Keynote Address: The Frontiers of AI Research

William J. Clancey

Introduction

I took the title that was offered to me very seriously—"Frontiers in AI Research"—and I'm going to use it as an opportunity to summarize for you a presentation (Clancey, 1988) that I made at the Carnegie Symposium on Cognition last May.

This was the twenty-second Carnegie Symposium that's been mainly organized by Newell, in the Psychology Department. There were two groups of people invited: those from the AI architectures area and those who were working on neural nets. I provided the commentary for the AI architectures area and Zenon Pylyshyn, who is a cognitive philosopher, talked about the neural net side. So if there's a little bit of criticism you constantly hear in my presentation, it's coming from the setting in which I prepared my thoughts, which was to look at my colleagues' work and to try to appraise what we had accomplished and what remained to be done.

In this all-too-brief talk, I am going to give you a whirlwind tour of two very basic ideas. I am going to start by saying that we need to do something different than we are doing in knowledge engineering, and part of that is to provide a better description of the engineering methods of our modelling techniques. I have a particular perspective on AI programming that I hope will help us put our field together better. That's probably the most relevant for your interests in using AI programming for scientific modelling.

The second part of the talk is more psychological, and I think it will probably be of equal interest to you, but maybe not as immediately relevant. That is, looking at artificial intelligence, not just as a modelling technique that any scientist or engineer could use, but taking it as it's original goal of trying to build an intelligent being. It turned out that most of the people in the Carnegie Symposium had that as their primary interest. I view that as more of a psychological question. I'm going to get into a few of the new perspectives that are evolving. Hopefully, I won't be too rushed so I can convey some new insights to you there.

William J. Clancey is Senior Research Scientist, Institute for Research on Learning, 2550 Hanover Street, Palo Alto, California.

Transcribed from talk presented at the IUFRO Forest Simulation Systems Conference, Berkeley, California, November 2–5, 1988.

The state of knowledge engineering

Knowledge engineering has progressed for about 10 years since the Heuristic Programming Project was cranking out programs based on MYCIN in the late 1970s. The state of knowledge engineering can be described in terms of three basic advances.

If we were to go back to the 1970s, we would find people working on learning, robotics, vision, and reasoning, with everyone working on programs that usually did not work together. There is a real attempt now to build, as they put it—I don't care for the term very much myself—an *autonomous agent,* a real being that has perception. It can move, it can learn, and it can solve problems. This turns out to be a very important connection, because a lot of learning work was done with what many of us would view as very impoverished problems, like "How do you recognize what a cup is?" as opposed to "How do you recognize a new disease when you see it out in the world?" So in the large sense, we are putting together a lot of components from before and treating the problem of reasoning in a much more comprehensive way. I'll have a little more to say about that as we go along.

At another, lower level, there is an integration of memory, problem solving, and learning, so people are seeing that you can't talk about learning without having a problem-solving component or without having some commitment to how the memory should relate to the knowledge.

And third, a little bit different, is some real progress in evaluating the systems, based mainly on set theoretic, graph theoretic, and logical analyses of the program. Again, none of these is a prevalent activity in the field. I would say these are all frontiers, so if you wanted me to point out the people who I think are advancing the field, I might pick out two or three people in each of these areas, as opposed to 50 or 60.

Qualitative systems modelling

So what do I think is missing then, if that's the reason to be cheered about the state of the field? First, there are a lot of people building expert systems out there who are not describing them in a way that allows us to codify a set of engineering practices. Part of this is because they think they're just building intelligent agents and don't see themselves as engaged in the traditional problems of scientific and engineering modelling, which has a certain requirement for failure analysis, boundary descriptions, and so on.

Second, in order to do that, it turns out we just can't apply traditional numeric analysis techniques. We need a different foundation for talking about these programs. These are programs that, as you've probably heard before, have things that we call qualitative models in them, representations, and not just equations and parameters, and we need a theory of that kind of model so we can say how you can be precise, how you can measure the accuracy of this kind of a model—how can you talk about its predictive capability? So in a sense, my talk falls into these two parts: the first, trying to show you what I think we've learned about these programs in terms of an engineering description; and the second, giving you some insight into what we know about representation that will allow us to build up a more traditional engineering methodology.

