1 Introduction

We are investigating computational models of dialog that can support robust, effective communication between people and computer systems [McRoy *et al.*, 1997, McRoy, 1995, McRoy, 1998, McRoy and Hirst, 1993, McRoy *et al.*, 1998a, McRoy *et al.*, 1998b, Restificar *et al.*, 1999a, Restificar *et al.*, 1999b]. Developing such methods requires:

- The specification and representation of information that affects learning and decision-making.
- The specification and implementation of algorithms for discriminating among alternatives, for recognizing errors, and for generating repairs.

The evaluation of this work involves the construction of computer programs that collaborate with people on tasks such as collaborative training or decision support. More specifically, we are developing collaborative tutoring systems for medical students to practice their decision-making skills (the B2 project) and for blood pressure health education (the ColTrain project).

The general model of processing for our work is one of an Intelligent Dialog System [Bordegoni *et al.*, 1997]. Intelligent Dialog Systems (IDS) are concerned with the effective management of an incremental, mixed-initiative interaction between the user and the system. This approach is in contrast with a presentation system, where the system's outputs are preplanned (*e.g.* driven by a fixed plan or grammar) and not adapted to the user's apparent understanding or lack thereof. In an IDS, content to be presented, as well as the system's model of the user, change dynamically during an interaction.

Reasoning about dialog, such as to determine what a user's actions mean in the context of the dialog, whether a user's actions indicate understanding and agreement, and how to respond to a user's action, requires representing and combining many sources of knowledge. To support natural communication (which may contain fragments, anaphora, or follow-up questions), as well as to reason about the effectiveness of the interaction, a dialog system must represent both sides of the interaction; it must also combine linguistic, social, and intentional knowledge that underlies communicative actions [Grosz and Sidner, 1986, Lambert and Carberry, 1991, Moore and Paris, 1993, McRoy and Hirst, 1995]. To adapt to a user's interests and level of understanding (*e.g.* by modifying the questions that it asks or by customizing the responses that it provides), a dialog system must represent information about the user and the state of the ongoing task.

Our work makes use of a uniform, declarative representation of dialog. *Uniformity* in a knowledge representation means that it allows the representation of different kinds of knowledge in the same knowledge base using the same inference processes. For example, the representation of a spoken `Huh` might be represented similarly to a typed `Huh?` and reasoned with identically. We favor uniformity because it eliminates knowledge interchange overhead and associated problems such as conflicting ontologies that often arise as implementations attempt to scale up (*cf.* [Traum *et al.*, 1996]). We favor a declarative approach because we wish to model dialog in a domain and modality independent way. Our approach differs from other declarative approaches to language processing and dialog, such as [Smith and Hipp, 1994], in that we use a full first-order logic based knowledge representation and represent dialog in far greater detail, at multiple levels of interpretation 3. This detail supports our approach's ability to handle failed expectations and to handle inputs from different modalities. The most important disadvantages of our approach is that it requires a significant investment in knowledge engineering and careful attention to pragmatic issues such as efficiency.

The architecture that we have been developing for building Intelligent Dialog Systems includes computational methods for the following:

- The representation of natural language expressions;
- The interpretation of context-dependent and ambiguous utterances;
- The recognition and repair of misunderstandings (by either the system or the user);
- The detection and rebuttal of arguments; and
- The generation of natural language responses in real-time.

In what follows, we present an architecture and computational model that addresses these issues. We present a detailed example of the representations and processing required to answer a question. We will then describe some of the components of our work and how they make use of the uniform, declarative representation of dialog.

2 A General Architecture for Dialog

Our architecture for Intelligent Dialog Systems is shown in Figure 1. The Input Manager and Display Manager deal with input and output, respectively. The input modalities will include typed text, spoken text, mouse clicks, and drawing. The output modalities will include text, graphics, speech and video. The Dialog Manager is the component through which all input and output passes. This is important because the system must have a record of everything that occurred (both user and system-initiated). If the user chooses to input language, the Language Manager will receive the text to parse and build the appropriate representation; the parsed input is then interpreted by another component of the dialog manager. The Domain Manager component contains general rules of the task as well as specific information associated with how the Content is to be presented. The content will be generated, prior to system use, by the use of Authoring Tools that allow the rapid development of the content.[1] Based on the ongoing interaction, as well as information provided by the user, User Background & Preferences are tracked. The status of the interaction is evaluated incrementally by the Evaluation Manager, which assesses the ongoing dialog and user model and makes changes to the domain model as appropriate. We will present some of these components in more detail in Sections 4 and 5.

[1]The content may be specified at varying levels of detail from simple facts about class membership or the properties of objects to whole paragraphs that should be output verbatim when requested by the user. Rules of the task include the teacher's plan for presenting a set of lessons as well as low-level rules that say how particular goals might be achieved. During the actual interaction, this content is generally static, but may be adapted or combined in response to dynamic changes in the system, such as those noted by the Evaluation manager, described below. For example, the system will provide a more sophisticated style of explanation to a nursing student than to a member of the general public — if it learns what type of student is using the system, and the domain model includes the necessary alternative explanations.

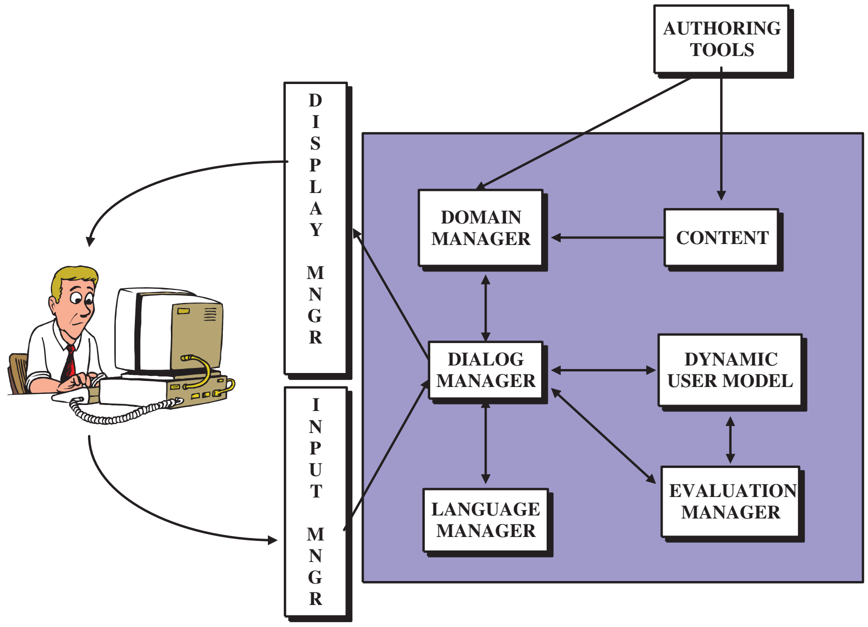

Figure 1: Our Architecture for Intelligent Dialog Systems

Fundamental to this architecture is the notion that all modalities and media are the same at the level of communicative act. In particular, non-text input will be mapped onto a propositional form (roughly a locutionary act) when the corresponding interface token is selected.[2] Similarly, multimedia output can be handled by including non-text primitives to the realization component, although our current implementations produce text output only.[3]

This architecture for dialog builds on our prior work, where the user is on a "thin" client personal computer interacting with a server that contains all the components described [McRoy *et al.*, 1997]. Most components of this architecture are general purpose; to re-target the system for a new domain, one would need to re-specify only the domain and content.[4]

As in our prior work [McRoy *et al.*, 1997], the core of the dialog system (the components within the large box on the right) all share a common representation language and a common inference and acting system. This

[2] Although in principal our model can thus handle a wide variety of input modalities, in practice, our current implementations work with typed text and mouse click actions. These actions are then mapped onto communicative acts such as *request*, *inform*, *askif* (yes-no questions) and *askref* (wh-questions) by means of interpretation rules at the third level of our dialog model (See Section 3.)

