Beyond Intelligent Interfaces:

Exploring, Analyzing and Creating Success Models of Cooperative Problem Solving

Gerhard Fischer, Brent Reeves

CU-CS-510-91 January 1991

Department of Computer Science
University of Colorado at Boulder
Campus Box 430
Boulder, Colorado 80309-0430 USA

(303) 492-7514
(303) 492-2844 Fax
gerhard@cs.colorado.edu

ANY OPINIONS, FINDINGS, AND CONCLUSIONS OR RECOMMENDATIONS EXPRESSED IN THIS PUBLICATION ARE THOSE OF THE AUTHOR AND DO NOT NECESSARILY REFLECT THE VIEWS OF THE NATIONAL SCIENCE FOUNDATION

Exploring and Analyzing Success Models of Cooperative Problem Solving

Gerhard Fischer and Brent Reeves
Department of Computer Science, Institute of Cognitive Science
University of Colorado at Boulder
Boulder, CO 80309-0430

Abstract. Cooperative problem-solving systems are computer-based systems that augment a person's ability to create, reflect, design, decide, and reason. Our work is focused on supporting cooperative problem-solving in the context of high functionality computer systems. Based on the limitations of earlier prototype systems, we conducted an empirical study of a success model of cooperative problem-solving between humans in a large hardware store. Insights gained are currently used in the design of integrated, domain-oriented, knowledge-based design environments that serve as a new generation of cooperative problem-solving systems.

1. Introduction

Our goal is to establish the conceptual foundations necessary to use the computational power now becoming available to create cooperative problem-solving systems. We explore conceptual frameworks, methodologies, and technologies in order to exploit the unique opportunity offered by powerful computer systems. The purpose is to augment human potential and productivity.

We characterize some of our older system-building efforts, which addressed isolated aspects of cooperative problem-solving. Next, we briefly describe our application domain— high functionality computer systems. The core of the paper discusses an empirical study analyzing a success model of cooperative problem-solving among humans as it takes places between customers and sales agents in a large hardware store. We conclude by describing the lessons learned from this study and their impact on our future work.

2. Previous Efforts to Address Issues of Cooperative Problem-Solving

Effective human-computer communication is more than creating attractive displays on a computer screen; it requires providing the computer with a considerable body of knowledge about the world, about users, about tasks and about communication processes.

During the last six years we have developed a number of prototype systems. We briefly characterize our major system building efforts, including their design rationale and their contribution toward cooperative problem-solving.

ACTIVIST: Information Volunteering by the System. Humans often learn by receiving answers to questions that they did not or could not pose. The active help system ACTIVIST [Fischer, Lemke, Schwab 85]) volunteers information not asked for. ACTIVIST looks "over the shoulder" of a user working with an editor, infers from user actions the plan that the user wants to achieve, and compares it with its own plan.

SYSTEMS' ASSISTANT: Information Volunteering by Users. Despite the fact that communication capabilities such as *mixed-initiative dialogues* [Carbonell 70a] have been found to be crucial for cooperative systems, the progress in achieving them has been rather modest. One model frequently used (especially in expert systems such as MYCIN [Buchanan, Shortliffe 84]) is the *consultation model.* From an engineering point of view, it has the advantage of being clear and simple: the program controls the dialogue. But empirical studies have shown that these programs are behaviorally unacceptable [Fischer, Stevens 87]. The SYSTEMS' ASSISTANT was an effort to support more mixed-initiative dialogues by allowing users to volunteer information.

LISP CRITIC: Critiquing Users' Work. LISP CRITIC is a knowledge based system that critiques LISP programs. The interaction is controlled by the user, who selects parts of programs and asks the systems for help in improving the code either for human comprehensibility or machine efficiency. This system served as a foundation for further research in critiquing systems [Fischer et al. 90].

HELGON: Incremental Construction of Queries by Reformulation. HELGON [Fischer, Nieper-Lemke 89] is based on the retrieval by reformulation paradigm [Williams 84], which was derived from a theory of human remembering. This theory postulates that humans incrementally construct queries and naturally think about categories of things in terms of specific examples. HELGON supports the incremental description of a desired object with multiple specification techniques.

Although each of these systems explored issues of importance and each made an identifiable step forward, they all fell short in supporting truly cooperative systems. The systems were isolated efforts and were built for relatively simple domains with little support for high functionality computer systems.