Figure 1 illustrates the most important thing I want you to take away from this talk. It is my perspective on AI programming. This is not by any means shared by most people in the field, really, though I think it is starting to take hold. This is what I said before, that we don't deal with an AI program as an intelligent agent, that we don't think of intelligence as what we're after—we think of programming as supplying a modelling technique, a means of modelling systems in the world qualitatively, rather than just numerically. So the first thing when we look at an expert system, or knowledge-based system, or any AI program, is we want to know what system is being modelled, and what it is we are trying to do with that system.

Systems and tasks

Here I want to shift my perspective a little bit to modelling processes, rather than just talking about systems. I list three kinds of processes that are modelled using AI techniques. The first one is what we would consider the subject matter system: a physical system (such as an electronic circuit or a forest), a mathematical system, something abstract (such as a computer program), or a social system (such as a group of people or an organization). And then there are other types of system. The inference task is, What are you trying to do with that system? Are you trying to design it, as you would try to design a VLSI circuit? Are you trying to assemble the parts of a car? Is it a manufacturing system that you're modelling? Are you interpreting an existing system? Are you auditing it, or diagnosing it? Are you predicting what is going to be next? Are you trying to control it?

There's a certain logic here, since here we have an input and an output, and we want to know what is the system. Here we have an input into a system, and we want to know what is the output, or here we have an output of a system, and we want to know what is the input. It is intriguing to see here that, of course, numerical techniques have been quite strong in prediction and control, but what AI brings into the field is the capability to produce other things automatically—to automate the process of specifying, designing, assembling, and interpreting systems using qualitative techniques.

The systems-modeling perspective

SYSTEMS AND TASKS

Synthesis (specification, design, assembly)

Analysis (Interpretation, prediction, control)

BASIC PROCESSES

Subject matter systems in the world
(physical, mathematical, or social),

Inference ("cognitive system")

Communication

PROCESS REPRESENTATIONS

Classification (e.g., a disease taxonomy)

Simulation
- Behavioral
  - (e.g., a causal-associational network)
- Functional (e.g., a procedural net)

Fig. 1. The systems-modelling perspective.

Basic processes

So, as I said, there will be a model of a system, and then there will be a model of the process for reasoning about that system. So how to do diagnosis, how to design, how to control (and notice, that has sort of a generic flavor to it) something that would apply to many different systems. That's why on the edge of expert systems today there are programs you can insert as models for many different types of systems, and this part remains the same. They will supply you with the language for describing the diagnostic knowledge, and you receive the diagnostic program as part of the shell.

Now I've separated out a third type of process here, because it's so important and in some sense it builds on the other two. This is the process of carrying out a consultation: explaining what you know, justifying what you know, learning something from someone else, teaching. So this communication process, which is really the modelling of a social system, is so important that we separate it out. If you think about it, that corresponds to what we call "natural language" research within AI.

Process representations

Providing the basis for all of this is really the core of what AI is bringing to the table. It's a modelling methodology for modelling processes. This is where "representation" comes in. I see two basic methodologies for modelling processes. One is that of classification, and a simple one that we're all familiar with is a disease taxonomy—a listing or a hierarchy of abnormal processes that might be occurring in some system. It doesn't model the internal states; it doesn't provide a structural explanation of how those processes occur; just a set of patterns—"under these conditions I might see symptoms like this, and this is the name I give to what is happening inside that system at this time."

Simulation models attempt to give some kind of an explanation in a structural or functional description of what's happening. There are two types of structure or function. One, the most prevalent type of simulation model in the AI programs, is what we call the causal-associational network, which describes states and state transition, given certain conditions. I actually like to think of this more as showing how abnormal substances are related to abnormal processes and linking those two together to a causal change.

Within AI, part of the confusion is that most of my colleagues don't call these things models at all. It's incredible. To them a model has to be a functional description. What is the purpose of the system? And the structure? How do all the parts work together to produce that function? They call this a "model-based system." This is something of a joke, since obviously all expert systems are model-based. You could not possibly have diagnosis or design without a model inside. That is partly why the field has had some trouble getting together what it has accomplished, because of this particular perspective it has on modelling.

I could say a lot more about this. I have written several papers summarizing it. I think that it's the most important perspective to carry away of what is a lasting contribution of AI programming, what's been done in the last 15 years or so.

Improved formalization

I didn't realize until I got to my hotel room and was looking over the topics here that you probably would have been interested in some talk about how numeric and qualitative modelling are coming together. To me, that is where the future lies. Any of the systems that have had major impact in AI over the last 15 years, it's very striking when you go back, you find the numeric models inside those systems. Now, we didn't talk about that because that wasn't what was new, and of course there wasn't this perspective on modelling methodology. We were talking about "intelligence," which confuses everything.