[3] For example, an implementation that uses the same knowledge representation framework, SNePS [Shapiro and Rapaport, 1992, Neal *et al.*, 1989], but not same content or dialog structure, has implemented simple speech and graphics type animations.

[4] Although the domain and content are the only components that would need respecification, we do not intend to imply that these are easy tasks. No approach, including our own, can avoid the inherent difficulty in specifying domain-specific content; however we do not exacerbate these tasks by requiring that the dialog model be respecified as well. We are also implementing authoring tools to assist domain experts.

representation and reasoning framework acts as a blackboard for intertask communication and cooperation. The representation includes the content and structure of the interaction at several different levels of interpretation. The particular framework that we use is a (partitioned) propositional semantic network formalism; see Section 2.3. We partition the network to improve efficiency.[5] Primitive acts specify which partitions are relevant to a particular reasoning problem.

In contrast to the work described in [McRoy *et al.*, 1997], little of the knowledge needed for the management of dialog is represented explicitly as procedural processing algorithms. Instead, knowledge about how to interpret information or how to select content to generate a response is represented declaratively, in the same, network-based framework. Moreover, utterances are interpreted and generated using the same declarative representation of communicative plans.

Control of the system resides in the dialog manager, which is a simple procedure that takes inputs sent by the interface (in our case, strings sent by a Java program) and calls the appropriate analyzer for the input type (*e.g.* text, menu selection, etc). For example, the analyzer for text is a linguistically-based parser, which, when initiated as an action, causes:

1. the creation of initial representations of the utterance;
2. the initiation of an interpretation action; and
3. the generation of a response, which may include actions that manipulate the knowledge base or actions that generate text via our sentence realizer, YAG (see Section 5).

2.1 Dialog Processing

Actions by the user are interpreted as communicative acts by considering what was observed and how it fits with the system's prior goals and expectations. First, a parser with a broad coverage grammar builds a mixed-depth representation of the user's actions [McRoy *et al.*, 1998a, McRoy *et al.*, 2000].[6] This representation includes a syntactic analysis and a partial semantic analysis. Mixed-depth representations are constructed incrementally and opportunistically. They are used to address the ambiguity that occurs in utterances, without sacrificing generality. Encoding decisions that require reasoning about the domain or about the discourse context are left to subsequent processing.

Second, the dialog manager uses domain knowledge to map linguistic elements onto domain elements and to refine some semantic structures. This level of processing includes the interpretation of noun phrases, the resolution

[5] For example, we partition knowledge that is associated with a particular processing module (such as the domain manager or the language manager), and we sub-partition knowledge for a particular module, such as knowledge that is most closely associated with particular processing tasks or that is associated with particular levels of our discourse model (see section 3).

[6] A *mixed-depth representation* is one that may be shallow or deep in different places, depending on what was known or needed at the time the representation was created [Hirst and Ryan, 1992]. Shallow representations include a representation of the interaction such as a sequence of time-stamped events. Deep representations include conventional first-order (or higher-order) AI knowledge representation. (The distinction is similar to the one between locutionary and illocutionary acts [Austin, 1962].)

of anaphora, and the interpretation of sentences. For example, the mixed-depth representation leaves the possessive relationship uninterpreted; at this stage, domain information is used to identify the underlying conceptual relationship (*i.e.* ownership, part-whole, kinship, or object-property), as in the following:

> The man's hat (ownership); the man's arm (part-whole);
> the man's son (kinship); the man's age (object-property).

Next, the dialog manager identifies higher-level dialog exchange structures and decides whether the new interpretation confirms its understanding of prior interaction. Exchange structures are pairs of utterances (not necessarily adjacent, because a sub-dialog may intervene) such as question-answer or inform-acknowledge. The interpretation of an exchange indicates how the exchange fits with previous ones, such as whether it manifests understanding, misunderstanding, agreement, or disagreement.

Finally, the assertion of an interpretation of an utterance triggers the appropriate actions (*e.g.* a question will normally trigger an action to compute the answer) to provide a response. In Section 3.2, we will illustrate our approach by working through the answer to a question: *What is Mary's age?*.

2.2 Interpretation as Reasoning

The creation of a mixed-depth representation of an utterance by the parser triggers the act of interpretation. Interpretation is a primitive act (implemented as a lisp procedure) that creates a new context[7] containing the discourse history and all the rules of interpretation and then triggers two types of inference: forward inference to map any vaguely specified entities onto objects in the domain and backward inference to deduce the communicative intent. In addition, we may invoke other types of primitive interpretation actions, such as argument detection, which similarly creates a new context and triggers forward inference in this context. Finally, the deduction of an interpretation initiates the generation of a response (also implemented as a primitive act). As with other acts, generation creates a new context containing the relevant knowledge (the pragmatic interpretation of the dialog) and all the rules for generation (which include rules for achieving goals and for constructing rebuttals) and triggers forward inference. Rules for generation include the plans needed achieve a goal such as that when the agent has a goal of having the user know some fact, one way of achieving it is to express it using a subtype of the communicative act `inform`, which is linked to a primitive act that invokes the text realization component. (The interface itself specifies the form of the inform, e.g. as a string of written text.)

