The Knowledge Engineering Review, No. 6:1, 1991, 21–47

A review and synthesis of user modelling in intelligent systems

ANDRÉ J. KOK

University of Amsterdam, Department of Social Science Informatics, Herengracht 196, 1016 BS Amsterdam, The Netherlands

Author's current address: MID/OIG, knowledge-based systems group, Verryn Stuartlaan lg, 2288lk Ryswyh, The Netherlands

Abstract

This paper gives a state-of-the-art overview of the rapidly expanding field of user modelling in artificially intelligent systems. After showing how a user modelling component can improve parts of the processing of, and the interaction with, a large number of systems, the current situation in this field is sketched by means of a number of short descriptions of systems employing user modelling techniques. Next, a synthesis of this review is made by discussing the aspects on which the existing methods differ. It turns out that these aspects can best be approached from two points of view: the technical level and a more abstract, functional level. This dichotomy results in eight dimensions on which to compare the modelling methods. The synthesis is then completed by describing the existing modelling techniques, and classifying the reviewed systems. In the final section, current trends in the field are outlined and prerequisites for acceptance of user modelling on a larger scale are discussed.

1 Introduction

User modelling deals with representing characteristics of users working with artificially intelligent systems. The application of user models can improve the interaction with these systems in an essential way. Adapting the system's behaviour to individual preferences becomes possible. Identifying misconceptions of the user can lead to actions undertaken by the system to repair these. The role of the system in the dialogue with the user can become more important by applying user modelling, i.e., more initiative is taken by the system. Instead of assuming a passive role, the system becomes a helpful supporter for the user in fulfilling his goals.

User modelling has become an active subfield of artificial intelligence and human computer interaction. International conferences are dedicated to it (Maria Laach, West Germany, 1986; Hawaii, 1990), the number of papers on (aspects of) user modelling increases every year (Kobsa, 1988), special issues of journals appear (e.g., *Computational Linguistics* **14**(3), 1988), and a new international journal *User Modeling and User-Adapted Interaction* has just been founded.

Because of the increasing volume of work in user modelling, getting a proper outlook on the field becomes hard. Overviews of user modelling exist but suffer from three problems: they are limited to specific areas (for instance, intelligent tutoring systems, [Clancey, 1986]; information retrieval, [Daniels, 1986]; natural language processing, [Wahlster and Kobsa, 1989]), they only describe systems that have now become obsolete, or they lack a synthesis and are therefore mainly an enumeration of some projects.

This paper tries to overcome these problems. It presents a review and synthesis of the current state-of-the-art of the user modelling field. The paper explains why constructing and maintaining a model of users is profitable, and what benefits can be expected in the processing or behaviour of the systems. Existing modelling techniques are outlined, and the position of a user modelling component in the dialogue between user and system is discussed.

1.1 Examples of user modelling applications

A general idea of possible uses of user modelling can be given by showing some simple examples of applications. The first example is from an imaginary tax advisory system

```
System: What is your assessable income?
User: I don't understand the question.
System: What is your income before taxes minus the deductions?
User: $34.500.
System: How much money do you have in cash, compared to your debts? . . .
```

The system noticed that its model of the user's expertise level was wrong, and simplified the formulation of its query. Note that the second question is also in relatively simple terms: it could instead have asked the user about his solvency, but decided that this was a too difficult term.

The next example is taken from Allen and Perrault (1980). A system is described which simulates a clerk at a railway station trying to answer questions from people meeting and boarding trains in a cooperative manner

```
User: The 3:15 to Windsor?
System: Gate 10.
```

In this example, the system is able to recognize the user's goal and the necessary information to resolve that goal. Here, the model of the user's goals is used for generating cooperative behaviour.

An example from the document retrieval field

```
User: Give me a list of the new books of this month.
System: Shall I use the same format for the list as last month?
User: Yes, please.
```

Here, the bookkeeping of user commands from the past is helpful as default information. The user does not have to repeat his preferences each time. The system recognized that even though the user has issued other commands during the past few weeks, this particular command was given a month ago and it could therefore predict that the user might want to reuse the same format.

A more detailed model of the user's interests and predictions of possible interests can improve a dialogue by having the system retrieve suggestions for the user (Siklóssy, 1979; Kok and Botman, 1988). These suggestions convey data that do not exactly meet the given constraints, but are possibly interesting too. This example interaction is between a database with information about secondhand car bargains and a user who wants to buy a car

```
User: Do you have a Mercedes, less than 10 years old, around $5,000?
System: Yes, a Mercedes from 1981, $4,800
        and a Mercedes from 1982, $5,500.
        But maybe you are also interested in
        a BMW from 1983, $4,500.
```

The system predicted an interest in BMWs based on the user model, and since the BMW bargain was newer and cheaper, suggested it to the user as a possible alternative.

The last example is taken from Kaplan (1982)

```
User: Which students got a grade F in CS200 in spring, '79?
System: CS200 was not given in spring, '79.
```

Instead of answering "None", which is formally a correct answer, the system modelled the presupposition of the user, recognized a wrong one (the presupposition that there *was* a course CS200 in spring, '79) and tried to correct it.

These examples show how the application of a user model changes what de Greef *et al.* (1988) have called the *modality* of the system, i.e., the distribution of tasks between the system and the user, and the control specification of the execution of the task distribution. The number of (sub)tasks performed by the system increases, while the number to be done by the user decreases. Also, the initiative of requiring the execution of a task moves in certain cases from user to system.

The examples can only give a superficial view of the range of possible uses of modelling methods. Therefore, in the next section a more detailed account is given of existing systems employing user models. The examples do show, however, what type of user models are dealt with in this paper. The user models of this review are explicit representations of characteristics of the people interacting with the system. The models are often used for more than one system function, i.e., more than one component adapts its behaviour on the basis of the user model contents. User modelling methods are techniques for constructing and maintaining the models.

A trivial example of a user model meeting these definitions is the ".profile" or ".login" file used by some operating system shells. Several things are adapted based on the contents of the model, such as layout of the screen, key bindings, command language (aliases), and default parameters of commands. The file is constructed by the user himself, and does not change in between editing. Although it is a user model according to these definitions, the emphasis in this paper will be on the more interesting user models that change over time due to interactions with some system.

Mainly because of the diversity of user modelling applications, a general methodology has not been defined yet. The approaches taken to representing users' characteristics are as diverse as knowledge representation approaches (Rivers, 1989). A survey of existing methods is therefore important to comprehend the current possibilities in user modelling. An outline of aspects of these methods will aid in analysing the requirements of new applications, and a synthesis of the field can indicate new directions for research.

In section 2 an overview of the field is given by means of short descriptions of systems employing user models, ranging over applications areas as diverse as dialogue systems, database retrieval and intelligent tutoring. With the exception of the GRUNDY system, all systems mentioned were built in the second half of the 1980s or are still under development. Therefore, this overview should be a good introduction into the current state of the art.

The different aspects of user modelling which can be recognized are outlined in section 3. A distinction is made on the basis of two points of view: analysing the user modelling part of systems from the technical level or from a more abstract, functional level. Examples are given of how to classify the systems of section 2 by means of these aspects. Also, a comparison is made with existing classification schemes from related work in the literature.

The synthesis of user modelling methods is then completed by categorizing the existing modelling techniques in section 4. First, the most often used technique, stereotyping, is discussed in some detail and then the others are described. The systems reviewed in section 2 serve as illustrations of the techniques.

Finally, in section 5 some current trends in user modelling are discussed. Important issues are providing modelling methods with formal foundations, designing rules for when to select which methods, and guaranteeing the privacy of the users of modelling systems.

2 Review of systems modelling users

In this section several systems employing user modelling techniques are described. The systems are classified into four categories: intelligent tutoring and help systems; dialogue systems; intelligent retrieval systems; and systems with the modelling of users themselves as their main task. Every subsection starts with an introduction of why user modelling is fruitful in that particular area. The system descriptions are kept very short, and the emphasis is on their user modelling parts. For more information about the systems, references are provided.

2.1 Intelligent tutoring and help systems

One of the main application areas of user modelling is intelligent tutoring systems. A tutoring system uses three models: a general model of the subject material or of some expert's knowledge, a model of the students' knowledge and beliefs concerning this subject model, and a model of the communication process containing information about teaching strategies (Clancey, 1986).

User modelling in tutoring systems consists of constructing and maintaining the student model. This model not only describes the student's factual knowledge, but also his abilities in applying it in problem solving situations. Modelling the student's inference processes can be done during problem solving or by reconstruction on the basis of the student's solutions, as is done in PROUST (section 2.1.1), for instance. In some tutoring systems the student model also includes a description of the learning process of the student (e.g., genetic graphs, [Goldstein, 1982]), although currently this has only been done for formal domains such as teaching mathematics and abstract computer programming (Clancey, 1986). In other areas, the learning models mainly consist of the classification of misconceptions and prerequisite ordering.

The construction of the student model is based on the model of the subject matter or the expert's knowledge model. According to the relation between this model and the student model, student models are usually classified into three types in the literature (e.g., Sleeman and Brown, 1982; Kass, 1987): overlay, differential, and perturbation models. The overlay model assumes that all differences between the student's behaviour and the behaviour of the expert model can be explained by a lack of skills on the part of the student. Differential modelling divides the class of things of which the system believes the student does not know into those which the student should know (if he is to behave like the expert), and those things which the student could not be expected to know. Both overlay and differential models assume that the knowledge of the student is only a subset of the knowledge of the expert, and can therefore not represent any belief of the user which is outside the domain knowledge of the expert. In perturbation models, the student model can differ, on the basis of small perturbations, from some of the knowledge in the expert model. A lack of knowledge is just one form of perturbation—for instance, common bugs or misconceptions another.

A category of systems with similar goals to intelligent tutoring systems are help systems. Help systems aid a user in his interaction with a computer system, for instance with editors, mail systems, operating systems, etc. Although a lot of systems have some component which can be consulted by the user if he runs into problems, the type of help systems considered here can take the initiative themselves. Help is offered if the user has made an error, if an inefficient sequence of commands is recognized, or periodically to update the user's knowledge of the capabilities of the system (Matthews and Biswas, 1985). Although advanced help systems exist which recognize user plans, possible misconceptions, learning styles, etc., as for instance in EUROHELP (Breuker *et al.*, 1987; Sandberg *et al.*, 1988), user modelling in a lot of help systems only consists of inferring which concepts and commands are known by the user, and remembering which ones are already explained by the system.

