From: AAAI Technical Report FS-93-05. Compilation copyright © 1993, AAAI (www.aaai.org). All rights reserved.

Collaboration in an Interactive Model of Plan Explanation

Susan M. Haller
Department of Computer Science
State University of New York at Buffalo
haller@cs.buffalo.edu

Abstract

A viable model of interactive discourse must explain how interlocutors share conversational control to construct the discourse, and recognize the contributions of others to it. In my work I focus on one mode of collaboration. I am implementing a system called the Interactive Disourse Planner (IDP) that plans text to describe and justify domain plans interactively. As a testbed for my model, IDP discusses driving routes as the domain plans. In this collaborative mode, the system is the primary speaker and the user is the primary listener. IDP is responsible for planning text to describe and/or justify a domain plan, and the user is responsible for providing feedback that lets the system know how to extend the discussion in a way that is helpful.

1 Introduction

A viable model of interactive discourse must explain how interlocutors share conversational control to construct the discourse, and recognize the contributions of others to it. In my work I focus on one mode of collaboration. I am implementing a system called the Interactive Disourse Planner (IDP) that plans text to describe and justify domain plans interactively [7]. As a testbed for my model, IDP discusses driving routes as the domain plans. In this collaborative mode, the system is the primary speaker (PS) and the user is the primary listener (PL). The PS is responsible for planning text to describe and/or justify a domain plan, and the PL is responsible for providing feedback that lets the PS know how to extend the discussion in a way that is helpful.

One insight that this work contributes is that attempts at designing "the collaborative discourse model" may be misguided. A better approach may be to identify the various modes of collaboration that communicators use, and build models for each one. Using this approach, we can make assumptions that lead to simpler, computationally tractable systems that provide for efficient and effective interaction.

In the next section, I describe the collaborative mode that IDP operates in. Next, I compare my approach to two others used for interactive generation. I follow this with a discussion of the theory behind processing that includes some IDP demonstrations. Next, I discuss how IDP addresses issues relevant to human-computer collaboration: sharing responsibility, knowledge requirements for collaboration, and interaction management. This is followed by a brief description of the implementation and its status.

2 The Collaborative Mode

In the role of the PS, IDP makes its own intentions clear, but it does so while trying to accommodate the informational needs of the PL. Because the PL is seeking plan advice, one assumption made in the IDP model is that the PL automatically believes what the PS has to say. This simplifies IDP's reasoning about the PL's beliefs and reduces informing acts to system primitives. Another simplifying assumption is that when the PS speaks, the PL knows the PS's intentions. This is based on the idea that discourse structures, and what they are used for are knowledge that language users share. Therefore, the order and proximity of clauses that the PS presents reveals his plan for the discourse, and thereby conveys his underlying intent.

In highly interactive settings, people are often called on to analyze and respond to vaguely articulated questions like *Why?* and *What?*, and ill-formed queries like *Why take Maple?* and *Why not Sheridan Drive?*. A third assumption made in the IDP model is that these kinds of questions are used by the PL to tell the PS efficiently what part of the text plan has failed, and how it can be replanned to succeed.

In the IDP model, the PS's *intentions* are synonymous with his discourse goals, and the PS's *discourse goals* are unachieved goals that have to do with the attitude or abilities of the PL. In the role of the PS, IDP can hold one or both of the following intentions: to have the PL adopt a domain plan, to have the PL be able to execute a domain plan. IDP plans text to try to achieve its intentions, and it interprets the PL's feedback in the context of three types of knowledge: its beliefs about the PL's knowledge of the domain, their shared knowledge of the text plan that the system has constructed so far, and their common, language-based knowledge of the ways it could be extended.

3 Related Work

In comparison to other interactive generators, the IDP approach to representing the discourse context and analyzing feedback is similar to that used in the Explanatory Discourse Generator (EDGE) [4], and unlike Moore and Swartout's treatment in the EES Text Planner [16]. EDGE formulates and represents a single text plan, and uses it as the discourse context when analyzing feedback. It uses interaction with the user to make decisions on restructuring and pruning parts of the plan yet to be executed. In the EDGE system, when the text plan fails as indicated by a question, the system examines the plan to see what assumptions it made while expanding its goals, and what alternative plans it could have used. It then selects one to expand and adds it to the discourse plan.

In contrast, the EES Text Planner does not represent the discourse context with a single text plan. The analyzer treats questions like *Why?* and *What?* using a stack of text plans from previous exchanges. It applies system rules to generate a set of likely interpretations for each utterance, and then it uses search heuristics to select one of them.