As a brief summary, there are four basic uses of expert systems tied to a numeric model. One is to set up or design the system that is going to be modelled numerically. One of the best examples I know is that by a student in Mechanical Engineering at Stanford who has an expert system that designs a manufacturing process, and who then runs a traditional mathematical model simulation of that manufacturing process to see how it would behave under certain constraints.

The second use of an expert system would be to generate a scenario, to reason about boundary conditions and actually carry out a systematic set of experiments using the numeric model. This in itself was also done in the Mechanical Engineering system that I was just mentioning.

The third use is very different. If the system that you're modelling contains a person or a group of people, you can simulate the reasoning of those people using an expert system. If there are heuristic decisions or alternative strategies or organizations of the people that you want to try out, you can have the expert system design that component and then carry out that role. For example, if you're talking about wildfire management, you can have the expert system play the roles of various people or groups doing things and run the simulation with that component inside making heuristic decisions.

The fourth use of expert systems in numeric modelling is for facilitating the analysis of the output. Here we have a particular slant in our knowledge-based programming, and that is the use of selective instrumentation to expose what is happening inside the model and to reapply or make concrete the processes that are occurring. Various graphical means have been developed within AI for going through a history of some processes and reasoning without it. There is probably a tremendous amount of opportunity in each of those four areas, and as I said, I think it is something that all of you would be excited about as I am.

Theory of representation

Well, let's continue my quick survey here of how can we talk about knowledge engineering in order to bring together what has been accomplished. Figure 2 shows the different layers of an expert system—in particular, an expert system that has been developed with a theoretical perspective of thinking about this modelling methodology. In developing the system, the theory and the model are clearly made explicit. Let me take you through these layers. I had to go back to my medical background, because that's something I can be very specific about and I think it fits our general experience so that you can all follow me.

If we were to look inside the program, we would find a general model of some system. For example, we would find a general model of the types of disease processes that can occur in people and, to be more specific, as in the program I developed, for some particular area of medicine. Then the program uses this general model with its inference procedure to produce a situation-specific model. That would be a model of a particular patient. What's happening now in the system that I'm modelling at this moment? I have a general model of this kind of system, and then I have a particular situation right now.

Now, these models are expressed in theoretical terms that would be like the following. I would have a model of how to do diagnosis and what the proper diagnostic knowledge is that would be necessary. Here we are still talking about medicine, but things become more and more general as we go up. I have the idea that I can do diagnosis, which is my theory, using a hierarchy of diseases in what's called an etiological hierarchy: abnormal processes, ultimate causes that can be explained by a congenital problem, toxic effect, a foreign agent that's come into a system, a very high level description of what can happen to a person.

I also have a theory of what constitutes a good diagnosis. For example, I might say that I have to explain all the abnormal findings, and that I have to get one process that explains everything. That would be optimal, and I would make that explicit in my program—that's part of my theory of what a good diagnosis is.

Then, I would have a particular diagnostic procedure. What observations should you make? In what order should you ask questions when you are interviewing the patient, or another physician? When you receive your information, what should you do with it? How do you relate it the situation-specific model that you've constructed so far? How do you know when to change your focus of attention? Most of this has to do with the focus of attention—shifting between what you're learning right now and what you know in general.

Architecture Layer	Example (From HERACLES)
Framework	
"General Framework"	Information-Processing Symbol Manipulation Production Systems
Representational Grammar (Constructs) & Interpreter = A programming language	Constraints, Tasks, Metarules, Relations, Definitions, Heuristic Rules, etc.
Theory	
Theory of Process Representations (Epistemological Structures)	Process Classification Hierarchy; State-transition Network
Theory of Task-Specific Knowledge & Control (general ontological structures)	Etiological Hierarchy; Dx Model-graph Constraints; Dx Inference Procedure
Model	
General Model of a System	Processes/States causing headache & other CNS findings
Situation-specific model of system	"Patient-specific Model"

Fig. 2. Architecture layers of an expert system.

Now this in turn is built on a more general model of types of process representation, so an etiological hierarchy is a kind of classification of processes. I would also incorporate a behavioral simulation of diseases represented as a state-transition network.

Getting one level higher, we see that this process-classification hierarchy is represented using a bunch of primitive objects that might be constraints, tasks, meta-rules, relations, or definitions. Actually, the process-classification hierarchy is represented using a set of relations, the inference procedure is represented using tasks, meta-rules, and so on. This is my programming language and these are the constructs in which I represent or implement my particular theory in models.