Below, the PROUST system is described as an example of an intelligent tutoring system. PROUST is one of the more interesting tutoring systems, since it does not just make simple inferences about the student's knowledge based on the interaction, but tries to reconstruct the intentions of the user leading to his behaviour. Next, a typical help system called SC-UM is described: a system which aids users of the SINIX operating system. SC-UM is chosen because of the emphasis on its user modelling part, and because it is representative of a large range of help systems.

2.1.1 PROUST

PROUST (Johnson and Soloway, 1984, 1985; Johnson, 1986) is a debugging and tutoring system of Pascal programs for novice programmers. The emphasis is currently on the automatic bug recognition part. PROUST recognizes non-syntactic bugs by constructing a model of the programmer's intentions. Experiments have shown bug detection results comparable to human debuggers.

PROUST differs from other diagnosis and debugging systems in its use of a model of the programmer's intentions. Knowledge of intentions makes it possible to distinguish buggy code from unusual but correct code. This means that PROUST can discover more complex bugs than most compilers. Programmers use plans to map their intended goals into code. PROUST tries to recreate the original mapping from goals to code, and uses these goals to recognize misconceptions. It then explains why the misconceptions are incorrect, or shows an input sequence for which the program clearly fails to work properly.

PROUST contains a description of the student exercises in its problem description language. Such a description consists of a set of goals which need to be fulfilled by means of the student's program. PROUST selects a goal from the problem description for analysis, and generates different realizations of this goal. This is done by retrieving a set of programming plans from a plan database, each of which might be used by novices to implement the goal. Such sets typically contain some expert plans, some novice plans, and some buggy novice plans. Programming plans describe stereotypic action sequences in programs, for instance, reading a sequence of numbers from a file. The plans may contain subgoals, which are recursively dealt with. PROUST tries to match each realization against the student program, and interprets the match failures in order to discover and identify bugs.

Without any expectations concerning what the code is supposed to do, there would be a combinatorial explosion of possible interpretations. PROUST's analysis is guided at all times by its model of what the student's intentions are likely to be, rather than by the behaviour that the buggy code happens to exhibit. Therefore, the top-down approach by means of the model of the user's plans is essential in this analysis.

An empirical evaluation of using PROUST for correction of about 300 student programs (implementing two programming exercises) gave promising results. More than 70% of the programs could be analysed, and in these 95% of the bugs were correctly identified, 20% of the programs received partial analysis, for which PROUST generated a limited form of bug explanation. Currently, it is being investigated whether PROUST can be used in a computer science curriculum.

2.1.2 SC-UM

SC-UM (Nessen, 1989) is the user modelling component of the SINIX Consultant SC, a help facility for the SINIX operating system with a natural language interface. SINIX is a UNIX derivate, and SC-UM has therefore the same function as the KNOME part of the UC system (Wilensky and Arens, 1984; Chin, 1989). The user models in SC contain information about the user's knowledge of the SINIX system and its concepts. The models are used to tailor the explanations offered by SC to each individual user's knowledge level, e.g., in the explanations only concepts are mentioned which the user actually knows, the same concept is not repeatedly explained, and concepts which are necessary to understand the effects of a command are outlined.

SC-UM uses an overlay model of the SC domain knowledge to represent a user's knowledge state. For every concept an *evidence value* is kept, reflecting the amount of evidence that the user knows of this concept. An evidence value of 1 means that the system thinks the user knows the concept, and a value of 0 means that it thinks he doesn't know it. These evidence values have two sources: if no information about the user's knowledge of a certain concept can be inferred, a default model is used by assigning the user to one out of four stereotypes. Secondly, the dialogue with the user is used to generate an individual model.

The stereotypes represent four types of users: novice, beginner, intermediate and expert. These stereotypes were constructed in accordance with a small experiment of 19 subjects filling out a questionnaire. The information in the stereotypes is only used as default: if anything can be derived from the interaction then that information is used. The assignment of the user to one of the four classes is not static. After each interaction a reevaluation of the applicability of each stereotype is made, and the best match is used. In this way, a wrong initial assignment is corrected quickly, and the process of becoming more experienced in the course of using SINIX and SC is reflected by a "higher" assignment.

Several methods are used to derive information from the dialogue with the user. Simple derivations are made from the use of certain concepts by the user, e.g., if a user asks for information about an item then it is assumed that he does not know that concept. If he uses a concept in the dialogue then it is assumed that he does know it, etc. More complex inferences are made by means of several heuristics. For instance, one set of rules is based on information about prerequisite knowledge for using certain commands. If the user knows, for instance, the "mail -q" command, then an inference is made that he knows the "mail" command in general. If he does not know a certain command, then he is assumed not to know commands for which this one is a prerequisite. If two inferences can be made about knowledge of the same concept, then the one with the highest evidence value is kept.

2.2 Dialogue systems

Dialogue systems, or question-answering systems, are the combination of a natural language interface and a problem solving system like intelligent databases or expert systems. For advanced natural language processing a model is necessary of the terminology the user comprehends, his goals and plans, and his beliefs.

The terminology information is needed for generating explanations of the user's expertise level. A simple first approach to this problem has been implemented in the UMFE system (Sleeman, 1985), where the user is classified on the basis of answers to classification questions posed by the system. A more interesting approach is to record the terms the user utters himself, and use that for a classification of his expertise level (as, for instance, in the XTRA system—see section 2.2.2). Sophisticated forms of explanation require a user model to include accurate information about the user's domain knowledge, the profoundness of explanations he likes best, and the type of explanation he prefers. Kass and Finin (1988b) give an overview of aspects of explanations.

The goals and plans of a user are needed to fully understand his utterances and to generate cooperative responses. For instance, the example in the introduction of this paper about the helpful railway station clerk system showed how its co-operative response was based on the knowledge that the user's aim was to meet or board a train. Dialogue systems also need goal and plan information for anaphora resolution.

A model of the user's beliefs is required for adapting to the user's view of the system's domain, and for correcting wrong presuppositions. The necessity of this is clear from an example dialogue taken from Kaplan (1982). An alternative version of this example was presented in the introduction

```
User: Which students got a grade of F in CS200 in spring 1979?
System: None.
User: Did anyone fail CS200 in spring 1979?
System: No.
User: How many people passed CS200 in spring 1979?
System: Zero.
User: Was CS200 given in spring 1979?
System: No.
```

The failure of communication (called "stonewalling") in this example is due to the user's wrong presupposition that the course CS200 was actually given in spring 1979. Recognizing this gives the system the possibility of correction by pointing out the wrong assumption. Work in the area of correcting user misconceptions is done, amongst others, by Webber (Webber and Mays, 1983; Webber, 1986), Hirschberg (1986). Motro (1986), and Wahlster (Wahlster *et al.*, 1983). Also, the ROMPER system described below deals with a particular type of mistaken presupposition.

In this section short descriptions of three dialogue systems are given: HAM-ANS, XTRA and ROMPER. These three are chosen because of their emphasis on user modelling and the difference in their modelling techniques. The subfield of dialogue systems is one of the main application areas of user modelling. For more information, Kobsa and Wahlster (1989) cite an extensive collection of papers on user models in dialogue systems.

2.2.1 HAM-ANS

The HAMburg Application-oriented Natural language System (Wahlster *et al.*, 1983; von Hahn and Wahlster, 1983; Wahlster, 1984; Hoeppner *et al.*, 1986; Wahlster and Kobsa, 1986, 1989; Morik, 1989) is a transportable natural language system. It has been interfaced to a hotel reservation system, an image-sequence analysis system of traffic scenes, and a relational database system with mass data on fishing expeditions. The interface is quite extensive, with methods for anaphora resolution, user modelling, the generation of extended responses, and focus shifting. For instance, a form of co-operative behaviour called *anticipation feedback* is exhibited when an exact answer to a question would very probably lead to follow-up questions, e.g.,

```
User: Does the room have any chairs?
HAM-ANS: Yes, two comfortable ones.
```

instead of the exact answer "Yes". Only the hotel manager application is described here, since it stresses the user modelling parts and is therefore the most interesting one for this paper.

In this application, HAM-ANS is an interface to a simple expert system simulating a hotel manager who tries to rent all his rooms. There are four categories of hotel room, and the goal is to establish the most suitable kind for a new guest. The system must therefore attempt to recognize the user's specific desires from his utterances. The user is assumed to have the overall goal of getting a hotel room which meets his requirements.

First, the system tries to find a room that meets all the explicitly stated requirements, for instance, about the period of stay, the number of beds, etc. If rooms of more than one category are applicable, a choice is made according to assumed interests of the user based on activated stereotypes. The entries in the stereotypes are simple attribute–value pairs. The information specified by the user leads to certain assumptions about his profession, the purpose of his trip, his solvency, etc., which trigger stereotypes. These stereotypes contain default information about the importance of the criteria describing the rooms. The important criteria are then used as conditions for the selectable rooms. This method is similar to GRUNDY's user modelling method (section 2.3.1).

The stereotypes contain, for instance, the information that managers need a telephone in their room, have a high budget and are therefore able to pay for an expensive room, and are interested in good entertainment facilities. Professors, on the other hand, do not have a high budget, and think a large table in their room more important than a telephone. People who plan to entertain guests think the comfort of the chairs important, and so on.

2.2.2 XTRA

The XTRA system (Allgayer *et al.*, 1989a; Allgayer *et al.*, 1989b) is a natural language front-end for expert systems. The design has been kept domain-independent, which means that XTRA can augment any expert system with its capabilities. The more the expert system fulfils certain characteristics (e.g., a declarative, or even better a frame-like representation), the more the knowledge of the expert system can be exploited by the natural language access system.

The main use of the system is to make the interaction between an expert system and its users easier, in particular for inexperienced users. It tries to achieve this goal by providing a mixture of natural language and pointing gestures as the interaction method, by being able to answer queries about the terminology used by the expert system, and by adapting its verbalization of results and explanations to the user's level of expertise. A first application of XTRA has been tested for an expert system on income tax, called LST-1.