Reasoning is triggered by input from the user through the interface. Input from the interface is sent to the appropriate manager. If the manager returns a response, the system will pass that information back to the interface object specified within the response. (That response need not use the same modality as the input.) In the case of text, the system invokes a

[7] We call a partitioning of the knowledge base a *context*. Contexts may share knowledge, contain more or less knowledge, or contain contradictory information, with respect to the primary knowledge base (from which these contexts are derived). Reasoning, in such contexts, is constrained to the knowledge in the context.

primitive act to interpret the utterance; the object of the act is the string corresponding to what was typed. This act has several steps:

1. The first step involves involves creating a temporary context, adding to it the information about who the speaker and hearer are, and calling the parser appropriate to the input type (*e.g.* a linguistically based parser for text). This step results in the assertion of a mixed-depth representation of the input, namely that some new discourse entity is an utterance with certain features. This assertion is then copied to a permanent context.
2. The assertion of this mixed-depth representation initiates:
   (a) a primitive act to add sequencing information, (*i.e.* that this utterance immediately follows the previous one) which is added to a permanent context,
   (b) another primitive act to partially interpret the content of the utterance. This creates its own temporary context, including the statement about the utterance and the rules for (domain independent) semantic interpretation, and then the resulting interpretation is added to a permanent context.
   (c) a primitive act to interpret any anaphora. Specifically, a temporary context is created that includes previous semantic interpretations, domain information, and the rules for interpreting anaphora. The results are then added to the permanent context that contains the previous semantic interpretations.
3. The final step in interpretation is triggered by the creation (by the previous step) of the partial interpretation of the input. This initiates a primitive act to interpret the illocutionary force of the utterance. This entails creating a temporary context that contains the semantics and the relevant interpretation rules. These rules include domain specific rules for describing command types or entities (such as that speakers may refer to the description of a medical case as a "story") as well as domain independent rules (such as that one way for a speaker to perform a request act is to "say" the content as an imperative sentence.)

Once a pragmatic level interpretation is asserted, it triggers new actions to address the goal of the communicative act; in the case of a question, there would be an attempt to find and provide an answer.

2.3 The Underlying Inference and Acting System

The inference and acting system provides services used for interpreting the user's actions and for constructing a response:

- It maintains a record of the events performed by the system and the user during their interaction (the discourse history).
- It keeps information needed by the system to select actions, including the relationships between plans and acts and between plans and goals.
- It represents and reasons over information about the domain and mediates between explicitly represented knowledge and any external knowledge sources.
- It partitions the knowledge for efficient reasoning.

All this information is represented as a propositional semantic network, that is, as a graph composed of nodes and labeled arcs, where the propositions are represented by the nodes. A propositional semantic network is a framework for representing the concepts of a cognitive agent who is capable of using language (hence the term *semantic*). The information is represented as a graph composed of nodes and labeled directed arcs. In a *propositional* semantic network, the propositions are represented by the nodes, rather than the arcs; arcs represent only non-conceptual binary relations between nodes.

The particular knowledge representation system that is used is SNePS [Shapiro, 2000, Shapiro and Rapaport, 1992]. SNePS provides facilities for building and finding nodes, as well as for (first- and second-order) reasoning, truth-maintenance, planning/acting, and knowledge partitioning (for user- and system-models). Our model is knowledge-intensive, knowledge partitioning allows tractable inference in real-time.

```
* (describe (assert member "Tweety" class "bird"))
  (M1!  (CLASS bird) (MEMBER Tweety))
```

Figure 2: `Tweety is a bird` as represented in the knowledge base.

Case frames are used to represent propositions. Case frames are conventionally agreed upon sets of arcs emanating from a node with specified semantics. For example, to express the proposition that `A isa B`, we use the MEMBER-CLASS case frame, which is a node with a MEMBER arc and a CLASS arc. Figure 2 shows the construction and representation of `Tweety is a bird` as node `M1!`. An extensive collection of standard case frames is provided in [Shapiro *et al.*, 1994] and additional case frames can be defined as needed. We use many standard case frames as well as several new ones.

3 The Representation of Discourse

The dialog manager builds five levels of representation, shown in Figure 3, to capture the content, structure, and sequence of the system's and the user's utterances. Together these levels capture everything that the user and the system have said, as well as how their utterances extend the ongoing discourse. They are needed to allow the system to interpret context-dependent utterances such as *Why?* or to deal with unexpected utterances, such as misunderstandings or arguments. All this knowledge is represented uniformly in one knowledge base; however we partition the knowledge for efficiency. For example, each level, as well as the rules for creating and manipulating each level, comprise their own context (however, nodes may belong to more than one context). The levels are created by the dialog manager by adding a new node to the utterance level (the one produced by the targeted input analyzer) and then initiating forward or backward inference over the nodes in each of the higher levels.

The utterance level is a (mixed-depth) representation of what the user typed or selected with a mouse, as produced by the parser. The second level corresponds to the sequence of utterances, which enables the system to reason about temporal ordering constraints. (This level is comparable to the linguistic structure in the tripartite model of [Grosz and Sidner,

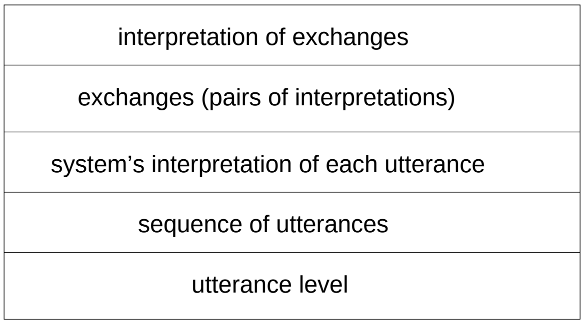

Figure 3: Five Levels of Representation

1986]). The third level comprises the system's interpretation of each utterance. Each utterance event (from level 1) will have an associated system interpretation, which corresponds to a communicative act (such as question or command) which may reference entities from the underlying task. The fourth and fifth levels of the discourse model are exchanges and interpretations of exchanges, respectively. These levels represent a key difference between our work and previous approaches. These structures are determined on the basis of a number of domain-independent schemata which are represented declaratively in a logical language. The starting point for these schemata was socio-linguistic accounts of dialog [Schegloff, 1992, Clark and Schaefer, 1989] and Grice's [Grice, 1975] notion of reflexive intentions. Most AI approaches to dialog are based on Searle's [Searle, 1969, Searle, 1979] account of speech acts and STRIPS [Fikes and Nilsson, 1971, Sacerdoti, 1977] style plan operators of traditional AI. Our approach is very flexible and can adapt to failed expectations (such as in misunderstanding or argumentation) [McRoy and Hirst, 1995, Restificar *et al.*, 1999a].[8]

3.1 The Levels in an Example Dialog

We illustrate some (but not all) of the levels of our dialog model for the dialog below:

User:	`Why does a positive HIDA suggest gallstones?`	1
B2	*In the case of Mr Jones, the pretest probability of gallstones is 0.135. A positive HIDA test results in a post-test probability of 0.307.*	2
User:	`I mean for what reason.`	3
B2:	*Oh. HIDA detects cholecystitis, which is caused by gallstones.*	4

Figure 4 illustrates the utterance sequence, interpretation, and exchange levels of representation that would result after this conversation.[9] Starting from the top of the figure, we see the following:

- Nodes `M150`, `M160`, and `M170` represent the sequencing relations that hold between utterances 1–4. (The first row of boxes gloss the subnet-

[8] Currently, our implementation incorporates schema for argumentation, but we are still working on incorporating the schema from our previous work on resolving misunderstanding.

[9] Currently, our system produces simplistic natural language output, rather than the output shown. We are still working on improving the text planning part of our system.

