The Knowledge Level

Presidential Address,
American Association for Artificial Intelligence
AAAI80, Stanford University, 19 Aug 1980

Allen Newell
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

I. Introduction

This is the first presidential address of AAAI, the American Association for Artificial Intelligence. In the grand scheme of history, even the history of artificial intelligence (AI), this is surely a minor event. The field this scientific society represents has been thriving for quite some time. No doubt the society itself will make solid contributions to the health of our field. But it is too much to expect a presidential address to have a major impact.

So what is the role of the presidential address and what is the significance of the first one? I believe its role is to set a tone, to provide an emphasis. I think the role of the *first* address is to take a stand about what that tone and emphasis should be—to set expectations for future addresses and to communicate to my fellow presidents.

Only two foci are really possible for a presidential address: the state of the society or the state of the science. I believe the latter to be the correct focus. AAAI itself, its nature and its relationship to the larger society that surrounds it, are surely important.* However, our main business is to help AI become a science—albeit a science with a strong engineering flavor. Thus, though a president's address cannot be narrow or highly technical, it can certainly address a substantive issue. That is what I propose to do.

I wish to address the question of knowledge and representation. That is a little like a physicist wishing to address the question of radiation and matter. Such broad terms designate a whole arena of phenomena and a whole armada of questions. But comprehensive treatment is neither possible nor intended. Rather, such broad phrasing indicates as intent to deal with the subject in some basic way. Thus, the first task is to make clear the aspect of knowledge and representation of concern, namely, what is the nature of knowledge. The second task will be to outline an answer to this question. As the title indicates, I will propose the existence of something called the *knowledge level.* The third task will be to describe the knowledge level in as much detail as time and my own understanding permits. The final task will be to indicate some consequences of the existence of a knowledge level for various aspects of AI.

II. The Problem of Representation and Knowledge

The Standard View

Two orthogonal and compatible basic views of the enterprise of AI serve our field, beyond all theoretical quibbles. The first is a geography of task areas. There is puzzle solving, theorem proving, game-playing, induction, natural language, medical diagnosis, and on and on, with subdivisions of each major territory. AI, in this view, is an exploration, in breadth and in depth, of new territories of tasks with their new patterns of intellectual demands. The second view is the functional components that comprise an intelligent system. There is a perceptual system, a memory system, a processing

I am grateful for extensive comments on an earlier draft provided by Jon Bentley, Danny Bobrow, H. T. Kung, John McCarthy, John McDermott, Greg Harris, Zenon Pylyshyn, Mike Rychener and Herbert Simon. They all tried in their several (not necessarily compatible) ways to keep me from error.

*I have already provided some comments, as president, on such matters (Newell, 1980a).

This research was sponsored by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory Under Contract F33615-78-C-1551. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

system, a motor system, and so on. It is this second view that we need to address the role of representation and knowledge.

Figure 2-1 shows one version of the functional view, taken from Newell & Simon (1972), neither better nor worse than many others. An intelligent agent is embedded in a *task environment;* a *task statement* enters via a *perceptual* component and is encoded in an initial *representation.* Whence starts a cycle of activity in which a *recognition* occurs (as indicated by the *eyes*) of a *method* to use to attempt the problem. The method draws upon a memory of *general world knowledge.* In the course of such cycles, new methods and new representations may occur, as the agent attempts to solve the problem. The *goal structure*, a component we all believe to be important, does not receive its due in this figure, but no matter. Such a picture represents a convenient and stable decomposition of the *functions* to be performed by an intelligent agent, quite independent of particular implementations and anatomical arrangements. It also provides a convenient and stable decomposition of the entire scientific field into subfields. Scientists specialize in perception, or problem solving methods, or representation, etc.

