GENERAL USER MODELING: A FACILITY TO SUPPORT INTELLIGENT INTERACTION

ROBERT KASS
Center for Machine Intelligence

TIM FININ
Paoli Research Center, Unisys Corporation

ABSTRACT

An important component of adaptable interactive systems is the ability to model the system's users. Previous systems have relied on user models tailored to the particular needs of that system alone. This paper presents the notion of a *general user model*, and describes some of our research on building a general user modeling facility that could be used by a variety of applications. This

work focuses on the representation, maintenance, and acquisition issues of modeling long-term beliefs of the user, and describes a general facility for accomplishing these tasks.

6.1 INTRODUCTION

User modeling is an important component of many systems that seek to adapt their behavior to users in order to interact more intelligently. This modeling may involve design criteria for an interface, such as human factors analysis [Norman86] (where the user is modeled by system designers when making decisions about the form of user-system interface), or it may involve using a model dynamically (where knowledge about the user is utilized to direct system behavior in an interaction). This paper discusses only the second aspect of user modeling. To provide a clear foundation for that discussion, the following definition, taken from [Wahlster86], will be used as a starting point.

A *user model* is a knowledge source in a system that contains explicit assumptions on all aspects of the user that may be relevant to the behavior of the system.[1]

Unfortunately, because user models are just one component that contributes to intelligent interaction, the user modeling aspects of interactive systems have frequently been left unexplored, or systems have employed simple, domain-specific models. This paper describes our ongoing research on the feasibility and effectiveness of *general user models*: models that have a well-defined set of capabilities that can be used in diverse situations and systems. To this end, Section 6.2 discusses when user models are needed for intelligent interfaces, and how they may be used. Section 6.3 describes the characteristics of an "ideal" general user modeling facility, whereas Section 6.4 and Section 6.5 present work we have done on the issues of user model maintenance and acquisition—focusing on models of the user's beliefs. Section 6.6 discusses how stereotypes, the classification techniques of Section 6.4, and with the implicit model acquisition techniques of Section 6.5, can be integrated, and describes a particular problem of arbitrating between conflicting beliefs about the user in such a system.

6.2 THE IMPORTANCE OF USER MODELING

User models are not needed for all man-machine interactions, or even all intelligent interfaces. User models are only beneficial to a system if it has one or more of the following characteristics [Kass86]:

---

[1] Wahlster and Kobsa's original definition was presented in the context of natural language systems only. The definition here has been expanded to include user models in any context.

- The system seeks to *adapt* its behavior to individual users;
- The system assumes *responsibility* (or shares responsibility with the user) for ensuring the success of user-system communication;
- The class of potential system users, or the potential uses of the system, is *diverse*.

Several contributors to this book describe systems that have these requirements. For example, the systems described by [Hollan88] and [Young88] modify their behavior based on their perception of user needs. Similarly, the Bridge tutor for novice programmers [Bonar88] requires a model of students' plans to help it decide what to do, and UC, the Unix Consultant [Chin88] must reason about user's goals and knowledge when generating advice in response to user questions. Further, user models may be employed by expert systems to tailor their explanations to individual users [Sleeman85; Kass88b; Paris88a]. In general, an interactive system may need to reason about a user's beliefs, goals and plans, preferences and attitudes, or capabilities to understand his or her actions and control the system's own behavior.
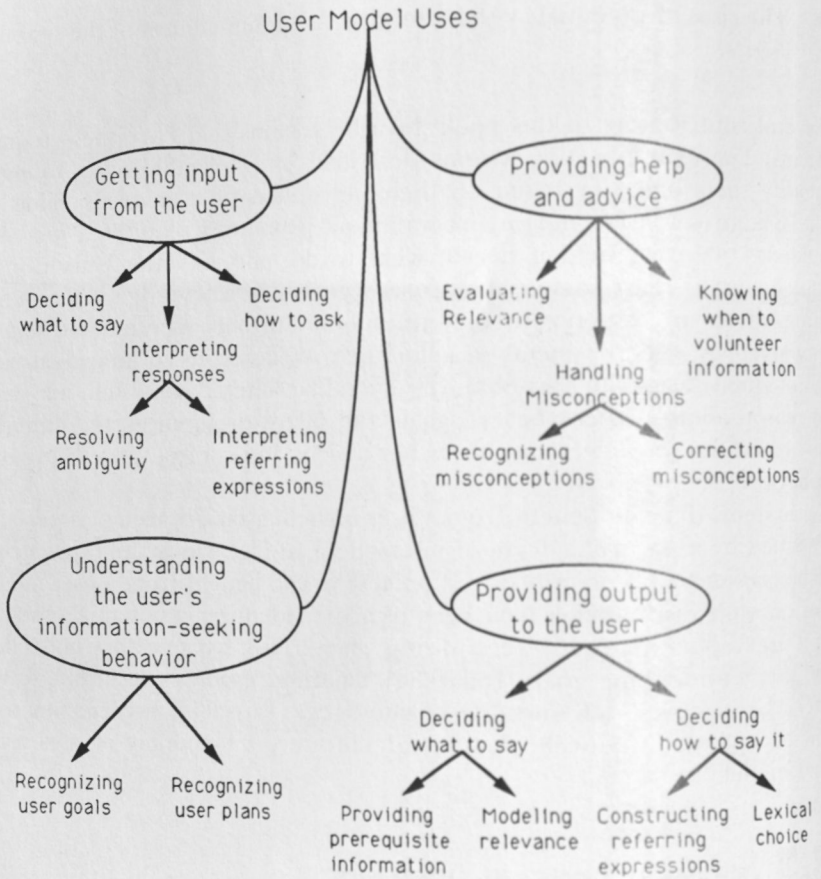
Systems that can benefit from a user modeling facility are not necessarily classified by a particular form of interaction. In fact, any system that strives to be *cooperative* (as described in [Cheikes88]) can benefit from a user model. For example, user models have been of help in identifying potential obstacles in a user's plan [Allen80], recognizing when a user's query does not reflect the user's underlying goals [Pollack86], tailoring responses according to the user's perspective [McKeown85b] or knowledge [Paris88b], or correcting user misconceptions [McCoy88]. Figure 6.1 illustrates a taxonomy of uses for a user model.

