COLLAGEN: Project Summary and Discussion Questions

for AAAI 1999 Fall Symposium on
Psychological Models of Communication in Collaborative Systems

http://www.merl.com/projects/collagen

Charles Rich
MERL–A Mitsubishi Electric Research Lab
201 Broadway
Cambridge, MA 02139 USA
rich@merl.com

Candace L. Sidner
Lotus Development Corporation
55 Cambridge Parkway
Cambridge, MA 02142 USA
csidner@lotus.com

The underlying premise of the Collagen™ (for *Collaborative agent*) project is that software agents, when they interact with people, should be governed by the same principles that govern human-to-human collaboration. To determine the principles governing human collaboration, we have relied on research in computational linguistics on collaborative discourse, specifically within the SharedPlan framework of Grosz and Sidner (Grosz & Sidner 1986, Grosz & Sidner 1990, Grosz & Kraus 1996, Lochbaum 1998). This work has provided us with a computationally-specified theory that has been empirically validated across a range of human tasks. We have implemented the algorithms and information structures of this theory (see Appendix for details) in the form of a Java middleware component, a *collaboration manager* called Collagen, which software developers can use to implement a collaborative interface agent for any Java application.

In the *collaborative interface agent* paradigm, illustrated abstractly in Figure 1, a software agent is able to both communicate with and observe the actions of a user on a shared application interface, and vice versa. The software agent in this paradigm takes an active role in joint problem solving, including advising the user when he gets stuck, suggesting what to do next when he gets lost, and taking care of low-level details after a high-level decision is made.

The screenshot in Figure 2 shows how the collaborative interface agent paradigm is concretely realized on a user's display. The large window in the background is the shared application, in this case, the Lotus eSuite™ email program. The two smaller overlapping windows in the corners of the screen are the agent's and user's *home windows*, through which they communicate with each other.

A key benefit of using Collagen to build an interface agent is that the collaboration manager automatically constructs a structured history of the user's and agent's activities. This *segmented interaction history* is hierarchically organized according to the goal structure of the application tasks. Among other things, this history can help re-orient the user when he gets confused or after

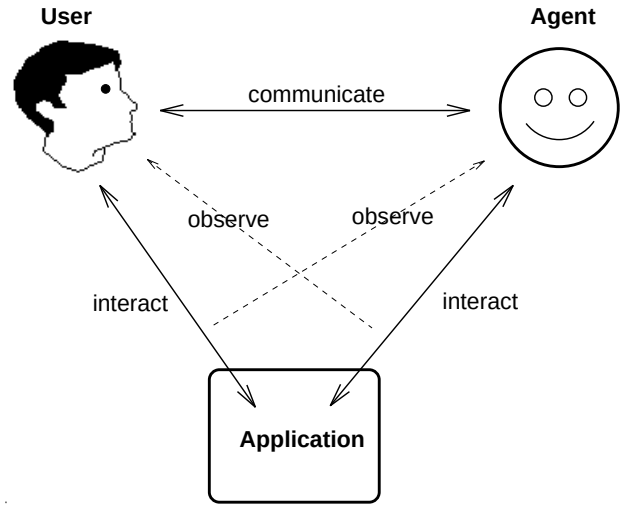

Figure 1: Collaborative interface agent paradigm.

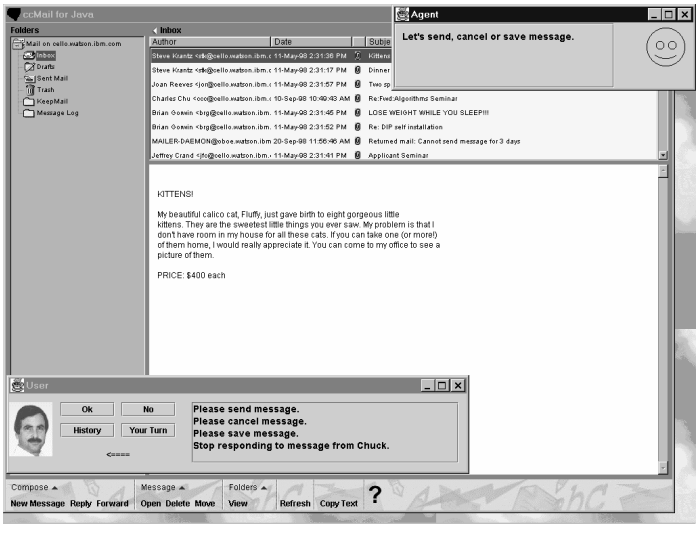

Figure 2: Graphical interface for Collagen email agent.