It is clear to us all what *representation* is in this picture. It is the data structures that hold the problem and will be processed into a form that makes the solution available. Additionally, it is the data structures that hold the world knowledge and will be processed to acquire parts of the

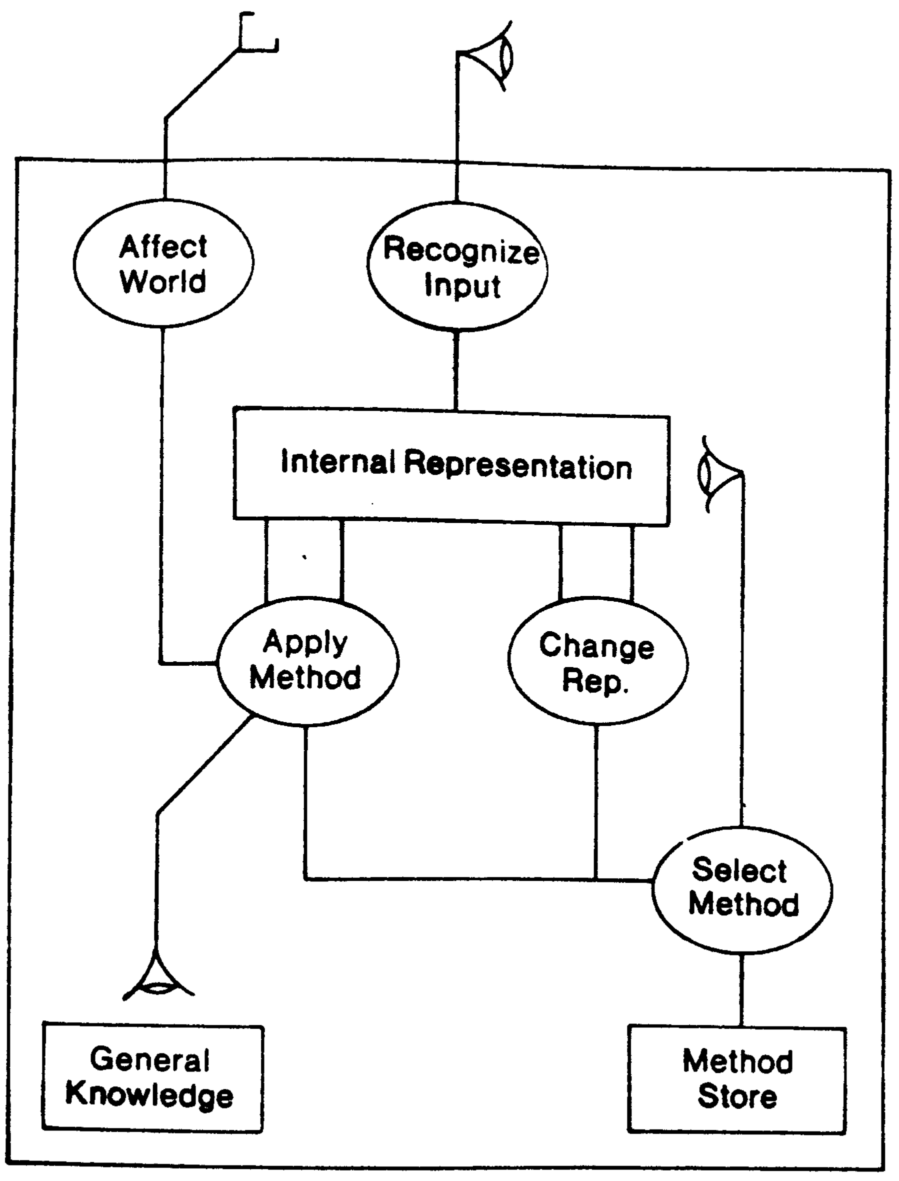

Figure 2-1: Functional diagram of general intelligent agent (after Newell & Simon, 1972).

solution or to obtain guidance in constructing it. The first data structures represent the problem, the second represent world knowledge.

A data structure by itself is impotent, of course. We have learned to take the representation to include the basic operations of reading and writing—of access and construction Indeed, as we know, it is possible to take a pure process view of the representation and work entirely in terms of the inputs and outputs to the read and write processes, letting the data structure itself fade into a mythical story we tell ourselves to make the memory-dependent behavior of the read and write processes coherent.

We also understand, though not so transparently, *why* the representation represents. It is because of the totality of procedures that process the data structure. They transform it in ways consistent with an interpretation of the data structure as representing something. We often express this by saying that a data structure requires an *interpreter*, including in that term much more than just the basic read/write processed, namely, the whole of the active system that uses the data structure.

