An Overview of Human-Computer Collaboration

Loren G. Terveeen
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
terveen@research.att.com

July 3, 1994

Abstract
This paper introduces the special issue of *Knowledge-Based Systems* on Human-Computer Collaboration (HCC). It derives a set of fundamental issues from a definition of collaboration, introduces two major approaches to HCC, and surveys each approach, showing how it formulates and addresses the issues. It concludes by proposing some themes that should characterize a unified approach to human-computer collaboration.

1 Introduction

Collaboration is a process in which two or more agents work together to achieve shared goals. Thirty researchers came together in Raleigh, North Carolina in October of 1993 for a AAAI Fall Symposium dedicated to this topic. The goal of the symposium was to achieve a better understanding of *Human-Computer Collaboration* (HCC), collaboration involving at least one human and one computational agent. In particular, the symposium sought to explore the fundamental nature of collaborative problem solving, understand the constraints brought to bear by the differing characteristics of human and computational agents, examine various approaches to modeling collaboration and designing collaborative systems, and draw lessons from implemented systems.

This special issue of *Knowledge-Based Systems* contains revised and extended versions of selected papers from the symposium. The last two issues of KBS have included Letters based on symposium papers. This overview paper surveys the field of HCC and discusses the papers in the special issue and the recent Letters. Let us begin by revisiting our definition of collaboration – *collaboration is a process in which two or more agents work together to achieve shared goals* – and considering a number of fundamental issues that arise directly from the definition.

1. *Agreeing on the shared goal(s) to be achieved.* Through direct discussion about goals or inference from statements and actions, agents must determine the shared goals they are trying to achieve. It is important to note that agents usually do not achieve a complete and unambiguous definition of their goals before beginning problem solving. Rather, problem solving often leads agents to further specify and even reformulate their goals.

2. *Planning, allocation of responsibility, and coordination.* Agents must decide how to achieve their goals, determine what actions will be done by each agent, and how to coordinate the actions of individual agents and integrate their results.

3. *Shared context.* Agents must be able to track progress toward their goals. They must keep track of what has been achieved and what remains to be done. They must evaluate the effects of actions and determine whether an acceptable solution has been achieved.

4. *Communication.* Any collaboration requires communication, for example, to define goals, negotiate over how to proceed and who will do what, and evaluate progress and results. Observation of other agents also plays a role.

5. *Adaptation and learning.* Collaboration has a history, both short term – within a single session – and long term – across many sessions. True collaboration over time seems to require partners to adapt themselves to each other. Learning is one way to adapt. In a collaborative interaction, one can learn from one's partner both directly, e.g., by being told or shown a new or better way of doing things, and indirectly, through the process of articulating, justifying, and explaining one's actions and attitudes to a partner.

The study of Human-Computer Collaboration is highly interdisciplinary. Its two basic parent disciplines are Artificial Intelligence (AI) and Human-Computer Interaction (HCI). From AI, it draws knowledge representation and reasoning techniques and a commitment to formal computational modeling. From HCI, it draws interaction and information presentation techniques and an awareness of the asymmetry in the abilities of people and computers. Other fields also have influenced HCC research, including philosophical studies of the nature of intention, goals, and beliefs [1, 2, 3, 4], and social science investigations of work practices, design, and the role of artifacts in human activity [5, 6, 7, 8, 9, 10, 11].

Historically, there have been two major approaches to Human-Computer Collaboration, one more closely allied to AI, the other to HCI. The two approaches are not completely distinct, and, as we shall see, there are interesting convergences between them. However, their historical roots and emphases are sufficiently distinct that it will be useful to organize our survey of HCC around the two approaches.

The first approach assumes that the way to get computers to *collaborate with* humans is to endow them with *human-like abilities*, to enable them to *act like humans*. It thus looks first to develop formal models of human-human collaboration, usually focusing on collaboration in language, then to apply the models to human-computer collaboration. Let's refer to this as the *Human Emulation* approach for short. The second approach assumes that the way to get computers to *collaborate with* humans is to exploit their *unique abilities*, to *complement humans*. It begins from the premise that computers and humans have fundamentally asymmetric abilities; therefore, it focuses on developing divisions of responsibility that assign each agent appropriate and distinct roles and on utilizing interaction techniques to facilitate effective human-computer communication. Let's refer to this as the *Human Complementary* approach for short.

Table 1 goes about here

Table 1 summarizes the set of issues derived from our definition of collaboration and lists some of the main topics and techniques from the two approaches that address the issues. Most of the paper is devoted to presenting an overview of each of the two approaches. The main focus is to show how each approach formulates the fundamental issues in collaboration and to identify the main techniques used to attack the issues. Along the way, we discuss the papers in this issue and mention the recent Letters, thus situating them in the larger context of human-computer collaboration research.

Considering each major approach separately tends to emphasize their differences. However, we also shall see a number of significant actual or potential convergences in how problems are formulated and what techniques are used to address them. Therefore, the paper concludes by suggesting the shape of a *unified* approach to human-computer collaboration that incorporates insights shared by both approaches. To give a brief preview, I argue that the following concerns should characterize a unified approach:

- *reification* – making knowledge, problem solving, and dialogue entities *visible* in the interface, *shared* and *manipulable* by user and system,

- *balancing representation & reasoning and interaction* – striking an effective balance between system reasoning and user-system interaction,

- *natural communication* – understanding the underlying aspects of human communication that make it successful and developing analogues for human-computer communication that exploit the unique properties of computational media,

- *collaborative adaptation* – making the process of adapting system behavior to an individual user a collaborative activity, with both user and system playing appropriate roles.

2 The *Human Emulation* Approach

This approach grew out of attempts to understand and model human communication. It focuses on intentions, beliefs, knowledge, and other attributes of agents' mental states. Humans are viewed as rational agents who form and execute plans for achieving their goals and infer other people's plans. Achieving goals might require communication, e.g., when an agent does not know how to achieve a subgoal, does not know a fact necessary to carry out an action, or has a goal concerning the mental state of another agent. Understanding requires inferring the goals and plans of other agents, and collaborative behavior consists of helping other agents to achieve their goals. The technical focus of this work has been on designing formalisms for representing beliefs, goals, plans, and actions, using these formalisms to develop models of collaboration, and developing algorithms for communication planning and plan recognition. These representation and reasoning problems are among the hardest in AI and cognitive science.

2.1 From Plan Recognition to Collaborative Planning

In the late 1970s, James Allen, Philip Cohen, and C. Raymond Perrault were pioneers in developing models of language based on AI planning theory [12, 13, 14, 15]. They drew on work in the philosophy of language [1, 2, 3, 4] that conceptualized language as goal-directed action and examined the intentions and beliefs necessary for successful communication. *Speech acts*, such as informing, requesting, asserting, and promising, are produced through a planning process. Each speech act is formalized in terms of its *preconditions* (conditions that must be true for it to be applied) and *effects* (changes it makes to the world being modeled). Acts are chosen for their effect on the hearer's mental state (or, more subtly, on some type of structure representing the shared discourse context). Understanding is an inference process in which an agent attempts to infer plans and intentions of other agents from observations of their actions, including speech acts. People tend to be helpful; in particular, when they recognize obstacles in another agent's plans, they will take action (e.g., provide information) to help overcome the obstacles.

A simple exchange from Allen[12] illustrates these points.

```
patron: When does the Montreal train leave?
clerk: 3:15 at gate 7
```

The patron wants to get on the Montreal train and constructs a plan to achieve this goal. However, he does not know the departure time. One way of finding out a piece of information is to have someone else inform you about it. And requesting someone to inform you often leads to them informing you. Using a formalization of these relationships, a planning process can lead the patron to ask the clerk when the Montreal train leaves. To respond to the request, the clerk essentially goes through the patron's planning process in reverse, attempting to reconstruct the plan that could have led him to make the request. The clerk finds that boarding the Montreal train is a plausible plan. In addition to the patron's specific request for the time of the train, the clerk finds another (potential) obstacle to carrying out the plan, namely that the patron may not know the departure gate. Therefore, the clerk plans to inform the patron of the gate. Thus, the clerk's answer both directly responds to the patron's question ("3:15") and includes additional information to overcome an inferred obstacle ("at gate 7").

It is important to note several assumptions of this work. First, one of the original motivations was to develop better (more human-like) question answering systems. This led to a focus on *single exchanges*, where one agent (the planning agent) asked a question, and the other agent (the inferring agent) provided a helpful answer.

Second, the planning agent and the inferring agent were viewed asymmetrically. The planning agent "owned" the plan (and the goal it was directed at) and was trying to execute it, while the inferring agent's job was to guess the plan and give some sort of help in executing it. In the context of a human-computer interaction, researchers naturally cast the human user as the planning agent and the computer system as the inferring agent. Is is important to realize, however, that this notion of asymmetry is not the same as the *Human Complementary* approach's assumption that computers and people have asymmetric abilities. In fact, the *Human Emulation* approach assumes a fundamental similarity between the agents it models, namely, that they are rational agents who plan to achieve their goals and infer other agents' plans.

