2 Motivation

Throughout the last decade, several studies in the field have shown attempts to make systems that fit individual users better. In an Internet context, adaptive hypertext systems are named as promising.

2.1 Background

With his MEMEX, Vannevar Bush tried to beat the information overload he was facing in 1945 [*Bush45*]. His ideas of storing and linking microfilm documents are the origin of the hypertext idea, which have developed to the collection of documents linked together in a non-linear manner today known as the World Wide Web [*Berners-Lee96*]. The information overload problem has become much larger than the generation of Bush ever imagined, and despite a dynamic conduct in terms of a continually changing document collection, the Web is in many ways a very static medium regarding the content presented to each user.

2.1.1 Challenges of HCI

From being a tool for the scientist or expert user only, the computer turned into common property a couple of decades ago. As the number of people that used a computer increased, so did the need for more user friendly systems. The society today is flooded with information and is commonly referred to as “the information society”. From a user point of view, computers are tools that provide services like money transactions, entertainment, information development and exchange. In order to fulfil their services, it is often required that the users have certain skills in how to use the systems. Unless being trained before interaction, the fellow man might therefore face troubles when interacting with computer systems. From the user point of view, one should, ideally, not need to have any technical knowledge on computers in order to survive in the digitized world.

An user interface constitute the intersection between the user and the software. In the area of Human Computer Interaction (HCI), the user is in focus. The field concerns the design of user interfaces, mental models to help the programmer in the design phase, and tools to support the development and implementation of interfaces. Schneiderman points out that due to the possibly broad spectrum of usage situations, the developers are forced to go for designs that benefit all users or the average user. The final product is not only constrained to user demands, but also to the technology available. Thus, for the user, the individual satisfaction of a piece of software may vary a lot [*Schneidermanoo*]. Due to the great challenge

when designing for a broad audience, several techniques have evolved during the past two decades in order to support developers to meet the goal of *usability.*

Usability concerns how well a system can be used. If a system can be easily learned, is efficient in use, its functionality easy to remember, its appearance pleasant to the user and the offered services are regarded trustable, the user is more likely to feel satisfied. E.g. command based systems that have no graphical user interface might please some users, but the vast majority will probably feel more comfortable with a graphical user interface, or GUI, using windows, icons, menus and pointing manoeuvres in order to navigate and act. Even though the graphical environments of today are somewhat more user friendly than their text-based predecessors, users still feel it challenging to learn new systems and programs. In order to write a document, a novice user must first learn how to navigate in the operating system and how start and use a text editor and its functionalities - which may indeed not seem too obvious. Thereafter, the typing begins and the user eventually discovers that it is not easy to instruct the computer on what to do. All details on which procedures to follow in order to initiate an action are likely to interrupt or confuse the user, hence removing focus from the task at hand, i.e. writing the content of the document.

According to Nielsen, the notion of usability is difficult, yet important, to realise [*Nielsen92*]. Moreover, the design and implementation of user interfaces is a complex matter. There is no silver bullet to make the design and implementation easier [*Myers94*], and many problems face designers, only but a few are mentioned here:

- designers have difficulties thinking like users do
- the tasks and domains in which to operate can be complex
- iterative design is difficult, but necessary
- the design process is creative rather than mechanised due to lack of theory and methodology
- there is a huge span in user knowledge, and the cost of making many differing versions of the software often is too expensive.

Nielsen further proposes an usability engineering *lifecycle* of design. In order to obtain successful user interfaces, the process of usability engineering should not only consider a larger context and analyse the user needs and characteristics *before* the design stage, but also collect user feedback *after* the release date. In other words, the result should be regarded as a prototype for the next version of the software.

2.1.2 Designing intelligent user interfaces

The difficulties involved in creating an interface that is easy to understand, yet flexible for the user, have inspired the growth of another field, namely one that adds intelligence to the system. The field of intelligent user interfaces (IUI) is an effort to solve the problems of usability, thereby increasing the user satisfaction of the interactive experience. Combining techniques from artificial intelligence (AI) with HCI yields interesting possibilities, including the ability for the system to reason on what the user really wants, adapt its behaviour to each individual user,

provide context sensitive help, understand inaccurate user input and generate presentations on the fly according to individual preferences or the knowledge level of each user.