6.3 THE IDEAL GENERAL USER MODEL

Although user models have been employed in many types of interactive systems, the models have been specifically crafted for each application, usually by the explicit coding of domain-related goals, plans, or knowledge that system users are expected to have. This hand-crafting is unfortunate, because building a user modeling facility requires a substantial amount of effort. This section examines the characteristics of an *ideal* general user modeling facility, focusing on dimensions for measuring the generality of a user modeling system, and the facilities that any general user modeling system should have.

Is it possible to produce a general user modeling facility so multiple systems can benefit from a single design effort? Our ultimate answer is yes, general user modeling is practical. Although the ideal system described here may not be realized, significant features of the ideal model can be achieved and used effectively.

FIGURE 6.1
USES FOR KNOWLEDGE OF THE USER

User Model Uses

Getting input from the user
- Deciding what to say
- Interpreting responses
  - Resolving ambiguity
  - Interpreting referring expressions
- Deciding how to ask

Providing help and advice
- Evaluating Relevance
- Handling Misconceptions
  - Recognizing misconceptions
  - Correcting misconceptions
- Knowing when to volunteer information

Understanding the user's information-seeking behavior
- Recognizing user goals
- Recognizing user plans

Providing output to the user
- Deciding what to say
  - Providing prerequisite information
  - Modeling relevance
- Deciding how to say it
  - Constructing referring expressions
  - Lexical choice

6.3.1. Dimensions of Generality

User models may be general with respect to three dimensions: the range of users, the forms of interaction, and the underlying system domain. *User generality* is usually a requirement of any user-modeling facility because user models are generally employed (and most beneficial) when a range of users deal with the system, or when the system strives to adapt its behavior to individual users. Thus, interaction and domain generality are the unique features of an ideal general user-modeling facility. A user model has *interaction generality* if it can be used with a variety of interaction modes, such as structured interactions or mixed initiative dialog, and can be used with various modes of communication, such as natural language, menus, speech, and graphics. A *domain general* user modeling facility can be used with applications having a

range of knowledge bases, such as diagnostic systems for medicine, mechanical devices, and electronic components.

6.3.2. User Modeling Facilities

A general user modeling system must provide three types of user modeling facilities: representation and maintenance facilities for the contents of the model, access facilities for other components of the system or interface, and acquisition facilities for building the model [Kass86].

Representation and Maintenance

Any user modeling facility requires a knowledge base to represent its beliefs about user goals and plans, preferences and attitudes, capabilities, and beliefs about the world and other agents. A user model's representation requirements differ from most knowledge-based systems, however, in that user models are inherently volatile. Not only is new information added as an interaction proceeds, but old knowledge about the user must be revised as well. Thus, the maintenance facilities for a general user model must be able to retract beliefs about the user and resolve conflicts in those beliefs.

Access

A user model is present in a system as a service; it provides information about the system's users. How other components in the system access this information can vary widely. The user model might provide information about users *descriptively*, serving as a knowledge base of information about the user and retrieving this information in response to queries from other components in the system. A user model might also be used *prescriptively*, to simulate the behavior of the user so that the system can "run the model" to see how a user might be expected to behave. In this case, other system components may propose hypothetical additions to the user model, seeking to learn the effects of these changes on the state of the user. An example of the prescriptive use of a user model is *anticipation feedback* in generating elliptical responses [Wahlster86]. In this case a potential system response is generated and given to the user model to test whether the response can be understood by the user.

Both descriptive and prescriptive methods for accessing the user model are *passive*; other system components initiate the interaction with the user model. The user modeling facility might also be an *active* participant in the system, volunteering information to other components when it decides the information is important. For example, an active user modeling facility might monitor the user model, notifying other system components when it achieves a certain state. Such a state might occur when the modeling facility notifies a system component that a belief revision that changes the belief status of information previously provided to the component has occurred in the user model.

Acquisition

Representation, maintenance, and access facilities are of little use if the user model contains no information about the user. Methods for acquiring knowledge about the user may vary: knowledge may be explicitly encoded in the user model before it is ever used, other system components may update the user model directly (by making assertions to the user model's knowledge base), or the general user modeling facility may actively acquire its own information. Active model acquisition, in turn, can be implicit or explicit. The user modeling facility may have access to some representation of the interaction between user and system and may use this to update its model of the user (implicit acquisition), or the user-modeling facility may have its own goals about what it would like to know about the user (such as information that would resolve a belief conflict) and may generate its own request for information from the user (explicit acquisition).

For a user modeling facility to be truly general, it must be able to support all of the capabilities described above. Thus, the ideal general user model is a "toolbox" of capabilities for user modeling. It has the versatility to support a variety of demands from the overall system and the flexibility to acquire the information it needs in the best manner possible.