IDP is like the EES Text Planner in one respect: IDP plans in reaction to user feedback. The EES Text Planner formulates text plans only as needed and then saves the used text plan on a stack with others. For plan discussions like giving route advice, there is no preset agenda. IDP must describe or justify a selected driving route until the PL's feedback indicates that she is satisfied. In contrast, EDGE plans tutorials. In teaching situations, the PS has a prespecified set of goals for the interaction. Therefore, EDGE formulates its text plan in advance, and then incrementally executes it.

The IDP model incorporates features of both systems. IDP plans only as needed in reaction to feedback. However, each plan that it formulates is incorporated into a single text plan that is used as the discourse context. This makes if possible for IDP to analyze feedback efficiently and generate concise replies.

4 The Theory Behind Processing

Following Carberry, when the PL knows the PS's intention, the PL has expectations for what will follow [3]. Motivated by Grice's cooperative principle and the maxims of Relevance and Quantity [5], IDP analyzes the PL's feedback using its discourse goal, its executed text plan, and the possible expansions of it as the discourse context. The discourse goal and the text plan are respectively, the PS's intent and how he has attempted to achieve his intent so far. The possible expansions to the text plan are the expectations that the PL has for how the PS will continue if he is required to do so. Because the PL is cooperating, the IDP model assumes that the PL's feedback addresses the system's intent by indicating which text plan expansion the PL would like the PS to use.

To analyze *Why?*, IDP uses the most recently executed (and focussed) portion of its text plan, possible expansions of it, and a set of propositions that comprise reasoning that is closely linked to the last proposition that IDP expressed. Of these propositions, ones that the PL is not known to know are called the *localized unknowns*. Processing proceeds by trying to find a way of expanding the focussed portion of the text plan that lets the PL know one of the localized unknowns. This approach produces results illustrated by the following IDP demonstration:[1]

User:	`Should I take Maple or Sheridan to go to the Eastern Hills Mall?`
System:	`you should take maple.`
User:	`Why?`
System:	`taking maple avoids heavy traffic.`
User:	`Why?`
System:	`since taking maple there are fewer businesses than taking sheridan,`

The demonstration also illustrates that by using its text plan as the discourse context, IDP can produce concise text that does not repeat old information along with the new. These types of interactions are coherent and collaborative because both participants know how their contributions fit in. This discourse property is what Levelt refers to as *discourse deixis* [12].

In comparison, current systems that plan reactively produce text that is cluttered with information from a previous exchange. This point is illustrated by the following fabricated interaction:

System:	You should take Maple.
User:	Why?
System:	Take Maple because it avoids heavy traffic.
User:	Why?
System:	Taking Maple avoids heavy traffic because taking Maple there are fewer businesses than taking Sheridan.
User:	Why?
System:	There are fewer businesses taking Maple than taking Sheridan because...

Without access to a single, evolving text plan, these systems tend to repeat information to relate new information to old in the text plan that is formulated for each exchange. Over an extended discussion, this can make explanations laborious, and difficult to follow.

Consistent with the Gricean Maxims, IDP processes feedback with the idea that the more the PL says, the more she feels she needs to say for the system to identify the continuation that is sought. Therefore, IDP uses any additional information that the PL supplies to try to recognize a text plan expansion, even though it may not be an immediate continuation of what the system said

[1] Interactions in roman font have been created to illustrate specific points. Actual interactions with IDP are shown in `courier` font.

last. The following demonstration illustrates this type of processing:

User:	`Should I take Maple or Sheridan to go to the Eastern Hills Mall?`
System:	`take maple.`
User:	`Why not go on Sheridan?`
System:	`you could take sheridan however, taking maple avoids heavy traffic.`

In this interaction, the PL's feedback indicates that she would like the system's response to include information about the feasibility of an alternative route. IDP uses the mentioned act, *go* ing *on Sheridan*, to identify a discourse entity and a text plan expansion that uses it. Therefore, IDP continues to address its own intentions while providing the implicitly requested information.

5 Human-Computer Collaboration

I developed the IDP model by taking one perspective on discourse collaboration. IDP demonstrates that human-computer collaboration might be dealt with effectively and efficiently by first identifying the types of collaboration (*collaborative modes*) that a computational system is expected to participate in.