XTRA includes a module which keeps records of the user's and system's beliefs, goals and plans.

The information in the user models is used for giving explanations in terms at the user's competence level, for recognizing and correcting wrong presuppositions, and will in future versions of XTRA be used for co-operative system actions.

The modelling formalism is based on Kobsa's work (Kobsa, 1985, 1985b; see also subsection 2.4.2) on nested models of beliefs and wants. Nesting means that a record is kept not only of the user's beliefs, but also of the system's beliefs of the user's beliefs, the user's beliefs of the system's beliefs of the user's beliefs, etc. A simple form of dialogue act analysis is carried out for this: if the user asks, for example, "Can I deduct my personal computer as a business expense?", then the following conclusions can be drawn (Allgayer *et al.*, 1989)

- The user is familiar with the concept "business expense".
- The user wants to know if his personal computer is deductible.
- The user believes that the system knows whether personal computers are deductible.

The user models are constructed based on three sources. First, a classification of the user's expertise level is made based on the terms he utters in the dialogue. The category determines which one out of three stereotypes is selected for default information. The second source is the simple dialogue analysis described above. The third is the contributions to the dialogue made by the system, e.g., if the system has recently explained a certain concept to the user then it will assume that he knows it.

2.2.3 ROMPER

As explained in the introduction of this subsection, much research on user modelling in dialogue systems concentrates on the recognition and correction of user misconceptions. The research of McCoy (1985, 1988, 1989) is an example of this: it deals with the generation of appropriate responses to recognized user misconceptions by a natural language interface. The emphasis in this research is not on the actual recognition of these misconceptions, but on determining the best response by using a highlighted model of the user's view of the domain. As an example of the type of responses dealt with, consider a natural language interface to a financial securities expert system. If the user asks the question

```
"What is the interest on this stock?"
```

a helpful response would be

```
"Stock does not have any interest. Did you mean bond?"
```

This type of response assumes that the system has knowledge about the similarity between bond and stock. McCoy defines the similarity of objects in terms of similarity of the attributes of the objects, but only the attributes which are salient in the current context are taken into account. This means that when talking about possible investments, stock and bond are considered to be similar, but when dealing with tax rules the two are considered to be different.

McCoy's ROMPER system (Responding to Object-related Misconceptions using PER-spective) simulates this phenomenon. It uses a domain model in the form of a hierarchy of objects, each with a set of attribute–value pairs describing the object. It takes as input a model of the user's view of the domain which is also a hierarchy of objects, but possibly different from the domain model. A misconception is defined to be some discrepancy between the system model and the user model, *viz.* the user may place an object in the object taxonomy where it does not belong (misclassification), or he may give an object an attribute–value it does not have (misattribution). ROMPER generates an appropriate response to point out the differences in the user's beliefs and the system model. Exactly how these differences are pointed out depends on the user's current *perspective* on the objects.

A perspective defines a particular way of looking at an object, i.e., different perspectives on the same object cause different properties of the user model to be highlighted. A perspective essentially acts as a filter on the properties which that object inherits from its superordinates. Thus properties of the object which are given a high salience rating by the perspective will be highlighted, while those which are given a low salience rating or do not appear in the perspective at all will be suppressed. Each perspective comprises a set of properties and their salience values. As an example, assume again a financial securities expert system; Figure 1 shows two perspectives for this domain.

The perspectives in Figure 1 depict that when dealing with certificates as savings instruments the maturity and denominations are very important. This means that if for two types of certificates the values on these attributes differ, they will be considered to be dissimilar. If, on the other hand, the perspective of the issuing company is taken, these attributes are not important and the same two types may be considered similar. This variable highlighting of the user model aspects results in different responses in different situations. For instance, if the currently active perspective is using certificates as savings instruments a dialogue could include

```
User: What is the penalty for early withdrawal on a Treasury Bill?
System: Treasury Bills do not have a penalty for early withdrawal.
Were you thinking of a Money Market Certificate?
```

while when using issuing company as the active perspective the same question would result in the response

```
System: Treasury Bills do not have a penalty for early withdrawal.
Were you thinking of a Treasury Bond?
```

The difference in response results from the fact that Treasury Bills and Money Market Certificates have similar maturity and denominations, but different issuers. On the other hand, Treasury Bonds and Treasury Bills differ in their maturity but have the same issuer.

The user model in ROMPER thus consists of the object hierarchy reflecting the user's beliefs, and the active perspective. ROMPER assumes that this user model is given as input, but McCoy (1989, p251) sketches some guidelines of how the active perspective could be determined by the system itself.

2.3 *Intelligent information retrieval*

Intelligent information retrieval has always been a fruitful application area of user modelling. Users of retrieval systems, and in particular the non-professional ones, find great difficulties in specifying exactly what data they are looking for and how important the expressed constraints are. Allowing a "vague" interpretation of commands, providing extra information, avoiding repetition

Savings Instruments
Maturity—high
Denominations—high
Safety—medium
Yield—medium

Issuing Company
Issuer—high
Safety—high
Purchase-place—medium
Yield—medium
Tax—medium

Figure 1 Sample perspectives (McCoy, 1989).

of commands, and showing results in an informative manner are all possibilities for alleviating these problems. These techniques, however, need detailed information about the user's characteristics.

For a number of years now, research has been conducted in trying to find methods to improve the quality of data retrieval based on the user's requests. An example is the SMART system (Salton, 1971), which uses a distance metric between queries and answers (in this system, a collection of document references) to retrieve extra references not exactly matching the specifications but probably useful. Another well-known technique is relevance feedback (Salton and McGill, 1983), in which the retrieved references are evaluated by the user and the query is adapted based on this evaluation.

These techniques mainly use domain knowledge about relations between concepts in a particular application, and short-term user information (e.g., his latest evaluation of the retrieved items). Current research concentrates on applying more long-term individual user information to improve the search requests. Empirical research done by Belkin *et al.* (see Daniels, 1986) showed that one of the main functions of a human search intermediary in document retrieval was making an assessment of the user's characteristics. In this research the user modelling component had five subfunctions: determine the status of the user (student, staff, etc.); determine his goals; his state of knowledge of the field; his familiarity with information retrieval systems; and establish his background. If the system can adopt some of these functions, then substantial improvements in retrieval can be expected.

In this section four systems are described in which a model of the personal preferences in data characteristics, subjects, search methods, and interaction style influence the processing of the system. These systems are GRUNDY, IR–NLI II, I^{3}R, and IMPACT. These systems are a good reflection of the user modelling techniques used in intelligent retrieval systems. Note that although two of these systems (IR–NLI II and I^{3}R) concern themselves with document retrieval, the term information is used here in the general sense, and is thus not restricted just to document references.

2.3.1 GRUNDY

One of the first retrieval systems exploiting explicit individual user models, and one of the most influential for the field of user modelling, was the GRUNDY system (Rich, 1979, 1979b, 1979c, 1983). GRUNDY recommends books according to its idea of what the user might like. A session with GRUNDY starts by entering several keywords describing the user's personality. Recognized keywords are, for instance, intelligent, tense, relaxed, open, humorous, adventurous, eccentric, catholic, liberal, etc. Based on these keywords a selection of the available stereotypes is made. These stereotypes describe typical interests in books for certain groups of users. The size of the selected set is usually small, e.g., less than 10 stereotypes. The information contained in them is combined to form a model of the current user.

The most salient feature in the user model is then used as the index into the database of book descriptions. The retrieved set of descriptions is ranked according to an overall match between the features of each book and the model of the user. The best match is suggested first, and the user is asked to give his opinion about this recommendation.

If the user likes the suggestion, then subsequent books of the ranked list can be recommended. If the user is not interested in the suggested book, then the model of the user is adapted. The feedback mechanism in GRUNDY consists of asking the user to specify what exactly he does not like about the recommendation, for instance is the amount of suspense not enough, is it too romantic, etc. Information in the model can then be changed based on his answers and the cycle of retrieving books from the database, ranking, and suggesting them starts again. Finally, when the user has seen enough suggestions and wants to stop the session, the most recent model of the user is stored for later use. This process is depicted in Figure 2.

A user model in GRUNDY consists of properties and their associated values, called *facets*. Each of the values of a property has a *rating* (a kind of certainty factor) and a list of reasons for that value.

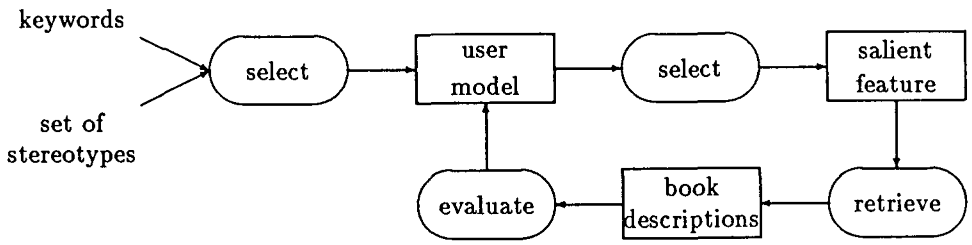

Figure 2 User modelling in GRUNDY.

Examples of facets are a user's gender, nationality, education, political position, piety, preference of plot speed, tolerance of violence, and preferred amount of romance. A description of a book has a similar structure, but the list of possible properties is slightly different. Each of the facets of a user model has an associated method for retrieving books based on the value of that facet, for instance the facet of preference of plot speed has a method that retrieves books with fast moving plots if the value is high.

Three sources of information are used to construct or augment the user models. The first is the answers to direct questions the system poses to the user, e.g., the description the user gives of himself at the start of the session. The second source is simple inferences, for instance male and female first-names are recognized. A second example occurs in feedback situations: if the user likes a certain book with a fast moving plot then the according facet of the model is adapted. The third and main information source is the invocation of stereotypes.

GRUNDY's stereotypes contain information about preferable features of books for certain classes of users. For instance, there is a stereotype for feminists, for sports persons, for catholics, etc. Table 1 shows an example stereotype. The stereotypes are organized in a generalization hierarchy. They contain triggers expressing in what situations their stereotype can be activated (see also section 4.1).