Our work has focused on the representation and maintenance, and acquisition facilities for general user modeling. Representation and maintenance of information about the user is central to any user modeling activity, whereas the acquisition of such information has been a major bottleneck to effective user modeling. We have also focused on modeling *long term* user information, such as the beliefs a user holds about the world or about the system domain. Such beliefs tend to persist over time, so the user model formed for an individual can be useful in many separate user-system interactions. The next two sections describe some of our efforts towards building a general user modeling facility.

6.4 GUMS—A GENERAL USER-MODELING SHELL

GUMS, a General User Modeling Shell, is intended to provide a basis for experimentation with the representation and maintenance issues of general user modeling [Finin86; Finin88]. GUMS is designed to serve as a utility for a set of application programs (see Figure 6.2). For each application GUMS keeps a knowledge base of user models relevant to that application. Applications are responsible for acquiring information about the user and supplying it to GUMS to update the user model. Likewise, the application queries GUMS to obtain information about the user, although demons can be used to accomplish the task of informing an application when specific changes occur in the user model.

FIGURE 6.2
A GENERAL ARCHITECTURE FOR A USER MODELING UTILITY
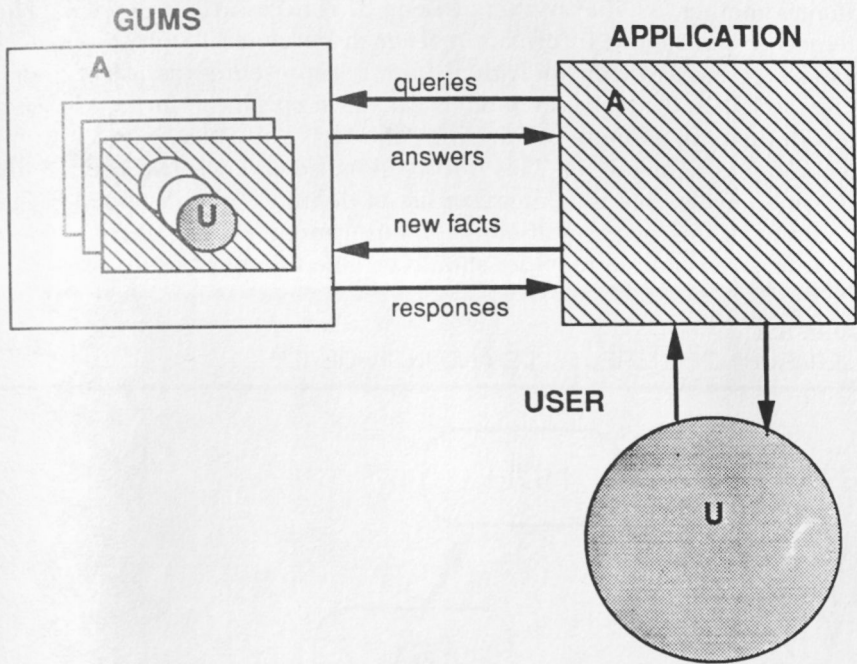

6.4.1. Representation

User modeling is most useful in situations in which a system must draw many plausible conclusions about the user on the basis of a small amount of definite knowledge. Thus, *default reasoning* [Reiter80] is an appropriate method for representing user model information. GUMS uses three default reasoning techniques to represent its beliefs about user knowledge: stereotypes, explicit default rules, and failure as negation. These three techniques capture generalizations of different grain size: they form a hierarchy with respect to the strength of their conclusions. Stereotypes capture generalizations about large classes of users. Within a stereotype, explicit default rules may express stereotypic norms that might vary for individuals of that class. Failure as negation is the weakest form of default reasoning, needed to gather weak evidence for beliefs about the user when stronger methods do not exist.

Stereotypes consist of a set of facts and rules believed to apply to a class of users. These facts and rules might be *definite*, meaning they necessarily apply to all users of that class, or *default*, specifying initial beliefs about users of that class that can be overridden. The definite information in a stereotype forms

a sort of definition for the stereotype, by specifying information that must be believed about users for them to be a member of that class.

Stereotypes can be organized in hierarchies, where one stereotype, S1, subsumes another, S2, if everything true in S1 is necessarily true in S2. Thus, a stereotype can inherit information from more general stereotypes in the hierarchy. A model of an individual user is represented as a leaf-node in the hierarchy. Individual user models can have specific information associated with them, in addition to inheriting the facts and rules from subsuming stereotypes in the hierarchy. This information, however, is constrained to be *definite* and *unitary*, that is, it must consist of definite, fully instantiated facts. A hierarchy of stereotypes and individual user models is illustrated in Figure 6.3. Ideally, users and stereotypes should be able to inherit information from