Third, both the planning agent and inferring agent are assumed to have access to the same complete and correct set of domain plans.

Partly in response to problems with these assumptions of early plan recognition work, Barbara Grosz, Candace Sidner, and Karen Lochbaum [16, 17, 18] have been developing a model of *collaborative* planning. Their work begins from the premise that agents collaborate to achieve a goal. That is, they explicitly reject the asymmetry between a planning and an inferring agent. Further, their model provides ways to represent and distinguish between agents' beliefs. This is important since agents have different beliefs about ways to achieve a goal and the actions necessary to do so, and each agent's beliefs, including beliefs about the other agent, may be incorrect or incomplete.

Their model takes "having a plan" to be central; this consists of "being in a certain mental state, namely, holding certain beliefs and intentions regarding acts and their executability" [18]. A fundamental difference of this work from earlier plan recognition work is that it allows for collaboration in the planning process itself. This means that there is no a priori plan (belonging to one agent) to be recognized (by the other agent). Rather, they seek to model how agents' beliefs about actions and intentions are augmented through communication. This also means that their model cannot focus on single exchanges; rather, it describes how extended discourse gradually leads to the mental state that constitutes having a *SharedPlan*.

The central technical accomplishments of this work have been to provide a formal definition of a SharedPlan and to develop algorithms for augmenting beliefs through communication, leading to the construction of a SharedPlan. Following Pollack [19], the SharedPlan model also distinguishes two notions of a plan that often had been conflated: a course-of-action to be carried out (often referred to as a "domain plan") and a set of beliefs and intentions. SharedPlans cover the latter, and the former are referred to as "recipes".

> A *KBS Letter* by *Sidner* analyzes a natural occurring dialogue using the SharedPlan model and an artificial language that represents the impact of utterances on the ongoing dialogue. A *KBS Letter* by *Rich* describes an experiment in applying the SharedPlan model to the design of a direct manipulation system, yielding benefits such as task suspension and resumption, mixed initiative, and clarification sub-dialogues.
>
> *McRoy* (this issue) describes a model of collaborative discourse that treats *misunderstanding* and *repair* as intrinsic parts of communication processing. When a misunderstanding is detected, the interpretation of the conversation must be revised; therefore, discourse processing must be nonmonotonic. In McRoy's model, interpretation is a process of *abductive* reasoning. Given a set of observations and a background theory, abduction generates a set of facts that are consistent with the theory and imply the set of observations. For the interpretation of an utterance, the hearer's current beliefs and assumptions are the background theory, and the identification of either the discourse role of the utterance or a misunderstanding that could have led to it being produced is the result. Thus, one important contribution of this work is the use of a single type of reasoning to account for both trouble free interpretation and the detection and repair of misunderstandings. Constraining inference is a major problem with models of language that require inferring beliefs and goals. The problem gets worse when prior interpretations may have to be revised. McRoy's method of addressing this problem leads to her second major contribution: integrating conventions and expectations derived from sociological analyses of language with the beliefs

and goals of AI processing models. Expectations constrain the set of possible interpretations the model must consider, and conventional repair patterns are implemented as interpretation rules.

2.2 Discourse

A second major line of research has addressed phenomena that arise in extended discourse. Central issues include what constitutes coherent discourse, how initiative is managed, the role of context, and the management of trouble.

Studies of coherence have attempted to characterize discourse in terms of *structures* and various types of *relations* that hold between structures [20, 21, 22] Based on empirical analysis of texts and dialogues, researchers identified relations such as *elaboration*, *justification*, and *exemplification*. Other work analyzed the issue of focus of attention – i.e., what entities were being discussed at any given point – and how focus shifted throughout the course of a task-oriented dialogue [23]. Grosz & Sidner [24] presented an influential model that unified work on focus of attention, discourse or linguistic structure, and discourse intentions. Litman [25, 26] extended plan recognition models to handle dialogue and also integrated the use of linguistic information. Work on what makes a discourse coherent is important in designing effective human-computer interactions.

Until the mid 1980s, the primary emphasis of AI work on language was on language *interpretation*. However, since then the issue of language *generation* has received significant attention. The basic model again is classical AI planning; McKeown's TEXT system[27] is a good early example. A system tries to achieve *communicative goals*, such as to define or describe an object or to persuade someone to carry out an action. These goals are achieved through the use of plans or *strategies*. Strategies bundle a set of communicative acts, often related to each other by rhetorical relations (the type of relations identified by studies of coherence). A planning algorithm controls how strategies are selected, instantiated, and combined.

A major motivation for work in this area has been to improve the explanations generated by expert advisory systems. Researchers have explored ways to use modern direct manipulation interface techniques to move from one-way *explanation* to mixed initiative *interaction*. This work also has sought ways to exploit the unique capabilities of direct manipulation interfaces to give users the benefits of human-human interaction – particularly its contextual, mixed-initiative nature – without solving the extremely difficult representation and reasoning problems posed by human language use. This move has brought discourse planning work square into the arena of human-computer collaboration.

Johanna Moore's research [28, 29, 30, 31] explores ways to use *discourse context* and the resources of direct manipulation interface to improve explanation. Her work has included algorithms for allowing a system to refer back to the discourse context in constructing its explanations. She also has looked at interface techniques for *reifying* the discourse context, making it a visible, manipulable text object. This means that when users do not understand a system explanation, it is easy for them to follow up using menus that offer relevant followups for particular clauses. It also facilitates users referring back to previous discourse in constructing new questions. The fundamental prerequisite for this is that *the system must understand its own explanations.* This is done by recording the intentional structure produced by the discourse planner in the system's internal discourse context. This structure records how each clause contributes to achieving communicative goals and how the clauses relate to each other. Possible followup questions for particular intentions, common ways to form new questions based on previous explanations, and heuristics for selecting relevant followup optionsall help users in asking questions that use previous context.

Enabling *mixed initiative* dialogues, in which both user and system can influence the direction of the interaction, is another fundamental research issue. Alison Cawsey [32] has worked to make the explanation process interactive by interleaving explanation *planning* with *execution*, i.e., the production of textual output. To do this, she designed an architecture in which communicative goals are placed in an agenda in a priority order. Goals on the agenda are expanded to produce system output. However, the user can interrupt at any point to ask a clarification question. This can lead to new goals being added to the agenda and thus to clarification sub-dialogues. User interruptions also can lead to revised assumptions about user knowledge, which will influence how goals remaining on the agenda are realized. When a sub-dialogue is completed, the system can return to carrying out any original goals still on the agenda.

Another important line of work extends *text* planning to *multimedia* planning. In addition to planning the content of the communication, the system must select media for realizing each piece of information and must determine how to coordinate the presentation of information in different media. Research in this area has raised interesting representation and architecture issues. In Feiner & McKeown's COMET system [33], a content planner produces a representation of the information to be communicated in a media independent logical form (LF). A media coordinator annotates the LF to specify which information should be in text, which in graphics, and how text and graphics should be coordinated. Text and graphics generators can communicate with each other by further annotating the LF. Maybury [34] presents a framework for representing three types of communicative acts in a media independent manner, then using a planner to produce coordinated text and map displays. Hovy & Arens [35] have argued that traditional top-down hierarchical planners are inadequate for multimedia presentations, and that several interacting *reactive* planners (similar to Pengi [36]) are required.

Finally, innovative work has been done in integrating natural language and direct manipulation for user input[37, 38, 39, 40, 41]. Each of these modalities has its own strengths and weaknesses. Attempts to integrate the modalities are based on combining them synergistically to exploit the advantages and counter the disadvantages of each. Direct manipulation and form-based input structure the interaction, give users direct access to objects, make options apparent, and provide means of easy feedback. Language is appropriate for constructing descriptions of objects that are not visible, do not yet exist, or are hard to indicate by pointing. Contextual referring techniques lead to more efficient communication. This work also has investigated how difficult representation and reasoning issues, e.g., involving the interpretation of anaphora and the use of context, can be avoided through the judicious use of direct manipulation and graphical interface techniques.