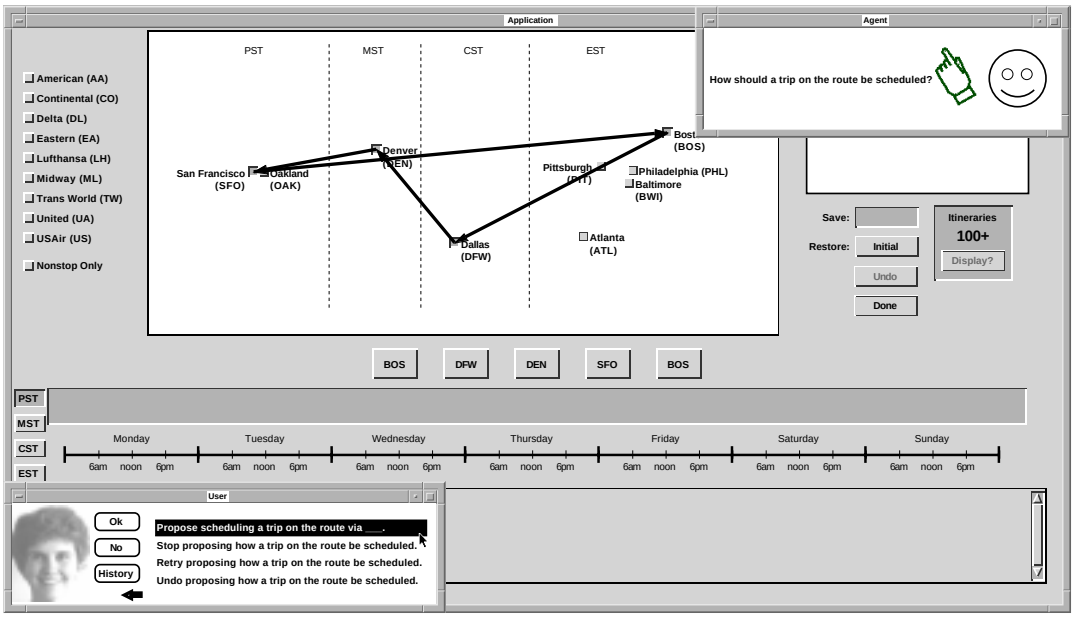

Figure 3: Graphical interface for Collagen air travel agent.

```
(Scheduling a trip on the route via working backward, allowing 4 itineraries)
   USER:  "Let's schedule a trip."
   AGENT: "Ok."
   (Done identifying route of scheduling a trip as Boston to Dallas to Denver
            to San Francisco to Boston, allowing 100+ itineraries)
      USER:  Add Boston to the route.
      USER:  Add Dallas to the route, allowing 97 itineraries.
      USER:  Add Denver to the route, allowing 100+ itineraries.
      USER:  Add San Francisco to the route, allowing 100+ itineraries.
      USER:  Add Boston to the route, allowing 100+ itineraries.
      USER:  "Your turn."
   (Done proposing a trip on the route be scheduled via working backward)
      AGENT: "How should a trip on the route be scheduled?"
      USER:  "Let's schedule a trip on the route via working backward."
   (Done working on San Francisco to Boston leg, allowing 70 itineraries)
      AGENT: "Let's work on San Francisco to Boston leg."
      USER:  Add Boston final interval with arrival Sun 11:30p.
      USER:  Change Boston final interval to arrival Fri noon.
      USER:  Add San Francisco stopover with arrival Wed 1:55a departure Wed 9:55a,
                allowing 70 itineraries.
      USER:  Change San Francisco stopover to arrival Thu 6:00p departure Fri 1:57a.
      USER:  "Your turn."
   (Done working on Denver to San Francisco leg, allowing 61 itineraries)
      AGENT: "Let's work on Denver to San Francisco leg."
      USER:  Add Denver stopover with arrival Thu 8:21a departure Thu 3:19p.
  ...
   (Done specifying airlines, allowing 4 itineraries)
      USER:  Add American specification, allowing no itineraries.
      (Done agent adding United specification, allowing 4 itineraries)
         AGENT: "I'm going to add United specification."
         USER:  "Ok."
         AGENT: Add United specification, allowing 4 itineraries.
   (Displaying and selecting itineraries)
      USER:  Display 4 itineraries.
      USER:  Select itinerary Wed AA/1099 Wed AA/313 Thu UA/343 Fri AA/928.
```

Figure 4: Sample segmented interaction history for air travel application.

an extended absence. It also supports high-level, task-oriented transformations, such as returning to an earlier goal. Figure 4 shows a sample segmented interaction history for an air travel planning application.