As an example, an intelligent interface to an image processing program could monitor the user's actions and then try to infer the user's goals. Thereafter, it would offer to complete the work for the user. Say, if the user resized a picture, saved it, and then repeated the action for another picture, the intelligent system would expect the user to repeat the same procedure on other pictures as well, and hence it should offer the user to automatically complete the task of resizing and saving a series of pictures. Identification of patterns from user behaviour is a simple example of intelligent behaviour. A wide range of potential application areas exist, however. Fine, if IUI is such a good thing with its main goal to meet the user in new, more intelligent ways - then why don't existing software apply such systems?

In order to improve efficiency and naturalness of the interaction, an IUI must represent, reason and act on models of users, domain, task, discourse and media [*Maybury+98*]. In short, the extended model needed for IUI development along with the difficulties involved in applying artificial intelligence techniques, makes the process of designing an intelligent user interface a lot more complex than that of traditional user interfaces. Chapter 4 contains a survey of some subordinate fields of IUI.

2.1.3 Design problems on the Web

Hypertext provides information in a non-linear manner (e.g. as opposed to the predefined curriculum of textbooks) which means the user can decide what to visit next and reach related information through visiting linked documents. The links is a structure of paths, and the paths available offer flexibility to the user, as illustrated in Figure 2–1. For a domain on a specific topic, notice that since readers might jump in on a document from somewhere else, it is not even certain that the user has followed one of the paths as outlined by the author [*Berners-Lee95*]. In his style guide for online hypertext, Berners-Lee emphasises that what is natural as links for one visitor may seem redundant for others [*Berners-Lee95*]. Due to the navigational freedom of hypertext environments and the vast amount of information available, users often feel a sense of getting lost in hyperspace, that is they are confused about where they are, or where the information they seek is

located. In general, publishing information online does not automatically make that sort of information more accessible.

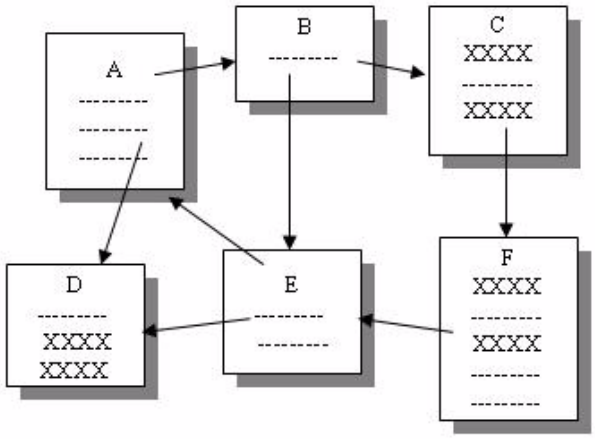

Figure 2–1: A typical hypertext graph structure. Some users might choose to jump directly from A to D, whereas others prefer to follow the path A to B to E to D or even the entire path A to B to C to F to E to D.

The implicit ordering of a document concerns the organisation of information within the document. The author of one domain does not have to know about the content of another, nor what the user knows in advance of interaction, and therefore a document essentially consists of the information the author finds most natural to group.

With the growth of the Internet, the usability and design problems have become particularly challenging. The huge span in user interest, level of knowledge, motivation and experience, makes it very hard to publish online material that is suitable for each particular user using conventional methods. The field of Adaptive Hypermedia, and in particular Adaptive Hypertext, promises to overcome such problems by offering hypertext presentations that are automatically tailored to each particular user. For instance, a domain on the Web offering articles on “Astronomy” may expect very different visitors ranging from the open-minded child, the novice student, the retired woman pursuing the hobby of star watching, to the Doctor of Astronomy. Obviously, what is comprehensible for the expert may be very difficult for the child, and what is made fit the novice is annoyingly trivial to a more skilled person. When accessing a document that contains difficult information, the novice should therefore be presented the concepts necessary to understand first. Likewise, an adaptive hypertext system should not present all the information with which the expert is already familiar, but trim it to a suitable level. Moreover, the system could deliver a wide range of presentational forms based on individual preferences, i.e. graphics, text, tabular information, animated sequences and the like.

