Knowledge Acquisition for Knowledge-Based System Workshop, Banff, UK, Nov. 1986, pp. 26.0–14.

B. McHugh, "Knowledge Acquisition and Nuclear Safety Advisors," *Proceedings of the First European Workshop on Knowledge Acquisition for Knowledge-Based Systems,* Reading University, UK, Sept. 1987, pp. C4.1–10.

Proceedings of the IJCAI Workshop on Knowledge Acquisition, Detroit, Mich., Morgan-Kaufmann, San Mateo, Calif., Aug. 1989.

M. Sharp, "Knowledge Acquisition for Advice Systems: FUSION, A Network Cognitive Support System," *Proceedings of the Fourth Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, Oct. 1989, pp. 28.1–19.

SIGART Newsletter, No. 108 [Special Issue on knowledge acquisition], (Apr. 1989).

S. Tanimoto, *The Elements of Artificial Intelligence,* Computer Science Press, Rockville, Md., 1987.

John H. Boose
Boeing Computer Services

KNOWLEDGE LEVEL

The knowledge level is a level of description for a complex system, such as a computer program. A knowledge-level description characterizes the system in terms of goals, beliefs, and reasoning capabilities, without regard for the actual physical mechanisms that cause behavior. Perhaps the most familiar knowledge-level description is a natural language grammar, which is used to describe patterns in speech. Saying that speakers know the grammar is a knowledge-level description. Knowledge-level descriptions are useful for explaining and predicting behavior. As abstractions, they are necessarily incomplete descriptions, but have proven useful for specifying what a computer program should do (Alexander and co-workers, 1986), comparing representation languages (Clancey, 1985), and predicting the learning capabilities of programs (Dietterich, 1986).

A knowledge-level description is to be distinguished from a symbol-level description, that is, the implementation or encoding of knowledge and reasoning procedures. A knowledge-level description is idealized. It is not concerned with the details of how computations are performed, just with what an agent (program or person) can do. In contrast, a symbol-level description is in terms of a representation language: stored structures and the matching and search algorithms that perform computations. This distinction is particularly useful for abstracting problem-solving architectures so they can be compared and reused, avoiding pragmatic issues of efficiency and storage that disguise computational equivalence.

Newell (1982) introduced the term knowledge-level. His intent was to resolve "mystification of the role of representation, the residue of the theorem-proving controversy, and the conflicting webwork of opinions on knowledge representation." First, encoding tricks (alternative representations) suggested that intelligence is not a matter of what is known, but the structures used to store it, which seems counterintuitive when considering that humans can effectively teach people without knowing how the brain stores knowledge. Second, arguments about the inadequacy of logic as an implementation language obscured its value as a specification language (Hayes, 1977). Third, the proliferating jargon of knowledge representation languages inhibited progress of the community. By highlighting the distinction between the knowledge level and symbol level, Newell sought to bring the question "What is knowledge?" to the forefront, making issues of representation and logic secondary. "Knowledge serves as a specification of what the symbol structure should be able to do" (Newell, 1982).

Newell uses a systems approach, characterizing the whole system (an agent) according to different levels of abstraction. Specifically, a computer system has these levels, viewed from the top down: knowledge, symbol, register–transfer, logic–circuit, circuit, and device. Crucially, knowledge is attributed by an observer to explain why the agent's behavior is rational. That is, the observer assumes that an agent, if it has certain knowledge, will use its knowledge to achieve its goals (the principle of rationality). Observing a pattern of behavior, the pattern is explained in terms of consistent goals and beliefs. For example, if an agent knows where a restaurant is located and the agreed meeting time, then the principle of rationality suggests that the agent will take actions to be at the restaurant at that time. Failures to accomplish goals are thus stated in terms of lack of knowledge or inability to take some action, not in terms of how the knowledge is represented.

Dietterich (1986) effectively applied Newell's levels of analysis to compare machine learning programs. He found that some existing programs are not capable of knowledge-level learning. Instead, the programs modify their symbol structures to access stored representations more efficiently. He called this symbol-level learning. Dietterich found that other programs are capable of learning knowledge, but their behavior cannot be predicted or described at the knowledge level. He called this nondeductive knowledge-level learning. Dietterich's analysis clearly demonstrates the advantage of subtracting away issues of implementation, to describe what a program can do. Levesque's (1984) analysis is similar, characterizing knowledge representations in terms of memory and inference functions.

Knowledge-level descriptions have been especially useful in the area of knowledge engineering (Stefik, in press). A special concern is the development of general-programming tools (generic shells) that can be used to develop expert systems in many domains. By early 1980s, attention turned to formalizing representation languages and problem-solving methods that were specialized for different tasks, such as diagnosis and design. A knowledge-level analysis permits the development of abstract methods in particular programs so they can be reused. Specifically, a knowledge-level description of an expert system indicates what knowledge is necessary for a task and what kinds of inferences can be made, without regard for the particular encoding (eg, rules and frames) or the methods of pattern matching and search control (eg, blackboard and constraint optimization).