FIGURE 6.3
A HIERARCHY OF STEREOTYPES AND INDIVIDUALS
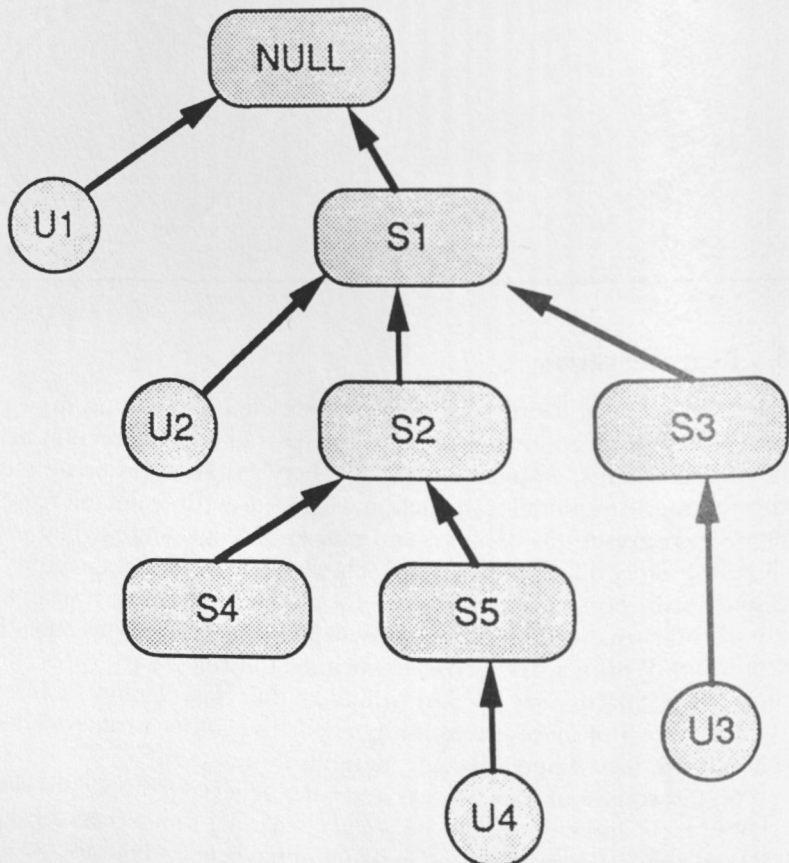
several immediate subsumers, as in a lattice, but this initial implementation limits the hierarchy to be a tree.

6.4.2. Maintenance

As new information about a user is supplied to GUMS, the individual user model must be updated, potentially creating inconsistencies in the model. The task of the maintenance facility is to update the individual user model and restore the consistency if necessary. Some inconsistencies are easy to resolve. If a new, definite fact about the user is asserted, contradicting a default assumption, the definite information is believed. Thus, default facts inherited from stereotypes, conclusions of default rules, and conclusions from failure as negation are overridden by definite facts about the user.

A more difficult conflict to resolve is one between a new, definite, fact and a definite fact inherited from a stereotype. In this case again the definite fact asserted about the user is believed, but steps must be taken to resolve the conflict with the stereotype. Because a definite fact is a defining characteristic of a stereotype, a conflict of this form means the user has been classified incorrectly in the stereotype hierarchy. Thus, this form of conflict requires a *reclassification* of the individual user model.

Reclassification can be either domain dependent or domain independent. Domain dependent methods are useful when reasons for a misclassification are understood. For example, as a user learns about the domain, the appropriate stereotype for representing his knowledge will change. Domain dependent reclassification can use knowledge of the user's expected "growth path" to select a new stereotype. A powerful domain independent reclassification method could implement a technique similar to concept classification in the KL-ONE family of representation languages [Brachman85]. The definite facts in a stereotype provide a set of features for classification. A KL-ONE style classifier would consider all possible stereotypes and find the set of most specific subsuming stereotypes with definite beliefs present in the individual user model. Although feasible, this approach may be computationally expensive. GUMS implements a simpler scheme; when a conflict is encountered, the ancestors of the current stereotype are searched in order of specificity (moving up the tree) until one is found that does not conflict with the individual user model.

GUMS enables general user modeling by providing applications with an environment containing a set of user modeling facilities. Applications using this environment take advantage of these facilities, instead of re-creating them. Thus, GUMS centralizes control of the access and maintenance of information about the user, similar to the way knowledge base systems and data base systems centralize the control of these functions for knowledge and data.

6.5 IMPLICIT USER MODEL ACQUISITION

In addition to the representation and maintenance of long term user beliefs addressed in GUMS, our work has also focused on the problem of *acquiring* a model of the user's beliefs, and included an implementation of GUMAC, a General User Model Acquisition Component [Kass87; Kass88a]. In GUMS, beliefs about the user were acquired in two ways: by explicitly encoding beliefs in the stereotypes and individual user models, and by assertions made by the application. Although GUMS can support the user-modeling requirements for many applications, a great deal of effort is required to make use of it. Not only must the application designers discover and implement a system of domain related stereotypes, but (because the application is responsible for populating the individual models of users with facts representing their beliefs) they must also design and implement some kind of knowledge acquisition strategy. Thus, the problem of acquiring user knowledge can be a significant bottleneck for general user modeling.

6.5.1. Explicit Versus Implicit Acquisition

The acquisition problem in GUMS, as in most user modeling systems, is the need to encode a large amount of information about the potential system users. Currently, several user model acquisition techniques that emphasize the *explicit* acquisition of information about the user are used. One approach, used in the BLAH system [Weiner80)] which generates tailored explanations of a system's reasoning, requires the users themselves to build the user model. A second approach is used by UMFE [Sleeman 85], a User-Modeling Front End for an explanation component of NEOMYCIN, that asks a series of questions to determine the user's knowledge of concepts that might appear in an explanation in order to present the explanation in terms the user understands. The most common approach, though, is to pre-encode a set of stereotypic user models and to try to identify which stereotypes are appropriate for particular users. Unfortunately, the first two techniques require explicit action by the user to build the user model and require self-representations that may not accurately describe the user. Although the use of stereotypes can avoid explicit model-building activity by users, it requires the system builders to encode a potentially large number of stereotypes—a task that may take more time than building the domain knowledge base itself.