As with IUI, models are needed in order to make adaptive hypertext systems, in particular models of the domain knowledge and the user knowledge. The number of adaptive (hypertext) systems is increasing, but it is, however, considered time

consuming to integrate existing web-pages into an adaptive environment since adaptivity requires that the system has knowledge of the content of each document in the domain, and this process is often plotted into the system's knowledge base by hand.

The challenging task of integrating adaptive behaviour into an Internet setting is of ever more interest. We therefore feel the need to decrease the time required to model the knowledge of the domain. With the basis that it is possible to, at least semi-automatically, *process* existing hypertext documents and extract the information within in such a way that a system can apply the resulting model so as to offer adaptivity to the end user, this thesis explores how.

2.2 Goals

According to Kobsa, there are two main problems with traditional hypertext [*Kobsa+94*]. First, the user often finds it difficult to orientate in large virtual spaces and therefore easily gets lost during the interaction. Second, it can be hard for novices to understand difficult concepts, since regular documents are written for the average audience.

2.2.1 Analysing, designing and bridging

The idea that the user should decide what to visit next through browsing has both positive and negative sides. In a traditional hypertextual context this facility provides freedom to the user, whereas in the adaptive world, it acts as a limitation. In advance of the interaction the user does not know exactly where in the hyperspace each particular chunk of knowledge exists, and in order to get to the information sought, the user has to rely solely on the descriptive names of the links, menus or sitemaps as organised by the author[1]. Furthermore, in situations where some concepts are assumed to be known prior to visiting a document, the information presented could be difficult to understand if the user lacks knowledge on these concepts.

If a system could both learn the content and the informational properties of a domain, and infer what each user knows at a particular time, it would be able to redefine the predefined (implicit or explicit) curriculum of the domain for each user at run-time. In order to prepare a domain for such an adaptive system, a model of the domain must be constructed and understood by the system. Moreover, the better and more automated the construction of the domain model, the less the threshold for an author to go for an adaptive solution while trusting that accurate adaptations will result. Based on the discussion so far, we define the main goal of this thesis as to design an adaptive hypertext system that can work behind the scenes and tailor the documents of a domain according to the knowledge of each

1. A sitemap shows the structure of the domain so that the user can more easily get an overview of how it is organized and jump directly to the area of interest.

individual user, doing so through an extensive analysis of existing HTML-documents without requiring the author to rewrite and reorganize new documents. A system would perform at best if the domain model is as complete as possible [*Davis+93*]. We therefore mainly concentrate on ensuring quality to the representation of the domain knowledge.

Table 2–1: The main goal of this thesis

The main goal of this thesis is to design an adaptive hypertext system in order to bridge the gap between the domain knowledge and user knowledge, and building the models needed in the system through an analysis of existing hypertext documents of the domain.

As will come clear throughout this thesis, an adaptive hypertext system has many target application areas. E.g. Schank stresses the need for 1-1 learning situations [*Schank95*]. As with the astronomy-example, electronic education often suffers from a lack of individual considerations. In an educational context, a teacher would benefit if offered means to turn an existing static course into dynamic presentations tailored to each student with very little effort required. In addition to the adaptive generations of content, the system could help students in planning how to browse the domain, or give advice along the road.

2.2.2 Methodology

Computer Science have historical bonds to mathematics, science and engineering. According to Denning these roots are the basis for the three paradigms of Computer Science, namely those of *theory*, *experimentation* and *design* [*Denning99*]. Furthermore, Denning divides Computer Science as a discipline into several subareas, each of which has activities in each of the three paradigms. The areas named range from Algorithms, Programming, Architecture, Operating Systems, Databases/Information Retrieval and Software Engineering, to Artificial Intelligence, Graphics, HCI, Computational Science and Bioinformatics.

Whereas the paradigm of theory concerns the construction of frameworks, the paradigm of experimentation explores and tests models of systems. The paradigm of design focuses on how to build computer systems that work in given application domains. Theoreticians often aim at deep, comprehensive analyses around formal models. Experimenters often use prototyping to quickly construct models of systems. Designers build systems according to accurate specifications which satisfy their customers.

This study belongs within the intersection of the paradigms of experimentation and design. Aiming at semi-automatically building a domain model from a collection of hypertext documents, we first experiment on how the use of certain techniques can help in the process, then validate the experiments through empirical prototyping, followed by a discussion of the results which is the basis for designing an adaptive hypertext system.