> Like McRoy, *Brennan & Hulteen* (this issue) are concerned with problems in communication. Where McRoy focuses on how misunderstandings can be repaired, Brennan & Hulteen deal with a prerequisite to repair – how a system can generate feedback that can be used to recognize that repair is necessary. Their model was developed in the context of a system that responds to spoken commands from a user to perform tasks like placing a telephone call. It is based on the process of *grounding* [42, 43, 44], the continuous seeking and providing of evidence about what has been said and understood that characterizes human conversation. The model specifies a sequence of states that a system goes through in interpreting and responding to a user command. If the system fails to reach the final state (of successfully carrying out the user's command), it generates negative feedback based on the last state it did reach, thus initiating repair. If the system generates positive feedback (i.e., it thinks it understood) that indicates to the user that it actually misunderstood, the user can initiate repair. Generating appropriate positive feedback is an important issue – it would be tedious for the user if the system always gave feedback for each processing state it passed through successfully. Therefore, the system uses a *grounding criterion* to control when and how much feedback it gives. The grounding criterion is dynamically updated based on the dialogue history, task model, and physical environment. The model was tested in a Wizard of Oz study and user studies with a prototype Lisp implementation, and parts of the model were implemented in an Apple speech interface.
>
> *Stein & Maier* (this issue) present a framework for describing, organizing, and producing multimedia discourse. They apply ideas from the study of human communication, most notably speech act theory and Mann & Thompson's [22] Rhetorical Structure Theory to user-system multimedia communication. All user and system actions are interpreted as communicative acts, categorized based on their purpose. These atomic, domain-independent acts can be arranged into "dialogue scripts" that capture coherent, empirically attested interaction patterns for a particular kind of task. Dialogue scripts

are used to organize the interaction. For example, after responding to a user request, the system generates icons and buttons for each of the possible next actions defined by the script. Other options generally available to the user, such as "help", "history", or "information on next step" are treated as initiations of a sub-dialogue or meta-dialogue. The attraction of Stein & Maier's approach is that it offers a theoretically motivated, modality independent framework for organizing human-computer interaction.

Walker (this issue) describes a method for testing theories of human-computer collaboration and presents the results of tests she has performed. She begins by analyzing a corpus of human dialogues to identify collaborative strategies. One interesting phenomenon is the abundance of utterances in which agents communicate facts that are mutually believed or that follow from mutual beliefs. To understand why redundant information might be communicated (which incurs extra communicative effort), Walker turns to cognitive theories, which suggest, for example, that this is an efficient way to make information accessible in working memory. However, testing such hypotheses in human-human collaboration is extremely difficult or impossible. Thus, Walker developed Design World, a computational simulation environment in which two computational agents carry out a dialogue about a simple design task. Agents' performance is controlled by a set of cognitive parameters that measure the cost of retrieving information from memory, making an inference, and communicating a fact. Performance is measured by the quality of the design the agents create. Different communicative strategies are defined and tested as cognitive and task parameters are varied. The results shed light on how the cognitive properties of agents affect collaborative behavior. This work offers a rigorous methodology for developing computational models from analysis of human data and testing the models in a computational simulation environment.

A *KBS Letter* by *Haller* discusses the use of different collaboration modes – assignments of roles and responsibilities – in a plan-based explanation system. A *KBS Letter* by *Smith* presents a theory of dialogue based on problem-solving, identifies several levels of shared initiative, and describes an empirical study that evaluates the results of using different levels. A *KBS Letter* by *Guinn*, building on the same dialogue theory, presents an algorithm for controlling initiative that lets an agent decide between its preferred problem solving step and that recommend by its partner.

2.3 User Modeling and Adaptive Systems

A major premise of AI language work has been that human participants in an interaction model their partners and tailor their contributions based on these models. For example, in an instructional dialogue, an expert might choose shorter or longer object descriptions and select or omit certain information based on assumptions about the learner's knowledge. Research on *user modeling* originated in the natural language community but has expanded to include the general problem of endowing interactive computer systems with the ability to model individual users and adapt their behavior based on these models [45, 46, 47, 48, 49].

Three fundamental issues in an adaptive system (derived from [49]) are:

- *Use* – How does the system use information about users? That is, what useful adaptations in its behavior can it make based on a model of a user?
- *Representation and Reasoning* – What information about a user is required? How should that information be represented? What sorts of reasoning facilities are appropriate for user information?
- *Acquisition* – How can the necessary information about a user be obtained?

1. *Use.*
The classic use of and original motivation for user model information was in natural language dialogue. Researchers showed how a system could tailor descriptions of objects [50], volunteer additional information [51], and correct user misconceptions [51, 52, 53], among other benefits. Recently, researchers also have explored ways that user models can be used in direct manipulation interfaces. For example, Goodman and Litman [54], focusing on plan-based user models, list uses that include (1) constraining user choices, e.g., by dimming out irrelevant menu items, (2) automatic task completion, e.g., to fill in plan steps known to follow from the plan attributed to the user, and (3) error prevention, e.g., to prevent users from taking actions that are not on the path to their goal. Other researchers have used user models to aid in hypertext navigation [55], adapt hypertext to a user's level of expertise [56], and filter email to people likely to be interested in it [57].

For most of the history of user modeling research, benefits of adapting system behavior based on a user model were claimed rather than demonstrated empirically. However, Kobsa [47] cites several recent empirical evaluations, including results that show that adapting hypertext can improve user comprehension and search speed [56] and that navigational assistance for hypertext also can improve search speed [55].

2. *Representation and Reasoning.*
Kobsa [47] identifies three major types of knowledge about users that could

be useful to a system, (1) *stereotypes* – subgroups of a user population likely to possess "homogeneous application-relevant characteristics", (2) users' plans, goals, and other intentional attributes, and (3) users' preferences, e.g., concerning types of email messages they are likely to be interested in.

The stereotype approach [49, 58, 59, 60] is popular and fairly easy to implement. A designer must define appropriate subgroups of the user population (the stereotypes), identify user behaviors that enable a system to categorize users into a subgroup, and represent the set of features that characterizes each stereotype and arrange the stereotypes into a hierarchy. Plan-based user models have the advantages and drawbacks of plan recognition, as discussed above. Work on modeling user preferences has a somewhat different flavor, for example, in exploring connectionist, learning, or statistical techniques.

A wide variety of representational formalisms have been used, including frames, attribute-value pairs, Prolog, and first order logic. Information about a user rarely can be guaranteed to be correct. Since one assumption about a user often leads a system to make other inferences, when an assumption is revised, the validity of other conclusions the system has drawn must be examined. This makes *truth maintenance* reasoning important.

3. *Acquisition.*
Information about users may be acquired *explicitly* – by engaging a user in an interaction expressly designed to acquire information – or *implicitly* – inferring information based on user actions [49]. Both methods have drawbacks. If users must answer system questions or fill out a form, they may find this obtrusive and may have a difficult time characterizing themselves accurately. On the other hand, implicit acquisition can be a difficult computational task, depending on the type of user model being constructed. Plan recognition, as previously mentioned, is a very difficult computational problem. Among other things, it is difficult to know when a user is starting a new plan (as opposed to continuing the current one), users may suspend and resume plans, actions may be part of more than one plan, and there may be multiple plans for a single goal [47]. Stereotypes can be easier to recognize – each stereotype generally has a triggering condition that, when satisfied, leads the system to categorize the user as a member of that stereotype. And some representations of user preferences are fairly simple, and thus can be computed easily. For example, simple statistics on what messages a user reads in Net News may allow a system to filter the messages the user sees in the future [61].

3 The *Human Complementary* Approach

The advent of direct manipulation interfaces, the growing usage and complexity of personal computer application software, and the coalescence of the Human-Computer Interaction research community in the early 1980s led to another perspective on human-computer collaboration. The *Human Complementary* approach seeks to improve human-computer interaction by making the computer a more intelligent partner. This is a very *pragmatic* goal, in contrast with the more philosophical goals that motivate the *Human Emulation* approach. However, achieving this pragmatic goal requires fundamental research into people's cognitive and perceptual abilities, how people work as individuals and in groups, and computational representation, reasoning, interaction, and information presentation techniques. This research assumes that humans and computers have fundamentally asymmetric abilities. To create systems that are intelligent partners, researchers invent interaction paradigms that divide responsibility between human users and computer systems in ways that exploit the strengths and overcome the weaknesses of both partners. Researchers also explore ways to use modern interface technology to design communication methods that are natural for both partners.

> A *KBS Letter* by *Stolze* discusses the importance of general perspectives for system design, describes the *information processing* and *situated action* perspectives, and advocates that designers integrate insights from multiple perspectives.

3.1 Direct, Incremental Specification of Goals and Plans

Many researchers have looked for ways to bypass the difficult problem of intent recognition, which requires making inferences about a user's mental state based on observed actions. We have encountered plan recognition and user modeling as instances of this problem. In the Intelligent Tutoring Systems (ITS) field, the problem of *student modeling* is very similar to user modeling. As traditionally formulated, it too involves difficult issues in intent recognition as well as the diagnosis of misconceptions.

However, ITS researchers have been active in designing interaction techniques that allow users to express their intent directly, rather than requiring the system to guess it [62, 63, 64].[1] By analyzing how a class of users works within a particular task domain, say symbolic integration, financial analysis, or medical diagnosis, researchers develop a set of goals and plans for achieving these goals. These goals and plans are represented formally in a knowledge representation language and also represented graphically as objects in a direct manipulation interface [40]. Users then directly specify their goals and plans. This benefits both user and system. The user is given a medium for making problem-solving explicit, rather than having to do it mentally or using

[1]Within the field of ITS, there is a distinction between systems that teach students a fixed curriculum and those that provide a *learning environment* in which students choose their own tasks to work on, and the system provides assistance as appropriate. The former type of system can fix the goals that a student works on; we are interested here in the latter type of system, where determining a student's goal becomes an important issue.

non-computational aids like paper and pencil. The system gains access to a high-level specification of what it is that the user is trying to accomplish, thus simplifying the computations required to play a useful role in the interaction. For example, computations to track and display finished and unfinished steps in the plan, to fill in low-level details required in executing a plan, and to determine whether a plan is inappropriate are very useful and much simpler than plan recognition. *Checklists* [65, 66], computerized versions of the everyday to-do list, are a specific interaction resource used for organizing interaction and tracking and displaying progress toward a goal.