The term representation is used clearly (almost technically) in AI and computer science. In contrast, the term *knowledge* is used informally, despite its prevalence in such phrases as *knowledge engineering* and *knowledge sources.* It seems mostly a way of referring to whatever it is that a representation has. If a system has (and can use) a data structure which can be said to represent something (an object, a procedure, ...whatever), then the system itself can also be said to have knowledge, namely the knowledge embodied in that representation about that thing.

Why Is There A Problem?

This seems to be a reasonable picture, which is serving us well. Why then is there a problem? Let me assemble some indicators from our current scene.

A first indicator comes from our continually giving to representation a somewhat magical role.* It is a cliche of AI that representation is the *real* issue we face. Though we have programs that search, it is said, we do not have programs that determine their own representations or invent new representations. There is of course some substance to such statements. What is indicative of underlying difficulties is our inclination to treat representation like a *homunculus*, as the locus of *real* intelligence.

A good example is our fascination with problems such as the *mutilated checkboard* problem Newell, 1965). The task is to cover a checkboard with two-square dominoes. This is easy enough to do with the regular board and clearly impossible to do if a single square is removed, say from the upper right corner. The problem is to do it on a (mutilated) board which has two squares removed, one from each of two opposite corners. This task also turns out to be impossible. The actual

*Representation is not the only aspect of intelligent systems that has a magical quality; learning is another. But that is a different story for a different time.

task, then, is to show the impossibility. This goes from apparently intractable combinatorially, if the task is represented as all ways of laying down dominoes, to transparently easy, if the task is represented as just the *numbers* of black and white squares that remain to be covered. Now, the crux for AI is that no one has been able to formulate in a reasonable way the problem of finding the good representation, so it can be tackled by an AI system. By implication—so goes this view—the capability to invent such appropriate representations requires intelligence of some new and different kind.

A second indicator is the great theorem-proving controversy of the late sixties and early seventies. Everyone in AI has some knowledge of it, no doubt, for its residue is still very much with us. It needs only brief recounting.

Early work in theorem proving programs for quantified logics culminated in 1965 with Alan Robinson's development of a machine-oriented formulation of first-order logic called *Resolution* (Robinson, 1965). There followed an immensely productive period of exploration of resolution-based theorem-proving. This was fueled, not only by technical advances, which occurred rapidly and on a broad front (Loveland, 1978), but also by the view that we had a general purpose reasoning engine in hand and that doing logic (and doing it well) was a foundation stone of all intelligent action. Within about five years, however, it became clear that this basic engine was not going to be powerful enough to prove theorems that are hard on a human scale, or to move beyond logic to mathematics, or to serve other sorts of problem solving, such as robot planning.

A reaction set in, whose slogan was "uniform procedures will not work." This reaction itself had an immensely positive outcome in driving forward the development of the second generation of AI languages: Planner, Microplanner, QA4, conniver, POP2, etc. (Bobrow & Raphael, 1974). These unified some of the basic mechanisms in problem solving—goals, search, pattern matching, and global data bases—into a programming language framework, with its attendant gains of involution.

However, this reaction also had a negative residue, which still exists today, well after these new AI languages have come and mostly gone, leaving their own lessons. The residue in its most stereotyped form is that logic is a bad thing for AI. The stereotype is not usually met with inpure form, of course. But the mat of opinion is woven from a series of strands that amount to as much: Uniform proof techniques have been proven grossly inadequate; the failure of resolution theorem proving implicates logic generally; logic is permeated with a static view; and logic does not permit control. Any doubts about the reality of this residual reaction can be stilled by reading Pat Hayes's attempt to counteract it in his *In Defence of Logic* (Hayes, 1977).

A third indicator is the recent SIGART *Special Issue of Knowledge Representation* (Brachman & Smith, 1980). This consisted of the answers (plus analysis) to an elaborate questionnaire developed by Ron Brachman of BBN and Brian Smith of MIT, which was sent to the AI research community working on knowledge representation. In practice, this meant work in natural language, semantic nets, logical formalisms for representing knowledge, and the third generation of programming and representation systems, such as AIMDS, KRL, and KL-ONE. The questionnaire not only covered the basic demography of the projects and systems, but also the position of the respondent (and his system) on many critical isues of representation—quantification, quotation, self-description, evaluation vs reference-finding, and so on.