In combining stereotypes conflicts can arise at two points. The first is when two stereotypes are opposites, for instance the stereotype for males and the one for females both seem applicable. In this case, GRUNDY selected the most certain one and dismisses the other one completely. The second place where conflicts might arise is at the facet level. The resolution of these conflicts is facet dependent, i.e., each facet has its own rules for resolving two conflicting values. For instance, facets with numeric values can resolve conflicts by taking the mean value, others by taking a generalization of the two values, etc.

Rich (1979) created the stereotypes herself. This was not based on empirical research but done *ad hoc*. Stereotypes can change based on the interaction with users, but only very gradually. GRUNDY cannot create new stereotypes automatically.

According to Rich's thesis the GRUNDY project fulfilled three important goals. The first was to increase the awareness of the potential usefulness of user models of the type as described above. The second goal was to show that for at least one system, in at least one domain, individual user

Table 1 Stereotype of a sports person (Rich, 1979)

Facet	*Value*	*Rating*
Excitement-as-motive	high	0.6
Interest-in-sports	very-high	0.8
Thrill	medium	0.7
Tolerate-violence	medium	0.6
Tolerate-suffering	medium	0.6
Romance	very-low	0.5
Education	low	0.5

models were improving the performance. The third was to suggest a possible technique for user modelling by means of stereotypes.

2.3.2 IR-NLI II

The Information Retrieval–Natural Language Interface IR–NLI (Guida and Tasso, 1982; Brajnik *et al.*, 1986, 1987, 1988, 1988b) has in its second version been extended to include a user modelling component called UM-Tool. The user models are used to interpret the search requests of the user and to improve them if necessary. The contents of the models are divided into two parts: one part contains information about specific features, attitudes and traits of the user (for instance, education level, professional background, experience with information retrieval, personality traits, and usual search requirements), while the second part contains information about the level of familiarity with the subject domains (coverage and depth), the databases (MEDLIN, INSPEC, etc.), and information retrieval systems in general.

The models are constructed by selecting one or more stereotypes from the stereotype database in accordance with data collected from a preliminary interview with the user. This interview includes questions about the user's education and professional background, as well as his search interests. The stereotypes include both declarative and procedural knowledge about which users belong to this class, what the characteristics are of users of this class, and about how to acquire and validate certain entries during system operation if they are not present in the model yet. The stereotypes are organized into a generalization hierarchy, in which the top node is an empty stereotype for the generic user. After every interaction, the selection of stereotypes is reevaluated. If a stereotype appears not to be applicable any more, then its contents and the depending information (recognized by means of a truth maintenance system) is deleted from the user model.

Besides the stereotypes, other sources for the user models are the current dialogue and a database of former sessions. In the interaction a reformulation of the search request can induce a change in the model. Some statistical processing of the search session histories can extract the most common features and traits of each individual user.

Currently, the models are mainly used for default information in search sessions (for instance, default databases to use, to aim for high recall or high precision, etc.), but in a future version they should also aid in input interpretation and output generation (explanations, justifications, tailoring of the output to the user's knowledge level). Figure 3 shows the architecture of the system and the place of the user modelling component in it.

IR–NLI II is currently in a prototype stage, and is still without the aimed-for natural language interface.

2.3.3 I^3R

The Intelligent Intermediary for Information Retrieval system I^3R (Croft and Thompson, 1987; Croft *et al.*, 1988; Thompson, 1989; Thompson and Croft, 1989) is a document retrieval system combining several retrieval methods. One of the interesting aspects of the system is its use of a partially user-dependent domain knowledge network for extending a request by means of spreading activation. This process can be driven automatically, or by hand.

The network consists of relations between concepts and documents. Two concepts can be related by being synonyms, by one being a comprising form of the other, by being a component, a nearest neighbour, or by being "related" in a more general sense. Two documents can be linked because of overlap in their reference lists, because they are co-cited in a third document, because one is cited in the reference list of the other, or because they are nearest neighbours. A concept and a document are related when the concept appears in the description list of the document. The links are typed, but not weighted.

The user can extend or improve his original specifications by using the system's suggestions of related terms and documents. For this purpose, the system graphically shows the part of the

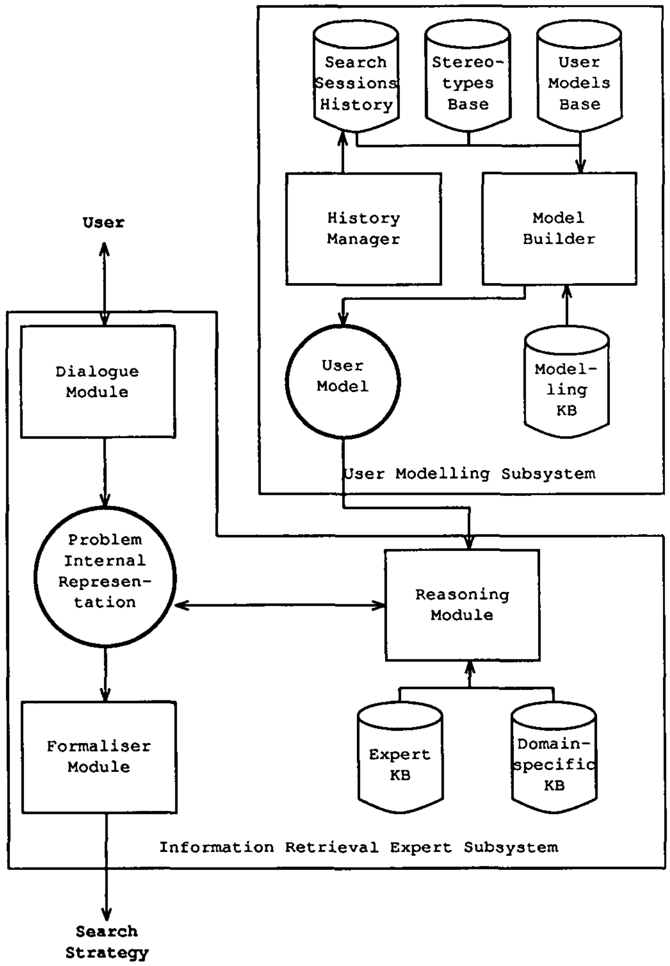

Figure 3 Architecture of IR–NLI II (Brajnik *et al.*, 1987).

network of concepts and documents which is currently being used for constructing a request. By clicking with the mouse on some of the nodes, interesting directions can be selected by the user. The browsing process is thus a semi-automatic version of a spreading activation search of the network. The specifications select a set of documents, which can be viewed by the user. He can then improve it by selecting some additional words from the listed documents. This iterative process stops as soon as the user is content with the number of relevant documents.

The user modelling function in I^3R consists of a simple classification of the user, and bookkeeping of user-specific information. At the start of a session three aspects of the user are determined by asking a few short questions. These aspects are the user's domain and system experience (values are "novice" and "expert"), and if the user wants to emphasize recall or precision. This classification is used for selecting the maximum number of searches and the number of wanted relevant references. The user-specific information consists of domain knowledge (i.e., links between concepts) added by the user, and a history of former sessions of this user. It is not possible for a user to add new concepts to the network.

The contents of the user model influence how the search is conducted (emphasis on recall or precision), what new domain knowledge can be entered by the user (a novice user has fewer facilities with which to enter information than an expert), which and how much information is displayed about the network, and how quickly the search controller will initiate a search while the user is browsing. The history of former sessions is used for constructing a version of the domain network adapted to the current user.

The knowledge base of I^3R contains the representation of 3200 document abstracts from the *Communications of the ACM.* Experiments were conducted using several different search strategies, showing that it is possible to obtain significant improvements in effectiveness by combining sources of evidence.

2.3.4 IMPACT

The IMPACT system (Botman *et al.*, 1987; Kok and Botman, 1988) is an implemented prototype of a user modelling interface to databases. The system's operation does not depend on the contents of the database, i.e., it is domain-independent. The user models are constructed based on interactions with the users. The aims of the IMPACT system are threefold

- It should provide an easy to learn and use interaction style, especially suited to non-professional users.
- Extra information, which is probably useful but not explicitly requested, should be shown.
- The retrieved items should be shown in a "user-friendly" way.

IMPACT realizes the first goal by providing an iterative browsing interaction style, in which the user can specify his interest and disinterest by simple selections with a mouse. The retrieval items, as well as aspects of them, can be evaluated and expressed to be interesting or not. The expressions are interpreted to only be indications, which means that they are a sort of "vague query" formulation. In contrast to, for instance, relevance feedback, not only the 'interestingness' of *items* can be indicated but also of attribute-values, attributes themselves, combinations of values, or even whole tables.

Extra information is shown based on the model of the user's interests and predictions of these interests. Items that almost match the expressed preferences, differing only on some insignificant attribute and possessing a feature that is predicted to be very interesting, will also be shown. The significance of an attribute is also based on the user model. Suggestions of possibly interesting attributes, attribute-values, and tables are displayed next to the retrieved items.

The retrieved information is sorted and the layout of the screen adapted to the user's interests as reflected in the user model. This means, for instance, that the most interesting tables are shown first, with the items within the tables sorted. Also, aspects of items that are not interesting according to the model will not be shown to avoid presenting too much information.

The user models in IMPACT thus contain interests of the user in (parts of) certain data, and predictions of those interests. The models are constructed based on the interaction with the current user and previous dialogues with this and other users. Patterns in the interactions are sought and stored as so-called *profiles,* small sets of interests with grades (how interesting is a particular datum) and certainty factors (how sure is the system that this interest is correct). The interest indications of the user at a particular moment define the context on which a selection of a set of these profiles is based. This set is combined to form a *focus,* the system's model of the current user interests. The modelling process is depicted in Figure 4.