An alternative to explicit user model acquisition is to build the model implicitly as the user interacts with the system. Implicit acquisition avoids the explicit encoding bottleneck and can reduce the burden on the application as well. If the user modeling facility takes full responsibility for acquiring the user model, applications do not need to reason about what information should be asserted about the user. Furthermore, if the user modeling facility's

acquisition capability is application independent, then general user modeling is a practical method for providing user modeling capabilities to a variety of applications.

Implicit user model acquisition has not been pursued extensively because it has generally been considered to be too slow to build a useful, robust model and too uncertain in the conclusions it makes. Our research suggests that this need not be the case. In particular, for specific forms of user-system interaction many basic assumptions about the user's behavior can be made that provide a foundation for drawing many conclusions about the user's beliefs.

6.5.2. User Model Acquisition Rules

This implicit acquisition approach has been implemented in GUMAC, which focuses on providing a set of user-modeling facilities for *cooperative advisory systems* (systems that advise the user and seek to be as helpful as possible) that allow the user to volunteer information and that communicate in natural language. Rather than providing a shell in which applications can be built, GUMAC implements the user-modeling facilities of a system as a separate module as illustrated in Figure 6.4. Given this model, GUMAC has four sources of information for acquiring a model of user beliefs: (1) the user's behavior that is observable by the system, (2) the system's behavior that the

FIGURE 6.4
THE GUMAC INTERACTIVE SYSTEM MODEL
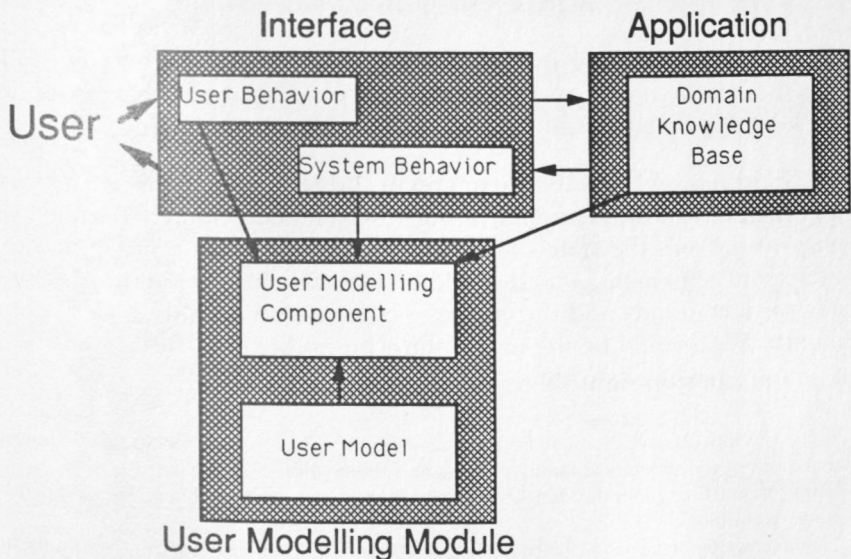
user can observe,[2] (3) the system's domain model, and (4) the current model of the user.

The implicit acquisition technique used by GUMAC has been implemented as a set of *user model acquisition rules*. These rules are domain independent, supporting the feasibility of general user modeling.[3] The rules were inspired by transcripts of over 100 conversations between a human expert and people seeking advice concerning their personal financial investments.[4] The rules capture reasonable methods that an expert might use to draw conclusions about the beliefs of the user. In fact, from a short dialogue between the system and a user, they are capable of building a model sufficiently robust to enable it to tailor its explanations to that user [Kass88a; Kass88b].

The implicit acquisition rules rely on basic assumptions about the user and the user's behavior. For example, one set of rules,[5] the *cooperativity* rules, assumes the user is cooperating with the system and thus is observing Grice's Maxims for cooperative communication [Grice75]. The *relevancy rule*, based on Grice's maxim of relation "Be relevant," allows the system to draw conclusions about the user's knowledge of the system's reasoning. If the maxim of relation is being obeyed then the system expects users to believe that the statements they make are relevant to their current conversational goals. In the type of expert system advisory interactions assumed in this work the system (which controls the interaction) establishes these goals, usually by asking the user a question. Thus, cooperative users will respond with information they believe is relevant to accomplishing the system's goals. In general form, the rule is stated as follows:

$$\text{coop} - \text{agent(User)} \wedge \text{do(User, Act)} \wedge \text{current} - \text{goal(System, G)} \rightarrow$$

$$\text{Bel(User, subgoal(Act, G))}.$$

Thus, in the following dialogue taken from the transcripts, the caller believes that all the information she provides is relevant to determining how to take her supplemental annuity.

(1) C.   I just retired December first, and in addition to my pension and social security I have a supplemental annuity, which I contributed to while I was employed, from the state of New Jersey mutual fund. I'm entitled to a lump sum settlement, which would be between \$16,800 and \$17,800, or a lesser life annuity and the choices of the annuity would be \$125.45 per month. That would be the maximum with no beneficiaries.

(2) E.   You can stop right there, take your money.

---

[2] Both (1) and (2) are obtained by gaining access to the interface component's internal representations of the statements made by users and the system.

[3] Thus, interaction generality has been abandoned in the effort to obtain domain-general user model acquisition.

[4] The transcripts were made by Martha Pollack and Julia Hirschberg from the radio talk show "Harry Gross: Speaking about Your Money" broadcast on station WCAU in Philadelphia, February 1–5, 1982.

[5] See [Kass88a] for a complete presentation of the rules.