The responses were massive, thoughtful and thorough, which was impressive given that the questionnaire took well over an hour just to read, and that answers were of the order of ten single-spaced pages. A substantial fraction of the field received coverage in the 80 odd returns, since many of them represented entire projects. Although the questionnaire left much to be desired in terms of the precision of its questions, the *Special Issue* still provides an extremely interesting glimpse of how AI sees the issues of knowledge representation
tion.

The main result was overwhelming diversity—a veritable jungle of opinions. There is no consensus on any question of substance. Brachman and Smith themselves highlight this throughout the issue, for it came as a major surprise to them. Many (but of course not all!) respondents themselves felt the same way. As one said, "Standard practice in the representation of knowledge is the scandal of AI."

What is so overwhelming about the diversity is that it defies characterization. The role of logic and theorem proving, just described above, are in evidence, but there is much else besides. There is no tidy space of underlying issues in which respondents, hence the field, can be plotted to reveal a pattern of concerns or issues. Not that Brachman and Smith could see. Not that this reader could see.

A Formulation of the Problem

These three items—mystification of the role of representation, the residue of the theorem-proving controversy, and the conflicting webwork of opinions on knowledge representation—are sufficient to indicate that our views on representation and knowledge are not in satisfactory shape. However, they hardly indicate a crisis, much less a scandal. At least not to me. Science easily inhabits periods of diversity; it tolerates bad lessons from the past in concert with good ones. The chief signal these three send is that we must redouble our efforts to bring some clarity to the area. Work on knowledge and representation should be a priority item on the agenda of our science.

No one should have any illusions that clarity and progress will be easy to achieve. The diversity that is represented in the SIGART *Special Issue* is a highly articulate and often highly principled. Viewed from afar, any attempt to clarify the issues is simply one more entry into the cacophony—possibly treble, possibly bass, but in any case a note whose first effect will be to increase dissonance, not diminish it.

Actually, these indicators send an ambiguous signal. An alternative view of such situations in science is that effort is premature. Only muddling can happen for the next while—until more evidence accumulates or conceptions ripen elsewhere in AI to make evident patterns that now seem only one possibility among many. Work should be left to those already committed to the area; the rest of us should make progress where progress can clearly be made.

Still, though not compelled, I wish to have a go at this problem.

I wish to focus attention on the question: *What is knowledge?* In fact, *knowledge* gets very little play in the three indicators just presented. *Representation* occupies center stage, with *logic* in the main supporting role. I could claim that this is already the key—that the conception of knowledge is logically prior to that of representation, and until a clear conception of the former exists, the latter will remain confused. In fact, this is not so. *Knowledge* is simply one particular entry point to the whole tangled knot. Ultimately, clarity will be attained on all these notions together. The path through which this is achieved will be grist for those interested in the history of science, but is unlikely to affect our final understanding.

To reiterate: What is the nature of knowledge? How is it related to representation? What is it that a system has, when it has knowledge? Are we simply dealing with redundant terminology, not unusual in natural language, which is better replaced by building on the notions of data structures, interpreters, models (in the strict sense used in logic), and the like? I think not. I think *knowledge* is a distinct notion, with its own part to play in the nature of intelligence.

The Solution Follows From Practice

Before starting on matters of substance, I wish to make a methodological point. The solution I will propose follows from the *practice* of AI. Although the formulation I present may have some novelty, it should be basically familiar to you, for it arises from how we in AI treat knowledge in our work with intelligent systems. Thus, your reaction may (perhaps even should) be "But that is just the way I have been thinking about knowledge all along. What is this man giving me?" On the first part, you are right. This is indeed the way AI has come to use the concept of knowledge. However, this is not the way the rest of the world uses the concept. On the second part, what I am giving you is a directive that your practice represents an important source of knowledge about the nature of intelligent systems. It is to be taken seriously.