Although there is some resemblance between the profiles and small stereotypes, IMPACT's modelling method is much more dynamic than the usual stereotype selection schemes. After every interaction, a complete new selection of the profiles is made, although the algorithm does assure that the focus changes "smoothly". The combining of such a large set of small dependencies between interests leads to a quick adaptation of the system to changing user interests, and the

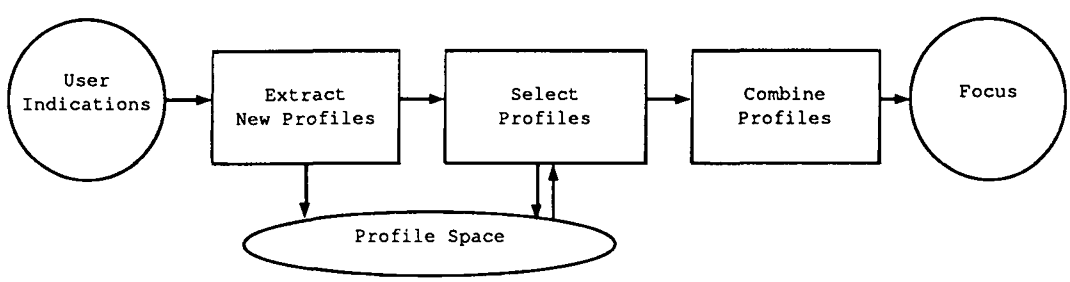

Figure 4 Model extraction and construction in IMPACT.

possibility of representing different dependencies in different situations. Note that IMPACT's modelling technique is different from multi-dimensional scaling, where only a very slowly changing global network is constructed between items instead of a network for the current situation between interests.

2.4 *Modelling systems*

The last application area of this paper covers systems for which the modelling of users is not just a part of the system's operation, but the main goal. For instance, in everyday life such modelling takes place in exams and personnel interviews. An example in the area of artificial intelligence is the ULLY system (Hayes and Rosner, 1976). ULLY simulates an inquisitive guest at a cocktail party who wants to find out as much as possible about the user. It starts out with some facts about the guests of the party, for instance their names, professions, who they like and dislike, etc., and tries to find out more by making 'small-talk' and inconspicuously asking questions. ULLY was just a proposal of a modelling system, an implementation was never realized.

A different type of system, but also with the modelling of users themselves as the aim, are user modelling shells. These systems contain mechanisms for creating and maintaining user models, and providing an application system with information about the user based on these models. Because these shells are designed to be usable in a wide range of applications, their methods have to be very general and are therefore often quite weak, i.e., an application designer has to provide much domain-dependent information himself. Kay (1990) describes a set of tools for constructing a user modelling component. In this subsection the user modelling shells GUMS and BGP-MS are described.

2.4.1 GUMS

The General User Modelling System GUMS (Finin and Drager, 1986; Finin, 1988, 1989) is a domain-independent add-on for application programs needing long-term models of individual users. The application system must itself take care of inferring information from the interaction. The main functions of GUMS are maintaining a database of facts about the user, inferring new facts by means of default rules, and maintaining the consistency of the inferences.

GUMS uses stereotypes for default values for the models. The stereotypes are organized into a hierarchy. The actual user models are represented as leaves of this tree, and have therefore only one associated stereotype. As soon as a fact is inferred which is inconsistent with the information in the user's stereotype, a more general one is sought that is not inconsistent. No combining of stereotypes takes place.

Inference rules need to be declared by the application designer. Examples of such rules from a tutoring system for computer science students are

default(knows(linkers) if knows(compilers))
default(understands(RAM) if understans(ROM)).

GUMS uses a four valued logic to reflect the difference between certain facts and defaults. These values are true, assume(true), assume(false), and false. Facts and rules can be stored in a model in four ways, as shown in Figure 5.

Currently, only a prototype of GUMS has been implemented in Prolog, called $GUMS_1$. According to the above cited articles, its truth maintenance part does not yet work. Also, the forward-chaining part that is needed for inferring new facts about the user has not been implemented.

2.4.2 BGP-MS

The Belief, Goal, and Plan Maintenance System BGP-MS (Kobsa, 1990b; Allgayer *et al.*, 1989: 174–176) provides a set of tools for constructing a user model in a particular application domain,

certain(P)	a definite fact: P is true.
certain(P if Q)	a definite rule: P is true if Q is definitely true and P is assumed to be true if Q is only assumed to be true.
default(P)	a default fact: P is assumed to be true unless it is known to be false.
default(P if Q)	a default rule: P is assumed to be true if Q is true or assumed to be true and there is no definite evidence to the contrary.

Figure 5 User model entry types in GUMS (Finin and Drager, 1986).

and for building a user modelling component for an application system. It comprises a KL-ONE-like representation language to express conceptual knowledge, a partitioning mechanism to define nested beliefs, some basic user-information inference mechanisms, and a stereotype management utility. A graphical interface is provided to aid a user model developer, and a functional interface for application systems to access and update the user model.

The represented knowledge is divided over several partitions, linked together in an inheritance hierarchy. By means of the partition mechanism it is possible to represent not only the beliefs of the user and of the system, but also shared knowledge and nested beliefs, e.g., the things the system believes that the user believes, the user believes the system believes that the user believes, etc. This can, for instance, be applied to represent the type of user misconceptions dealt with by the ROMPER system (section 2.2.3). The creation, deletion and revision of knowledge is checked by BGP-MS to keep the partitition hierarchy consistent.

Not only (nested) beliefs of individual users and the system can be represented in BGP-MS, but also beliefs of groups of users, i.e., stereotypical beliefs. The stereotypes can be linked into a hierarchy or a directed acyclic graph. A graphical interface is provided to construct a stereotype hierarchy interactively. The stereotype selection and retraction rules should be provided by the user model designer, but several predefined rules are at the user's disposal, e.g., select the stereotype if a certain concept is known by the user, retract it when a certain percentage of the domain concepts is known by the user, and others.

In BGP-MS some basic domain-independent acquisition rules have already been built in. An example is the rule: "If the user knows that concept A possesses attribute X, and B specialises A, then he knows that B also possesses attribute X". Domain-dependent rules can be specified by the user model builder in the form of condition–action pairs.

BGP-MS has been used in XTRA (section 2.2.2) to build the user modelling component of that system.

3 Dimensions of user modelling

The wide range of application areas dealt with in the previous sections makes a comparison of existing methods difficult. The use of a set of "dimensions" for classifying them facilitates this task. In the literature, several classification schemes of user modelling methods are defined, for instance by Carbonell (1983), Rich (1983), Sleeman (1985), Sridharan (1985), and Kass and Finin (1988). Unfortunately, all these schemes appear to intermix two levels of analysis: some dimensions mention aspects on the technical level of user modelling, while others are defined on a more abstract, and functional level. This section contains a new proposal for classifying user modelling methods on the basis of four aspects (why, who, what and how), with for every aspect a functional level dimension and a technical level dimension. These eight dimensions can serve as an analysis scheme, as shown in section 3.5. In section 3.6 a comparison is made with the existing schemes mentioned above.

The proposed dimensions are

1 Why are users modelled?
   (a) What is the overall aim of modelling users.
   (b) Which parts of the system need user information.
2 Who is modelled?
   (a) What is the role of the user being modelled in relation to the system.
   (b) How individual are the user models, i.e., which and how many users are represented by a model.
3 What is modelled?
   (a) What aspects of the user are meant to be represented by the user model contents.
   (b) What is contained in the user models, and what are the interpretation methods of the contents.
4 How are users modelled?
   (a) What are the methodologies of modelling and what are its sources.
   (b) Which user modelling techniques are used.

Note that for each of the four aspects the first dimension is on the functional level and the second one on the technical level. Each of the aspects is now described in more detail.

3.1 Why are users modelled?

How is adapting to individual users affecting the system's processing? Reasons to model users include improving the user friendliness (for instance, by giving suggestions of possibly interesting data, or by aiding the user in his use of the system), improving the economy of interaction (for instance, by tailoring the dialogue to the user's expertise level, or by focussing the dialogue to his current interests), and affecting the processing of the system itself (for instance, to improve recall and precision, to interpret the utterances of the user, or to construct a teaching sequence).

While the first dimension of the why-aspect covers the general aim, the second one is concerned with the technical part, *viz.* which system modules need data about the users' characteristics. As stated before, a characteristic of the type of explicit user models described in this paper is that they are often applicable in more than one part of the system. For instance, in natural language processing the module interpreting the user's utterances needs to have some information about his beliefs, and the language generation module needs to know the terms the user understands. In information retrieval the module computing the layout of the retrieved information needs a model of the user's preferences, and his attention focus. Tutoring systems attempt to obtain a model of the missing knowledge of the user so that the tutoring component can decide which new concepts to explain, and to determine the level on which these explanations should be generated.

3.2 Who is modelled?

From the functional point of view, the question of who is modelled comprises the role of the modelled persons in relation to the system. Although in this paper the term "user model" is used throughout, not only models of the persons actually employing the system are meant to be covered, but every explicit model of somebody related to the system's processing. For instance, in the HAM-ANS system (section 2.2.1) the hotel room reservations are not necessarily made by the guests themselves, but can be made by their secretary. It may therefore be more appropriate to use the term "agent model" instead of user model (Kobsa, 1985). The question of who is being modelled should take this difference into account, as well as aspects such as what category of users are modelled (for instance, first-time users or experienced ones, etc.), and what their aim is in using the system.

From a technical point of view, the question of who is modelled covers the issue of how individual the constructed models are. Is there a model for each user, or are certain groups of users modelled (for instance, by means of stereotypes) and the user-specific information consists only of a classification in this set of models, or is a generic model constructed for the average user? The variation along this dimension can be seen as going from a concentration of information around a particular user, clustering together everything known about that person, to a concentration around a particular topic, clustering together what is known about all users with respect to that issue.

What is modelled?

The functional dimension of this aspect concerns itself with what user characteristics are modelled. In most applications one or more of the following aspects are modelled: the knowledge or the level of expertise of a user, his beliefs (including wrong ones: for instance, mistaken presuppositions), his goals, his plans or intentions, his interests or preferences, his capabilities, or one of these but belonging to the system or another person (i.e., nested models). An important issue is whether the models are meant to be descriptive or predictive; for instance, are preferences modelled or predictions of preferences? Another issue is the psychological validity of the modelled information. Most descriptions of user modelling systems explicitly state that the inferences are not necessarily psychologically valid. Using labels like "interest" or "belief" may lead to some confusion (McDermott, 1981), but it does indicate the purpose of the user models.