Of course, it's different from just an ordinary programming language, since it does have these two distinct components. It has a representational grammar and an interpreter, which is the stuff you've probably heard about—backward-chaining, resolution, and things like that.

At the next level, there is an implicit framework in which I am doing all of this work. The idea of qualitative systems modelling, the idea of a system, the idea of symbol manipulation, and so on. You don't usually find that in a program. What I'm claiming is that if we look at any expert system out there and really try to pin people down, they really have to make commitments all along the line here if they're going to be advancing our technique of modelling methodology. Otherwise they're just "hacking."

There's another angle. It goes back to this idea of the intelligent agent. We can view the program as a set of nested functions. On the outside, it corresponds to what I've called *communication* before: the function the program has in interacting with people. What role does it play in some social setting? Here is where the capabilities of the program can be discussed or described in terms of observing and acting in an uncertain world, for example. That turns out to be one of our main themes that keeps coming up over and over again. Down one level, supporting that capability, is what I would call the modelling functions of the program: the ability to construct that model of what's happening out there right now.

What is my map of the world that I am going to use as a basis for taking action? I'm going to model the situation, and then I'm going to come up with some planned course of action. Here's where we have the capability to form hypotheses, test them, refine them, discriminate alternative models, make predictions, explain what's occurring, and so on. These are all the things you would find, for example, in a typical diagnostic system. If we were talking about a design problem, there's a somewhat different set of modelling functions that have to do with tying components to one another, forming a consistent and coherent system, and so on.

Now, one level down, surprisingly, is where we find most of the AI architecture research going on, and this is what we would call the *representation, inference,* and *control* aspects of the program. What are the primitives for representing the model? Before, I said constraints, tasks, and meta-rule relations; this is where that level of work is going on. Here, we talk about a virtual machine, having certain memory, states, transitions, a scheduler, and so on. Here is where the computer scientists are having all the fun. If you go back to figure 1, here is also where we think we will make the most progress in developing a complete agent. You have to make certain commitments—how does memory work, how does that commitment about memory relate to your commitment about learning, and how do you get knowledge when you need it in order to formulate hypotheses?

So there is a set of constraints inside here that limits what can go on later. For example, if we look at the Soar work going on that was started by Rosenbloom, Newell, and Laird (1988) at Carnegie-Mellon University, their model of memory today is that of a set of re-write rules or production rules that can't be inspected directly by the program. You can only give it a certain situation, and things come out, conclusions come out. That's a particular commitment to what memory is like, and has some very important implications for a model of learning.

When I looked at these six programs on which I was asked to comment last May, I found that all these people were reaching for some foundation. They wanted to build their theories of intelligence on something solid, and what was striking was that there were three distinct approaches, even though there was only one person down in the third camp. And that's the second part of my talk, actually.

One of them took interaction with the world as the most important constraint. If I'm going to say what constitutes intelligence, I really want to start with the environment. What is it about the situation in which this agent is going to be placed? How will that constrain what this agent has to do? The realtime component—how fast, or how much time you have to make a decision—has a very important constraint on the design of the system. For example, Barbara Hayes-Roth's (1988) system that deals with an intensive care unit has more data than it can analyze in any minute, and yet it has to make decisions within every few minutes. It has to decide when to process data and when to ignore it. She has several processors as a result, and she's doing parallel processing in order to handle that. That shows you how a certain perspective constrains the architecture.

Most of the people are proceeding from a "formal" framework. Some of them are genuine mathematicians who are giving us theorems and a set of axioms about the world and trying to prove what rationality is, what reflection is, and so on. But in general, I would say the way to talk about these people is that they are trying to come up with a set of primitives and a set of axioms that would allow them to build an agent from the ground up. How do I get these things that we know are important in an intelligent agent (flexibility in new situations, learning efficiency, and so on)?

The third approach is very different, and I'm going to spend a few minutes talking about the work of Rod Brooks (1988) because it illustrates this approach. Here's where we get into the neural net and connectionist arena. These are people who say the basis of intelligence is not the manipulation of a set of symbols, but the capability of the program to act directly with its environment. The essence is that they don't build in maps of the world. They don't start with a model of a system that the agent is trying to manipulate. They design the program so that it goes more directly from sensation to action through an internal network representation.