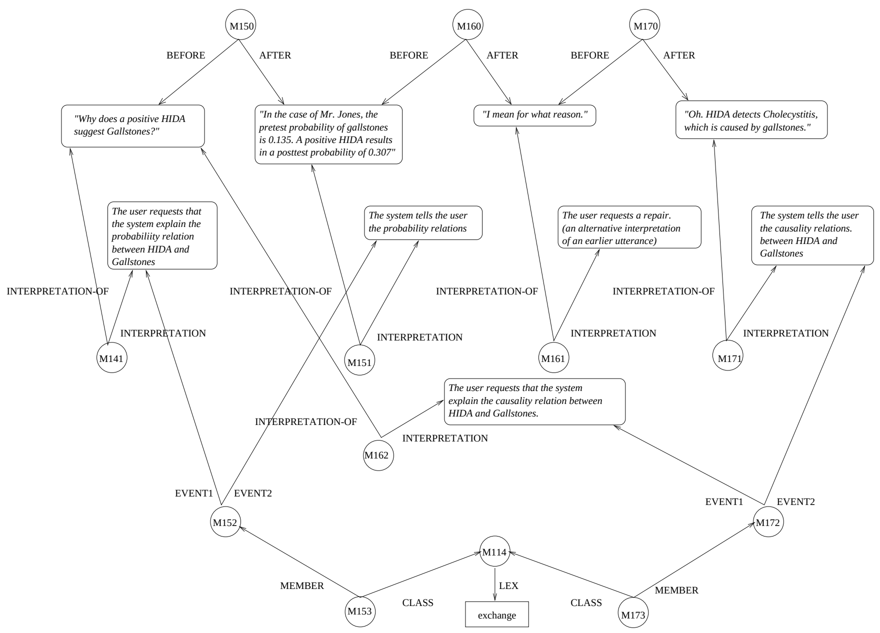

Figure 4: This figure shows (a gloss of) the utterance level representations, the utterance sequence-level representations, (a gloss of) the interpretation level representations, and the exchange-level representations for a dialog containing a repair.

works corresponding to the utterance representations.) The sequence nodes would be constructed using the utterance level representations produced by the parser.

- Nodes `M141`, `M151`, `M161`, and `M171` represent the interpretations that B2 gives to utterances 1–4, at the time that it parsed them. (The second row of boxes gloss the subnetworks corresponding to the interpretations.) The interpretations themselves correspond to representations of the speaker's actions on the domain or the discourse; these interpretations are derived from the utterance representations on the basis of linguistic information, social conventions, and domain-specific plans. For example, to derive the interpretation of `M141`, the system reasons that when the user asks a why-question about an unspecified relation between two concepts in the Bayesian network, then it can be interpreted as a `request` for the system to perform the action `describe-probability-chain` on the two nodes. (Other interpretations are possible, the system need only find one that applies.) In the terminology of the previous section, `M141` is explained as a case of plan-adoption, `M151` is explained as acceptance, `M161` is a self-misunderstanding (of utterance 1), `M171` is a repair, which also acts as the acceptance for the interpretation of `M141`.

- Nodes `M152` and `M172` identify the exchange structure of utterances 1 and 2 (the original interpretation) and the exchange structure of utterances 1 and 4 (the repaired interpretation). When a node is recognized as the acceptance of another node, those two nodes can be taken as a complete exchange. In the figure, `M153` and `M173` are propositions that the two structures, respectively, are indeed exchanges.

The exchange structure is significant because, the system will step back through this structure if it needs to reason about alternative interpretations. The exchange structure indicates how each speaker displayed their understanding of the other's previous utterances. (The utterance sequence will not always provide this information because exchanges can be nested inside each other, *e.g.* to ask a clarifying question.)

The interpretation of `M161` is special, because it neither begins a new exchange nor completes an open one. (This would be determined by its linguistic form and by the expectations created by the previous interaction.) When the system fails to find a domain plan (*e.g.* a request to display part of the network) or a discourse plan (*e.g.* a request to clarify a previous utterance or an answer to a question from the system), then it considers evidence of failure. In this case, the surface form of the utterance suggests looking back in the conversation to consider an alternative domain plan. Finding an utterance that admits an alternative interpretation–corresponding to an alternative domain plan—a new interpretation (`M162`) is constructed, which results in a repair action by the system to accept it. (If there had been no alternative domain plan, then the system would not be able to form any interpretation of utterance 3—a case of non-understanding—and would subsequently ask the user to provide more information about the problem.)

The result of dialog processing is thus a detailed network of propositions that indicates the content of the utterances produced by the system or the user, their role in the interaction, and the system's belief about what has been understood. If necessary, the system will be able to explain why it produced the utterances that it did and recover from situations where communication has failed.

3.2 A Detailed Example with Representations

Computationally, the system processes dialog by parsing communicative acts into mixed-depth representation, the construction of these representations triggers inference to determine an interpretation, and finally the derivation of an interpretation triggers an acting rule that performs an an action that satisfies the user and system intentions.

To illustrate, we will now consider the underlying representations that are used when processing the question: *What is Mary's age?*. The steps that occur in answering this question are:

- The parser produces a mixed-depth representation of the utterance (where the utterance is assigned the discourse entity label `B4`).
- The addition of the mixed-depth representations triggers inference to:
    1. Invoke content interpretation rules, to deduce that age is an attribute of *Mary*, and that the utterance is about an object-property relationship (between *what* and *Mary's age*).
    2. Invoke anaphora interpretation rules to find a known entity named *Mary*.
    3. Invoke pragmatic interpretation rules that derive that the communicative act associated with the utterance is an *askref* and that it initiates a new (question-answer) exchange.

  For efficiency, this reasoning occurs in distinct contexts; interpretation(s) of the utterance are shared in contexts where further interpretation occurs.

- The resulting interpretation of the question triggers an acting rule that answers the question.
- Finally, this leads to a goal whose plan calls for the system to say *42* (the answer).

All interpretation and acting is done with the same representation language, thus a complete record of all of these events is maintained. We now consider this example in more detail, showing many of the representation(s) used.[10]

3.2.1 Parsing, content, and anaphora interpretation

As mentioned above, the question is parsed by a broad-coverage grammar which builds the utterance-level (mixed-depth) representation(s) as shown in Figure 5. For clarity, the semantic networks are shown as simplified feature structures. Propositions are labeled as M*j* and (potential) discourse entities are labeled as B*k*. In Figure 5, three propositions are produced from the initial parse of the question. Proposition `M10` represents the fact that there was an utterance whose label is `B4`, whose form and attitude was interrogative copula, and whose content (`M9`)is some unknown *is* relation between `B2` and `B1`. `B1` corresponds to the pronoun `what` and `B2` to `age`. Proposition `M4` states that `B2` is a member of the class of `age`. Finally, proposition `M5` represents the fact that there is an unknown possessive relationship between `B2` (an age) and `B3` (an entity whose proper name is `Mary`).