This point can use expansion. Every science develops its own ways of finding out about its subject matter. These get tidied up in meta-models about scientific activity, eg, the so-called *scientific method* in the experimental sciences. But these are only models; in reality, there is immense diversity in how scientific progress is made.

For instance, in computer science many fundamental conceptual advances occur by (scientifically) uncontrolled experiments in our own style of computing.* Three excellent examples are the developments of time-sharing, packet switched networks, and locally-networked personal computing. These are major *conceptual* advances that have broadened our view of the nature of computing. Their primary validation is entirely informal. Scientific activity of a more traditional kind certainly takes place—theoretical development with careful controlled testing and evaluation of results. But it happens on the details, not on the main conceptions. Not everyone understands the necessary *scientific* role of such experiments in computational living, nor that standard experimental techniques cannot provide the same information. How else to explain, for example, the calls for controlled experimental validation that speech understanding will be useful to computer science? When that experiment of style is finally performed there will be no doubt at all. No standard experiment will be necessary. Indeed, none could have sufficed.

As an example related to the present paper, I have spent some effort recently in describing what Herb Simon and I have called the *Physical symbol system hypothesis* (Newell & Simon, 1976, Newell, 1980b). This hypothesis identifies a class of systems as embodying the essential nature of symbols and as being the necessary and sufficient condition for a generally intelligent agent. Symbol systems turn out to be universal computational systems, viewed from a different angle. For my point here, the important feature of this hypothesis is that it grew out of the practice in AI—out of the development of list processing languages and Lisp, and out of the structure adopted in one AI program after another. We in AI were led to an adequate notion of a symbol by our practice. In the standard catechism of science, this is not how great ideas develop. Major ideas occur because great scientists discover (or invent) them, introducing them to the scientific community for testing and elaboration. But here, working scientists have evolved a new major scientific concept, under partial and alternative guises. Only gradually has it acquired its proper name.

The notions of knowledge and representation about to be presented also grow out of our practice. At least, so I assert. That does not give them immunity from criticism, for in listening for these lessons I may have a tin ear. But in so far as they are wanting, the solution lies in more practice and more attention to what emerges there as pragmatically successful. Of course, the message will be distorted by many things, eg, peculiar twists in the evolution of computer science hardware and software, our own limitations of view, etc. But our practice remains a source of knowledge that cannot be obtained from anywhere else. Indeed, AI as a field is committed to it. If it is fundamentally flawed, that will just be too bad for us. Then, other paths will have to be found from elsewhere to discover the nature of intelligence.

III. The Knowledge Level

I am about to propose the existence of something called the

*Computer science is not unique in having modes of progress that don't fit easily into the standard frames. In the heyday of paleontology, major conceptual advances ocurred by stumbling across the bones of immense beasties. Neither controlled experimentation nor theoretical prediction played appreciable roles.

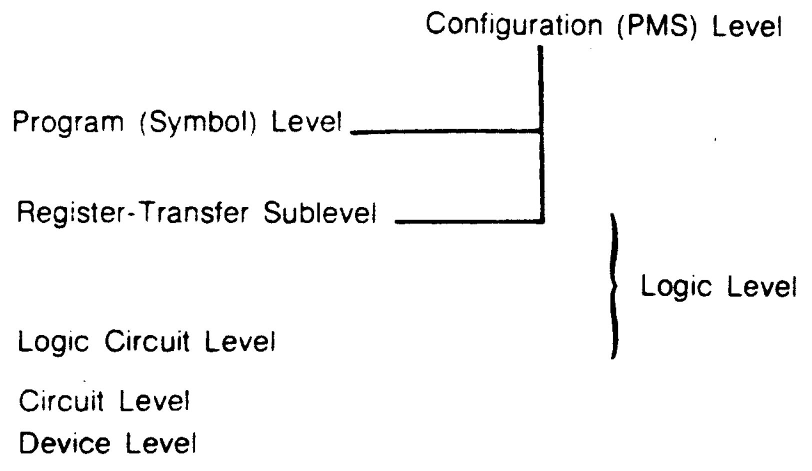

Figure 3-1: Computer system levels.

knowledge level, within which knowledge is to be defined. To state this clearly, requires first reviewing the notion of computer systems levels.