To complete this knowledge engineering section, as I said, there is a lot of progress being made in formalization, and what I tried to do here was to summarize the best techniques that we're following. You can view these as evaluation and testing methods for improving or describing qualitative models. The most powerful one that we've developed—and it's being used fairly systematically throughout AI today—is taking a given program and varying the input, the different situations that it has to cope with, the different types of users (or, defined in a teaching program, the many different types of students), the environment in which it has to act, whether it has to be realtime or can be batch processing, or even taking the shell and implementing a different model from a different domain. It is very common to take a diagnostic system that we developed for medicine and put in an electronic diagnosis problem that tests the generality of the model. That's one of the most powerful tests that we have.

Something that's becoming increasingly popular is for me to take the knowledge that someone has represented in their system and re-represent it in my framework. We can then get a better idea of the theoretical constraints: What is the architecture in which we are working? It forces us to be more explicit about our language.

One thing that I am especially interested in—and I've been very much inspired by Petrosky's book *To Engineer Is Human* (St. Martin's Press, New York)—is the role of failure in successful design. That emphasis on failure, I think, is the essence of what engineers are good at: taking a model and being able to manipulate it and being able to describe how it's going to behave under different situations. It's quite indicative of an AI system that, usually, it works the first time, you step back, wipe off your brow, and you write your paper about it. There are more and more people now within the field, especially those with some experience, who are saying we should not allow any papers to be published on expert systems that don't have a failure analysis that says here are 10 cases where the program failed and here's why. An interesting failure is not one that can be fixed by some incremental little change but something that requires you to change your theory, and remember that model theory and framework—

something that says, at a high level, my idea of how to represent processes—is probably not quite right. One method has been to simplify the program and determine how it degrades. But again, that's not done very much.

The idea of synthetic data sets is of course very common in a lot of computer science, but it hasn't been followed so much in expert systems work. One case where it's especially interesting is where we have an expert system that is being used as some kind of a standard inside a larger program. For student modelling, for example, we can simulate a student's behavior by modifying the expert system. We can take one rule of an expert system and ask the student modeller to say what knowledge does this person have, this simulated person, and now I have an explicit model of that knowledge and my student modeller had better say that there's one rule missing, and what the rule is. That's a very powerful technique for relating these model manipulation programs to each other.

As I said, when we look at how people were searching for foundations, there are a few outliers out there who say, "Hey, everything we're doing is fundamentally wrong." This idea that memory has symbols and grammar stored in it is not how people achieve their intelligence. And as I say here, in particular as a psychological theory, this idea of knowledge structures, that knowledge consists of a set of physical structures, is rapidly losing support. In particular, if you're familiar with the neural net and connectionist area, that's what they identify with. If this is right, if we are going to have a very dramatic paradigm shift here, that says that our interpretation of expert systems—as literal models of what people carry around in their brains—that that is wrong. This is going to dramatically change our idea of how to perform knowledge acquisitions, how to evaluate these systems, and how to talk about how expert systems will relate to people. I do think that, as I say here, the general hypothesis people have been following is wrong.

I'm just going to go through Brooks's work (1988) to make this concrete, so you can get some idea of what this alternative approach is. Brooks last May showed us video tapes of his programs. They are what he calls *insects,* that can move around the room, avoid obstacles, seek out heat or light sources, move through doorways, explore multiple rooms, collect Coke bottles, and so on.

What's interesting is that he does it without building a map or representation of those rooms in his programs. He builds the program as a set of layered parallel machines, of finite-state automata. Each one of these boxes is a finite-state automaton or a state within which each layer is actually another machine. The states are related to one another by inhibition and stimulation links, and there is a threshold for any particular state transition that has to receive so many inputs before it makes the transition.

Obviously, I would need another half hour to explain that to you in more detail, but the important thing is that these three levels—one that avoids obstacles, the other that hugs along the wall, and the third that surveys the robot through a doorway—these are all running in parallel, and what any particular level does depends on the state of the entire vehicle. In some sense the history of where this thing has been and what it is doing now is implicit in each of these states.

Again, the fundamental idea is that there is no map of the room here, that the program is not inspecting a map, it's just feeling forces and moving according to the current forces and the past state.

Conclusion

Why do we think we can make basic progress in understanding the brain now? That is something people have been working on for thousands of years. One of the most fundamental reasons is that knowledge engineering has taken this view of structures in the memory and made it concrete so we can sit there and look at the program and we can say, "That's what you think is in my head." That's something a philosopher could not have done 2,000 years ago. It's very intriguing that that commitment was Plato's commitment, this idea of ideal forms stored in the head that lead to all behavior. That's what these grammars are all about in AI systems.