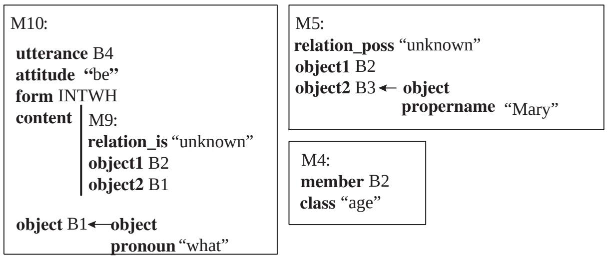

Figure 5: Initial Mixed-Depth Representation of: *What is Mary's age?*

As can be seen from Figure 5, the utterance-level propositions produced by the parser are the weakest possible interpretations of the utterance. Any question of this form would parse into similar utterance-level propositions; the subsequent interpretation(s) would vary.

In the next step of interpretation, `M5` is further interpreted as specifying an attribute (`B2`, *i.e.* `age`) of an object (`B3`, *i.e.* `Mary`). This is a domain-specific interpretation and is deduced by an interpretation rule (not shown here for space reasons). The rule encodes that `age` is an attribute of an entity (and is not, for example, an ownership relation as in `Mary's dog`).

Figure 6 shows the interpretation rule used to deduce a partial interpretation of the utterance `B4`. A partial interpretation of an utterance is a semantic interpretation of the content of the utterance, apart from its communicative (pragmatic) force. This relationship will also be represented

[10] As might be expected, our declarative model uses and creates many knowledge structures. Here we focus on the important ones; at the level of detail we must use in our model, presenting increasingly more of these structures quickly detracts from the understandability of the whole.

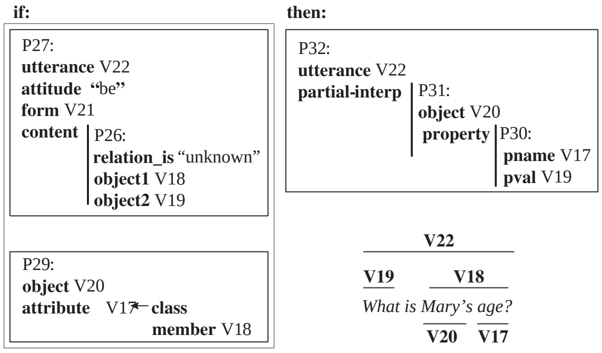

Figure 6: Partial Interpretation Rule for the Utterance B4

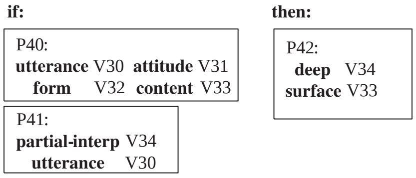

Figure 7: Derivation Rule for Making Explicit the Relation between an Utterance's Content and Partial Interpretation

explicitly as a deep-surface relationship, which is derived using the rule shown in Figure 7.[11] In addition, a separate rule (not shown) will be used to establish an equivalence relationship between `B3` (the Mary mentioned in the utterance) and `B0` (the Mary known to the system).[12] As a result of the rule in Figure 6, the semantic content of the utterance is interpreted as an object-property relationship (pragmatic processing, discussed in the next subsection, will determine that the force is as a particular subclass of question `askref`).

In a rule such as in Figures 6 and 7, variables are labeled as `V`n and, for clarity, the bindings of the variables of the rules are shown relative to the original question in the lower right corner. The `if` part of the rule in Figure 6, has two antecedents: (1) `P27`, requires that there be an copula utterance whose content is an unknown *is* relation between an entity (`V19` *i.e.* `What`) and another entity (`V18`), (2) `P29`, requires that the latter entity (`V18` *i.e.* `age`) is an attribute of another entity (`V20` *i.e.* `Mary`). The consequent of this rule `P32` stipulates that, should the two antecedents hold, then a partial interpretation of the utterance is that `V20` (*i.e.* `Mary`) has a property whose name is `V17` (*i.e.* `age`) and whose value is `V19` (*i.e.* `what`). The rule of Figure 6 allows the interpretation of the mixed-depth representations of Figure 5 as a proposition, which expressed in a logical formula, is *has-property(*`Mary, age, what`*)*

[11] Elements of the deep-surface relation may also be asserted as part of the domain knowledge, to express differences in terminology among utterances of the user, *e.g.* high blood pressure, and concepts in the domain, *e.g.* hypertension.

[12] Currently, the system makes the simplifying assumption that all objects with the same name are the same entity; we are exploring rules for anaphora.

```
if:                  then:
P43:                 P44:
deep    V35          act   | P45:
surface V36                | action   "askref"
                           | dobject  V35
                     plan  | P45:
                           | action   "surface"
                           | content  V36
                           | form     "intwh"
                           | attitude "be"
```

Figure 8: Text Planning Rule

3.2.2 Pragmatic interpretation

After parsing and content interpretation, discussed above, the system will have constructed Levels 1 and 2 of the discourse model and will be ready to consider the construction of the third level. At the third level, there might be several possible interpretations, but only one will be believed (which one is believed will be determined by inferences performed during the construction of the fourth and fifth levels of the discourse model). A communicative action is a possible interpretation of the user's literal action if the system believes that user's action is one of the known ways of performing the communicative act. We consider two rules, shown in Figures 8 and 9 that the system uses to derive a possible interpretation.

The rule in Figure 8 specifies the relationship between an utterance and the way it may be realized as an utterance. In this case, whenever there is a deep-surface relationship between two propositions `V35` and `V36`, that is, `V36` is a representation of how the user might express a proposition and `V35` is a representation of how the system represents the concept in its model of the domain, then an agent (either the system or the user) may perform an `askref`[13] by performing the (linguistic) action called "surface" to output the content `V36` with a surface syntax of "intwh" and attitude "be". We call this type of rule a "text planning rule" because it may be used by the system either to interpret an utterance by the user or to generate an utterance by the system.

Figure 9 is a rule that specifies a possible interpretation of an utterance. It says that if a speaker makes an utterance, and that utterance is part of a plan that accomplishes an action, then an interpretation of the utterance is that the speaker is performing the action. This rule relies on the results of the text planning rule mentioned above, where `P52` is matched against a text plan whose act is the following:

```
(M23 (ACTION "askref")
     (DOBJECT (M24 (OBJECT B0)
                   (PROPERTY (M25 (PNAME "AGE")
                                  (PVAL B1))))))
```

and `P50` is matched against the output of the parser with `form = intwh`, `attitude = be`, and

```
content = (M9 (RELATION_IS "unknown")
```

[13] An `askref` is a type of communicative act that is used to ask for the referent of some expression, akin to asking for the hearer's binding of some variable.