To apply Collagen to a particular application, the application developer must provide an abstract model of the tasks for which the application software will be used. This knowledge is formalized in a *recipe library*, which is then automatically compiled for use by the interface agent. This approach also allows us to easily vary an agent's level of initiative from very passive to very active, using the same task model. For more details on the internal architecture of Collagen, see Figure 7 and (Rich & Sidner 1998).

We have developed prototype interface agents using Collagen for several applications, including air travel planning (Rich & Sidner 1998), resource allocation, industrial control, and common PC desktop activities using email and calendar applications (Gruen *et al.* 1999).

The desktop agent is the first Collagen-based agent we have built that supports spoken-language interaction. Our other agents avoided the need for natural language understanding by presenting the user with a dynamically-changing menu of expected utterances, which was generated from the current discourse state according to the predictions of the the SharedPlan theory. The desktop agent, however, incorporates a speech and natural language understanding system developed by IBM Research, allowing users to collaborate either entirely in speech or with a mixture of speech and graphical actions

Discussion Questions

What is the role of plan recognition in collaboration?

Early versions of Collagen had no plan recognition capabilities. Users were required to explicitly announce each goal before performing a primitive action which contributed to achieving it. We recently added (intended) plan recognition to Collagen (Lesh, Rich, & Sidner 1999), which significantly reduced the amount of communication required of the user, since the agent can now infer the intent of many user actions.

Adding plan recognition to Collagen brought up a number of questions at the "boundary" between the plan recognition algorithms and the discourse interpretation algorithms provided by SharedPlan theory. In particular, we need better principles for deciding what to do when the plan recognizer returns too many or too few explanations.

In the case of ambiguity, i.e., too many explanations, there would appear to be a tradeoff between trying to clarify the ambiguity now (e.g., by asking the user) and waiting (to see if future events resolve the ambiguity). On the one hand, always bothering the user is undesirable (especially if it can be avoided by a little waiting). On the other hand, the longer (i.e., over more events) the ambiguity persists, the more complicated the discussion to resolve it will ultimately be; plus, there may be missed opportunities to be helpful and collaborative along the way.

At the other end of the spectrum, if the plan recognizer finds no explanations, how should the agent determine whether the unexplained event is an interruption of the current goal versus the start of a new toplevel goal?

Which general negotiation strategies should a collaborative agent use?

In even the most non-adversarial situations, negotiation is a common phenomenon. The participants in a collaboration almost never have completely identical goals and beliefs. Even when they have agreed upon a shared goal and a recipe for achieving it (i.e., a SharedPlan), they may not agree on the details of what is currently true about the world around them and the likely outcomes of their actions. Negotiation is the process by which they resolve these kinds of differences in goals and beliefs throughout a collaboration.

For example, negotiation may be required when an interface agent proposes an action and the user rejects it (perhaps because the action is optional or because the user believes its effect has already has achieved). Negotiation may also occur when one collaborator is uncertain of a proposed action or belief and wishes more information. A collaborator may wish to modify an already agreed-upon goal or recipe choice. Finally, negotiation is needed when one collaborator has rejected a proposal due to a misunderstanding (Chu-Carroll & Carberry 1995), especially of a linguistic nature. We have thus far in Collagen addressed these kinds of negotiation in an unsatisfactory, piecemeal fashion.

How do we evaluate the implementation of a collaborative theory?

Several possible kinds of evaluation come to mind for a system like Collagen. One kind of evaluation, determining whether an interface agent implemented using Collagen can actually help a user in some application task, is relatively straightforward; and we plan to carry it out in near future.

However, we are also interested in evaluating the extent to which our system is a sound implementation of the underlying theory and how that implementation bears on the claims of that theory.

SharedPlan theory is sufficiently rich to include many concepts (such as contracting out) that are not implemented in the current version of Collagen. We would like to understand whether our "take" on the theory captures its essence, or perhaps mis-interprets portions of it. For example, we decided to treat communication

actions (utterances) and domain actions uniformly in Collagen, an issue which is not directly addressed in SharedPlan theory.