Clancey (1985) showed that a wide variety of expert systems can be described at the knowledge-level by a problem-solving method called heuristic classification. These programs relate data (observable information) to hypotheses (conjectured problem solutions) in systematic ways: abstracting data, relating data categories heuristically to hypothesis categories, and refining hypotheses. A key feature of heuristic classification is the selection of a solution from a preenumerated list or hierarchy. Such a description shows how concepts are typically related by the program, without regard for how they are encoded. As Newell intended, this allows researchers to compare their work and separate out symbol-level issues of efficiency and storage.

Clancey argued further that expert systems could be related to traditional systems analysis in this way. Knowledge bases are characterized in terms of the system being modeled, the purpose for constructing a model (the task), the way in which processes are modeled (eg, chronological stage classifications, state-transitions, and functional composition), the modeling method (heuristic classification selecting a process model vs configuration of new process descriptions), encoding in a representation language (eg, production rules), and finally the encoding in a programming language (eg, C and LISP) (Clancey, 1985; in press *a*). These ideas have been applied for developing expert system shells and specialized knowledge acquisition tools (Alexander and co-workers, 1986; McDermott, 1988). Chandrasekaran (1985) and Sticklen (1989) have extended Newell's analysis to describe a problem-solving agent in terms of a cooperating society of more primitive agents. This decomposition deals with complexity by exploiting and formalizing the different roles knowledge plays.

More recently, Clancey (1991) argues that the knowledge level has relativistic properties. A knowledge-level description is of an agent in its environment. It is an observer's theory, not representations possessed by the agent being studied.

Knowledge-level descriptions concern emergent patterns that develop over time, not plans (eg, schemas and grammars) that are represented inside the agent before its history of interaction begins. Knowledge-level descriptions are useful and necessary to describe, predict, explain, etc patterns of interaction that develop between the agent and its environment. Crucially, knowledge-level descriptions cannot be reduced to (replaced by) symbol-level descriptions. That is, they are a level above individual agents. Unlike the relation between register–transfer and logic–circuit levels, for example, the knowledge level cannot be realized as physical structures in an agent in isolation. Extending this idea, Clancey argues that classification models of any system interacting with its environment (eg, disease models) in principle cannot be replaced by structure–function models that describe how the system is constructed (ie, deep models). The identification of such frame of reference issues is currently having a major effect on the design of robots and models of human behavior in what is called situated cognition.

BIBLIOGRAPHY

J. H. Alexander, M. J. Freiling, S. J. Shulman, J. L. Staley, S. Rehfuss, and M. Messick, "Knowledge Level Engineering: Ontological Analysis," *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, Pa., 1986, AAAI, Menlo Park, Calif.

B. Chandrasekaran, "Generic Tasks in Knowledge-Based Reasoning: Characterizing Systems at the 'Right' Level of Complexity," *Proceedings of the IEEE Second Annual Conference on Artificial Intelligence Applications*, 1985.

W. J. Clancey, "Heuristic Classification," *Artif. Intell.* **27**, 289–350 (1985).

W. J. Clancey, "Model Construction Operators," *Artif. Intell.*, in press.

W. J. Clancey, "The Frame of Reference Problem in the Design of Intelligent Machines," in K. vanLehn, ed., *Architectures for Intelligence: The Twenty-Second Carnegie Symposium on Cognition*, Erlbaum, Hillsdale, N.J., 357–422 (1991).

T. Dietterich, "Learning at the Knowledge Level," *Machine Learning* 1(3), 287–315 (1986).

P. Hayes, "In Defence of Logic," in *Proceedings of the Fifth IJCAI*, Cambridge, Mass., Morgan-Kaufmann, San Mateo, Calif., 1977.

H. J. Levesque, "Foundations of a Functional Approach to Knowledge Representation," *Artif. Intell.* **23**(2), 155–212 (1984).

J. McDermott, "Preliminary Steps Toward a Taxonomy of Problem-Solving Methods," in S. Marcus, ed., *Automating Knowledge Acquisition for Expert Systems*, Kluwer Academic Publishers, Boston, 1988.

A. Newell, "The Knowledge Level," *Artif. Intell.* **18**(1), 87–127 (1984).

M. Stefik, *Introduction to Knowledge Systems*, in preparation, Morgan-Kaufmann, Publishers.

J. Sticklen, "Problem-Solving Architecture at the Knowledge Level," *J. Exper. Theor. Artif. Intell.* 1(4), 233–248 (1989).

WILLIAM J. CLANCEY
Institute for Research on Learning

KNOWLEDGE REPRESENTATION

The dominant paradigm for building intelligent systems since the early 1970s has been based on the premise that intelligence presupposes knowledge. Thus to build a program that performs, say, diagnosis for an infectious disease, it is necessary to identify the units of knowledge used by human experts as they perform the diagnostic task and to make them available to the system under development. Moreover, it is necessary to characterize the patterns of reasoning used by the human expert (deductive, hypothetical, or heuristic) and endow the intelligent system with analogous reasoning capabilities. This methodology is applicable for systems intended to perform any task requiring intelligence, be it diagnosis, planning, design or interpretation. Generally, knowledge is represented in the system's *knowledge base*, which consists of data structures and programs. In addition, the intelligent system is expected to have a program called an *inference engine* that implements the reasoning patterns necessary for the task at hand. Thus current AI theory and practice dictate that intelligent systems be knowledge based, consistent with this simple knowledge base plus inference engine architecture. This emphasis on knowledge has led