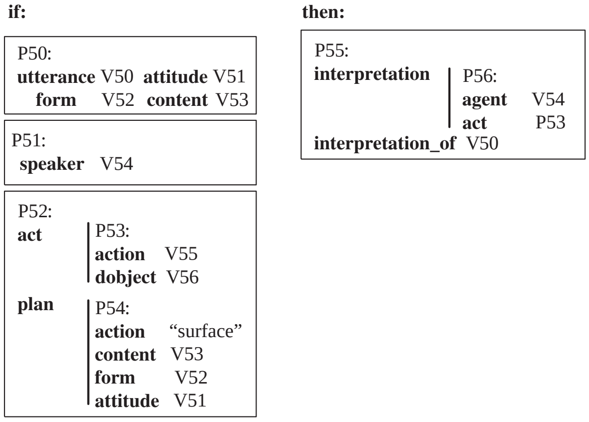

Figure 9: Pragmatic Interpretation Rule to Identify Communicative Act

```
(OBJECT1 B2)
(OBJECT2 B1))
```

The final interpretation of the original utterance `B4` is shown in Figure 10. `M22` is the (level 3) interpretation, namely that the user is performing an `askref` on *what is Mary's age* (`M24`) and the `system`. More concisely, the system has interpreted the original utterance *what is Mary's age* as the user asking the system: *what is Mary's age?*.

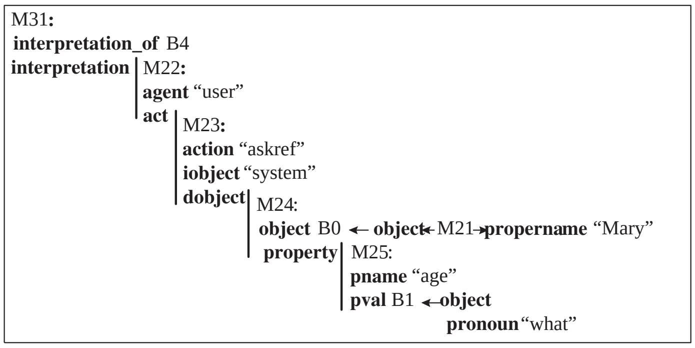

Figure 10: An Interpretation of the Utterance *What is Mary's age?*

At this point our discussion of interpretation is complete (we will not consider, here, the possibility of misunderstandings or arguments and hence can safely ignore the construction of the fourth and fifth levels of the discourse model). Next, we will consider response generation, as it illustrates the link between inference and action in the underlying knowledge base.

3.2.3 Answering the question

The assertion of an interpretation of the utterance as an `askref` and its acceptance as a coherent continuation of the dialog leads to an action by the system to answer the question.

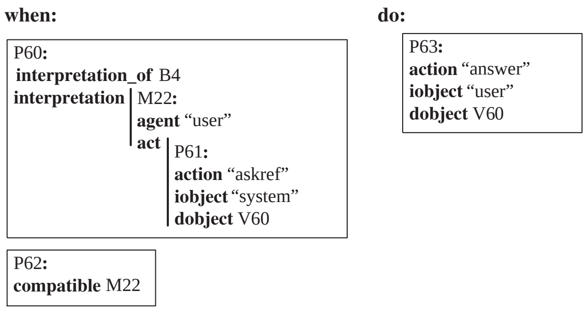

Figure 11: An Acting Rule for Responding to a Question

Figure 11 is an acting rule (by contrast to the inference rules discussed previously), which glosses as: if the user asks the system a question (`P60`) and the system believes that it is compatible with the dialog to answer the question (`P62`) then do the action of answering the question.[14] To achieve the latter action (`answer`) the system uses a plan in which the system deduces possible values of `Mary's age` by replacing the `what` in the question with a variable, and responds by saying the answer (if any). This answer is realized as a natural language expression using our real-time generator, YAG (discussed in Section 5).

4 Interpreting Exchanges

We have been working on computational methods for recognizing and repairing misunderstandings (RRM) and for detecting an rebutting arguments (ARGUER). The current model extends the notion developed in [McRoy and Hirst, 1995, McRoy, 1998] that interpretation of communicative acts is driven by expectations and these expectations are based on a combination of knowledge including linguistic, intentional, and social conventions. Reasoning about misunderstanding and about argument involves non-monotonic reasoning, that is, plausible, but possibly unsound inferences that we make based on the information that is available at the time and which are revised when contradictory evidence is received. Both the current and earlier work use schema to specify these plausible rules.

For example, in ARGUER [Restificar *et al.*, 1999a, Restificar *et al.*, 1999b] argument schemata are used to characterize important patterns of argument that are used to establish whether propositions expressed by utterances *support* or *attack* other propositions. For example, if agent A's interpretation of agent B's utterance is inconsistent with a consequence of an interpretation of one of A's previous utterances, then the B's utterance can be considered an attack to A's. (The schema for this rule is shown

[14]Compatibility is a notion that is related to the coherence of dialog and the expression of reflexive intentions (see [McRoy and Hirst, 1995]). In this case, a question expresses the lack of knowledge about some referent and an intention to know it. This interpretation of the original utterance is compatible because the neither the user nor the system has indicated that the user already knows the answer—which might be the case, if, for example, the system had previously answered the question. If one interpretation is incompatible, another response might be possible (*e.g.* the generation of a repair), but that possibility is beyond the scope of this paper.

S1: Uncontrolled high blood pressure can lead to heart attack.

U1: But I feel healthy.

S2: Unfortunately, there are no signs or symptoms that tell whether your blood pressure is elevated.

U2: Ok, I'll get it checked.

Attacking a consequence:
Let s_1, s_2 be speakers; and u_1, u_2 be utterances by s_1 and s_2, respectively, and u_1 precedes u_2.
If `interp_of`(s_1, u_1, `implies(`a_1, a_2`)`),
and `interp_of(`s_1, u_2, a_3`)`,
and `implies(`a_1, `not(`a_3`))`
Then `attacks(`s_1, a_3, `implies(`a_1, a_2`))`

s_1 = **system**, s_2 = **user**, u_1 = **S1**, u_2 = **U1**
a_1 = User might have uncontrolled high blood pressure,
a_2 = User might have a heart attack
`implies(`a_1, a_2`)` = Uncontrolled HBP can lead to heart attack (**S1**)
a_3 = User is healthy (**U1**)

Figure 12: A Schema for Detecting an Argument

in Figure 12.) During interpretation, these patterns are instantiated by propositions expressed by the agents during a dialog, as well as related beliefs that the agents might hold. (Argument schema can also be used to generate a rebuttal.) The result of argument detection is that new attack and support nodes will be added to the fifth level of the discourse model. Inferences about misunderstanding are handled similarly, using schematic rules derived from those in [McRoy and Hirst, 1995].