The next dimension concerns the technical aspects of what is modelled. Issues applicable here are: what kind of structure does the model contain (for instance, is a user model just a point in a stereotype hierarchy, is it an overlay model of the domain knowledge, etc?), what inference engine or interpretation method is used for retrieving information from the user model*, is the information purely qualitative or is there a kind of grading involved, does the information include a notion of its certainty, and is the information in the user model necessarily consistent, or are inconsistencies allowed?

3.4 How are users modelled?

The last aspect concerns the user modelling methods. Again, two viewpoints can be distinguished. The functional one conveys the methodology and sources of the modelling, while the technical point of view is about the actual techniques employed, the tenability of the information, and the determination of the contexts in which a particular datum is applicable.

It is possible to talk about modelling methods without referring to the specific algorithms used or a specific implementation (Newell, 1982; Clancey, 1985; Breuker and Wielinga, 1987). In this vein one can, for instance, state that the modelling method uses abstraction, classification, or a model provided by the system designer. A determination of the information source used by the methods also belongs at this level. The possible sources include preprogrammed information, information supplied directly by the user, information extracted indirectly from the interaction, and information inferred from past interactions and sessions.

Examples of user modelling techniques defined at the technical level are stereotype selection, correlation computation, heuristic rules with backward-chaining, similarity-based learning, etc. An important issue concerning the inferred information is its tenability and the determination of the situations in which it is applicable. Modelling techniques should account for changes in the user's characteristics, such as his preferences, so it is important to recognize when extracted data become obsolete. In section 4 a more detailed description of existing modelling techniques is given.

3.5 Analysis of example systems

In this subsection, the eight questions resulting from the proposed dimensions are answered for the GRUNDY and IMPACT systems. GRUNDY is chosen because it is the "classic" user modelling

*Note that the inference engine is not part of the user modelling method. Methods are covered by the "how" aspect.

system of this paper, and IMPACT is chosen because the author of this paper is one of IMPACT's designers so its details are available. After these two examples, an analysis of the remaining systems discussed in section 2 is summarized.

The aim of modelling users in the GRUNDY system (1a) is to be able to give useful book references based on personal preferences. The modules that need this information (1b) are the retrieval module and the module explaining why it makes these specific suggestions. The readers looking for an interesting novel are modelled (2a), and the models are initially based on stereotypical users, but adapted individually (2b).

Aspects of the user that are meant to be conveyed by GRUNDY's user models are his preference of book characteristics (3a). The models contain a grading and certainty factor for each possible characteristic (3b). They are constructed by classifying users based on personality traits, given by the users themselves (4a). The technique used for this (4b) is posing a query to the user asking to describe himself, selecting the most specific stereotypes based on the results, and combining them.

Note how even this very short analysis by means of the eight dimensions gives a well-balanced account of the user modelling part of GRUNDY (compare it, for instance, with section 2.3.1). IMPACT can be analysed just as quickly, first from the functional point of view and then from the technical (refer to section 2.3.4 for a comparison).

The overall aim of modelling users in IMPACT is to make the retrieval of interesting data from a database easier. The emphasis is on helping non-professional users, needing the information themselves. The interests of the user in particular data and aspects of that data are modelled, based on patterns in the dialogues with this and other users.

The modules that need user information are the command interpreter, the data retriever, the suggestions generator, and the layout determiner. Models are constructed for each individual user. They contain an interest grade and certainty for every datum that is of interest in the current situation. Interests can be computed by a simple look-up in the model, or by means of default rules. IMPACT's user modelling technique extracts and stores patterns in the interests indications of the users, retrieves them when the same context is entered, weights them based on several heuristics, and combines them to form the overall model.

In Table 2 the functional point of view is summarized for the other eight systems described in section 2.

3.6 Comparison with related work

As stated at the start of section 3, several classification schemes of user modelling already exist in the literature. In this paper a new one has been defined because the existing ones suffer from two problems: they mix the two abstraction levels, which have been kept separated here, or their dimensions overlap, i.e., the classification scheme is not orthogonal. In this subsection some examples of these problems are given.

The most often cited set of dimensions is that defined by Rich (1983)

- One model of a single, canonical user versus a collection of models of individual users.
- Models specified explicitly either by the system designer or by the users themselves versus models inferred by the system on the basis of users' behaviour.
- Models of fairly long-term user characteristics such as areas of interest or expertise versus models of relatively short-term user characteristics such as the problem the user is currently trying to solve.

Sleeman added a fourth dimension (Sleeman, 1985)

- The nature and form of the information contained in the user model and the inference engine needed to interpret that information.

Although this classification scheme was useful to characterize the GRUNDY system, it does mix up the abstraction levels indicated in this section. Compare, for instance, the first and fourth dimensions which deal with technical issues with the more abstract second and third dimensions, e.g., the third dimension mentions high-level characteristics of the modelled information, while the fourth is more concerned with the actual representation of that information.

The problem of non-orthogonality of dimensions appears, for instance, in the list of Kass and Finin (1988), where the "modifiability" and "temporal extent" dimensions overlap. Brajnik *et al.* (1988b) themselves admit that their list of four dimensions is not orthogonal. In their list the "given versus inferred" dimension and the "static versus dynamic" dimension partially overlap.

4 User modelling techniques

The previous section has given a set of aspects on which to compare and analyse the user modelling part of intelligent systems. The specific modelling techniques were mainly covered by the "how"

Table 2 Functional dimensions of the modelling systems

System	*Dimension*	
PROUST	Why	Analysing user knowledge and beliefs
	Who	Student using the system as teacher
	What	User's intentions
	How	Reconstructing intentions by matching stereotype plans
SC-UM	Why	Providing help in interacting
	Who	User of general system needing support
	What	User's knowledge of system concepts
	How	Classification and inferences from the dialogue
HAMS-ANS	Why	Providing help in retrieving data
	Who	User requiring information from the system
	What	User's preferences
	How	Classification
XTRA	Why	Providing support in interacting
	Who	User of underlying system needing support
	What	User's beliefs and goals
	How	Classification and inferences from the dialogue
ROMPER	Why	Recognizing and correcting user misconceptions
	Who	User of underlying system needing support
	What	User's beliefs and perspective
	How	Model is given beforehand
IR–NLI	Why	Making interaction with information systems easier
	Who	User of information system needing advice
	What	User's search preferences
	How	Classification and bookkeeping of dialogue
I^3R	Why	Making interaction with information systems easier
	Who	User of information system needing advice
	What	User's view of data relations and search requirements
	How	Classification and bookkeeping of dialogue
GUMS and BGP-MS	Why	Providing user information to application system
	Who	User of application system
	What	Long term user information
	How	Classification and application-dependent inferences

aspect. This section describes the techniques used in the modelling systems in more detail. First, a subsection is dedicated to the most often used technique: stereotyping.

4.1 Stereotyping schemes

Even though the word 'stereotype' has a negative connotation, stereotypes appear to play an important role in the generation of co-operative behaviour (McCauley *et al.*, 1980). Mainly since Rich's work on GRUNDY, stereotyping has been one of the most widely applied user modelling techniques. Section 2 already indicated this: most of the systems mentioned there use some form of stereotypes, e.g., SC-UM, HAM-ANS, XTRA, IR-NLI II. Murray (1988) even concludes that stereotype modelling is the most useful technique investigated so far.

A stereotype contains information that is common to a particular category of users, which is specified in the stereotype's *trigger*. Stereotypes are often organized into a generalization hierarchy, whereby specific categories inherit information from the more general ones by default. Because of the popularity of stereotyping, systems have been designed which provide tools to create such a hierarchy and to store and manipulate stereotypical information. The GUMS and BGP-MS systems described in section 2.4 are an example of this. Tang *et al.* (1989) have designed tools which aid a system designer in implementing a stereotyping scheme for the dialogue part of tutoring systems.

In stereotyping techniques three issues are of particular interest: how are the appropriate stereotypes selected, where do the stereotypes come from (e.g., on which basis were they created), and are the stereotypes adapted in the course of their use.

In some schemes a selection of the appropriate stereotypes is made only once, at the beginning of the session, while in others the selection is reevaluated during the dialogues with the user and, if necessary, adapted. The necessity of frequent reevaluations depends on the user aspects that are tested in the triggers. Most stereotyping schemes base their selection on user characteristics which are not likely to change during a dialogue, e.g., gender, profession, income, religion, etc. Others, however, use patterns in the behaviour of the user as selection criterion. For instance, they exploit stereotype information of users who ask about certain concepts, who are interested in certain data, who appear to have a certain experience level because of their mastering of the system commands, and so on. In these cases it is necessary to regularly reevaluate the stereotype selection during the dialogue.

The second interesting issue concerns the source of the stereotype information. For instance, the stereotypes used in the GRUNDY system were created *ad hoc* by Rich herself. Although other systems also use such an *ad hoc* approach, a more empirical approach is of course preferable. As an example, SC-UM used stereotypes created on the basis of the results of a small empirical study of the target user group. An analysis of the user group for which the system is meant is often part of the design phase anyway, even when no user modelling component is added (Kidd, 1987), which means that it does not always have to be extra work to take such an approach. A third technique is to let the system extract stereotypical information from the dialogues by using patterns in the interactions with the users and correlating these with other user information.

Most schemes use a predefined set of stereotypes, which are not changed in the course of time. GRUNDY, however, adapts its stereotypes on the basis of feedback from the interactions, both positive feedback that the predictions based on a particular stereotype are correct and should be used again in the future in similar circumstances, and evidence that the predictions were incorrect and should not be made any more. An adaptation technique must deal with the problem of how to assign blame and credit among all the factors that made a given prediction and how much weight should be given to the initial values in the stereotype and to each additional piece of evidence (Rich, 1989). As an example, the GRUNDY system gradually adapts its stereotypes on the basis of evaluations made by the user. If a certain recommendation of a book turns out to be inappropriate, then the rating of the facet accountable for this prediction is lowered. Also, the ratings of the trigger aspects are lowered, since the reason for the mistake might be an inappropriate selection of the stereotype in this situation.

4.2 Categorization of techniques

In this section the techniques used in current user modelling systems are described from relatively simple to advanced. All the systems of the review section are classified to illustrate an application of the techniques.

4.2.1 Using a model supplied before or a default model

The first method is the application of a model supplied at the start of the dialogue by the system designer, or the use of a generic default model based on the characteristics of the average user as defined by the designer. The system designer can base these definitions on results of empirical research or create them *ad hoc.* Another type of generic model keeps a model of the average user's behaviour and uses that for adapting the system. The ROMPER system is an example of a system which has the user model supplied beforehand.