Systems by Self for logic tutoring [63] and Singley for algebra rate of change problems [67] explored the use of "goal posting". For example, in Singley's system, a user selects a goal such as "find dp/dt in terms of t". She next chooses a plan operator for achieving the goal, such as using the chain rule. Several subgoals might have to be satisfied before the operator can be applied. At all times, the system keeps track of which goals have and have not been satisfied, visually differentiating the current goal, satisfied goals, and unsatisfied goals. Empirical studies showed that the goal posting technique improved user performance and facilitated learning.

The goals and plans in this type of system generally are very domain specific. The power of domain-specificity comes at a cost – each new domain requires a different set of goals and plans and often a different set of visual representations. Researchers have attacked the cost of developing domain-specific systems in several ways.

One approach is to take a multi-level or "layered" approach in which plans are arranged in hierarchies. For example, in Bonar & Liffick's BRIDGE tutor [62], the highest level is a set of English phrases derived from studies of how novice programmers conceive of procedures. A second level consists of programming plans represented graphically as icons, which can be arranged into various well-defined configurations. The third level is the Pascal programming language. In such an approach, lower levels are more general purpose, and higher levels are more specialized, encapsulate domain and task knowledge, and are easier to use. Entities at one level are built from entities and operations of the next lowest level.

A second approach is found in the work of Bonnie Nardi and colleagues at Hewlett-Packard [68, 69]. They carried out empirical studies that illustrated the power of domain-specific programming systems such as spreadsheet formula languages and CAD extension languages. They then defined a set of common *visual formalisms*, such as tables, charts, graphs, and control panels, and created a *toolkit* for building domain-specific systems that apply and combine the appropriate components.

In addition to techniques for directly specifying goals, another important notion is the *incremental* specification of goals. People typically do not form precise definitions of goals to accomplish, then plan to achieve their goals, then carry out their plans. Rather, acting, planning, and forming and pursuing goals are interleaved. A number of interaction techniques have been explored that support incremental specification of goals in exploratory activity. The *retrieval by reformulation* [70, 71, 72] information retrieval technique interleaves query definition, querying, and evaluation of results. Experiments with critics (discussed in detail in the next section) have shown that users may refine their goals based on the delivery of advice from a system about user actions. They begin by specifying information they know and care about, then gradually refine and elaborate it in response to system advice.

3.2 Dividing Responsibility: The Critic Paradigm

Determining an effective division of responsibility between a user and an intelligent computer system is a fundamental issue in human-computer collaboration. In the systems we have just discussed, a user is responsible for selecting high-level plans, and the system is responsible for performing low-level details and tracking and displaying progress. *Critiquing* [73, 74, 75, 76, 77, 78, 79, 80] is another well known paradigm for collaboration that addresses the division of responsibility issue. Intuitively, a *critic* is a program that "looks over the shoulder" of users as they perform tasks in a computational environment and offers critiques (advice) from time to time. Critics compute their advice by using a domain knowledge base to examine the actions users perform and the products they create.

Critics exploit the asymmetry in human and computer abilities. Humans have common sense knowledge and know the tasks they are trying to accomplish using a computer system. Computers can be given much specialized domain knowledge and excel at maintaining consistency and in bookkeeping operations. Therefore, in a critiquing interaction, humans select goals to pursue, attempt to achieve the goals, and retain control of the interaction. Critics detect potential problems in user's problem solving and suggest solutions, propose additional relevant issues to consider, and automatically perform routine or low-level aspects of the problem solving.[2] Users then evaluate system critiques and decide how to respond, based again on their knowledge of the task.

It is important to realize that critics do not necessarily solve problems for users. Rather, their role is to stimulate and inform user problem solving [74, 79, 81]. As Fischer and colleagues put it [74](pp 124, 126), they must present a "reasoned opinion about a user's product or action" and must "recognize and communicate debatable issues concerning a product". Critics also have access to a user's (partial) solutions in computing their assistance. That

[2]Relying on the strict definition of a critic as a system that critiques user actions, one might question whether a system that does problem solving should be called a critic. However, the problem solving of critics such as those described in [81, 79, 80] is done in response to user actions and is presented to users for judgement and possible modification. Therefore, I believe that it does not violate the spirit of the critiquing approach to allow critics some limited problem solving ability.

critics have access to user solutions and do not need to solve complete problems automatically distinguishes them from expert systems [77].

The notions of presenting a *reasoned* opinion and of issues being *debatable* emphasize that in many interesting domains there is no "right answer". Experts may disagree, evaluation criteria may be somewhat subjective, and different factors may trade off against each other. In such cases, a critic's role is to engage the user in a reasoned "argument" in which issues are made explicit and alternative solutions and their pros and cons are considered [82]. Ideally, an "argumentation" interaction such as this should lead to learning on the part of both the human and computer partners; this issue is discussed below.

Critics have been widely used in design applications. For example, in Janus [74], users design a kitchen floor plan by selecting and positioning appliances. Critics use expert knowledge about kitchen design to detect potential problems with the floor plan such as the stove and refrigerator being too far apart. Both Silverman[77] and Terveen[78, 79, 80] discuss critics that aid users in knowledge-base construction. Rules that ensure knowledge structures are unambiguous, consistent, and complete are used to critique a user's partial specifications.

One of the advantages of the critiquing approach is that it is fairly simple to implement. The minimum requirements are *product analysis* and *presentation* components to compute and deliver the critiques [77]. Of course, more complexity may be added: for example, Fischer et al's [74] process model includes goal recognition, user modeling, and user modification capabilities. However, these capabilities can be added incrementally to the basic approach to ensure a favorable ratio of cost (of knowledge engineering and runtime computation) to benefit (enhanced user performance). In addition, both more and less complicated versions of these capabilities exist and can be applied. For example, goal recognition can be done the hard way, i.e., using AI plan recognition techniques. However, a very simple technique is to hard-wire in one or more domain goals. As discussed above, a third approach (of intermediate complexity) is to represent domain goals and plans in the interface and allow users to specify them explicitly. Fischer and colleagues [75, 76] discuss how combining a goal specification component with a critiquing system improves the quality of system advice.[3] For another example, product analysis may be done *analytically* – checking products with respect to predefined features and consequences – or via *differential analysis* – comparing the user's solution to one generated by the system using an expert module. The second approach is more complicated and works only when the domain can be formalized sufficiently to allow the system to compute full solutions.

> *Nakakoji & Fischer* (this issue) present a model of human-computer collaboration in design that integrates specification of design require-

[3]This approach is a form of *explicit* user model acquisition. Users respond to a set of domain-specific questions, then rate the importance of each factor. The system tailors its advice based on the user answers and ratings.

ments, construction of design artifacts, critiquing, and knowledge acquisition. Designers work by specifying requirements that their designs must satisfy and building artifacts in the construction area. The system uses the requirements and (partial) design artifacts to deliver three types of assistance: (1) messages indicating potential problems, (2) indices into an argumentation base that explains issues raised by the current design, and (3) a set of examples of previous designs, retrieved from a catalog by their relevance to the current design. When critics have access to user requirements, they can deliver advice more tailored to the particular design situation, allowing a designer to focus on its distinctive features. Empirical studies showed that system assistance is useful even if designers disagree: for example, they may articulate reasons to disagree or conditions that define more precisely when the system's advice is valid. When designers do articulate new knowledge, they can add it to the system by modifying the argumentation, adding new items to the requirements component, modifying items in the design palette, or adding new design analysis rules. Criticism thus leads to the growth of user and system knowledge (see [83, 84] for other perspectives on the growth of knowledge through system use).

Rogers (this issue) presents a model and implemented system of human-computer collaboration in visual problem solving – tasks involving the understanding and interpretation of visual images. Her work began with extensive empirical study of how radiologists use chest x-rays in diagnosis, and she developed a general methodology for analyzing cognitive data to guide system design. The studies identified types of visual objects radiologists looked for, visual features used to detect objects, the role of attention and expectations, types of reasoning performed, evidence used in reasoning, and common errors. These results served as the basis for a cognitive model of visual problem solving and a system architecture. A blackboard architecture was judged appropriate because of the diverse sources of knowledge and types of reasoning involved and the need for flexible, opportunistic problem solving. The blackboard approach allows both user and system to offer interpretations, hypotheses, and suggestions and to query each other. An important role for the system is to present suggestions that help avoid common types of oversights and errors. Empirical studies showed that using the system increased solution quality. As in Walker's paper, Rogers develops a computational model based on close analysis of human collaborative behavior; however, where Walker tested her model through computational simulation, Rogers tested hers by performing user studies.