RRM addresses possible misunderstandings (as well as expected interpretations), while respecting the time-constraints of Intelligent Dialog Systems, by combining intentional and social accounts of interaction to capture the expectations that help constrain interpretation. From intentional accounts, it includes the notion speakers use their beliefs, goals, and expectations to decide what to say and, when they interpret an utterance, speakers identify goals that might account for it. From social accounts, it includes the notion that turns of dialog form adjacency pairs [Schegloff, 1992] and that the recognition and repair of misunderstanding are normal activities, driven by the formal expectations that are established by conventions such as adjacency pairs. An action is considered a manifestation of misunderstanding if there is no coherent link apparent and there is a reason for supposing that misunderstanding has occurred.

Figure 13 shows a schema for detecting a misunderstanding (in particular a misunderstanding that is detected by the agent who has misunderstood, *i.e.* a *self-misunderstanding* [Schegloff, 1992]). Figure 14 shows a schema for repairing a misunderstanding (in particular, for making a repair of a self-misunderstanding after hearing an unexpected reply, *i.e.* a *fourth-turn repair* [Schegloff, 1992]). (For clarity of presentation, we are not showing the detailed representations corresponding to these schema; they are similar to those of section 3.2.) ColTrain uses these RRM schemata to handle misunderstandings.

Figure 13 shows the dialog discussed in Section 3.1, the schema that is used to detect a misunderstanding and the bindings that are used when matching the schema against the knowledge base. The system's interpretations of the user's utterances U1 and U2 are not compatible. Additionally, there is an alternative interpretation of U1. This allows the system to de-

U1: Why does a positive HIDA suggest gallstones?
S1: In the case of Mr. Jones, the pretest probability of gallstones is 0.135. A positive HIDA results in a posttest probability of 0.307.
U2: I mean for what reason.
S2: Oh. HIDA detects cholecystitis which is caused by gallstones.

Self-misunderstanding:
Let s_1, s_2 be speakers; and u_1, u_2 be utterances by s_1 where u_1 precedes u_2.
If **interp_of**(s_2, u_1, a_O)
and **interp_of**(s_2, u_2, a_N)
and a_N is not compatible with a_O,
and **textplan**(u_1, a_O) and **textplan**(u_1, a_I)
Then **disbelieve** (**interp_of**(s_2, u_1, a_O))
interp_of(s_2, u_1, a_I)
mistake (s_2, a_O, a_I)

s_1 = **user**, s_2 = **system**, u_1 = **U1**, u_2 = **U2**
a_O = User requested probability relations: **interp_of**(**system**, **U1**, a_O).
a_N = User requested a repair: **interp_of**(**system**, **U2**, a_N).
a_I = User requested causality relations: **interp_of**(**system**, **U1**, a_I).

Figure 13: A Schema for Detecting Misunderstanding

U1: Why does a positive HIDA suggest gallstones?
S1: In the case of Mr. Jones, the pretest probability of gallstones is 0.135. A positive HIDA results in a posttest probability of 0.307.
U2: I mean for what reason.
S2: Oh. HIDA detects cholecystitis which is caused by gallstones.

Fourth-turn repair:
Let s_1, s_2 be speakers;
If **mistake** (s_2, a_O, a_I),
and **expect** (s_1, s_2, a_I, a_R),
and a_R is compatible with the dialog,
Then **Do**(s_2, **repair** (a_R))

s_1 = **user**,
s_2 = **system**,
a_O = User requested probability relations: **interp_of**(**system**, **U1**, a_O).
a_I = User requested causality relations: **interp_of**(**system**, **U1**, a_I).
a_R = System informs user of causality relations: **interp_of**(**system**, **S2**, a_R).

Figure 14: A Schema for Repairing Misunderstanding

cide that its original interpretation of U1 was incorrect. In summary, this schema allows the detection of the system's misunderstanding.

Figure 14 shows the same dialog, the schema that is used to repair a misunderstanding, and the bindings that are used when matching the schema against the knowledge base. This schema allows the system to perform a repair. In this case, the system has detected that it was a mistake to provide the probability relations and that the user wanted the causality relations. The repair is the (conventionally expected) reply to a request for causality information.

5 Generating Natural Language Responses in Real-Time

YAG (Yet Another Generator) is our system for generating natural language in real-time, as required for an Intelligent Dialog System. YAG performs realization of text—it maps representations of content onto sequences of words. YAG combines a template-based approach for the representation of

Pluto is a dog.

```
(((M2 (CLASS "dog")
      (MEMBER B2))
  (M5 (OBJECT B2)
       (PROPERNAME "Pluto")))
 ((form decl)
  (attitude be) )
)
```

Figure 15: Example input to YAG

text with knowledge-based methods for representing content (*i.e.*, inputs can be concepts or mixed-depth propositions along with optional annotations to specify syntactic constraints).

YAG differs from previous approaches to realization in two respects: first its functionality is geared toward use by knowledge based systems. It accepts under-specified syntactic and semantic representations. The latter fits well with the semantic representations that we use for reasoning about dialog, in contrast to systems that expect a nearly complete description of syntax. The second key distinction is that our template based approach is fast and robust—so it can keep up with real-time dialog. Unlike a unifier or systemic grammar, realization time does not depend on size of the grammar because the application or the knowledge engineer specifies which template to use. (If the application does not specify a template, then the realizer will retrieve it from a hash-table that includes an entry for each type of proposition; the entry includes the name of the template to use and the mapping from arguments of the predicates to slots in the template.) Thus, the speed of YAG depends on the complexity of the particular template that is selected and not on the number of template types. Short utterances are always produced more quickly than longer ones.

Templates in YAG are declarative representations of text structure. Each form in a YAG template is a rule that expresses how a surface constituent should be realized, given features present in the input. YAG can accept feature structures directly, or can map propositions represented as SNePS case frames onto their feature structure equivalents and select an appropriate template for their realization.

Inputs to YAG may include multiple propositions as well as a list of control features, as shown in Figure 5. When processing this input, YAG treats the first proposition as the primary proposition to be realized. YAG will map the `MEMBER-CLASS` proposition to the template shown in Figure 16. The control features, **form** = `decl` and **attitude** = `be`, are also used in selecting the template. (If the form had been `interrogative`, a template for generating a yes-no question would have been used.)

Prior to realization, a mapping from each type of proposition to the name of the corresponding template in a mapping table is specified. (This is the primary task in constructing a new knowledge representation realization component for other knowledge representations.) Each mapping entry provides a declarative specification for constructing a feature structure from

```
((EVAL member)
 (TEMPLATE verb-form
          ((process "be")
           (person (member person))
           (number (member number))
           (gender (member gender))) )
 (EVAL class)
 (PUNC "." left) )
```

Figure 16: A `member-class` Template.

```
(member-class
 ((decl
    (be (template member-class)
        (slot-map ((class class)
                   (member member)) )
        (feature nil)
    )
 )))
```

Figure 17: A Simplified Mapping Entry for the `member-class` Case Frame.

the propositions and control features. A sample entry of a mapping table is given in Figure 17.