4.2.2 Model supplied by the user

As explained in the introduction of this paper, a trivial method of user modelling is letting the user define a model himself. This can, for instance, be done by using a "login" or ".profile" file, but parameter setting is also an example of this kind. These models do not change often, because keeping a dynamic model of this nature is usually not worth the user's effort in constructing it.

4.2.3 Model editing

A third modelling method is the direct manipulation of the model by the user. The initiative for this can be taken by the user (e.g. starting an editing session of the model to correct a mistaken inference of the system), or by the system (e.g., posing questions like "Do you want to emphasize recall or precision?", or "Do you know what *Diplococcus Pneumoniae* is (y/n)?"). As an example, the GRUNDY system allows in its evaluation state an almost direct editing of its user model. Also, the tools described by Kay (1990) permit users to directly edit their user models, which are represented as text files.

If the user is given the opportunity to edit his model a problem might be that it is not always easy or even possible to adequately explain what the contents of the model mean. Also, making clear why a model is maintained in the first place, and what the influence is of the model contents on the behaviour of the system can be difficult (see also Kobsa, 1985, 1990).

4.2.4 Stereotype selection

The fourth technique of the list is stereotype selection based on user characteristics such as profession, or on the user's behaviour. This type of technique has been dealt with in the previous subsection. From the systems described in the review, HAM-ANS, GRUNDY, I^3R, IMPACT, SC-UM, IR-NLI II, and GUMS all use some form of stereotyping.

4.2.5 Bookkeeping of the interactions to use as defaults

The simplest technique using the dialogue with the user is bookkeeping. The user's statements in a dialogue can be used as defaults for future sessions. For instance, it can be assumed that a user wants to access the same database as before, as long as this assumption is not explicitly refuted.

When a contradiction is stated to a particular default, then the new value becomes the default. This technique is used by IR-NLI II, IMPACT and I^3R.

4.2.6 Simple inferences from the interactions

A more interesting technique using the dialogue is making inferences from an interaction to user model contents. The type of influences can be deductive, but heuristic associations also fall into this category. For instance, from the utterance "Which courses were taught during spring 1979?" the system can infer that the user wants to get a list of courses taught in that period, that the user believes there were courses in spring 1979, and that he assumes that the system has information about courses in certain time periods and is able to answer queries about them. A correct use of a certain term in an utterance can be used as an indication that the user knows the meaning of that term, or indicates a certain expertise level, while an incorrect use of a command is an indication that the beliefs of the user about the command need to be repaired.

If a user knows a command, then it can be assumed that he knows the prerequisites for that command, and if he knows a specific variant of a command, then he probably knows the general form. As an example, if the command "ls -al" is used correctly, then the user can be assumed to know the more general command "ls" as well. Dialogue contributions made by the system can also be used to update the user model. If the system has, for instance, just explained a certain concept, then the model will have to reflect that the user is now familiar with it. This type of technique is employed by SC-UM and XTRA, and GUMS allows its definition.

4.2.7 Inferences from patterns in the dialogue

More complex inferences can be based on patterns in the dialogue, as is done in the IMPACT system. Other examples include some plan recognition techniques, which apply forms of pattern matching to determine the plans of the user. Also, most help and tutoring systems use some form of inference based on patterns in the interaction. For instance, PROUST determines intentions on the basis of patterns in the student's program.

4.2.8 Plan recognition

In the field of user modelling two major plan recognition techniques have been proposed: plan operator chaining, and plan operator decomposition. In the former technique, a chain or directed acyclic graph of operators is sought, for which the preconditions of each operator are met by the effects of one of the directly preceding operators, and all observables are explained in the effects of the operator chain. An example application is described by Carberry (1989). The technique is influenced by the STRIPS approach.

In the plan operator decomposition technique, plan operators may be recursively decomposed into sequences of lower-level operators. The preconditions of an operator must meet the effects of its predecessors, and the observables must be contained in the effects of the lowest level. Maybury (1990) uses this approach to plan text generation for tailored responses.

4.2.9 Abstraction and generalization over interactions

The last type of technique concerns statistical and inductive inferences over interactions and sessions. These techniques try to recognize patterns in the behaviour of the users, which can be used to predict future behaviour. Examples are techniques extracting links between interests in data based on the correlation in commands of several dialogues (as in IMPACT), and abstracting over dialogues with different users to induce the areas in a domain where most mistakes are made.

Note that most modelling systems use a combination of the techniques mentioned in this section, for instance, a default model for initializing the user model and deductive inferences from the interaction for maintaining it. This phenomenon is reflected by the appearance of several of the reviewed systems in more than one category.

5 Conclusion and discussion

This paper has presented a review and synthesis of the user modelling part of existing systems in the AI field. The review should provide some handles for a knowledge engineer on how to design a user modelling component for his application. Analysis of a new application can be done with the dimensions of section 3. The range of existing modelling techniques has been outlined in section 4.

In practice it is not feasible to supplement each and every AI system with a fully fledged user modelling component. For most applications a simple user classification by means of stereotypes is enough, and is more practical than constructing original models for every individual (Swartout, 1985). Creating this set of preprogrammed stereotypes is part of the knowledge engineering effort when designing the system, and should be based on an empirical analysis of the target user group. However, for areas such as intelligent tutoring systems and intelligent data retrieval, more advanced modelling techniques than stereotyping are a necessity.

A trend in the user modelling field that can be recognized is that, as in all fields of artificial intelligence, emphasis is more and more placed on formal approaches. In user modelling formal work is mainly found in knowledge and belief systems (e.g., Hintikka, 1962; Moore, 1984). Formalizing methods will aid in comparing and combining them. Also, more formal definitions of the semantics of the user models are needed, since in current applications it is not always clear what the models' contents mean exactly.

An issue that is important to consider when designing a practical application is the impact of user modelling, in particular the implications it has in terms of user privacy. In practical applications the users will have to be informed about the modelling function of the system and its purpose. The user must be given the opportunity to stop the modelling part, even though this would result in a less co-operative system. It might also be interpreted to explain to the user what have been inferred from his behaviour and why. Of course, in that case the user must be given the opportunity to change inferences which are in his opinion wrong. For application areas such as tutoring systems some form of modelling will be expected by the user, and will therefore probably be less threatening than in applications where it is not always expected such as data retrieval systems. Other potential social impacts are discussed by Kobsa (1990).

Acknowledgements

This work is supported by the Netherlands Organization for Scientific Research (NWO) under grant 619-319-008. Thanks are due to Professor Bob Wielinga and Radboud Winkels for extensive comments on earlier drafts, and for discussing this paper. The author also wishes to thank *The Knowledge Engineering Review* referees for their very helpful comments and suggestions.

References

Allen, JF and Perrault, CR, 1980. "Analyzing intention in utterances" *Artificial Intelligence* **15**(3) 143–178.

Allgayer, J, Harbusch, K, Kobsa, A, Reddig, C, Reithinger, N and Schmauks, D, 1989. "XTRA: a natural-language access system to expert systems" *International Journal of Man–Machine Studies* **31** 161–195.

Allgayer, J, Jansen-Winkeln, R, Reddig, C and Reithinger, N, 1989. "Bidirectional use of knowledge in multi-modal NL access system XTRA" *Proc. 11th International Joint Conferences On Artificial Intelligence,* pp 1492–1497, Detroit.

Botman, AM, Kok, AJ and Siklóssy, L, 1987. "Using user models and automatic expertise acquisition in man–machine interaction" *Proc. 7th International Workshop on Expert Systems and their Applications,* pp 389–398, Avignon.

Brajnik, G, Guida, G and Tasso, C, 1986. "An expert interface for effective man–machine interaction" In: Bolc, L and Jarke, M, eds., *Cooperative Interfaces to Information Systems,* pp 259–308, Springer-Verlag.

Brajnik, G, Guida, G and Tasso, C, 1987. "User modelling in intelligent information retrieval" *Information Processing and Management* **23**(4) 305–320.

Brajnik, G, Guida, G and Tasso, C, 1988. "IR-NLI II: applying man–machine interaction and artificial intelligence concepts to information retrieval" *Proc. 11th International Conference on Research and Development in Information Retrieval,* pp 387–400, Grenoble.

Brajnik, G, Guida, G and Tasso, C, 1988b. *User Modeling in Expert Man–Machine Interfaces: A Case Study in Intelligent Information Retrieval* Technical Report UDMI/06/88/RT, University of Udine, Italy.

Breuker, J and Wielinga, B, 1987. "Use of models in the interpretation of verbal data" In: Kidd, AL, ed., *Knowledge Acquisition for Expert Systems: A Practical Handbook,* pp 17–44, Plenum Press.

Breuker, J, Winkels, R and Sandberg, J, 1987. "A shell for intelligent help systems" *Proc. 10th International Joint Conference on Artificial Intelligence,* pp 167–173, Milan.

Carberry, S, 1989. "Plan recognition and its use in understanding dialog" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 133–162, Springer-Verlag.

Carbonell, JG, 1983. *The Role of User Modelling in Natural Language Interface Design* Technical Report, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, USA.

Chin, DN, 1989. "KNOME: modeling what the user knows in UC" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 74–107, Springer-Verlag.

Clancey, WJ, 1985. "Heuristic classification" *Artificial Intelligence* **27** 289–350.

Clancey, WJ, 1986. "Qualitative student models" In: Traub, JF, ed., *Annual Review of Computer Science,* pp 381–450, Annual Review Inc., Palo Alto, CA.

Croft, WB, Lucia, TJ and Cohen, PR, 1988. "Retrieving documents by plausible inference: a preliminary study" *Proc. 11th International Conference on Research and Development in Information Retrieval,* pp 481–494, Grenoble.

Croft, WB and Thompson, RH, 1987. "I^3R: a new approach to the design of document retrieval systems" *Journal of the American Society for Information Science* **38**(6) 389–404.