Regarding the second half of the question above, namely, using the implementation to evaluate the theory, one might argue that a theory about what people do cannot be critiqued based on a computer implementation. However, it has long been a tenet of AI and is often the case in other scientific fields, that computational embodiments of a theory can often yield much insight. We are interested in what kinds of experiments with Collagen might confirm or discount aspects of SharedPlan theory.

Appendix

This appendix provides a brief overview of the collaborative discourse theory and algorithms on which Collagen is based. Readers are referred to the bibliography for more details.

Collaboration is a process in which two or more participants coordinate their actions toward achieving shared goals. Most collaboration between humans involves communication. *Discourse* is a technical term for an extended communication between two or more participants in a shared context, such as a collaboration.

Collaborative discourse in Grosz and Sidner's framework (Grosz & Sidner 1986, Grosz & Sidner 1990) is understood in terms of three interrelated kinds of discourse structure:

- intentional structure, which is formalized as partial *SharedPlans*.
- linguistic structure, which includes the hierarchical grouping of actions into *segments*.
- attentional structure, which is captured by a *focus stack* of segments.

We summarize the key features of each of these structures below, followed by a concrete example of Collagen's discourse state representation. Finally, we describe the discourse interpretation and generation algorithms, which are the heart of Collagen's discourse processing.

SharedPlans

Grosz and Sidner's theory predicts that, for successful collaboration, the participants need to have mutual beliefs[1] about the goals and actions to be performed and the capabilities, intentions, and commitments of the participants. The formal representation (Grosz & Kraus 1996) of these aspects of the mental states of the collaborators is called a SharedPlan.

[1] A and B mutually believe p iff A believes p, B believes p, A believes that B believes p, B believes that A believes p, A believes that B believes that A believes p, and so on. This is a standard philosphical concept whose infinite formal definition is not a practical problem.

As an example of a SharedPlan in the air travel domain, consider the collaborative scheduling of a trip wherein participant A (e.g., the user) knows the constraints on travel and participant B (e.g., the software agent) has access to a data base of all possible flights. To successfully complete the collaboration, A and B must mutually believe that they:

- have a common goal (to find an itinerary that satisfies the constraints);
- have agreed on a sequence of actions (a *recipe*) to accomplish the common goal (e.g., choose a route, specify some constraints on each leg, search for itineraries satisfying the constraints);
- are each capable of performing their assigned actions (e.g., A can specify constraints, B can search the data base);
- intend to do their assigned actions; and
- are committed to the overall success of the collaboration (not just the successful completion of their own parts).

Several important features of collaboration should be noted here.

First, due to partial knowledge of the shared environment and each other, participants do not usually begin a collaboration with all of the conditions above "in place." They typically start with only a *partial* SharedPlan. An important purpose of the communication between participants is to determine (possibly with the help of individual information gathering) the appropriate recipe to use, who should do what, and so on.

Second, notice that SharedPlans are recursive. For example, the first step in the recipe mentioned above, choosing a route, is itself a goal upon which A and B might collaborate.

Finally, planning (coming to hold the beliefs and intentions required for the collaboration) and execution (acting upon the current intentions) are usually interleaved for each participant and among participants. Unfortunately, there is currently no generally accepted domain-independent theory of how people manage this interleaving. (The current best candidates for a generic theory are the so-called belief/desire/intention frameworks, such as (Bratman, Israel, & Pollack 1988).) Collagen therefore does not currently provide a generic framework for execution. Another way of saying this is that we provide a generic framework only for *recording* the order in which planning and execution occur, not for *deciding* how to interleave them.

Discourse Segments and Focus Stack

The concept of discourse segments is at the very foundation of discourse theory. Analysis of discourses from a range of human interactions has resulted in general agreement that discourse has a natural hierarchical

structure. The elements of this hierarchy are called *segments*. A segment is a contiguous sequence of communicative actions that serve some purpose. For example, a question and answer sequence constitutes a discourse segment whose purpose is (usually) to achieved shared knowledge of some fact.

The existence of segments can be seen in everything from pitch patterns in spoken discourse to the way that pronouns are interpreted. Automatic segmentation (i.e., the segmented interaction history) has therefore been our first milestone in applying discourse principles to human-computer interaction.

