From the Programmer’s Apprentice to Human-Robot Interaction: Thirty Years of Research on Human-Computer Collaboration

Charles Rich, Candace L. Sidner
TR2006-052  July 2006

Abstract
We summarize the continuous thread of research we have conducted over the past thirty years on human-computer collaboration. This research reflects many of the themes and issues in operation in the greater field of AI over this period, such as knowledge representation and reasoning, planning and intent recognition, learning, and the interplay of human theory and computer engineering.

AAAI Conference on Artificial Intelligence

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 2006
201 Broadway, Cambridge, Massachusetts 02139
From the Programmer’s Apprentice to Human-Robot Interaction: Thirty Years of Research on Human-Computer Collaboration  

Charles Rich and Candace L. Sidner  
Mitsubishi Electric Research Laboratories  
201 Broadway  
Cambridge, MA, 02139, USA  
rich,sidner@merl.com

Abstract
We summarize the continuous thread of research we have conducted over the past thirty years on human-computer collaboration. This research reflects many of the themes and issues in operation in the greater field of AI over this period, such as knowledge representation and reasoning, planning and intent recognition, learning, and the interplay of human theory and computer engineering.

Human-Computer Collaboration
Figure 1 illustrates our overall research methodology, which has been to model human-computer collaboration on what is known about human-human collaboration. Furthermore we have focused almost exclusively on the special case of two copresent collaborators, i.e., where each collaborator is able both to communicate with and observe the actions of the other. Examples of such collaborations include two mechanics working on a car engine together or two computer users working on a spreadsheet together. To a first approximation, our approach has been simply to substitute a computer agent for one of the human collaborators, keeping as much else the same as possible.

Due to space limitations, we will not attempt to review all research on human-computer collaboration, but limit ourselves to viewing this topic through the lens of our own work and that of our immediate collaborators. Consistent with this, note that bibliography below contains only publications by ourselves and our immediate collaborators.

Chronological Summary
The chronology of our research begins in 1976 with the publication of Rich and Shrobe’s joint M.S. thesis on the Programmer’s Apprentice [1,3]: “As compared to automatic programming research, the programmer’s apprentice emphasizes a cooperative relationship between the computer and the human programmer...” Shortly thereafter, Sidner began work on modeling how natural language is used in the context of pairs (and later groups) of people achieving tasks together. Her first paper on this topic dealt with the interpretation of discourse purposes in the Personal Assistant Language Understanding Program [2].

Under the direction of Rich and Shrobe, and later Waters, the Programmer’s Apprentice project [4,15,16] lived at the MIT AI Lab from 1976 until Rich and Waters left MIT in 1991. Even though the concept of human-computer collaboration was the bedrock of the project, we never developed a deep theoretical understanding of what collaboration meant. Instead, most of the Programmer’s Apprentice research concentrated on how to represent and reason with the shared knowledge necessary for successful human-computer collaboration in the domain of software engineering, including requirements analysis [14], design, and implementation.

In retrospect, the choice of software engineering as a domain, as compared to, for example, medical diagnosis (which was another popular AI application domain at the time) may have been unwise. We were initially attracted by the fact that we already knew a lot about software engineering (as compared to having to spend the equivalent of a year in medical school to learn enough to do research). However, it turned out that the knowledge underlying software engineering is particularly hard to codify, in part because it is difficult to separate from knowledge about the world in which the software is intended to function.

Meanwhile, Grosz and Sidner [5,8,11,12,17,19,21] were delving deeply into the nature of human collaboration, culminating in the SharedPlan theory of collaborative discourse. By 1994, Rich and Sidner [22,23] had begun developing a practical application-independent tool, called Collagen (for collaborative agent), which implemented parts of this theory (see Figure 2). In a sense, Collagen was “the Programmer’s Apprentice without the programming.”

Collagen continues to evolve [24,26,27,28,32] and has been used to build prototype human-computer collabora-
tive systems, both at MERL and at several other institutions, for a wide range of applications, including air travel planning [28], email [30], gas-turbine operator training [34], programmable thermostat operation [35], power system operation [37], airport landing path planning [38], GUI design [41], and personal video recorder operation [45].

Most recently, we have concentrated on applications of Collagen in the area of feature-rich digitally-enabled home appliances, such as home entertainment systems and programmable combination washer-dryers. In particular, we have developed a new user interface design, called DiamondHelp [50,52,51], which reinforces the collaborative metaphor through the use of a “chat window” between the user and a collaborative agent.