5.1 Sharing Responsibility and Knowledge Requirements for Effective Collaboration

A collaborative mode is defined by what the various participants have responsibility for in that mode. For a given mode, if we define the computational system's responsibilities, and delegate other responsibilities to the user, we can make appropriate assumptions and identify the knowledge requirements. For example, during interactive generation the system's role is active and the user's role is passive. A computational system has responsibility for planning the text to realize and achieve its own intentions. Recognizing the system's intention, the user is responsible for providing feedback that lets the system know how to plan additional text that achieves the system's intention while satisfying the user's informational needs. In this mode, the computational system's text plan is the relevant discourse context.

In contrast, during interactive plan recognition the user controls the conversation, and the system's role is passive. The system has responsibility for trying to recognize the user's domain plan from her utterances, and thereby providing a helpful answer. In this mode, it is the set of possible domain plans that the user could be considering that comprise the relevant discourse context. Researchers in plan recognition have developed analysis techniques that track the user's discourse focus to piece together the user's unstated plan [1, 2]. They have also developed tools for recognizing the user's domain plans by first recognizing her discourse plans [13, 3] and her plan construction plans [17].

The point is that when the system's responsibility shifts, so does the knowledge that is used as the context for analyzing user contributions. Defining collaborative modes in terms of system responsibility determines which assumptions are appropriate, and what knowledge is relevant. This leaves open the question of how responsibility shifts, that is, *how* and *why* the system moves from one mode to another. With respect to *how*, one possibility is that the knowledge requirements for each mode are really highlighted aspects of an overarching model. As the system moves from one collaborative mode to another, the overarching model would need to be updated.

5.2 Managing Human-Computer Interaction

The question of *why* the system moves from one mode to another is beyond the scope of this work. However, the IDP model does demonstrate that shifts in discourse initiative can be detected from a single collaborative mode. The following interaction demonstrates one way that IDP recognizes user-imposed digressions:

User:	`Should I take Maple or Sheridan to go to the Eastern Hills Mall?`
System:	`take maple.`
User:	`Why should I take Maple?`
System:	`taking maple avoids heavy traffic.`
User:	`Why?`
System:	`taking maple there are fewer businesses than taking sheridan.`
User:	`Why is there heavy traffic now?`
System:	`since now is rush-hour, as I was saying, taking maple is a plan for avoiding heavy traffic.`

The user's third question relies on an inference that she made from the proposition that the system conveyed in its second response. IDP uses its text plan to detect that this question, unlike the user's first two questions, initiates a digression. A digression is one of three types of interruptions that Grosz and Sidner distinguish [6]. IDP makes this determination when it cannot find a way of expanding the focussed portion of its text plan to answer it.

Since I do not have a theory of managing initiative, the question of how to decide if digressions should be allowed, and for how long, is not addressed. When the system detects a user-imposed digression, it answers the question, and then it immediately shifts the discourse back to its own intention. In this demonstration, that intention is to have the user adopt the plan of taking the Maple Road route. As demonstrated by the last two lines of IDP's final response, it can use its own text plan as content to do this. This is possible because the system's text plan is represented uniformly with the domain plans that are under discussion.

IDP can distinguish digressive questions from ones that are incoherent in the discourse context:

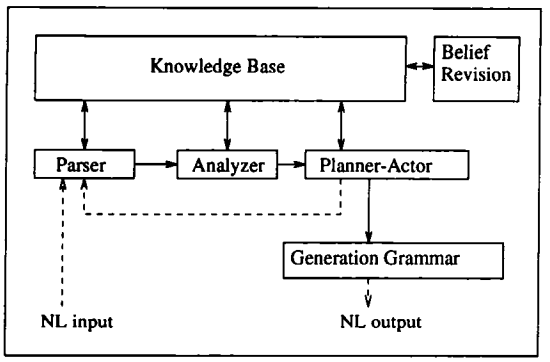

Figure 1: System Architecture

User: `Should I take Maple or Sheridan to go to the Eastern Hills Mall?`
System: `take maple.`
User: `Why is there heavy traffic now?`
System: `huh?`

In this example, IDP cannot find any way to associate the question's answer with the existing text plan, therefore, IDP signals incoherency by responding with *huh?*. In the previous example, IDP answered this question, however the way that the system had to extend its text plan to incorporate its response indicated that the question was digressive. See [8] for details.

