a subset of that of the expert, illustrated in Figure 4–4. We note that due to student misconceptions such a model may be imperfect.

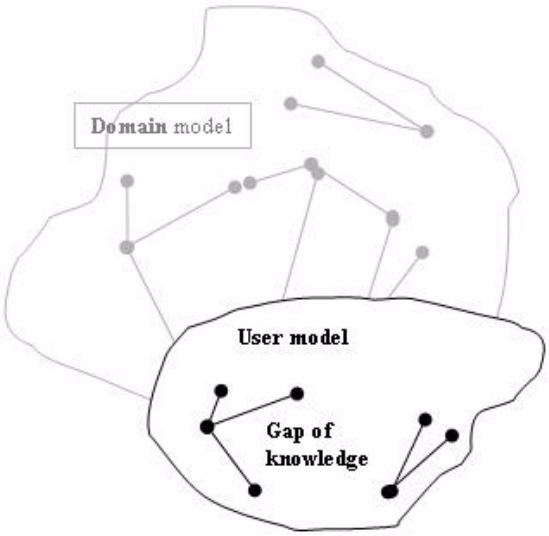

Figure 4–4: The user model is a subset of the domain model.

4.5.3 Generating adaptive presentations

For adaptation to take place, the system needs to modify documents so that they allow for the present knowledge of each user based on information about what the user already knows. The system could perform a real-time analysis of the contents of the document requested for, but there are two problems to overcome. First, going through the processes of parsing, conceptualising and building a network representation again would likely result in an imperfect temporal representation of that document. Secondly, as mentioned before, it is expensive to perform such an analysis in a real-time environment and requiring the user to wait too long would be unacceptable. As a solution, adaptive documents of high quality can be generated simply by comparing the incomplete user model to the perfect domain model. Given that the domain has been analysed in advance, this approach is both inexpensive and flexible. Each presentation would therefore consist of information represented by the concepts selected by the system, accounting for individual preferences of the users. Hence in order to bridge the gap of knowledge, the system would need to reason on which concepts the user has insufficient expertise and

tailor adaptive documents with the required information. The steps of this process is visualised in Figure 4–5.

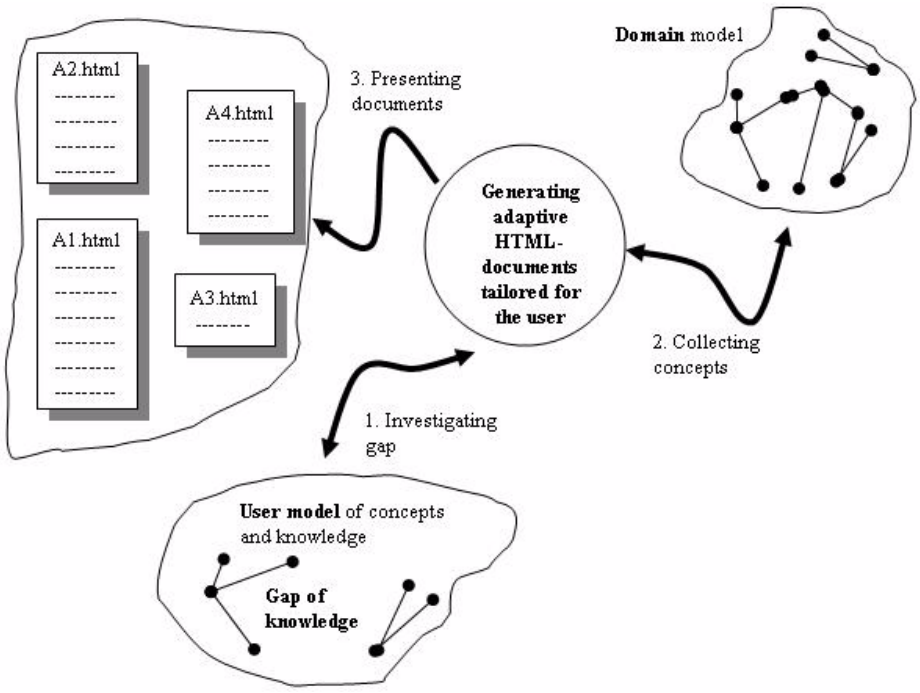

Figure 4–5: Adaptive generations are based on both DM and UM.