If this dialogue took place between a user and an investment advisory system, the user-modeling module would assert that the user believes that each of these items is relevant to the goal of deciding how to take the annuity, even though the system knows that some of the information is, in fact, not relevant.

Other acquisition rules make assumptions about reasoning that the user is likely to do. In general, one cannot assume that the user will believe all of the logical consequences of his current beliefs (i. e., assume *consequential closure*). Instead, an approximate model of the inferences any user would be likely to draw is needed. Our model includes, for example, rules to cover the transitivity of subsumption relations (if the users know A is a kind of B, and B is a kind of C, then they will infer that A is a kind of C) and inheritance of properties of concepts.

Another group of rules focuses on assumptions about the user's human behavior. For example, the *agent rule* considers the situation where the system knows the user has performed a particular action. The fact that the user was the agent of this action is significant, because it means the user himself must know of the action, know of any necessary substeps of the action, and know all the information related to performing that action. For example, if the user says, "I just rolled over two CD's," he or she must not only know about the rolling over action, but also that CD's have a due date, that they are obtained from banks, and so on. Thus, the Agent Rule can be a quite powerful way of determining a large amount of information about the user's beliefs. The rule states:

$$achieved(User, Act) \rightarrow$$

$$Bel(User, goal(Act)) \wedge \forall p(property(Act, p) \supset Bel(User, property(Act, p)))\wedge$$

$$\forall g(necessary - subgoal(g, Act) \supset (Bel(User, subgoal(g, Act))\wedge$$

$$Bel(User, goal(g)) \wedge \forall p(property(g, p) \supset Bel(User, property(g, p)))))).$$

Here, "necessary-subgoal(A, B)" is needed because there may be more than one way to achieve a goal. A necessary subgoal is one that must be performed, no matter what plan is used to accomplish the goal. Thus, the necessary subgoals constitute the intersection of the steps of all possible plans to achieve the goal.

In summary, a significant problem in acquiring user models can be overcome through the use of implicit acquisition techniques. The acquisition rules developed in our work are domain independent, thus they enable more practical general user models to be built.

6.6 INTEGRATING STEREOTYPES AND IMPLICIT ACQUISITION

Although the contrasts between explicit and implicit user model acquisition techniques are emphasized in Section 6.5, these methods can complement each other. Despite the problems with explicit acquisition and the advantages of implicit acquisition methods, in many situations it is still desirable to encode

domain-specific knowledge about users. This section describes how GUMAC integrates the implicit acquisition rules with the GUMS framework, resulting in a powerful, extensible general user modeling facility that benefits from both acquisition approaches.

The key to integrating the implicit acquisition rules with GUMS is recognizing that the rules are *default* rules, sanctioned by specific assumptions about the user. Thus, the acquisition rules can be viewed as elements of very general stereotypes. For example, a stereotype for the class of "cooperative agents" contains the cooperativity rules, including the Relevancy Rule, as default rules. Other general stereotypes include "rational agent," containing rules modeling the user's reasoning capabilities, and "communicative agent," containing rules specific to communication. These stereotypes distinct from the domain specific stereotypes, form an independent hierarchy in a stereotype lattice. Thus, for an investment advisor system, a hierarchy of stereotypes such as in Figure 6.5 might be used.

With this notion of general stereotypes it is useful to distinguish between two types of rules a stereotype may contain: rules believed to be used *by* the users in their own reasoning (user inference rules) and rules *about* users needed by the system to make conclusions about them (model acquisition rules). For example, the Relevancy Rule reasons *about* users' beliefs, but a transitivity rule, although it can be used to draw conclusions about the user's beliefs, is assumed to be used *by* the users in their own reasoning. Any of these rules may be held as definite or default beliefs.

FIGURE 6.5
A HIERARCHY CONTAINING GENERAL AND DOMAIN INDEPENDENT
STEREOTYPES
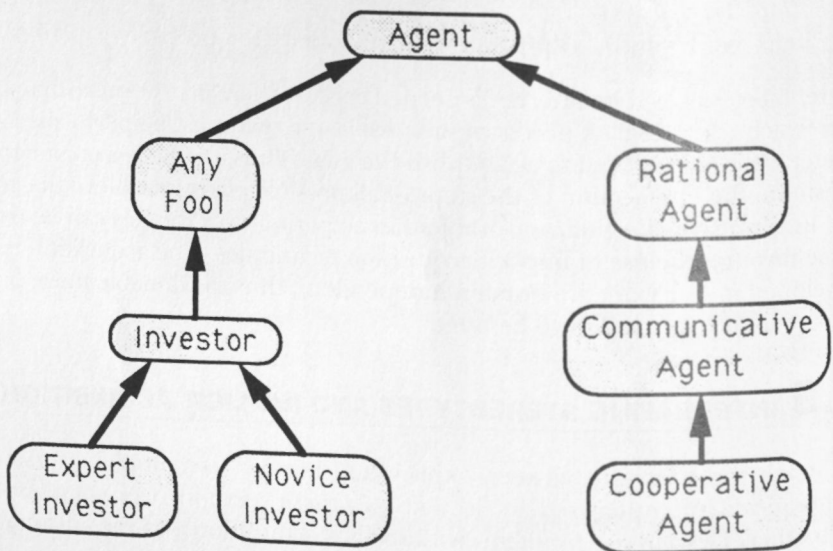
No representational distinction is made between domain independent and domain dependent stereotypes, enabling two additional, potentially powerful user modeling capabilities. First, model acquisition and user inference rules that are domain-specific can be included in stereotypes. One of the problems with building stereotypes is the number of facts that must be included in the stereotype. Domain specific model acquisition rules enable the user-modeling facility to infer these facts implicitly, so they do not need to be encoded explicitly.