6 The System Overview

6.1 Architecture

There is a one-way flow of control though three of the five components of IDP: the parser, the analyzer, and the planner-actor (Figure 1). During a complete user-system exchange, this flow starts with, and cycles back to, the parsing component. After selecting and structuring content, the planner-actor sends a text message to a fourth component, the generation grammar written in the Generalized Augmented Transition Network (GATN) formalism [18]. The same formalism has been used for the parser. Finally, these four components all have access to a fifth component, a knowledge base, which they consult to find or deduce information.

6.2 The Knowledge Base

In the knowledge base, the several sources of information that IDP needs to analyze and plan the text are all represented uniformly using the Semantic Network Processing System (SNePS) [20, 19]. This includes knowledge of the text plan operators, the domain plans, entities in the domain, the user model, the text plan executed so far, and rules for reasoning about all of the above.

6.3 The Planner-Actor

The planning-acting component is based on the SNePS Actor [11]. The SNePS Actor models a cognitive agent operating in a single-agent world. It integrates inference and acting by representing beliefs, plans, and acts as structured intensional entities in the network formalism. The agent has access to its concepts and beliefs about the world as SNePS structured individuals and propositions. The agent's concepts include plans, acts, and goals, and some of the agent's propositional beliefs are about how these concepts relate to one another. That is, the agent has propositional beliefs about plans for goals, and the preconditions, effects, and plan expansions of various acts. Because the agent's world and planning knowledge is represented uniformly, the agent can discuss, reason about, formulate, and also execute its plans [10].

6.4 The Domain

Based on the TOUR model [9], the various driving routes that IDP can discuss are represented as plans that are composed of two types of actions: *going* and *turning*. The domain plans are represented at various levels of detail and, as conceptual entities, can have properties. Whenever the system reasons about the domain, the reasoning that leads to deductions is recorded in the knowledge base along with the deductions themselves. This is available as content for explanations.

6.5 The Text Plan Operators

Most of IDP's text plan operators (TP-operators) are based on Rhetorical Structure Theory (RST) [14]. In the SNePS Actor formalism, plan operators are written as rule propositions that state what consequents can be deduced from a set of antecedents. Figure 2 shows a *text-plan operator* (TP-operator) for the *motivate* act. [2] In the formalism, an *act* decomposes into one or more structures of other acts called *plans*. IDP instantiates plans, preconditions, and effects for a given act by satisfying a rule's antecedents. These are the constraints on the plan, and the process of constraint satisfaction selects new content for the text. For TP-operators that are based on rhetorical relations, this new content is a satellite proposition that is appropriate for the relation and a given nuclear proposition. The TP-operator in Figure 2 states that if there is a domain goal-act ?g that is enacted by a plan ?p, and a secondary goal-act ?g2 that is also enacted by plan ?p, then a plan for the act of motivating the user to do ?p is a sequence of four other acts.

6.6 The Kinds of Text Plans

Mann and Thompson describe a two-way division of the rhetorical relations. Each *presentational relation* relates two text spans for the purpose of increasing an inclination in the hearer. In contrast, a *subject-matter* relation is used by a speaker to inform the hearer of the rhetorical relation itself.

I use the above classification of rhetorical relations to distinguish two kinds of text plans: *discourse text plans* (DTPs), and *content-selection text plans* (CTPs). In the

[2] Arguments enclosed in braces,{...}, are unordered set arguments.

```
ANT-CSQNT(  { GOAL-ACT( ?g ),
              ACT-PLAN( ?g, ?p),
              SECONDARY-GOAL-ACT( ?g2 ),
              ACT-PLAN( ?g2, ?p) },

            ACT-PLAN(motivate(user, DO(user, ?p)),
                        snsequence( advise (user, DO(user, ?p)),
                                    circumstantiate (ACT-PLAN(?g2, ?p)),
                                    say  ( ACT -PLAN( ?g2, ?p)),
                                    restate  ( ACT -PLAN( ?g2, ?p)))))
```

Figure 2: A TP-operator for Motivate

```
ACT-PLAN(   motivate  (user,
                      DO(user, *maple-plan)),

            snsequence  ( advise (user, DO(user, *maple-plan)),
                          circumstantiate (ACT-PLAN (avoid(heavy-traffic),
                                                     *maple-plan)),

                          say  ( ACT -PLAN( avoid( heavy-traffic),
                                            *maple-plan)),

                          restate  ( ACT -PLAN( avoid( heavy-traffic),
                                                *maple-plan)))))
(a)
-------------------------------------------------------------------------
ACT-PLAN(   circumstantiate  ( ACT-PLAN( avoid( heavy-traffic),
                                         *maple-plan)),

            snsequence  ( say ( OBJECT-PROPERTY( *maple-plan,
                                                 fewer-businesses)),

                          forget ( CONTENT-GOAL ( KNOW( user,
                                                        O'JT-PR'TY(...)))))))
(b)
```