3.3 Visual Objects as Communication Medium and Context

One of the key goals of the *Human Complementary* approach has been to explore ways to use new interface technology for effective human-computer communication. The direct manipulation interface paradigm stimulated much interest with its ability to constitute a *model world* for a particular domain, in which objects and relationships are represented graphically, and actions are performed by manipulating the objects.

In Don Norman's work on cognitive artifacts [11], he showed how artifacts such as simple to-do lists expand human cognitive capacities by serving as *external memories*. Artifacts can store more information than can be stored in short term memory, they store it permanently, and the spatial arrangement of items can represent conceptual relationships. The use of interface representations as external memories is exploited to some degree by all direct manipulation interfaces. For example, the desk top metaphor for file systems makes it unnecessary for users to remember the names of their directories and files or issue a command to list them before they can perform an operation. Intelligent systems that dynamically compute information to communicate to the user present opportunities for a more sophisticated type of external memory. An additional motivation for exploring the communicative potential of direct manipulation techniques has been to avoid some of the problems in language interaction, such as constructing references and instructions, deciding whether and when a user should be interrupted, managing initiative, and ordering the information to be communicated.

Several researchers have explored ways to deliver information that exploit the interactive potential of computational media and avoid the problems faced by language-based communication [79, 80, 85, 86, 87, 88]. The basic idea is to communicate as much information as possible by modifying the display of graphical objects rather than generating text. For example, the HKE system [78, 79, 80] computes various types of critiques as a user adds and links objects in a knowledge diagram, including *troubles* – inconsistencies that must be addressed – and *suggestions* – optional issues to consider. Color, shading, and fonts are used to modify the display of objects in the diagram to communicate this information. DETENTE [85] partially automates task management and uses a similar set of display techniques to communicate task status. In HKE and DETENTE, the work objects on the user's screen were transformed into an *implicit agenda*, from which users could "read off" work they needed to do. The PetriNED system [87] detects problems as a user constructs a Petri net. In addition to modifying the visual properties of objects, it also changes the shape of the cursor and uses lines to indicate potentially problematic relationships between objects.

The benefits of visual information delivery include (1) reducing the need for language generation, (2) leaving initiative with users – they can deal with advice whenever they want by interacting with the appropriate display object[4], (3) avoiding unnecessary sequentiality – a system can display many critiques at once, and users can respond to them in whatever order they want, and (4) avoiding memory errors – it is much less likely that users will forget about an issue when it is encoded persistently as a visible property of a display object. Of course, not all interaction can be managed around objects. For example, in HKE, users can interact with a visually distinguished object to get a *repair resource* that further explains the issue(s) involving the object, encapsulates a set of methods for resolving the issue, and guides the user through the process of selecting and applying a resolution method. These resources are *reified dialogues*. They are associated with graphical objects, and users can return to them and re-interact with them whenever they desire, e.g., to change their responses and take an alternative path through the dialogue.

3.4 Adaptation through Collaboration

When two people work together to solve a problem, adaptation and learning occur naturally. For example, if two children are working together to solve an algebra word problem, one might offer a solution. In response to the other's questions, she would be forced to articulate how she reached the solution, justify the steps, state assumptions, etc. A similar dynamic occurs when several professionals work together. Each might offer a solution. They then would have to identify the rules and principles they used to reach their solutions, determine how the two solutions related to each other, and identify exceptions to rules and tradeoffs between principles. Notice how both partners benefit from this process, one from clarifying and elaborating her own knowledge, the other from access to a much richer knowledge structure.

Researchers have explored ways to bring comparable adaptation and learning to human-computer collaboration, for both the human and computer partner. The critiquing paradigm naturally offers a form of learning to users [78, 89, 90, 91]. That is, a well-designed critic offers an alternative perspective on what a user has done by pointing out potential problems, suggesting additional relevant issues to consider, and making reasonable guesses to fill in low-level details. This brings not just the immediate benefit of improving the current problem solving process; it also exposes users to (potentially) new knowledge that they can apply themselves in the future without need for system critiquing.

Intelligent Tutoring Systems researchers have explored the notion of *collaborative learning* [92, 93]. They have extended the traditional two agent model of ITS – a learner and automated tutor – by adding a *simulated peer* or *co-learner*. This work draws on pedagogical and psychological research that details the

[4] Of course, when it truly is necessary, e.g., a dangerous situation has been detected that the user must deal with, a system still can take the initiative and force the user to respond. However, visual information delivery makes this an option that the system designer (or the system itself) can select, rather than the default behavior.

benefit of social interaction for individual learning. In Vygotskian [94, 95, 96] terms, a dialogue or argument is *internalized*, leading to enhanced knowledge for the individual. The technical challenge for producing a co-learner is that it must *learn* to keep the level of its knowledge relatively close to that of the human learner. Both machine learning techniques and "simulated" learning, in which the co-learner advances through a pre-programmed sequence of knowledge states, have been explored.

Other researchers have investigated additional techniques that enable systems to adapt and learn from the collaboration process, including:

- Extending argumentation dialogues to allow users to modify the system's argument base.
  The IBIS method and derivatives [97, 98] are commonly used to organize argumentation. Information is represented as hypertext, with a specialized set of nodes such as *issues*, *answers* to the issues, *arguments* pro and con for the answers, and relationships between the nodes. As users interact with such an argument structure, it is easy to allow them to add new issues, answers, and arguments and to create new links. Subsequent users then receive the benefit of interacting with the enhanced argumentation structure.

- Providing *end-user modification* facilities that allow users to modify system knowledge structures.
  The limit of the type of argumentation structure just described is that is largely up to users to traverse it. If a system, say a critic, is to automatically direct users to relevant arguments, it needs *rules* that map from user actions and situational features to the argument base. In general, any knowledge representation that a system uses to drive its interaction with the user may have to be modified. The Information Lens/Object Lens/Oval line of research by Malone and colleagues [99, 100, 101] explored techniques for end users to edit and modify objects in frame-based knowledge bases and to write and modify rules that used a domain-specific vocabulary of conditions and actions. Fischer & Girgensohn [102] and Candy, O'Brien & Edmonds [89] also explored techniques for letting users modify object hierarchies, properties, and rules.

- Using machine learning techniques to infer patterns in users' activities.
  People are better at modifying existing artifacts than creating new ones. Therefore, rather than requiring users to create new rules and objects from scratch, it would be better for a system to infer rules, then present them to users for approval or any necessary modification. Bailin [103] describes a software design system in which designers demonstrate faulty design patterns and the system uses failure-based learning to learn detection and repair rules. Crow & Smith [104] describe a system that monitors user actions in a command language interface and uses pattern recogni-

tion techniques and simple domain knowledge to recognize patterns of commands. The system then engages the user in a dialogue, allowing him or her to verify and modify the patterns, after which they are available to the user as new "meta" commands. Maes and colleagues [105, 106] also have explored machine learning techniques for letting agents learn about users' personal preferences, e.g., for scheduling meetings.

Eisenberg (this issue) explores solutions to the tension between short-term and long-term goals of a user of an interactive system. In the short term, performing the task at hand quickly and easily is paramount. In the long term, however, mastering the tool and learning the domain may be more important. Thus, from one perspective, *learnability*, *ease of use*, and *efficient problem solving* are central, while from the other perspective, *expressiveness*, *flexibility*, and *learning* are key. Eisenberg addresses this dilemma with the notion of *programmable design environments* (PDES). PDEs combine the domain-oriented design environment [107, 108] and critiquing paradigms with a "domain-enriched" programming language. The SchemeChart PDE integrates direct manipulation tools for creating charts and graphs with a version of Scheme enriched with drawing primitives. Example charts and "query-able objects" index Scheme code used to produce them, thus providing entry points for learning programming concepts as users' needs or curiosity lead them in this direction. Critics also help users to design better charts and to learn Scheme. The most important aspect of the SchemeChart PDE is how it integrates mechanisms that facilitate ease of use and effective problem solving with support for learning the Scheme programming language. Users learn as they are motivated, in the context of concrete problems they care about [69].

4 Toward a Unified Approach to Human-Computer Collaboration

The goal of this section is to sketch the shape of a unified approach to human-computer collaboration Our discussion highlighted differences between the *Human Emulation* and *Human Complementary* approaches, but also revealed potential convergences. Table 2 summarizes four themes from the the *Human Emulation* approach, four themes from the *Human Complementary* approach, and four themes from the proposed unified approach.

Table 2 goes about here

The first two lines of the table show two unresolved differences between the approaches:

- *intent recognition* vs. *intent specification*,
- *symmetric agents* vs. *asymmetric agents*.