3. High Functionality Computer Systems

The more powerful systems become, the more difficult they are to use. In order to take advantage of the power of high functionality computer systems, the cognitive costs of mastering them must be reduced. The following problems of high functionality systems (as identified by Lemke [1989] and Draper [1984]) must be overcome:

Users do not have *well-formed* goals and plans. Problem-solving in ill-structured problem domains can be characterized by the fact that no precise goals and specifications can be articulated. Users of high-functionality computer systems suffer especially from a lack of knowledge of the interdependencies between problem specification and the knowledge of which tools exist to solve these problems. Unfamiliarity with this mapping leads users to concentrate too quickly on implementation issues, and they often overlook alternative solutions.

Users do not know about the *existence* of tools. Users cannot develop complete mental models of high functionality systems. Without complete models, users are in many situations *not aware of the existence of tools.* A passive type help system will be of no assistance in these situations. Active systems, critics, and browsing tools are needed to let users explore a system or point out useful information.

Users do not know *how to access* tools. Knowing that something exists does not necessarily imply that users know how to find it.

Users do not know *when to use* tools. In many cases, users are lacking the *applicability conditions* for tools or components. Features of a computer system may have a sensible design rationale from the

viewpoint of system engineers, but this rationale is frequently beyond the grasp of most users, or even a user familiar with the basic functions of the system. Systems seem imponderable because not only do users have to search through a large list of options, they also do not yet know how to choose amongst them.

Users cannot combine, adapt, and modify tools according to their *specific* needs. Even after having overcome all of the previous problems (i.e., a tool was found, its functioning was understood, etc.), in many cases the tool does not exactly do what the user wants. This problem requires system support to carry out modification at an operational level with which the user is familiar [Fischer, Girgensohn 90].

To deepen our understanding of the problems of high functionality systems and find ways to overcome these problems, we engaged in a search for success models of such systems. The success model idea previously has proven to be of great value. We have analyzed skiing as a success model [Burton, Brown, Fischer 84] and derived architectural components for computer-based learning environments [Wenger 87] from this analysis. In a similar fashion, the ideas behind spreadsheets were used as guiding principles in system building efforts in other domains [Lai, Malone 88; Wilde, Lewis 90]. A preliminary analysis clearly indicated that McGuckin Hardware in Boulder might be an ideal candidate for a success model. McGuckin carries more than 350,000 different line items on 33,000 square feet of retail space. The store's superior reputation among its customers and its continued growth and profitability make it a success model. In the next section, we describe a study done at McGuckin, argue why it is relevant to these issues, and show how the store has successfully addressed the difficult problems mentioned previously.

4. McGuckin: A Success Model for Cooperative Problem-Solving

To get a better understanding of just how the "system" operates, we asked McGuckin Hardware for permission to observe and record interactions between customers and sales agents. The interesting dialogues were transcribed on audiotapes and carefully analyzed. Videotapes would have been the superior medium, but they would have interfered too much with store operations.

Space limitations prevent us from illustrating each of the above problems with its own dialogue. A detailed account of our study can be found in Reeves [1990] . We will illustrate key principles of cooperative problem-solving and refer to other problems as they arise in these three dialogues.

Applicability Conditions. Figure 4-1 illustrates how the sales agent helps customers to know *when* to use a given tool, even if that use was not intended by the designer of the tool. Mapping the problem of "weighing small animals" to "weighing food" happened so quickly that it seems almost trivial, yet behind this mapping lies knowledge that has eluded designers of high functionality computer systems: extracting the knowledge crucial to the task at hand and mapping it to another domain, temporarily ignoring or reinterpreting the "intended" uses.

The customer describes the intended item "differentially" in terms of an example, building on what the environment has to offer. Our goal here is not to suggest in detail how to solve this problem with computer systems, but to show that applicability conditions and differential descriptions play an important role in cooperative systems.

Incremental Problem Specifications. The knowledge that people have about a given problem emerges only when the situation "asks" for it [Schoen 83; Williams 84; Fischer, Nieper-Lemke 89]. Figure 4-2

1. C: *I'm looking for some scales and I saw some little tiny ones over here, but I need something that has a large platform on it, to weigh small animals on.* Holding hands about 18 inches apart
2. S: I would think something in our housewares department, for weighing food and things like that. Go on down to the last isle on the left.

Figure 4-1: Applicability Conditions and Differential Descriptions

In this and the following dialogues, *C: means customer* and S: means sales agent. Our explanations and comments are in this typeface.

shows that there is a close relationship between defining specifications incrementally, as seen here, and establishing shared knowledge, as will be seen in the next dialogue. The distinction is a subtle, yet useful one. Shared knowledge has more to do with establishing a common reference point with which to discuss a situation, and less to do with the specific process of identifying relevant parts of the problem domain.