Computer Systems Levels

Figure 3-1 shows the standard hierarchy, familiar to everyone in computer science. Conventionally, it starts at the bottom with the *device level*, then up to the *circuit level*, then the *logic level*, with its two sublevels, *combinatorial and sequential circuits*, and the *register-transfer level*, then the *program level* (referred to also as the *symbolic level*) and finally, at the top, the *configuration level* (also called the *PMS* or *Processor-Memory-Switch level*). We have drawn the configuration level to one side, since it lies directly above both the symbol level and the register-transfer level.

The notion of levels occurs repeatedly throughout science and philosophy, with varying degrees of utility and precision. In computer science, the notion is quite precise and highly operational. Figure 3-2 summarizes its essential attributes. A level consists of a *medium* that is to be processed, *components* that provide primitive processing, *laws of composition* that permit components to be assembled into *systems*, and *laws of behavior* that determine how system behavior depends on the component behavior and the structure of the system. There are many variant instantiations of a given level, eg, many programming systems and machine languages and many register-transfer systems.*

Each level is defined in two ways. First, it can be defined autonomously, without reference to any other level. To an amazing degree, programmers need not know logic circuits, logic designers need not know electrical circuits, managers can operate at the configuration level with no knowledge of programming, and so forth. Second, each level can be reduced to the level below. Each aspect of a level—medium, components, laws of composition and behavior—can be defined in terms of systems at the level next below. The *architecture* is the name we give to the register-transfer level system that defines a symbol (programming) level, creating a machine language and making it run as described in the programmers manual for the machine. Neither of these two definitions of a level is the more fundamental. It is essential that they both exist and agree.

Some intricate relations exist between and within levels. any instantiation of a level can be used to create *any* instantiation of the next higher level. Within each level, systems hierarchies are possible, as in the subroutine hierarchy at the programming level. Normally, these do not add anything special in terms of the computer system hierarchy itself. However, as we all know, at the program level it is possible to construct any instantiation within any other instantiation (modulo some rigidity in encoding one data structure into another), as in creating new programming languages.

There is no need to spin out the details of each level. We live with them every day and they are the stuff of architecture textbooks (Bell & Newell, 1971), machine manuals and digital component catalogues, not research papers. However, it is noteworthy how radically the levels differ. The medium changes from electrons and magnetic domains at the device level, to current and voltage at the circuit level, to bits at the logic level (either single bits at the logic circuit level or bit vectors at the register-transfer level), to symbolic expressions at the symbol level, to amounts of data (measured in data bits) at the configuration level. System characteristics change from continuous to discrete processing, from parallel to serial operation, and so on.

Aspects	Register-Transfer Level	Symbol Level
Systems	Digital Systems	Computers
Medium	Bit Vectors	Symbols, expressions
Components	Registers Functional Units	Memories Operations
Composition Laws	Transfer Path	Designation, association
Behavior Laws	Logical Operations	Sequential interpretation

Figure 3-2: Defining aspects of a computer system level.

Despite this variety, all levels share some common featues. Four of these, though transparently obvious, are important to us:

1) Specification of a system at a level always determines completely a definite behavior for the system at that level (given initial and boundary conditions).
2) The behavior of the total system results from the local effects of each component of the system processing medium at its input to produce its output.
3) The immense variety of behavior is obtained by system structure, ie, by the variety of ways of assembling a small number of component types (though perhaps a large number of instances of each type).
4) The medium is realized by state-like properties of matter, which remain passive until changed by the components.

*Though currently dominated by electrical circuits, variant circuit level instantiations also exist, eg, fluidic circuits.

Computer systems levels are not simply levels of abstraction. That a system has a description at a given level does not necessarily imply it has a description at higher levels. There is no way to abstract from an arbitrary electronic circuit to obtain a logic-level system. This contrasts with many types of abstraction which can be uniformly applied, and thus have a certain optional character (as in abstracting away from the visual appearance of objects to thier masses). Each computer system level is a *specialization* of the class of systems capable of being described at the next level. Thus, it is a priori open whether a given level has any physical realizations.