A simple example of segments in a task-oriented human discourse is shown in Figure 5, which is adapted from (Grosz [Deutsch] 1974). In this discourse, participant A is instructing participant B how to repair an air compressor. Notice that this analysis of discourse structure includes not only the participants' utterances, but also their actions (e.g., B removes belt). This is appropriate in a context, such as collaborative interface agents, where all actions on the shared artifact are known and intended to be mutually observable.

The toplevel segment and three embedded segments in Figure 5 are indicated by the brackets and indentation shown (further subsegments are elided). In Grosz and Sidner's theory, the segment structure of such a discourse is accounted for by assigning a *purpose* to each segment, such that each segment's purpose contributes to successful collaboration on the parent segment's purpose via the SharedPlan conditions described above.

For example, the purpose of the toplevel segment in Figure 5 is to replace the pump and belt, which is the common goal of the collaboration. The purpose of the first subsegment is to remove the belt, which is one of the steps in the recipe for replacing the belt. The purpose of the first subsubsegment is to identify a parameter of the removal action, i.e., the belt to be removed.

A: "Replace the pump and belt please."

- B: "OK, I found a belt in the back."
  - B: "Is that where it should be?"
  - A: "Yes."
- B: Removes belt.
- B: "It's done."

A: "Now remove the pump."
...
A: "First you have to remove the flywheel."
...
A: "Now take off the base plate."
B: "Already did."

Figure 5: Segments in a task-oriented human discourse.

The purpose of the second subsegment is to remove the pump, which is also one of the steps in the recipe for the toplevel purpose.

The shifting focus of attention in a discourse is captured by a *focus stack* of discourse segments. In the natural flow of a a collaborative discourse, new segments and subsegments are created, pushed onto the focus stack, completed, and then popped off the stack as the SharedPlan unfolds in the conversation. Sometimes participants also interrupt each other, abandon the current SharedPlan even though it is not complete, or return to earlier segments.

Thus, as we will see more concretely in the discussion of Figure 6 below, the attentional (focus stack) and intentional (SharedPlan) aspects of discourse structure theory are connected through the discourse segment purpose: each segment on the stack is associated with a SharedPlan for its purpose (Lochbaum, 1994; 1998).

Discourse State Representation in Collagen

The *discourse state* in Collagen is a concrete representation of the three kinds of discourse structure described above. Figure 6 shows an example discourse state.

The lower part of Figure 6 shows a *plan tree*, which is an approximate representation of a partial SharedPlan. Plan trees are composed of alternating act and recipe nodes as shown. Both acts and recipes have *bindings*, shown as labelled stubs in the figure, with constraints between them specified in their recipe library definitions. An act node has a binding for each of its parameters, who performs it and, if it is non-primitive, a

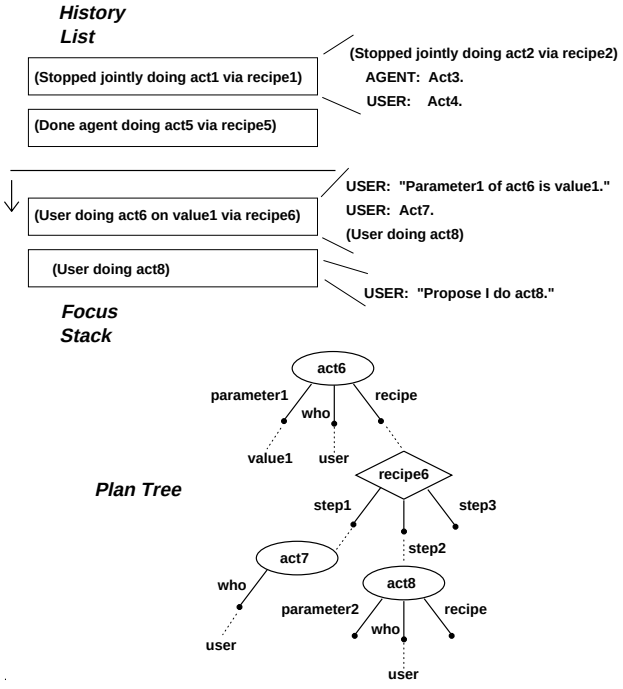

Figure 6: Internal discourse state representation.

recipe node. A recipe node has a binding for each step in the recipe. To support the nonmonotonic changes in discourse state required for negotiation and history-based transformations (Rich & Sidner 1998), bindings and the propagation of logical information in the plan tree are implemented using a truth-maintenance system.