1. C: *I need a cover for a barbecue.*
2. S: (Leading customer down an isle where several grills are lined up and accessories are displayed) Okay... what have we got here... chaise, chair barbecue grill cover... Does that look kind of like what you got? (Pointing to one of the grills) Similar? No?
3. C: *No.*
4. S: Take any measurements?
5. C: *No.*
6. S: That's a good guess there. (Pointing to a one-burner grill)
7. C: *It's a double burner one.*
8. S: 52 inches that's the total length it'll cover. (Measuring with the tape and holding the tape over one of the grills)
9. C: *Yeah. I know its not that big at all.*
10. S: You saying about 18 by 18. Well, this is 27, it'll cover up to here. (Using measuring tape again and pointing)
11. C: *I need two.*
12. S: A couple... in that brand, that's all I have. Here are these Weber ones, thicker material and all that. Here are some smaller ones.
13. C: *I'll take this one.*
14. S: We'll be getting more of these pretty soon.
15. C: *You'll have them by Christmas?*
16. S: Hopefully Thursday.

Figure 4-2: Incrementally Refining a Query

The specifications are given incrementally: (1) a cover for a barbecue, (2) dissimilar to certain examples, (3) double burner, (4) not that big, (5) I need two, and (6) by Christmas.

Achieving Shared Understanding. Between the time a customer begins to interact with a sales agent and the time the customer leaves with a satisficing solution, a shared understanding must be created between the two cooperating agents. The customer must begin to appreciate relevant parts of the solution domain and the sales agent must understand the problem in enough depth to make reasonable suggestions. Figure 4-2 shows how establishing shared understanding is a gradual process in which each person participates, sometimes ignoring questions, sometimes volunteering information, and sometimes identifying miscommunications. Illustrated also are the problems of knowing about the existence of tools and understanding the results that they produce.

In addition to the points illustrated specifically by these dialogues, the following issues were observed (for details see [Reeves 90]):

1. C: *I'm looking for a small fastener. Maybe one sixteenth.*
2. S: Okay. Plastic? Metal?
3. C: *Well, what I've got is to fasten a sign onto a square pole. I've got a hole in the top and it fits fine and I've got to get one on the bottom.*
4. S: Pole have holes in it?
5. C: *Yeah. I had a one eighth bolt, but it's too big. Need something smaller than that.*
6. S: You don't want to go clear through the pole and put a nut on the other end?
7. C: *No.*
8. S: Round pole? Square Pole?
9. C: *Square pole.*
10. S: (Picking up a fastener and showing it) You tried these?
11. C: (Scrutinizing the fastener) *Hmmmm*
12. S: You've got to have a five sixteenths hole and you fold this thing up and stick it in. Would that work?
13. C: *It's got to be five sixteenths?*
14. S: HmHm. The size of the shaft on this thing. (Pointing to the fastener)
15. C: *Yeah. It's not that big.*
16. S: No way to drill it?
17. C: *No.*
18. S: No. What did you use the first time?
19. C: *I tried a one eighth inch.*
20. S: How thick is the metal in the pole?
21. C: *Oh, probably about one eighth inch.* (Pointing to a certain fastener) How about these?
22. S: (Picking one up and showing the moving parts) These work on hollow-core doors.
23. C: *Yeah.*
24. S: (walking over to a different kind of fastener and picking it up) I don't know if this would be strong enough. Still need a three sixteenth hole. If the wind is blowing hard it might give way. Just putting it in with a screw-driver?
25. C: *Yeah.*
26. S: How about a self-tapping bolt? (Picks one up and shows it) Put that in, tighten it down, (points to tip), that's a thread cutting thread there.
27. C: *Well, what, uhmm, well, this would probably do it. What about, would it come back out?*
28. S: Oh sure. It would come back out.
29. C: *But once it's in?*
30. S: As long as the hole is smaller than this thing, you can thread it in and out. Metal pole, right?
31. C: *Yeah.*

Figure 4-3: Achieving Shared Understanding

Natural Language vs Natural Communication. People rarely spoke in complete, grammatical sentences, yet managed to communicate in a natural way. The support for natural communication (which allows for breakdowns, clarifying dialogues, explanations, etc.) is more important for cooperative problem-solving than being able to parse syntactically complex sentences.

Mixed Initiative Dialogues. People are flexible in the roles they play during a problem-solving episode. They easily switch from asking to explaining, from learning to teaching. The structure of these dialogues was neither determined by the customer nor by the sales agent, but clearly indicated mixed initiatives [Carbonell 70b] determined by the specifics of the joint problem-solving effort.