6 Conclusions

We have proposed a computational model of dialog based on a uniform, declarative, representation language. Interpretation in this model is based on reasoning. We have specified a multi-level architecture for our model. The different levels of the model represent the content, structure, and sequence of dialog, along with interpretations of these representations. This architecture also incorporates mixed-depth representations which allow for the opportunistic interpretation of vaguely articulated or fragmentary utterances. We have illustrated, in detail, the use of reasoning in this model by considering the end-to-end interpretation of a fragment of dialog. The practicability of the model arises from our choices in structuring the complex knowledge used in interpretation, in such a way that reasoning uses only salient information. The model is flexible, extendible, and reusable.

Acknowledgements

This work has been supported by the National Science Foundation, under grants IRI-9701617 and IRI-9523666 and by a gift from Intel Corporation. The authors wish to thank Richard McConachy, Ingrid Zuckerman and the anonymous reviewers for their thoughtful comments.

References

[Austin, 1962] John L. Austin. *How to Do Things with Words.* Oxford University Press, London, England, 1962. Reprinted 1975.

[Bordegoni *et al.*, 1997] M. Bordegoni, G.Faconti, M. T. Maybury, T. Rist, S. Ruggieri, P. Trahanias, and M. Wilson. A standard reference model for intelligent multimedia presentation systems. In *Proceedings of the IJCAI '97 Workshop on Intelligent Multimodal Systems*, 1997.

[Clark and Schaefer, 1989] Herbert H. Clark and Edward F. Schaefer. Contributing to discourse. *Cognitive Science*, 13:259–294, 1989.

[Fikes and Nilsson, 1971] R. E. Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[Grice, 1975] H. P. Grice. Logic and conversation. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics 3: Speech Acts.* Academic Press, New York, 1975.

[Grosz and Sidner, 1986] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linquistics*, 12, 1986.

[Hirst and Ryan, 1992] Graeme Hirst and Mark Ryan. Mixed-depth representations for natural language text. In Paul Jacobs, editor, *Text-Based Intelligent Systems.* Lawrence Erlbaum Associates, 1992.

[Lambert and Carberry, 1991] Lynn Lambert and Sandra Carberry. A tripartite plan-based model of dialogue. In *29th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 47–54, Berkeley, CA, 1991.

[McRoy and Hirst, 1993] Susan W. McRoy and Graeme Hirst. Abductive explanation of dialogue misunderstandings. In *6th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 277–286, Utrecht, The Netherlands, 1993.

[McRoy and Hirst, 1995] Susan W. McRoy and Graeme Hirst. The repair of speech act misunderstandings by abductive inference. *Computational Linguistics*, 21(4):435–478, December 1995.

[McRoy *et al.*, 1997] Susan McRoy, Susan Haller, and Syed Ali. Uniform knowledge representation for nlp in the b2 system. *Natural Language Engineering*, 3(2):123–145, 1997.

[McRoy *et al.*, 1998a] Susan W. McRoy, Susan M. Haller, and Syed S. Ali. Mixed Depth Representations for Dialog Processing. In *Proceedings of Cognitive Science '98*, pages 687–692. Lawrence Erlbaum Associates, 1998.

[McRoy *et al.*, 1998b] Susan W. McRoy, Alfredo Liu-Perez, and Syed S. Ali. Interactive Computerized Health Care Education. *Journal of the American Medical Informatics Association*, 5(4):76–104, 1998.

[McRoy *et al.*, 2000] Susan W. McRoy, Syed S. Ali, and Susan M. Haller. Mixed depth representations for dialog processing. In Lucja Iwanska and Stuart C. Shapiro, editors, *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language.* AAAI Press/The MIT Press, 2000.

[McRoy, 1995] Susan W. McRoy. Misunderstanding and the negotiation of meaning. *Knowledge-based Systems*, 8(2–3):126–134, 1995.

[McRoy, 1998] Susan McRoy. Achieving robust human-computer communication. *International Journal of Human-Computer Studies*, 48:681–704, 1998.

[Moore and Paris, 1993] Johanna Moore and Cécile Paris. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics*, 19(4):651–695, 1993.

[Neal *et al.*, 1989] J. G. Neal, C. Y. Thielman, Z. Dobes, Haller S. M., and S. C. Shapiro. Natural language with integrated deictic and graphic gestures. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 410–423. Morgan Kaufmann, 1989. Reprinted in Readings in Intelligent User Interfaces. M. Maybury and W. Wahlster (eds), 1998.

[Restificar *et al.*, 1999a] Angelo C. Restificar, Syed S. Ali, and Susan W. McRoy. ARGUER: Using Argument Schemas for Argument Detection and Rebuttal in Dialogs. In *Proceedings of User Modeling 1999.* Kluwer, 1999.

[Restificar *et al.*, 1999b] Angelo C. Restificar, Syed S. Ali, and Susan W. McRoy. Argument Detection and Rebuttal in Dialog. In *Proceedings of Cognitive Science 1999*, 1999.

[Sacerdoti, 1977] Earl D. Sacerdoti. *A Structure for Plans and Behavior.* American Elsevier, New York, 1977.

[Schegloff, 1992] Emanuel A. Schegloff. Repair after next turn: The last structurally provided defense of intersubjectivity in conversation. *American Journal of Sociology*, 97(5):1295–1345, 1992.

[Searle, 1969] John Searle. *Speech Acts.* Cambridge University Press, Cambridge, England, 1969.

[Searle, 1979] John Searle. A taxonomy of illocutionary acts. In his *Expression and Meaning: Studies in the Theory of Speech Acts*, pages 1–29. Cambridge University Press, London, 1979. Previously published as "A Classification of Illocutionary Acts", in *Language and Society*, 1975.

[Shapiro and Rapaport, 1992] Stuart C. Shapiro and William J. Rapaport. The SNePS family. *Computers & Mathematics with Applications*, 23(2–5), 1992.

[Shapiro *et al.*, 1994] Stuart Shapiro, William Rapaport, Sung-Hye Cho, Joongmin Choi, Elissa Feit, Susan Haller, Jason Kankiewicz, and Deepak Kumar. A dictionary of SNePS case frames, 1994.

[Shapiro, 2000] Stuart C. Shapiro. SNePS: A logic for natural language understanding and commonsense reasoning. In Lucja Iwanska and Stuart C. Shapiro, editors, *Natural Language Processing and Knowledge Representation: Language for Knowledge and Know ledge for Language*, pages 175–195. AAAI Press/The MIT Press, 2000.

[Smith and Hipp, 1994] R Smith and D Hipp. *Spoken Natural Language Dialog Systems: A Practical Approach.* Oxford University Press, 1994.

[Traum *et al.*, 1996] David R. Traum, Lenhart K. Schubert, Massimo Poesio, Nathaniel G. Martin, Marc N. Light, Chung Hee Hwang, Peter A. Heeman, George M. Ferguson, and James F. Allen. Knowledge representation in the trains-93 conversation system. *International Journal of Expert Systems*, 9(1):173–223, 1996.