The third and fourth lines show differences that have been reconciled into unifying themes:

- *natural language* vs. *direct manipulation* $\Rightarrow$ *natural communication*,
- *adaptive systems* vs. *adaptable systems* $\Rightarrow$ *collaborative adaptation*.

Finally, the last two items in the table show two unifying themes that cut across previously discussed issues and distinctions:

- *reification*,
- *balancing representation and reasoning with interaction*.

Let's first consider the two unresolved differences, then discuss the four unifying themes.

Intent recognition vs. intent specification.
Human Emulation models have presumed that agents must infer each other's intent; thus, the development of algorithms and heuristics for inferring mental state based on observed actions is central. While these algorithms and heuristics are domain independent, they require much rich domain-specific knowledge to work. *Human Complementary* work has attempted to avoid the problem of intent recognition by developing domain-specific languages and interfaces that allow users to specify their goals and plans directly.

Symmetric vs. Asymmetric agents.
Human Emulation models presume rational agents with symmetric abilities, in particular, to form goals, to plan to achieve goals, and to infer the plans of their partners. *Human Complementary* work assumes that computer and human agents have asymmetric abilities. Responsibility for accomplishing a task is divided to maximize the strengths of each agent, with computer agents designed to complement humans.

Theme 1 – Reification.
A great power of graphical interfaces is to *reify*, making previously invisible entities visible and providing a concrete representation of abstractions. User and system share access to entities in the interface and can manipulate these entities. Our interest is in reifying the *models* and *processing* of a collaborative system. We have seen examples of the following objects being reified:

- *goals, plans, and tasks* [40, 62, 63, 64, 66, 67, 85, 91] – Users can directly examine and select goals, plans, and tasks, reducing the need for system inference. Once selected, these structures become a visible context useful for tracking and displaying the state of problem solving and giving users control over the problem solving process.
- *inference steps* [78, 79, 80] – Users are made aware of and can respond to system inferences by accepting, modifying, or rejecting them.
- *dialogue and dialogue context* [28, 29, 30, 31, 37, 38, 40, 78, 79] – This aids users in constructing followup questions or new questions that refer to previous context. Users can return to a previous topic or explore a different path through the dialogue.
- *user models* [63, 104] – Reifying user models enables *collaborative adaptation*, a theme considered separately below.

Theme 2 – Balancing representation & reasoning and interaction.
Representation and reasoning are necessary for a collaborative system. However, some models of collaboration subsume many of the hardest and most fundamental problems in AI and cognitive science. For example, Self [63] describes how the "student modeling problem" for Intelligent Tutoring Systems can expand to encompass control of reasoning, representation of commonsense knowledge, plan recognition, mental models, episodic memory, and individual differences. Similar remarks could be made about work in user modeling and natural language discourse.

If collaboration *inherently* involves solving these hard problems, then we have no choice but to make the attempt. However, I think there is a way out – to exploit the power of interaction to overcome limits on reasoning. We have encountered this approach in various guises. For example, Johanna Moore's work illustrates this point with respect to user models. Her systems exploit user model information when available; however, she notes the practical difficulties in obtaining reliable information about users and reasoning with such information. Therefore, rather than concentrating solely on developing reasoning techniques that produce just the right explanation given just the right user model, she has explored ways to use interaction and feedback to make communication successful.

The power of interaction also is seen in an emphasis on incremental processing. For example, Grosz, Sidner, & Lochbaum's SharedPlan model explicitly rejects the notion that one agent must *recognize* a previously existing and complete plan of another agent (a difficult inferential task). Rather, plans are constructed incrementally through interaction. Within a very different tradition, critics have been shown to stimulate users to refine their plans incrementally. Users begin by specifying information they know and care about, then gradually fill in details and refine their original specifications in response to system advice.

Finally, the retrieval by reformulation paradigm is an incremental method for specifying a query (which embodies the user's goal of retrieving certain information) that also depends on reifying intermediate formulations of the query.

Theme 3 – Natural communication.
Once upon a time, natural language and direct manipulation were seen as two distinct and often competing interaction paradigms. However, investigations of what makes *natural communication* [109] effective have helped to shape an emerging synthesis. Studies of human conversation revealed the importance of properties such as shared context, mixed initiative, clarification sub-dialogues, and mechanisms for the management of trouble – all properties that are independent of the communication medium. We have discussed research that applied these properties to create multimedia dialogues, integrating the strengths of natural language and direct manipulation and reifying discourse context in a direct manipulation interface. We also have seen how these insights have been exploited by explorations in the use of modern interface techniques to organize interaction, e.g, structuring dialogue around the graphical objects that comprise the shared work context and reifying dialogues as interface objects. The common goal is to combine the best of natural language communication and modern interface technology to create interactions that are natural for both people and computers.

Theme 4 – Collaborative adaptation.
Within the *Human Emulation* approach, there has been a heavy focus on techniques for systems to model their users and automatically adapt their behavior to the individual user. Within the *Human Complementary* approach, the emphasis has been more on methods for allowing users to adapt systems. However, there is work that points the way toward synthesizing these two perspectives [110]. Adapting system behavior to an individual user should itself become a collaborative activity, with user and system playing appropriate roles.

The modeling process may begin with the system observing the user and doing its best to create a model. This might result in categorizing the user as belonging to some stereotype, inferring the user's plans, generating a pattern that summarizes the user's actions, or producing an email or Net News filter [61]. This model then should be *reified*, making it available for the user to inspect and edit as necessary. Further, it may be possible for the user to request a "simulation" of system behavior to understand better how the user model drives system adaptation. Thus, adaptation is an incremental, collaborative activity, in which the user model becomes more refined and accurate over time.

To summarize and conclude, we have taken a critical look at the field of human-computer collaboration. After deriving a set of fundamental issues from a definition of collaboration, we considered how two major, historically distinct approaches addressed the issues. Finally, we sketched out a set of themes that we suggest should be central to a unified approach to HCC.

Acknowledgements

I would like to thank all the participants in the 1993 AAAI Fall Symposium on Human-Computer Collaboration for three days of stimulating and productive discussion. I especially thank the other members of the organizing committee, Gerhard Fischer, Lewis Johnson, Johanna Moore, Chuck Rich, and Candy Sidner, for all their work in making the symposium a success. I also am grateful to them for their thoughtful comments on an earlier version of this paper.

References

[1] Austin, J L *How to Do Things with Words*' Harvard University Press, USA, (1962)

[2] Grice, H P 'Utterer's Meaning and Intentions' *Philosophical Review* Vol 68 No 2 (1969) pp 147-177

[3] Grice, H P 'Logic and Conversation' *in* Cole, P and Morgan, J L, (Eds.) *Syntax and Semantics: Speech Acts* Academic Press, USA (1975)

[4] Searle, J R *Speech Acts: An Essay in the Philosophy of Language* Cambridge University Press, UK (1969)

[5] Schön, D *The Reflective Practitioner* Basic Books, USA (1983)

[6] Suchman, L A 'Office Procedures as Practical Action: Models of Work and System Design' *ACM Transactions on Office Information Systems* Vol 23 No 4 (1983) pp 320-328

[7] Suchman, L A *Plans and Situated Actions: The Problem of Human-Machine Communication* Cambridge University Press, UK (1987)

[8] Lave, J *Cognition in Practice* Cambridge University Press, UK (1988)

[9] Hutchins, E L 'The Technology of Team Navigation' *in* Galegher, J, Kraut, R, and Egido, C (Eds.) *Intellectual Teamwork: Social and Technological Foundations of Cooperative Work* Lawrence Erlbaum Associates, USA (1990)

[10] Hutchins, E L 'The Social Organization of Distributed Cognition' *in* Levine, J, Resnick, L B, and Teasley, SD (Eds.) *Perspective on Socially Shared Cognition* APA, USA (1991) pp 283-307

[11] Norman, D A 'Cognitive Artifacts' *in* Carroll, J (Ed.) *Designing Interaction: Psychology at the Human-Computer Interface* Cambridge University Press, UK (199) pp 17-38

[12] Allen, J F 'Recognizing Intentions from Natural Language Utterances' *in* Brady, M and Berwick, R C (Eds.) *Computational Models of Discourse* MIT Press, USA (1983) pp 107-166

[13] Allen, J F and Perrault, C R 'Analyzing Intention in Utterances' *Artificial Intelligence* Vol 15 (1980) pp 143-178

[14] Cohen, P R and Perrault, C R 'Elements of a Plan-Based Theory of Speech Acts' *Cognitive Science* Vol 3 (1979) pp 177-212

[15] Perrault, C R and Allen, J F 'A Plan-Based Analysis of Indirect Speech Acts' *Computational Linguistics* Vol 6 No 3 (1980) pp 167-182

[16] Grosz, B J and Sidner, C L 'Plans for Discourse' *in* Cohen, P R, Morgan, J L, and Pollack, M E (Eds.) *Intentions in Communication* MIT Press, USA (1990)

[17] Grosz, B J and Kraus, S 'Collaborative Plans for Group Activities' *International Joint Conference on Artificial Intelligence, IJCAI'93* (1993)