Multiple Specification Techniques. Customers used a great variety of different specification techniques such as bringing in an example part, pointing to an item in a catalog or in the store, and giving general descriptions such as "I need a lock that qualifies for cheaper insurance rates."

Management of Trouble. Many breakdowns and misunderstandings occurred during the observed problem-solving episodes, but in almost all cases, clarifying dialogues allowed their recovery, illustrating the important feature that problem-solving among humans cannot be characterized by the absence of trouble, but by the identification and repair of breakdowns [Suchman 87].

Simultaneous Exploration of Problem and Solution Spaces. Customers and sales agents work within

both problem and solution spaces simultaneously, or at least alternatively. Typically the problem owner (customer) has a better grasp of the problem space and the problem solver (sales agent) has a better understanding of the solution space, but over time these spaces converge until there is a large enough intersection of shared knowledge within which potential solutions can be evaluated.

5. Lessons Learned from the McGuckin Study

The most important factor in making McGuckin a success model for cooperative problem-solving is its knowledgeable sales agents. McGuckin has a special kind of sales agent called a "roamer." Roamers provide a global view of the store. McGuckin hires experts in the various departments and considers previous experience within a field, such as plumbing, to be more important than previous *sales* experience in that field. The roamer is responsible for knowing how the various departments combine to solve problems. A suitable architecture for high functionality computer systems might be to have relatively independent domain specific systems, made coherent and accessible by a global knowledge-based component. The knowledge requirements for this overriding component are formidable: the *least* experienced roamer has been working there more than 11 years.

The study clearly indicated that problems in realistic problem situations are not fixed targets. The combination of a large selection of objects and knowledgeable sales agents creates an environment in which customers can produce partial solutions and get feedback from the items in the store and from the sales agents in the form of critiques [Fischer et al. 90]. As problem solvers tentatively explore possible solutions and evaluate how those affect their perception of the original problem, they shape the situation, in accordance with their initial appreciation of it, the situation "talks back," and they respond to the situation's back-talk [Schoen 83].

These findings parallel closely the history of our system building efforts. Originally, we believed that domain-oriented construction kits would be powerful enough to talk back by themselves, which is unfortunately not the case. Construction kits support the construction of an artifact, but they do not provide any feedback on the quality of the design. To obtain this, we have to have design environments that can judge the quality of a design with the help of critics [Fischer, McCall, Morch 89].

Comparing current computer systems to what we have seen at McGuckin led to the following claim: "*High functionality computer systems offer the same broad functionality as large hardware stores, but they are operated like department stores,*" i.e., what is missing from them is the cooperative support of knowledgeable sales agents.

Any use of a success model in one domain cannot be transferred to another domain without great care and additional work. The most obvious difference is that human-human interaction is *not* human-computer interaction. The analogies will certainly break down along a number of dimensions. We should also not overlook that McGuckin does not address all the possibilities that a computing environment offers, not being present in a hardware environment. One aspect of a cooperative problem-solving in which users deal with software rather than hardware is that users should be able to change the functionality as time goes on and new knowledge and new artifacts become available.

6. Future Work

If we take seriously the idea that problem setting and problem-solving have a dialectical relationship, then examples can play a major role in shaping the problem. This is a difficult issue to study, because one cannot know beforehand how examples will affect the problem, and a post-hoc analysis will be blind to the subtle changes in the problem definition that examples have induced. As much as sales agents use analogies and examples to narrow the search down, it would be desirable to get a better grasp of what is involved in creatively generating partial solutions that best clarify the problem itself.

Lave [1988] has shown that a crucial ingredient in real problem-solving is that people are allowed to redefine the problem itself, but one cannot redefine a problem that one does not own. Only the owner of a problem is free to decide whether another description of the problem is legitimate. The difficulty with delegating ill-defined problems is that the owner of the problem interacts only indirectly with the emergent problem solution and is not able to foresee implications that certain specifications and assumptions are having on the final solution. The way this manifests itself at McGuckin is that in one case problem owners themselves go to the store whereas in other cases they send another person. Studying these cases will hopefully provide us with more insights into constructing computing systems with which problem owners can directly interact.

The McGuckin study has influenced our system building efforts. The roamer/specialist distinction has shown how important it is to build components that are specialized to certain domains and, at the same time, provide a global view of the whole system. Future problem-solving systems must incorporate the resources that help the design situation to "talk back," signalling suboptimal designs and raising those issues to the users. Support systems must acknowledge that people often have to design problems and solutions iteratively and incrementally.