For example, in Figure 6, act6's sole parameter has been bound to value1 and act6's recipe has been bound to recipe6. If a history-based transformation "undoes" act7 and act8, then act6's recipe binding will be retracted. Similarly, act6's parameter binding will be retracted if the first communication act in its segment is undone.

The upper part of Figure 6 shows the *focus stack* (illustrated as growing downward) and the *history list*, which contains toplevel segments that have been popped off the focus stack. When a segment is popped off the focus stack, it is added to the history list if and only if it has no parent segment. In the figure, there are two segments on the focus stack and two segments in the history list. The elements of one of the segments on the stack and one of the segments in the history list are shown expanded to the right.

Segments on the stack are called *open*, because they may still have acts added to them. Segments that have been popped off the stack are called *closed*. All the segments in the history list and their subsegments are closed. Segments on the stack may have closed subsegments.

Usually, the root of the plan tree (e.g., act6 in Figure 6) is the purpose of the base segment of the focus stack, and each subsegment (e.g., act8) corresponds to a subtree, recursively. The exception is when there is an interruption, i.e., a segment which does not contribute to its parent, in which case we have a disconnected plan tree for that segment. Plan trees remain associated with segments even after they are popped off the stack.

The two key discourse processing algorithms used in Collagen are *discourse interpretation*, which is a reimplementation of Lochbaum's (1994, 1998) rgraph augmentation algorithm, and *discourse generation*, which is essentially the inverse of interpretation.

Discourse Interpretation

The main job of discourse interpretation in Collagen is to consider how the current direct communication or observed manipulation action can be viewed as contributing to the current discourse purpose, i.e., the purpose of the top segment on the focus stack. This breaks down into five main cases.[2] The current act either:

- directly achieves the current purpose,
- is one of the steps in a recipe for the current purpose (this may involve retrieval from a recipe library),
- identifies the recipe to be used to achieve the current purpose,
- identifies who should perform the current purpose or a step in the current recipe, or
- identifies an unspecified parameter of the current purpose or a step in the current recipe.

If one of these cases obtains, the current act is added to the current segment (and the partial SharedPlan representation is appropriately updated). Furthermore, if this act completes the achievement of the current discourse segment purpose, the focus stack is popped.

If none of the above cases holds, discourse interpretation concludes that the current action starts an *interruption*, i.e., a segment that does not contribute to its parent. A new segment is pushed onto the focus stack with the current act as its first element. The purpose of this new segment may or may not be known, depending on the specific content of the initiating action.

The occurence of interruptions may be due to actual interruptive material in the ongoing discourse or due to an incomplete recipe which does not include the current act even though it ought to. We take the view that, in general, the agent's knowledge will never be complete and it therefore must deal gracefully with unexpected events.

Another phenomenon which manifests itself as interruptions in the current discourse interpretation algorithm is when the user and/or agent are pursuing two (or more) goals in parallel, e.g., arbitrarily interleaving steps from both recipes. In this situation, some higher level representation of the parallel goals would be preferable to the alternating structure of pushes (interruptions) and pops currently required. This is an area for future work.

It is tempting to think of discourse interpretation as the plan recognition problem, which is known to be exponential in the worst case (Kautz 1990). However, this misses a key property of normal human discourse, namely that speakers work hard to make sure that their conversational partners can understand their intentions without a large cognitive search. Notice that only search performed by the discourse interpretation algorithm above is through the steps of the current recipe or all known recipes for the current segment's purpose (and this is *not* done recursively). We think it will be reasonable to expect users to communicate enough so that the agent can follow what is going on without having to do general plan recognition.

Discourse Generation

The discourse generation algorithm is, as mentioned above, essentially the inverse of interpretation. It looks at the current focus stack and associated SharedPlan and produces a prioritized *agenda* of (possibly partially specified) actions which would contribute (according to the five cases above) to the current discourse segment purpose. For example, if the current purpose is to jointly schedule a trip, the agenda includes an action

[2]The last three cases are instances of a larger class of explanations that Lochbaum (1995) calls "knowledge preconditions."