In fact, computer systems at all levels are realizable, reflecting indirectly the structure of the physical world. But more holds than this. Computer systems levels are realized by *technologies*. The notion of a technology has not received the conceptual attention it deserves. But roughly given a specification of a particular system at a level, it is possible to construct by *routine* means a physical system that realizes that specification. Thus, systems can be obtained to specification limits of time and cost. It is not possible to invent arbitrarily additional computer system levels that nestle between existing levels. Potential levels do not become technologies, just by being thought up. Nature has a say in whether a technology can exist.

Computer system levels are *approximations*. All of the above notions are realized in the real world only to various degrees. Errors at lower levels propagate to higher ones, producing behavior that is not explicable within the higher level itself. Technologies are imperfect, with constraints that limit the size and complexity of systems that can actually be fabricated. These constraints are often captured in design rules (eg, fan-out limits, stack-depth limits, etc), which transform system design from routine to problem solving. If the complexities become too great, the means of system creation no longer constitute a technology, but an arena of creative invention.

We live quite comfortably with imperfect system levels, especially at the extremes of the heirarchy. At the bottom, the device level is not complete, being used only to devise components at the circuit level. Likewise, at the top, the configuration level is incomplete, not providing a full set of behavioral laws. In fact, it is more nearly a pure level of abstraction than a true system level. This accounts for both symbol level and register-transfer level systems having configuration (PMS) level abstractions.*

These levels provide ways of describing computer systems; they do not provide ways of describing their environments. This may seem somewhat unsatisfactory, because a level does not then provide a general closed description of an entire universe, which is what we generally expect (and get) from a level of scientific description in physics or chemistry. However, the situation is understandable enough. System design and analysis requires only that the interface between the environment and the system (ie, the inner side of the transducers) be adequately described in terms of each level, eg, as electrical signals, bits, symbols or whatever. Almost never does the universe of system plus environment have to be modeled in toto, with the structure and dynamics of the environment described in the same terms as the system itself. Indeed, in general no such description of the environment in the terms of a given computer level exists. For instance, no register-transfer level description exists of the airplane in which an airborne computer resides. Computer system levels describe the internal structure of a particular class of systems, not the structure of a total world.

To sum up, computer system levels are a reflection of the nature of the physical world. They are not just a point of view that exists solely in the eye of the beholder. This reality comes from computer system levels being genuine specializations, rather than being just abstractions that can be applied uniformly.

A New Level

I now propose that there does exist yet another system level, which I will call the *knowledge level*. It is a true systems level, in the sense we have just reviewed. The thrust of this paper is that distinguishing this level leads to a simple and satisfactory view of knowledge and representation. It dissolves some of the difficulties and confusions we have about this aspect of artificial intelligence.

A quick overview of the knowledge level, with an indication of some of its immediate consequences, is useful before entering into details.

The system at the knowledge level is the *agent*. The components at the knowledge level are *goals*, *actions* and *bodies*. Thus, an agent is composed of a set of actions, a set of goals and a body. The medium at the knowledge level is *knowledge* (as might be suspected). Thus, the agent processes its knowledge to determine the actions to take. Finally, the behavior law is the *principle of rationality:* Actions are selected to attain the agent's goals.

To treat a system at the knowledge level is to treat it as having some knowledge and some goals, and believing it will do whatever is within its power to attain its goals, in so far as its knowledge indicates. For example:

- "She knows where this restaurant is and said she'd meet me here. I don't know why she hasn't arrived."
- "Sure, he'll fix it. He knows about cars."
- "If you know that 2 + 2 = 4, why did you write 5?"

The knowledge level sits in the hierarchy of systems levels immediately above the symbol level, as Figure 3-3 shows. Its components (actions, goals, body) and its medium (knowledge) can be defined in terms of systems at the symbol level, just as any level can be defined by systems at the level one below. The knowledge level has been placed side by side with the configuration level. The gross anatomical description of a knowledge-level system is simply (and only) that the agent has

*See Bell, Grason & Newell (1972) for a PMS approach to the register-transfer level.