Daniels, PJ, 1986. "Cognitive models in information retrieval—an evaluative review" *Journal of Documentation* **42**(4) 272–304.

de Greef, P, Breuker, J and de Jong, T, 1988. *Modality: An Analysis of Functions, User Control and Communication in Knowledge-based Systems* ESPRIT Project P1098, Deliverable D6 (task A4), University of Amsterdam, Netherlands.

Finin, T, 1988. "Default reasoning and stereotypes in user modelling" *International Journal of Expert Systems* **1**(2) 131–158.

Finin, TW, 1989. "GUMS—a general user modeling shell" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 411–430, Springer-Verlag.

Finin, T and Drager, D, 1986. "$GUMS_1$: a general user modeling system" *Proc. 1986 Conference of the Canadian Society for Computational Studies of Intelligence* pp 24–30, Montreal.

Goldstein, IP, 1982. "The genetic graph: a representation for the evolution of procedural knowledge" In: Sleeman, D and Brown, JS, eds., *Intelligent Tutoring Systems,* pp 51–77, Academic Press.

Guida, G and Tasso, C, 1982. "NLI: a robust interface for natural language person–machine communication" *International Journal of Man–Machine Studies* **17** 417–433.

Hayes, PJ and Rosner, MA, 1976. "ULLY: a program for handling conversations" *Proc. 1976 AISB Summer Conference,* pp 137–147, Edinburgh.

Hintikka, J, 1962. *Knowledge and Belief* Cornell University Press.

Hirschberg, J, 1986. "Anticipating false implicatures: cooperative responses in question–answering systems" In: Kerschberg, L, ed., *Expert Database Systems* Benjamin/Cummings Publishing Co, Inc.

Hoeppner, W, Morik, K and Marburger, H, 1986. "Talking it over: the natural language dialog system HAM-ANS" In: Bolc, L and Jarke, M, eds., *Cooperative Interfaces to Information Systems,* pp 189–258, Springer-Verlag.

Johnson, WL, 1986. *Intention-Based Diagnosis of Novice Programming Errors. Research Notes in Artificial Intelligence* Pitman.

Johnson, WL and Soloway, E, 1984. "Intention-based diagnosis of programming errors" *Proc. National Conference on Artificial Intelligence,* pp 162–168, Austin, TX.

Johnson, WL and Soloway, E, 1985. "PROUST—an automatic debugger for Pascal programs" *BYTE* **10**(4) 179–190.

Kaplan, SJ, 1982. "Cooperative responses from a portable natural language query system" *Artificial Intelligence* **19** 165–187.

Kass, R, 1987. *The Role of User Modelling in Intelligent Tutoring Systems* Technical Report MS-CIS-86-58, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA.

Kass, R and Finin, T, 1988. "Modeling the user in natural language systems" *Computational Linguistics* **14**(3) 5–22.

Kass, R and Finin, T, 1988b. "The need for user models in generating expert system explanations" *International Journal of Expert Systems* **1**(4) 345–375.

Kay, J, 1990. "UM: a toolkit for user modelling" *Second International Workshop on User Modeling* Hawaii.

Kidd, AL, 1987. "Knowledge acquisition—an introductory framework" In: Kidd, AL, ed., *Knowledge Acquisition for Expert Systems: A Practical Handbook,* pp 1–16, Plenum Press.

Kobsa, A, 1985. *Benutzermodellierung in Dialogsystemen* Springer-Verlag.

Kobsa, A, 1985b. "VIE-DPM: a user model in a natural language dialogue system" In: Laubsch, J, ed., *GWAI-84, 8th German workshop on artificial intelligence* Springer-Verlag.

Kobsa, A, 1988. "A selected bibliography of the field of user modeling in artificial intelligence dialog systems" *SIGART Newsletter* **106** 31–37.

Kobsa, A, 1990. "User modeling in dialog systems: potentials and hazards" *AI and Society* **4** 219–231.

Kobsa, A, 1990b. "Modelling the user's conceptual knowledge in BGP-MS, a user modeling shell system" *Computational Intelligence* **6**(4).

Kobsa, A and Wahlster, W, eds., 1989. *User Models in Dialog Systems. Symbolic Computation* Springer-Verlag.

Kok, AJ and Botman, AM, 1988. "Retrieval based on user behaviour" *Proc. 11th International Conference on Research and Development in Information Retrieval,* pp 343–358, Grenoble.

Matthews, MM and Biswas, G, 1985. "Raising user proficiency through active assistance: an intelligent editor" In: Weisbin, CR, ed., *Artificial Intelligence Applications,* pp 358–362, North-Holland.

Maybury, MT, 1990. "Custom explanations: exploiting user models to plan multisentential text" *Second International Workshop on User Modeling* Hawaii.

McCauley, C, Stitt, CL and Segal, M, 1980. "Stereotyping: from prejudice to prediction" *Psychological Bulletin* **87** 195–208.

McCoy, KF, 1985. "The role of perspective in responding to property misconceptions" *Proc. 9th International Joint Conference On Artificial Intelligence,* pp 791–793, Los Angeles.

McCoy, KF, 1988. "Reasoning on a dynamically highlighted user model to respond to misconceptions" *Computational Linguistics* **14**(3) 52–63.

McCoy, KF, 1989. "Highlighting a user model to respond to misconceptions" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 233–254, Springer-Verlag.

McDermott, D, 1981. "Artificial intelligence meets natural stupidity" In: Haugeland, J, ed., *Mind Design,* 143–160, The MIT Press.

Moore, RC, 1984. "A formal theory of knowledge and action" In: Hobbs, JR and Moore, RC, eds., *Formal Theories of the Commonsense World,* pp 319–358, Ablex.

Morik, K, 1989. "User models and conversational settings: modeling the user's wants" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 364–385, Springer-Verlag.

Motro, A, 1986. "Query generalization: a method for interpreting null answers" In: Kerschberg, L, ed., *Expert Database Systems,* pp 597–616, Benjamin/Cummings Publishing Co. Inc.

Murray, DM, 1988. *A Survey of User Cognitive Modelling* Deliverable D1, Alvey Project MMI 110 NPL Report DITC 92/87, National Physical Laboratory, Middlesex, UK.

Nessen, E, 1989. "SC-UM: user modelling in the SINIX consultant" *Applied Artificial Intelligence* **3** 33–44.

Newell, A, 1982. "The knowledge level" *Artificial Intelligence* **19**(2) 87–127.

Rich, E, 1979. "Building and exploiting user models" PhD thesis, Computer Science Department, Carnegie-Mellon University, USA.

Rich, E, 1979b. "Building and exploiting user models" *Proc. 6th International Joint Conference On Artificial Intelligence,* pp 720—722, Tokyo.

Rich, E, 1979c. "User modelling via stereotypes" *Cognitive Science* **3** 329–354.

Rich, E, 1989. "Stereotypes and user modelling" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 35–51, Springer-Verlag.

Rich, E, 1983. "Users are individuals: individualizing user models" *International Journal of Man–Machine Studies* **18** 199–214.

Rivers, R. 1989. "Embedded user models—where next?" *Interacting with Computers* **1**(1) 14–30.

Salton, G, ed., 1971. *The SMART Retrieval System—Experiments in Automatic Document Processing* Prentice-Hall.

Salton, G and McGill, MJ, 1983. *Introduction to Modern Information Retrieval* McGraw-Hill.

Sandberg, J, Breuker, J and Winkels, R, 1988. "Research on HELP-systems: empirical study and model construction" *Proc. European Conference on Artificial Intelligence ECAI,* pp 106–111, Munich.

Siklóssy, L, 1979. "Passive vs. active question-answering" *Proc. First International Symposium on Policy Analysis and Information Systems PAIS,* pp 271–276, Durham, NC.

Sleeman, D, 1985. "UMFE: a user modelling front-end subsystem" *International Journal of Man–Machine Studies* **23** 71–88.

Sleeman, D and Brown, JS, eds., 1982. *Intelligent Tutoring Systems* Academic Press.

Sridharan, NS, 1985. "User modelling and plan recognition" *Proc. 9th International Joint Conference on Artificial Intelligence,* pp 1301–1302, Los Angeles.

Swartout, B, 1985. "Explanation and the role of the user model: how much will it help?" *Proc. 9th International Joint Conference on Artificial Intelligence,* p 1299, Los Angeles.

Tang, HCY, Major, NP and Rivers, R, 1989. "From users to dialogues: enabling authors to build an adaptive, intelligent system" In: Sutcliffe, A and Macauley, L, eds., *People and Computers V. Proceedings of the Conference of the British Computer Society,* pp 121–135, Cambridge University Press.

Thompson, RH, 1989. "The design and implementation of an intelligent interface for information retrieval" PhD thesis, Computer and Information Science Department, University of Massachusetts, Amherst, MA.

Thompson, RH and Croft, WB, 1989. "Support for browsing in an intelligent text retrieval system" *International Journal of Man–Machine Studies* **30**(6) 639–668.

von Hahn, W and Wahlster, W, 1983. "HAM-ANS: a German language dialog system for knowledge-based access to relational data bases" *SIGART Newsletter* **86** 41–42.

Wahlster, W, 1984. "Cooperative access systems" *Future Generation Computer Systems* **1**(2) 103–111.

Wahlster, W and Kobsa, A, 1986. "Dialogue-based user models" *Proc. IEEE* **74**(7) 948–960.

Wahlster, W and Kobsa, A, 1989. "User models in dialog systems" In: Kobsa, A and Wahlster, W, eds., *User Models in Dialog Systems,* pp 4–34, Springer-Verlag.

Wahlster, W, Marburger, H, Jameson, A and Busemann, S, 1983. "Overanswering yes–no questions: extended responses in a NL interface to a vision system" *Proc. 8th International Joint Conference on Artificial Intelligence,* pp 643–646, Karlsruhe.

Webber, BL, 1986. "Questions, answers and responses: interacting with knowledge-base systems" In: Brodie, ML and Mylopoulos, J, eds., *On Knowledge Base Management Systems,* pp 365–402, Springer-Verlag.

Webber, BL and Mays, E, 1983. "Varieties of user misconceptions: detection and correction" *Proc. 8th International Joint Conference on Artificial Intelligence,* pp 650–652, Karlsruhe.

Wilensky, R and Arena, Y, 1984. "Talking to UNIX in English: an overview of UC" *Communications of the ACM* **27**(7) 574–593.