[18] Lochbaum, K E, Grosz, B J, and Sidner, C L 'Models of Plans to Support Communication: An Initial Report' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 485-490

[19] Pollack, M E 'Inferring Domain Plans in Question-Answering' *PhD Thesis* Department of Computer and Information Science, The University of Pennsylvania, USA (1986)

[20] Hobbs, J 'Coherence and Co-reference' *Cognitive Science* Vol 3 No1 (1979) pp 67-82

[21] Hovy, E H 'Planning Coherent Multisentential Text' *Proc. of Association for Computational Linguistics* (1988)

[22] Mann, W C and Thompson, S A 'Rhetorical Structure Theory: A Theory of Text Organization' *in* Polanyi, L, (Ed.) *Discourse Structure* Ablex, USA (1987) pp 85-96

[23] Grosz, B J 'Focusing and Description in Natural Language Dialogues' *in* Joshi, A, Webber, B, and Sag, I (Eds.) *Elements of Discourse Understanding*, Cambridge University Press, UK (1981)

[24] Grosz, B J and Sidner, C L 'Attention, Intentions, and the Structure of Discourse' *Computational Linguistics* Vol 12 No 3 (1986) pp 175-204

[25] Litman, D J 'Plan Recognition and Discourse Analysis: An Integrated Approach for Understanding Dialogues' *PhD Thesis* Department of Computer Science, The University of Rochester, USA (1985)

[26] Litman, D J and Allen, J F 'A Plan Recognition Model for Subdialogues in Conversation' *Cognitive Science* Vol 11 (1987) pp 163-200

[27] McKeown, K R *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text* Cambridge University Press, UK (1985)

[28] Moore, J D 'Responding to "Huh": Answering Vaguely Articulated Follow-Up Questions' *Human Factors in Computer Systems, CHI'89 Conference Proceedings* (1989) pp 91-96

[29] Moore, J D and Swartout, W R 'Pointing: A Way Toward Explanation Dialogues' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 457-464

[30] Carenini, G and Moore, J D 'Generating Explanations in Context' *Proc. International Workshop on Intelligent User Interfaces* (1993) pp 175-182

[31] Lemaire, B and Moore, J D 'An Improved Interface for Tutorial Dialogues: Browsing a Visual Dialogue History' *Human Factors in Computer Systems, CHI'94 Conference Proceedings* (1994) pp 16-22

[32] Cawsey, A 'Generating Interactive Explanations' *Proc. National Conference on Artificial Intelligence, AAAI'91* (1991) pp 86-91

[33] Feiner, S K and McKeown, K R 'Coordinating Text and Graphics in Explanation Generation' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 442-449

[34] Maybury, M T 'Planning Multimedia Explanations Using Communicative Acts' *Proc. National Conference on Artificial Intelligence, AAAI'91* (1991) pp 61-66

[35] Hovy, E and Arens, Y 'The Planning Paradigm Required for Automated Multimedia Presentation Planning' *Human-Computer Collaboration: Reconciling Theory, Synthesizing Practice* papers from the 1993 Fall Symposium Series, AAAI Technical Report FS-93-05

[36] Agre, P E and Chapman, D 'Pengi: An Implementation of a Theory of Activity' *Proc. National Conference on Artificial Intelligence, AAAI'87* (1987) pp 196-201

[37] Cohen, P R, Dalrymple, M, Moran, D B, Pereira, F C N, Sullivan, J W, Gargan, R A, Schlossberg, J L, and Tyler, S W 'Synergistic Use of Direct Manipulation and Natural Language' *Human Factors in Computer Systems, CHI'89 Conference Proceedings* (1989) pp 227-233

[38] Cohen, P R 'The Role of Natural Language in a Multimodal Interface' *Proc. of the Second FRIEND-21 Symposium on Next Generation Human Computer Interfaces* (1991)

[39] Neal, J G and Shapiro, S 'Intelligent Multi-Media Interface Technology' *in* [111] (1991) pp 11-44

[40] Tyler, S W, Schlossberg, J L, Gargan, R A, Cook, L K, and Sullivan, J W 'An Intelligent Interface Architecture for Adaptive Interaction' *in* [111] (1991) pp 85-109

[41] Wahlster, W 'User and Discourse Models for Multimodal Communication' *in* [111] (1991) pp 45-68

[42] Clark, H H and Schaefer, E F 'Contributing to Discourse' *Cognitive Science* Vol 13 (1989) pp 259-294

[43] Clark, H H and Brennan, S E 'Grounding in Communication' *in* Levine, J, Resnick, L B, and Teasley, S D (Eds.) *Perspective on Socially Shared Cognition* APA, USA (1991) pp 127-149

[44] Brennan, S E 'Seeking and Providing Evidence for Mutual Understanding' *PhD Thesis* Stanford University, USA (1990)

[45] Kobsa, A and Wahlster, W (Eds.) *Computational Linguistics, Special issue on User Modeling* Vol 14 No 3 (1988)

[46] Kobsa, A and Wahlster, W (Eds.) *User Models in Dialog Systems* Springer, Germany (1989)

[47] Kobsa, A 'User Modeling: Recent Work, Prospects and Hazards' *in* Schneider-Hufschmidt, M, Kuehme, T, and Malinowski, U (Eds.) *Adaptive User Interfaces: Principles and Practice* North-Holland, The Netherlands (1993)

[48] Kass, R and Finin, T 'Modeling the User in Natural Language Systems' *in* [45] (1988) pp 5-22

[49] Kass, R and Finin, T 'General User Modeling: A Facility to Support Intelligent Interaction' *in* [111] (1991) pp 111-128

[50] Paris, C L 'Tailoring Object Descriptions to a User's Level of Expertise' *in* [45] (1988) pp 64-78

[51] Chin, D N 'Intelligent Interfaces as Agents' *in* [111] (1991) pp 177-206

[52] McCoy, K F 'Reasoning on a Highlighted User Model to Respond to Misconceptions' In [45] (1988) pp 52-63

[53] Quilici, A, Dyer, M, and Flowers, M 'Recognizing and Responding to Plan-Oriented Misconceptions' *in* [45] (1988) pp 38-51

[54] Goodman, B A and Litman, D J 'On the Interaction between Plan Recognition and Intelligent Interfaces' *User Modeling and User-Adapted Interaction* Vol 2 (1992) pp 83-115

[55] Kaplan, C, Fenwick, J and Chen, J 'Adaptive Hypertext Navigation Based on User Goals and Context' *User Modeling and User-Adapted Interaction* Vol 3 No 3 (1993) pp 193-220

[56] Boyle, C D B and Encarnacion, A O 'An Adaptive Hypertext Reading System' *User Modeling and User-Adapted Interaction* (1993)

[57] Jennings, A and Higuchi, H 'A User Model Neural Network for a Personal News Service' *User Modeling and User-Adapted Interaction* Vol 3 No 1 (1993) pp 1-25

[58] Rich, E 'User Modeling via Stereotypes' *Cognitive Science* Vol 3 (197) pp 329-354

[59] Rich, E 'Stereotypes and User Modeling' *in* [46] (1989) pp 35-51

[60] Chin, D N 'KNOME: Modeling what the User Knows in UC' *in* [46] (1989) pp 74-107

[61] Stevens, C 'Helping Users Locate and Organize Information' *PhD Thesis* Department of Computer Science, The University of Colorado, USA (1993)

[62] Bonar, J and Liffick, B W 'Communicating with High-Level Plans' *in* [111] (1991) pp 129-156

[63] Self, J A 'Bypassing the Intractable Problem of Student Modeling' *in* Frasson, C and Gauthier, G (Eds.) *Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education* Ablex, USA (1990) pp 107-123

[64] Soloway, E, Guzdial, M, and Hay, K E 'Learner-Centered Design: The Challenge for HCI in the 21st Century' *Interactions* (April 1994) pp 36-48

[65] Lemke, A C and Fischer, G 'A Cooperative Problem Solving System for User Interface Design' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 479-484

[66] Terveen, L G and Wroblewski, D A 'A Collaborative Interface for Browsing and Editing Large Knowledge Bases' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 491-496

[67] Singley, M K 'The Effect of Goal Posting on Operator Selection' *Proc. of Artificial Intelligence in Education* (1987)

[68] Johnson, J A, Nardi, B A, Zarmer, C L, and Miller, J R 'ACE: Building Interactive Graphical Applications' *Communications of the ACM* Vol 4 (1993) pp 40-55

[69] Nardi, B A *A Small Matter of Programming* MIT Press, USA (1993)

[70] Williams, M D 'What Makes RABBIT Run? *International Journal of Man-Machine Studies'* Vol 21 (1984) pp 333-352

[71] Fischer, G and Nieper-Lemke, H 'HELGON: Extending the Retrieval by Reformulation Paradigm' *Human Factors in Computer Systems, CHI'89 Conference Proceedings* (1989) pp 357-362