The second capability involves the encoding of explicit facts in general stereotypes. A problem with many interactive systems is their limited knowledge of the world. If a body of common sense knowledge that all users are assumed to know is available in a very general stereotype, it could relieve the brittleness problem of such systems. Such a stereotype of common sense knowledge might be called the *any fool stereotype*, after McCarthy's notion that this is the knowledge "any fool" would know [McCarthy80].

6.6.1. Integration Problems

Some problems arise, however, when implicit acquisition methods are integrated with the stereotype hierarchy of GUMS. Two that we have encountered are discussed here, with our thoughts on potential solutions for the problems.

One problem with very general stereotypes, such as "cooperative agent," is the lack of defining facts about members of that class. In GUMS, the hierarchy of stereotypes is determined by the definite facts contained in the stereotypes, but for a stereotype such as "cooperative agent," useful defining facts are rare. Users might belong to the class of cooperative agents if the property "cooperative" can be applied to them, but such a user property is as difficult to define and as hard to acquire. Thus, it is not feasible to wait to discover the fact "cooperative(U)" before classifying U as a cooperative agent.

This difficulty can be avoided by initially classifying the user beneath all of the domain independent stereotypes. Consequently, GUMAC initially assumes the user is a rational, communicative, and cooperative agent, so the defining facts of these stereotypes are not needed in order to use them. Thus, for an investment advisory system, a user might originally be assumed to be an "initial agent" and a "novice investor," as in Figure 6.6. It might be necessary, however, to retract the assumption that a general stereotype applies to a user. For example, this might be necessary when the cooperativity rules consistently make conclusions that are contradicted by other, more certain, beliefs about the user. We do not know of a general method for distinguishing when one or more default rules in a stereotype should be retracted, and when belief in the stereotype as a whole should be retracted.

A second problem involves conflicting beliefs about the user. In GUMS new information about the user was supplied by the application and GUMS assumed it was definite knowledge. Thus, most of the belief conflicts encoun-

FIGURE 6.6
INITIAL CLASSIFICATION FOR AN INVESTMENT ADVISORY SYSTEM USER
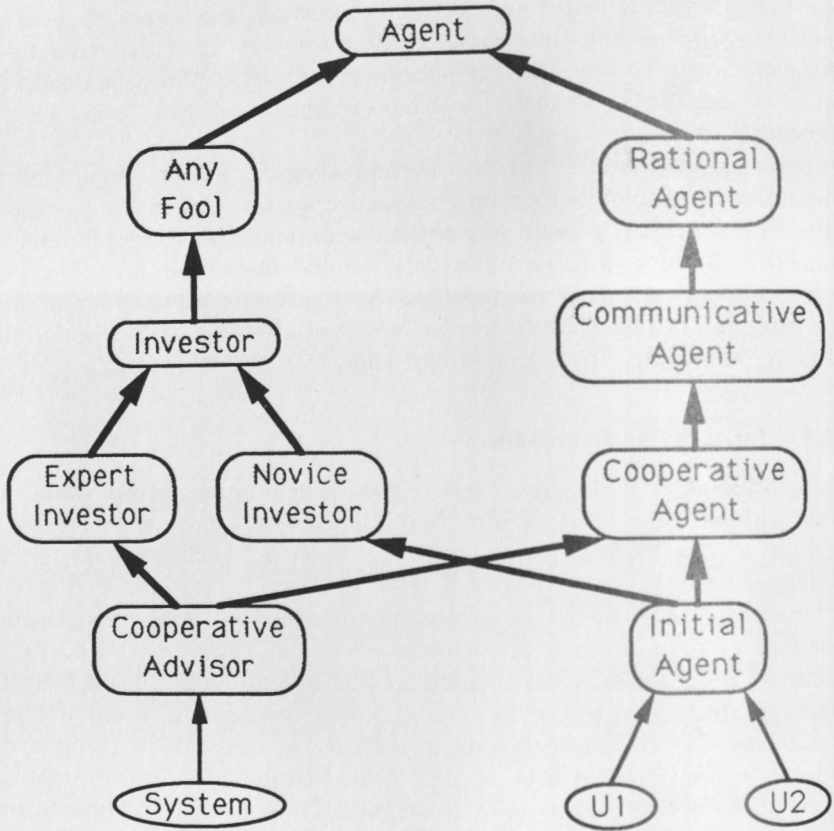
tered by GUMS are between definite knowledge from the application, and definite or default knowledge assumed by stereotype subsumption. Conflicts between definite and default knowledge are easy to resolve: believe the definite knowledge. In GUMS a conflict with definite knowledge from a stereotype is also straightforward; belief in the stereotype is dropped, and the hierarchy is traversed upward until a stereotype consistent with the definite knowledge about the user is found.

The simple resolution techniques in GUMS do not extend well to the integrated version of the system. First, the assumption that assertions from the application are definite is a simplification. Most of the assertions made by the implicit acquisition rules are default conclusions. Second, the stereotype hierarchy in GUMS is a tree, whereas GUMAC requires a lattice to reflect the inheritance of several independent sets of assumptions about the user. Thus, GUMAC may have conflicts between stereotypes that are incomparable.