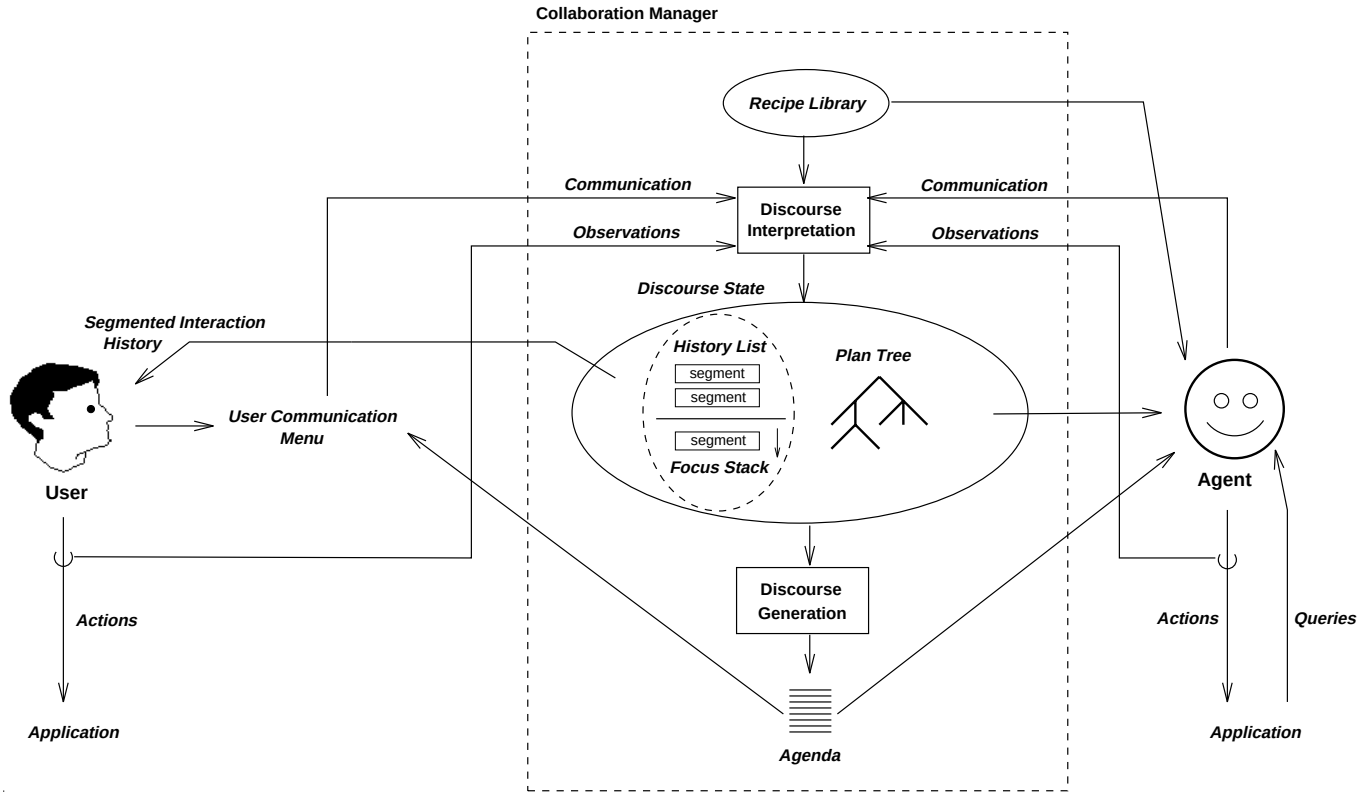

Figure 7: Collagen architecture.

in which the agent asks the user to propose a route. In Collagen, the agenda contains communication and manipulation actions by either the user or agent, which would advance the current problem-solving process.

The main reason we believe that a menu-based approach to user communication demonstrated may be workable is because the discourse generation algorithm typically produces only a relatively small number of communication choices, all of which are relevant in the current discourse context.

Collagen Architecture

This section summarizes how the theory-grounded algorithms and information structures described above have been embodied in a practical architecture (Figure 7) for building collaborative interface agents. Figure 7 is essentially an expansion of Figure 1 in which the Collagen discourse manager is made explicit as the mediator of all communication between the agent and the user.

All of the internal data flow in Figure 7 takes place using Collagen's artificial discourse language (Sidner 1994). Whenever there is a need for the user to see information, such as in display of the user communication menu or the segmented interaction history, these internal representations are given an English gloss by simple string substitution in templates defined in the recipe library.

We refer to Collagen as a *collaboration manager*, because it provides a standard mechanism for maintaining the flow and coherence of agent-user collaboration. In addition to saving implementation effort, using a collaboration manager provides consistency across applications, and to the extent it is based on good principles, leads to applications that are easier to learn and use.