[72] Yen, J, Neches, R, DeBellis, M, Szekely, P, and Aberg, P 'BACKBOARD: An Implementation of Specification by Reformulation' *in* [111] (1991) pp 421-444

[73] Miller, P *Expert Critiquing Systems: Practice-Based Medical Consultation by Computer* Springer-Verlag, USA (1986)

[74] Fischer, G, Lemke, A C, Mastaglio, T and Morch, A I 'The Role of Critiquing in Cooperative Problem Solving' *ACM Transactions on Information Systems* Vol 9 No 3 (1991) pp 123-151

[75] Fischer, G Nakakoji, K, Ostwald, J, Stahl, G, and Sumner, T 'Embedding Computer-Based Critics in the Contexts of Design' *Human Factors in Computer Systems, INTERCHI'93 Conference Proceedings* (1993) pp 157-164

[76] Fischer, G, Nakakoji, K, Ostwald, J, Stahl, G, and Sumner, T 'Embedding Critics in Design Environments' *The Knowledge Engineering Review Journal* Vol 4 No8 (1993) pp 285-307

[77] Silverman, B G 'Survey of Expert Critiquing Systems: Practical and Theoretical Frontiers' *Communications of the ACM* Vol 35 No 4 (1993) pp 106-127

[78] Terveen, L G 'Person-Computer Cooperation through Collaborative Manipulation' *PhD Thesis* Department of Computer Sciences, The University of Texas at Austin, USA (1991) (Also MCC Technical Report ACT-AI-048-91)

[79] Terveen, L G 'Intelligent Systems as Cooperative Systems' *International Journal of Intelligent Systems* Vol 3 No 2-4 (1993) pp 217-250

[80] Terveen, L G, Wroblewski, D A, and Tighe, S N 'Intelligent Assistance Through Collaborative Manipulation' *International Joint Conference on Artificial Intelligence, IJCAI'91* (1991) pp 9-14

[81] Clarke, A A and Smyth, M G G 'A Co-operative Computer Based on the Principles of Human Co-operation' *International Journal of Man-Machine Studies* Vol 38 (1993) pp 3-22

[82] Fischer, G, Lemke, A C, McCall, R and Morch, A I 'Making Argumentation Serve Design' *Human-Computer Interaction* Vol 6 No 3-4 (1991) pp 393-419

[83] Ackerman, M S and Malone, T W 'Answer Garden: A Tool for Growing Organizational Memory' *Proc. of Conference on Office Information Systems* (1990) pp 31-39

[84] Terveen, L G, Selfridge, P G, and Long, M D 'From Folklore to Living Design Memory' *Human Factors in Computer Systems, INTERCHI'93 Conference Proceedings* (1993) pp 15-22

[85] Wroblewski, D A, Hill, W C, McCandless, T 'DETENTE: Practical Support for Practical Action' *Human Factors in Computer Systems, CHI'91 Conference Proceedings* (1991) pp 195-202

[86] Hill, W C, Hollan, J D, Wroblewski, D A, and McCandless, T 'Edit Wear and Read Wear' *Human Factors in Computer Systems, CHI'92 Conference Proceedings* (1992) pp 3-9

[87] Stolze, M 'Visual Critiquing in Domain Oriented Design Environments: Showing the Right Thing at the Right Place' *Proc. of AI and Design* (1994)

[88] Terveen, L G and Selfridge, P G 'Intelligent Assistance for Software Construction: A Case Study' *Ninth Knowledge-Based Software Engineering Conference, KBSE'94* (1994)

[89] Candy, L, O'Brien, S M, and Edmonds, E A 'End-user Manipulation of a Knowledge-Based System: A Study of an Expert's Practice' *International Journal of Man-Machine Studies* Vol 38 (1993) pp 129-145

[90] Fischer, G 'Supporting Learning on Demand with Design Environments' *Proc. of the International Conference on the Learning Sciences* (1991)

[91] Fischer, G, Lemke, A C, and McCall R 'Towards a System Architecture Supporting Contextualized Learning' *Proc. National Conference on Artificial Intelligence, AAAI'90* (1990) pp 420-425

[92] Chan, T W and Baskin, A B 'Learning Companion Systems' *in* Frasson, C and Gauthier, G (Eds.) *Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education*, Ablex, USA (1990) pp 6-33

[93] Dillenbourg, P and Self, J A 'A Computational Approach to Socially Distributed Cognition' *European Journal of Psychology of Education* Vol 8 No (1992) pp 353-372

[94] Vygotsky, L S *Thought and Language* MIT Press, USA (1962)

[95] Vygotsky, L S *Mind in Society* Harvard University Press, USA (1978)

[96] Wertsch, J V (Ed.) *Culture, Communication, and Cognition: Vygotskian Perspectives* Cambridge University Press, UK (1985)

[97] Kunz, W and Rittel, H W J 'Issues as Elements of Information Systems' *Working Paper 131* Center for Planning and Development Research, University of California, USA, (1970)

[98] Conklin, J and Begeman, M 'gIBIS: A Hypertext Tool for Exploratory Policy Discussion' *Proc. of Computer-Supported Cooperative Work, CSCW'88* (1988) pp 140-152

[99] Malone, T W, Grant, K R, Turbak, F A, Brobst, S A, and Cohen, M D 'Intelligent Information Sharing Systems' *Communications of the ACM* Vol 30 No5 (1987) pp 390-402

[100] Lai, K Y, Malone, T W, and Yu, K C 'Object Lens: A Spreadsheet for Cooperative Work' *ACM Transactions on Office Information Systems* Vol 6 No 4 (1988) pp 332-353

[101] Malone, T W, Lai, K Y, and Fry, C 'Experiments with Oval: A Radically Tailorable Tool for Collaborative Work' *Proc. of Computer-Supported Cooperative Work, CSCW'89* (1992) pp 289-297

[102] Fischer, G and Girgensohn, A 'End User Modifiability in Design Environments' *Human Factors in Computer Systems, CHI'90 Conference Proceedings* (1990) pp 183-192

[103] Bailin, S 'Learning in a KBSE – The HENDRIX Tool (Demonstration)' *Eighth Knowledge-Based Software Engineering Conference, KBSE'93* (1993) p 200

[104] Crow, D and Smith, B 'The Role of Built-in Knowledge in Adaptive Interface Systems' *Proc. International Workshop on Intelligent User Interfaces* (1993) pp 97-104

[105] Kozierok, R and Maes, P 'A Learning Interface Agent for Scheduling Meetings' In *Proc. International Workshop on Intelligent User Interfaces.* (1993) pp 81-88

[106] Maes, P and Kozierok, R 'Learning Interface Agents' *Proc. National Conference on Artificial Intelligence, AAAI'93* (1993) pp 459-465

[107] Fischer, G 'Domain-Oriented Design Environments' *Seventh Knowledge-Based Software Engineering Conference, KBSE'92* (1992) pp 204-213

[108] Fischer, G, Girgensohn, A, Nakakoji, K, and Redmiles, D 'Supporting Software Designers with Integrated, Domain-Oriented Design Environments' *IEEE Transactions on Software Engineering, Special Issue on Knowledge Representation and Reasoning in Software Engineering* Vol 18 No 6 (1992) pp 511-522

[109] Fischer, G and Reeves, B 'Beyond Intelligent Interfaces: Exploring, Analyzing, and Creating Success Models of Cooperative Problem Solving' *Applied Intelligence, Special Issue on Intelligent Interfaces* Vol 1 (1992) pp 311-332

[110] Fischer, G 'Shared Knowledge in Cooperative Problem-Solving Systems – Integrating Adaptive and Adaptable Components' *in* Schneider-Hufschmidt, M, Kuehme, T, and Malinowski, U, (Eds.) *Adaptive User Interfaces: Principles and Practice* North-Holland, The Netherlands (1993)

[111] Sullivan, J W and Tyler, S W (Eds.) *Intelligent User Interfaces* ACM Press, USA (1991)

Collaboration Issue	*Human Emulation*	*Human Complementary*
1. Determining shared goals	goal/plan recognition	goal specification languages exploratory environments
2. Allocating responsibility, planning, and coordination	goal/plan recognition (collaborative) planning system as advisor	fixed, appropriate divisions of responsibility system as critic
	system as problem solving partner	
3. Shared context	(reified) discourse context	model world external memory implicit agenda
4. Communication	natural language NL & DM integration discourse planning multimodal dialogue	visual information delivery reified dialogues
5. Adaptation and learning	*adaptive* – user modeling	*adaptable* – end user mod. argumentation learning on demand collaborative learning
	machine learning	

Table 1: Collaboration Issues and Approaches

Human Emulation	*Human Complementary*	*Unified*
intent recognition	intent specification	
symmetric agents	asymmetric agents	
natural language	direct manipulation	natural communication
adaptive systems	adaptable systems	collaborative adaptation
		reification
		representation & reasoning vs. interaction

Table 2: Toward a Unified Approach to Human-Computer Collaboration