Some of these problems can be handled by applying a heuristic that attempts to maximize the number of assumptions about the user after a conflict is resolved—this heuristic is used implicitly in GUMS. The stereotype hierarchy in GUMS represents an ordering based on the amount of information assumed about the user: stereotypes at the top of the tree contain few assumptions, whereas those at the bottom inherit the assumptions from higher stereotypes in addition to their local assumptions, so they contain more assumptions. When a conflict is encountered that requires a stereotype to be dropped, GUMS traverses up the tree, trying to find the stereotype with the most information that is consistent with the definite beliefs about the user.

A similar method can be used in GUMAC: when a conflict between stereotypes arises, the system should retract the stereotype that leaves the maximum number of assumptions about the user intact. Because several stereotypes may concurrently draw conclusions about some facts, this means the stereotype that makes the least number of unique conclusions about the user should be retracted.

6.7 CONCLUSION

General user modeling is an attractive approach to providing interactive systems with information about their users. Our work in this area indicates that general user modeling is not only feasible, but practical. The GUMS system demonstrates how a set of facilities for user modeling can be provided in the scope of an environment for building interactive applications. Further, work on implicit acquisition indicates that the acquisition bottleneck can be overcome in a domain independent manner. Thus, a general user modeling facility for supporting cooperative advisory systems is a practical possibility.

REFERENCES

[Allen80] Allen, J. F., and Perrault, C. R. 1980. Analyzing Intention in Utterances. *Artificial Intelligence*, 15, 143–178.

[Bonar91] Bonar, J., and Liffick, B. W. 1991. Communicating with High-Level Plans. In present volume.

[Brachman85] Brachman, R. J., and Schmolze, J. G. 1985. An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9, 171–216.

[Cheikes88] Cheikes, B. A. and Webber, B. L. 1988. The design of a cooperative respondent. In *Proceedings of the Workshop on Architectures for Intelligent Interfaces: Elements and Prototypes*, pp. 3–18.

[Chin89] Chin, David N. 1989. KNOME: Modeling What The User Knows in UC. In Kobsa, A. and Wahlster, W. (eds.), *User Models in Dialog Systems*. Berlin and New York: Springer Verlag, pp. 74–107.

[Finin86] Finin, T., and Drager, D. 1986. GUMS₁: A General User Modeling System. In *Proceedings of the 1986 Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 24–30.

[Finin89] Finin, T. 1989. GUMS—A General User Modeling Shell. In Kobsa, A., and Wahlster, W. (eds.), *User Models in Dialog Systems*. Berlin and New York: Springer Verlag, pp. 411–430.

[Grice75] Grice, H. P. 1975. Logic and Conversation. In Cole, P., and Morgan, J. L. (eds.), *Syntax and Semantics*, 3, pp. 64–75. New York: Academic Press.

[Hollan91] Hollan, J., Rich, E., Hill, W., Wroblewski, D., Wilner, W., Wittenburg, K., and Grudin, J. 1991. An Introduction to HITS: Human-Interface Tool Suite. In present volume.

[Kass86] Kass, R., and Finin, T. 1986. User Models in Intelligent Interfaces: Why They Are Needed, Problems with Their Implementation. In B. Neches and T. Kaczmarek, (eds.), *AAAI-86 Workshop on Intelligent Interfaces*, pp. 48–50.

[Kass87] Kass, R., and Finin, T. 1987. Rules for the Implicit Acquisition of Knowledge about the User. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pp. 295–300.

[Kass88a] Kass, R. J. 1988. *Acquiring a Model of the User's Beliefs From a Cooperative Advisory Dialog*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

[Kass88b] Kass, R., and Finin, T. 1988. The Need for User Models in Generating Expert System Explanations. *International Journal of Expert Systems*, 1(4), 345–375.

[McCarthy80] McCarthy, J. 1980. Circumscription—A Form of Nonmonotonic Reasoning. *Artificial Intelligence*, 13(1), 27–40.

[McCoy88] McCoy, K. F. 1988. Reasoning on a Highlighted User Model to Respond to Misconceptions. *Computational Linguistics*, 14(3), 53–63.

[McKeown85] McKeown, K. R. 1985. Tailoring Explanations for the User. In *Ninth International Conference on Artificial Intelligence*, pp. 794–798.

[Norman86] Norman, D. A., and Draper, S. W., (eds.) 1986. *User Centered System Design: New Perspectives in Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

[Paris88a] Paris, C. L. 1988. Generation and Explanation: Building and Explanation Facility for the Explainable Expert System Framework, July. Presented at the Fourth International Workshop on Natural-Language Generation.

[Paris88b] Paris, C. L. 1988. Tailoring Object Descriptions to a User's Level of Expertise. *Computational Linguistics*, 14(3), 64–78.

[Pollack86] Pollack, M. E. 1986. *Inferring Domain Plans in Question-Answering*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

[Reiter80] Reiter, R. 1980. A Logic for Default Reasoning. *Artificial Intelligence*, 13(1), 81–132.

[Sleeman85] Sleeman, D. H. 1985. UMFE: A User Modeling Front End Subsystem. *International Journal of Man-Machine Studies*, 23, 71–88.

[Weiner80] Weiner, J. L. 1980. BLAH, a System Which Explains its Reasoning. *Artificial Intelligence*, 15, 19–48.

[Wahlster86] Wahlster, W., and Kobsa, A. 1986. Dialog-based User Models. In *Proceedings of the IEEE*, 74(7).

[Young91] Young, R. L., 1991. A Dialogue Model for User Interfaces. In present volume.