Acknowledgments. We thank McGuckin Hardware (Especially Robb Hight, Randy Dilkes, and Larry Kemmer) for allowing us to record and analyze interactions between customers and sales agents, and the customers who participated in the study. The research was partially supported by grants No. CDA-8420944, IRI-8722792, and IRI-9015441 from the National Science Foundation, grant No. MDA903-86-C0143 from the Army Research Institute, grants from the Intelligent Systems Group at NYNEX, Software Research Associates (SRA), Tokyo, and the Colorado Institute of Artificial Intelligence.

References

[Buchanan, Shortliffe 84]
B.G. Buchanan, E.H. Shortliffe, *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley Publishing Company, Reading, MA, 1984.

[Burton, Brown, Fischer 84]
R.R. Burton, J.S. Brown, G. Fischer, *Analysis of Skiing as a Success Model of Instruction: Manipulating the Learning Environment to Enhance Skill Acquisition*, in B. Rogoff, J. Lave (eds.), *Everyday Cognition: Its Development in Social Context*, Harvard University Press, Cambridge, MA - London, 1984, pp. 139-150.

[Carbonell 70a]
J.R. Carbonell, *AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction*, IEEE Transactions on Man-Machine Systems, Vol. MMS-11, No. 4, December 1970.

[Carbonell 70b]
J.R. Carbonell, *Mixed-Initiative Man-Computer Instructional Dialogues*, Report 1971, BBN, May

1970.

[Draper 84]
S.W. Draper, *The Nature of Expertise in UNIX*, Proceedings of INTERACT'84, IFIP Conference on Human-Computer Interaction, Elsevier Science Publishers, Amsterdam, September 1984, pp. 182-186.

[Fischer et al. 90]
G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch, *Using Critics to Empower Users*, Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA), ACM, New York, April 1990, pp. 337-347.

[Fischer, Girgensohn 90]
G. Fischer, A. Girgensohn, *End-User Modifiability in Design Environments*, Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA), ACM, New York, April 1990, pp. 183-191.

[Fischer, Lemke, Schwab 85]
G. Fischer, A.C. Lemke, T. Schwab, *Knowledge-Based Help Systems*, Human Factors in Computing Systems, CHI'85 Conference Proceedings (San Francisco, CA), ACM, New York, April 1985, pp. 161-167.

[Fischer, McCall, Morch 89]
G. Fischer, R. McCall, A. Morch, *JANUS: Integrating Hypertext with a Knowledge-Based Design Environment*, Proceedings of Hypertext'89 (Pittsburgh, PA), Association for Computing Machinery, New York, November 1989, pp. 105-117.

[Fischer, Nieper-Lemke 89]
G. Fischer, H. Nieper-Lemke, *HELGON: Extending the Retrieval by Reformulation Paradigm*, Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), Association for Computing Machinery, New York, May 1989, pp. 357-362.

[Fischer, Stevens 87]
G. Fischer, C. Stevens, *Volunteering Information -- Enhancing the Communication Capabilities of Knowledge-Based Systems*, Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction (Stuttgart, FRG), H.-J. Bullinger, B. Shackel (eds.), North-Holland, Amsterdam, September 1987, pp. 965-971.

[Lai, Malone 88]
K.-Y. Lai, T.W. Malone, *Object Lens: A "Spreadsheet" for Cooperative Work*, Proceedings of the Conference on Computer-Supported Cooperative Work (CSCW'88), ACM, New York, September 1988, pp. 115-124.

[Lave 88]
J. Lave, *Cognition in Practice*, Cambridge University Press, Cambridge, MA, 1988.

[Lemke 89]
A.C. Lemke, *Design Environments for High-Functionality Computer Systems*, Unpublished Ph.D. Dissertation, Department of Computer Science, University of Colorado, July 1989.

[Reeves 90]
B. Reeves, *Finding and Choosing the Right Object in a Large Hardware Store -- An Empirical Study of Cooperative Problem Solving among Humans*, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, 1990.

[Schoen 83]
D.A. Schoen, *The Reflective Practitioner: How Professionals Think in Action*, Basic Books, New York, 1983.

[Suchman 87]
L.A. Suchman, *Plans and Situated Actions*, Cambridge University Press, New York, 1987.

[Wenger 87]
E. Wenger, *Artificial Intelligence and Tutoring Systems*, Morgan Kaufmann Publishers, Los Altos, CA, 1987.

[Wilde, Lewis 90]
N. Wilde, C.H. Lewis, *Spreadsheet-Based Interactive Graphics: From Prototype to Tool*, Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA), ACM, New York, April 1990, pp. 153-159.

[Williams 84]
M.D. Williams, *What Makes RABBIT Run?*, International Journal of Man-Machine Studies, Vol. 21, 1984, pp. 333-352.