The Agent

Note that the agent itself is a "black box" in Figure 7. We are not trying to provide tools for building a complete agent. At the heart of the agent there may be a rule-based expert system, a neural net, or a completely ad hoc collection of code—whatever is appropriate for the application. What Collagen does provides is a generic framework for recording the decisions made and communicated by the agent (and the user), but not for *making* them. We believe this is a good software engineering modularity.

As can be seen in Figure 7, Collagen also provides some important new resources (inputs) for a developer to use in implementing the decision-making part of an interface agent: the discourse state, the agenda, and the recipe library.

The default agent implementation that "comes with" Collagen always simply chooses to perform the highest priority action in the current agenda for which the actor is either unspecified or itself. Our air travel planning agent, for example, was constructed by extending this

default implementation with only a page of application-specific logic, which sometimes proposed other actions based on querying the discourse and application states.

The Basic Execution Cycle

The best way to understand the basic execution cycle of the architecture in Figure 7 is to start with the arrival of a communication or observation event (from either the agent or the user) at the discourse interpretation module at the top center of the diagram. The interpretation module updates the discourse state as described in Section above, which then causes a new agenda of expected communication and manipulation acts (by either the user or agent) to be computed by the discourse generation module. As mentioned above, the agent may decide to select an entry in this new agenda for execution.

A subset of the agenda is also presented to the user whenever the user opens the communication menu in his home window. Specifically, the user communication menu is constructed by selecting all the communication actions in the agenda for which the actor is either unspecified or itself. What we are doing here is using expectations generated by discourse context to replace natural language understanding. The user is not allowed to make arbitrary communications, but only to select from communications expected by the discourse interpretation algorithm. Thus, unlike usual ad hoc menu-driven interaction, the user menu in Collagen is systematically generated from an underlying model of orderly discourse.

If the user selects one of the communication menu entries, it becomes input to the discourse interpretation module, thus closing an execution cycle.

References

Bratman, M. E.; Israel, D. J.; and Pollack, M. E. 1988. Plans and resource-bounded practical reasoning. *Computational Intelligence* 4(4):349–355.

Chu-Carroll, J., and Carberry, S. 1995. Response generation in collaborative negotiation. In *Proc. 33rd Annual Meeting of the ACL*, 136–143.

Grosz, B. J., and Kraus, S. 1996. Collaborative plans for complex group action. *Artificial Intelligence* 86(2):269–357.

Grosz, B. J., and Sidner, C. L. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics* 12(3):175–204.

Grosz, B. J., and Sidner, C. L. 1990. Plans for discourse. In Cohen, P. R.; Morgan, J. L.; and Pollack, M. E., eds., *Intentions and Communication*. Cambridge, MA: MIT Press. 417–444.

Grosz [Deutsch], B. J. 1974. The structure of task-oriented dialogs. In *IEEE Symp. on Speech Recognition: Contributed Papers*, 250–253.

Gruen, D.; Sidner, C.; Boettner, C.; and Rich, C. 1999. A collaborative assistant for email. In *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems*.

Kautz, H. 1990. A circumscriptive theory of plan recognition. In Cohen, P. R.; Morgan, J. L.; and Pollack, M. E., eds., *Intentions and Communication*. Cambridge, MA: MIT Press. 105–133.

Lesh, N.; Rich, C.; and Sidner, C. 1999. Using plan recognition in human-computer collaboration. In *Proc. 7th Int. Conf. on User Modelling*, 23–32.

Lochbaum, K. E. 1994. Using collaborative plans to model the intentional structure of discourse. Technical Report TR-25-94, Harvard Univ., Ctr. for Res. in Computing Tech. PhD thesis.

Lochbaum, K. E. 1995. The use of knowledge preconditions in language processing. In *Proc. 14th Int. Joint Conf. Artificial Intelligence*, 1260–1266.

Lochbaum, K. E. 1998. A collaborative planning model of intentional structure. *Computational Linguistics* 24(4).

Rich, C., and Sidner, C. 1998. COLLAGEN: A collaboration manager for software interface agents. *User Modeling and User-Adapted Interaction* 8(3/4):315–350.

Sidner, C. L. 1994. An artificial discourse language for collaborative negotiation. In *Proc. 12th National Conf. on Artificial Intelligence*, 814–819.