The second reason we are making rapid progress is that over the last 15 years, anthropologists, linguists, and sociologists have been picking up on the cognitive science of this work. They've been reading this work, reading the psychology that came out of AI and saying, "No that's not what we're seeing out there. That's not how people behave. That's not how organizations are set up."

Some of this work in fact goes back to the 1920s and 1930s, when anthropologists were developing theories of their theories: theories of knowledge, what you know about another person. There are some very basic theoretical works that we're now looking at.

The third reason I think we are about to make some dramatic progress is that an alternative model has come about. We have something else that is concrete that we can look out and say, "What can we do to that to make it behave in an intelligent way? Let's throw out the idea about the map of the world being in memory and see what we can get."

There is a whole bunch more I could say, and I'm not going to go into it today, but I will show you one picture that I hope will make this alternative view clear to you. The accepted view within AI is that there are these symbols, things like words—like tree and forest, and patient and disease, and pest and whatever—that are stored as little tokens, strings, or something in the neurons, and that there are rules that link these things together.

The important thing is that this is just the old linguistic view of language. The original view of Noam Chomsky is that we have these grammars that are stored in memory, and when we speak and when we behave we are manipulating these grammars. I could go through another analysis to show you, but I think you can see pretty quickly that the expert system view of knowledge is a purely grammatical view. It's based on primitive terms and rules for rewriting those terms and rules for controlling the rewriting of the use of those rules. The alternative view, which is what I am putting forth by having synthesized a lot of things that I've read in linguistics and in philosophy, is that what memory's capability is, what its purpose is, is to *replay* sequences of activities—that you think about memory as a bunch of processes that are always running and that there is some resolution, some competition that is going to be necessary to know "What's the sequence I am going to follow now?"

This is highly layered; there is a lot of detail. The important thing is that when I say *A* implies *B*, it should be viewed as my saying *A* and then my saying *B*, not my reading out something that I've already said inside, somehow. So this is how we avoid this recursion where somehow I had to piece together *A* and *B* inside, and then I read that back out. Of course, it is the people who piece these together in programs. And so what we say instead is that representations are in some sense always out here with us. They are written down, they are the sounds I am making now, they are the marks on the paper, they are my silent speech when I read things, they are always perceived. These representations are not stored away in memory. Memory allows us to make those representations.

The frontier

What is on the frontier of AI research? First, there is some progress in describing programs. I showed you the simulation systems modelling perspective, I talked about the architectural layers, I talked about functional layers. Those are three perspectives for describing an expert system so that we can advance our theory of process representation. That will allow us to build more appropriate tools.

Second, I said a little bit about the evaluation of the models. It's my belief that the formal, mathematical approaches, especially using set graphs and logic, are advancing the field in that way.

Third, I think there is going to be a dramatic change in how we think about our programs, specifically in a theory of what these representations are and how they relate to human knowledge and how representations relate to the environment. In particular, this is the part I wasn't able to go into, but the idea that we are modelling a social world that inherently is always open to change and always open to interpretation and cannot be pinned down quite as well as a physical system.

Literature cited

Brooks, R. A. 1988. How to build complete creatures rather than isolated cognitive simulators. In: *Architectures for intelligence: The Twenty-second Carnegie Symposium on Cognition,* K. vanLehn and A. Newell, editors. Lawrence Erlbaum Associates, Hillsdale, California. (In preparation).

Clancey, W. J. 1988. The frame of referenc problem in the design of intelligent machines. In: *Architectures for intelligence: The Twenty-second Carnegie Symposium on Cognition,* K. vanLehn and A. Newell, editors. Lawrence Erlbaum Associates, Hillsdale, California. (In preparation).

Hayes-Roth, B. 1988. Making intelligent systems adaptive. In: *Architectures for intelligence: The Twenty-second Carnegie Symposium on Cognition,* K. vanLehn and A. Newell, editors. Lawrence Erlbaum Associates, Hillsdale, California. (In preparation).

Rosenbloom, P. S., A. Newell, and J. E. Laird. 1988. Toward the knowledge level in Soar: The role of architecture in the use of knowledge. In: *Architectures for intelligence: The Twenty-second Carnegie Symposium on Cognition,* K. vanLehn and A. Newell, editors. Lawrence Erlbaum Associates, Hillsdale, California. (In preparation).