Finally, recognizing the importance of physically embodied, nonverbal behavior [33], such as looking and nodding [42,46,48,49,53], in human collaboration, Sidner has recently embarked on a research program in human-robot interaction to study and apply these phenomena, using hosting activities as a domain.

Knowledge Representation and Reasoning

As in any AI undertaking, the issues of how to represent and reason with the knowledge have been a central concern. For human-computer collaboration, these issues can be broken down along several dimensions. First, there is application-independent versus application-specific knowledge. A major contribution of the theoretical work on discourse has been to show that, given the appropriate abstractions, a significant amount of structure and computation can be captured in an application-independent tool like Collagen.

A second important dimension is computational cost. We are interested in interactive collaborative systems, which means that the system needs to respond in a reasonably short period of time. We have therefore been driven to explore hybrid knowledge representations [7,9] which combine expressive general logical formalisms with specialized data structures for fast computation. We are also employing fast, sound, but incomplete inference methods [20].

Collaboration also involves changing your mind and and/or adapting to changes in your collaborator’s beliefs. To support this behavior, we make use of dependency-directed inference methods, such as truth maintenance systems [13].

Collaborative Discourse

Collaboration is a process in which two or more participants coordinate their actions toward achieving shared goals. Discourse is an extended communication between two or more participants in a shared context, such as a collaboration. From our point of view, discourse and collaboration are virtually coextensive concepts. It is possible to have a discourse without collaboration or a collaboration without discourse, but these are the unusual cases.

Grosz and Sidner [12] developed a tripartite framework for modeling collaborative discourse structure. The first (intentional) component records the beliefs and intentions of the discourse participants regarding the tasks and subtasks to be performed. This component, which came to be known as SharedPlans [17], was further formalized and generalized by Grosz and Kraus [25]. The second (attentional) component captures the changing focus of attention in a discourse using a stack of focus spaces organized around the discourse purposes. The third (linguistic) component consists of the contiguous sequences of utterances, called segments, which contribute to a particular task or subtask.

This model of collaborative discourse has implications for both natural language and speech understanding [45] and generation [47]. Lochbaum [29] developed algorithms for discourse interpretation based on this model, which made it possible to begin implementing Collagen.

More recently, Sidner and colleagues have begun to study the role of nonverbal behavior in collaborative discourse, specifically engagement, the process by which participants in an interaction start, maintain, and end their perceived connection to one another in a physical setting. Based on observation of human-human interactions, they have developed a computational model of how nodding and looking behaviors contribute to engagement and tested this model using a physical robot that interacts with a human [42,48,49,53].

Planning and Intent Recognition

A key insight in the Programmer’s Apprentice work was the need to represent the programmer’s intent at a more abstract level than source code. The Plan Calculus [6] formalism combined concepts from planning and software engineering for this purpose. There was also a significant effort in so-called “program understanding” (also called “reverse engineering”) to automatically recover programmer’s intent from extant source code [18].

These two themes of planning and intent recognition continue in the Collagen project. Key application-independent services provided by Collagen include: plan-based response generation [44], plan execution monitoring, and limited plan repair and replanning. Furthermore, even though plan recognition is in general NP-complete, Collagen uses a plan recognition algorithm [10,31,39] which is tractable by virtue of exploiting distinguishing properties of the collaborative
setting: the focus of attention, the use of partially elaborated hierarchical plans, and the possibility of asking for clarification.

**Intelligent Tutoring Systems**

Collaboration is a very broad concept which, depending on the relative knowledge and initiative of the participants, spans interactions from helping to teaching; or to put it in human-computer terms, from intelligent assistants to intelligent tutoring systems. Rickel used Collagen to develop PACO (Pedagogical Agent for COllagen) [34, 43] for teaching procedural tasks, and as the first step in building a bridge between the intelligent tutoring and the collaborative discourse communities [36].

**Learning**

Learning is a hallmark of intelligence. The need for many different forms of learning naturally arise in the process of developing human-computer collaborative systems. One obvious case we have pursued is learning hierarchical task models from examples [40]. There is also “learning by being told,” which is not as simple as it sounds, since it may involve negotiation about conflicting beliefs [21]. Other forms of learning during collaboration, such adapting to interaction style of the other participant(s), are still open research problems, some being pursued by others.

### Chronological Bibliography