Figure 3: (a) A DTP for Motivate (b) A CTP for Circumstantiate

IDP system, DTPs are used to attempt and reattempt the achievement of the system's discourse goals. Since these goals have to do with affecting the attitudes and abilities of the user, DTPs are based on speech acts and presentational rhetorical relations. The DTPs describe how to try to achieve discourse goals by selecting some minimal text content. IDP can augment this content with additional content by using one or more CTPs.

Figure 3(a) shows a DTP for *motivate*. Each act expands into one or more plans. The motivate act takes nuclear information (the user taking the Maple Road route) as one of its arguments. The plan expansion for motivate incorporates new, satellite information (taking Maple Road avoids heavy traffic) which is the motivation. This is how the IDP model separates old information from new.

In Figure 3(a), the DTP for motivate includes references to two additional CTPs that are potential growth points: *circumstantiate* and *restate*. A plan for circumstantiate is given in Figure 3(b). Since CTPs are not executed to affect the user in any way other than to provide information, the user is not an argument to acts for CTPs. IDP can only deduce a CTP for an act when there is an active *content-goal* (CG) that the plan satisfies. A constraint on all CTP-operators requires there to be an active CG to let the user know the proposition that will be the satellite in a subject-matter rhetorical relation.

As shown in Figure 3(b) as the second step in executing the circumstantiate CTP, the CG is retracted. Because the ACT-PLAN proposition is deducible only when the CG exists, the SNePS Belief Revision component (SNeBR) [15] retracts the ACT-PLAN proposition from the knowledge base as part of the execution of the CTP. This precludes the same CTP from being used twice if the system reattempts a DTP, or uses another DTP for which the same CTP is appropriate.

6.7 The TP

Figure 4 shows IDP's TP for the first demonstration run in this paper. The TP has been formulated to achieve the discourse goal of having the user adopt the plan to take Maple Road. The high-level plan is a DTP which can include other DTPs and CTPs. The TP always bottoms out in one primitive act, *say*. The argument to *say* is a text message which includes a proposition as the content to be expressed. In the TP, the checks ($\surd$) mark the *active path*. The active path is the most recently expanded (and focussed part of the TP). Note that the plan for motivate has not been executed in the order indicated by the sequencing act *snsequence* (see Figure 3(a)). In particular, the second act expands to an optional CTP, which IDP does not use until it responds to *Why?* a second time.

6.8 The Analyzer

The analyzer uses the existing TP, the active path, and the set of localized unknowns. First, in compliance with Carberry's notion of discourse expectation, IDP analyzes *Why*-questions in relation to its own intent as represented by the DTPs along the active path. The analyzer starts with the most recently executed DTP (the last DTP on the active path), and the localized unknowns associated with it. It tries to find another DTP to expand the current DTP that lets the PL know a localized unknown. The analyzer backs up the active path testing each DTP in turn. If this fails, the analyzer considers the informational level next. This level is reflected in CTPs. The analyzer examines the most focussed DTP on the active path to see if it can be expanded with a CTP to let the PL known a localized unknown. If this fails, the analyzer tries to expand a used, or unused CTP that expands the DTP with another CTP. IDP recognizes this kind of expansion as a digression.

7 Current Status and Future Work

The system I have demonstrated currently uses nine TP-operators to formulate TPs for justifying domain plans. Future work will extend the system to engage in domain plan descriptions and system-imposed topic shifts. Since the TP is represented uniformly with the domain plan that is under discussion, another objective is to extend IDP so that it can engage in full-blown, meta-level discussions of its own TP.

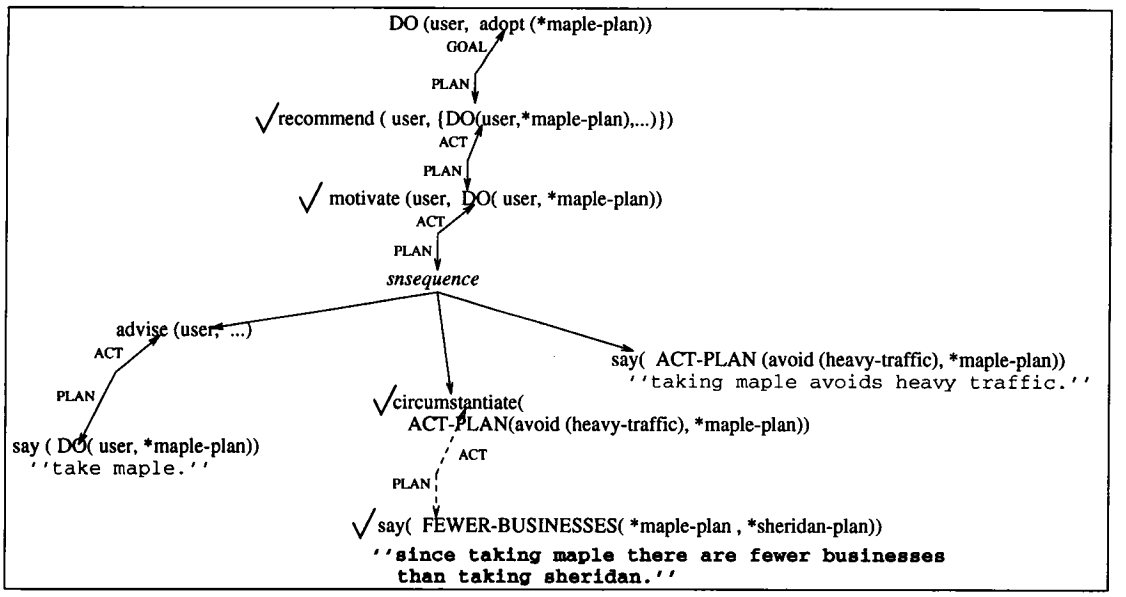

Figure 4: The TP for the First IDP Demonstration

References

[1] J. Allen and C. R. Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15:143–178, 1980.

[2] S. Carberry. Modeling the user's plans and goals. *Computational Linguistics*, 14(3), 1988.

[3] S. Carberry. A pragmatics-based approach to ellipsis resolution. *Computational Linguistics*, 15(4), 1989.

[4] A. Cawsey. Generating explanatory discouse. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*, chapter 4, pages 75–101. Academic Press, 1990.

[5] 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.

[6] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linquistics*, 12:175–204, 1986.

[7] S. M. Haller. *Interactive Generation of Plan Descriptions and Justifications*. PhD thesis, Department of Computer Science, State University of New York at Buffalo, 1993. In progress.

[8] S. M. Haller. An interactive model for plan explanation. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, Melbourne, Australia, November 1993. to appear.

[9] B. Kuipers. Modeling spatial knowledge. *Cognitive Science*, 2:129–153, 1978.

[10] D. Kumar, S. Ali, and S. C. Shapiro. Discussing, using and recognizing plans in SNePS preliminary report - SNACTor: An acting system. In *Proceedings of the Seventh Biennial Convention of South East Asia Regional Confederation*, pages 177–182, New Delhi, India, 1988. Tata McGraw-Hill.

[11] D. Kumar and S. C. Shapiro. Architecture of an intelligent agent in SNePS. *SIGART Bulletin*, 2(4):89–92, August 1991.

[12] W. J. M. Levelt. *Speaking: From Intention to Articulation*. MIT Press, Cambridge, 1989.

[13] D. Litman and J. Allen. A plan recgnition model for subdialogues in conversations. *Cognitive Science*, 11:163–200, 1987.

[14] W. C. Mann and S. A. Thompson. Rhetorical structure theory: A theory of text organization. Technical report, Information Sciences Institute, 1987.

[15] J. P. Martins and S. C. Shapiro. A model for belief revision. *Artificial Intelligence*, 35(1):25–79, 1988.

[16] J. Moore and W. Swartout. A reactive approach to explanation: Taking the user's feedback into account. In C. Paris, W. Swartout, and W. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, chapter 1, pages 3–48. Kluwer Academic Publishers, 1991.

[17] L. A. Ramshaw. A metaplan model for problem-solving discourse. In *Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics*, 1989.

[18] S. C. Shapiro. Generalized augmented transition network grammars for generation from semantic networks. *American Association of Computational Linguistics*, 8:12–25, 1982.

[19] S. C. Shapiro and The SNePS Implementation Group. *SNePS-2.1 User's Manual*. Department of Computer Science, SUNY at Buffalo, 1992.

[20] S. C. Shapiro and W. J. Rapaport. SNePS considered as a fully intensional propositional semantic network. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*, pages 263–315. Springer–Verlag, New York, 1987.