Expert Systems: Principles and Practice*

Edward A. Feigenbaum
Professor of Computer Science
Stanford University

Most applications of the Artificial Intelligence (AI) science and technology are of a type called Expert Systems. An Expert System (ES) is a computer program that reasons using knowledge to solve complex problems. This overly brief caricature will be expanded upon below, but it serves to indicate an alignment of ES with AI's long term goals. Traditionally, computers solve complex problems by arithmetic calculation (not reasoning using logic); and the knowledge needed to solve the problem is known only by the human programmer and used to cast the solution method in terms of algebraic formulas.

The Emergence of the Principles and the Technology

One way to approach an understanding of the principles of expert systems is to trace the history of the emergence of ES within AI. Perhaps AI's most widely shared early goal had been the construction of extremely intelligent computers. That is, in addition to other goals (such as learning and language understanding), most researchers shared a goal to produce programs that performed at or beyond human levels of competence. For example, an early program took the NY State Regents Examination in Plane Geometry (late 1950s) to validate its human-level abilities in this domain. Also famous predictions were made regarding what year an AI program would be chess champion of the world. A scientific viewpoint about knowledge representation, declarativism, was formulated by McCarthy during this period, and proved to be robust and important. Declarativism insists that a program's knowledge about objects and relations be encoded explicitly so that other programs can access and reason with that knowledge.

---

* This article is adapted from text in [1] and will appear in *The Encyclopedia of Computer Science and Engineering,* 1992.

This work was supported by NASA Cooperative Agreement No. NAG2-581 (under ARPA Order 6822), by NASA Cooperative Agreement NCC2-537 and by National Library of Medicine Grant NLM-EP 3 P41 LM05208-18S1.

In the early 1960s, AI's focus shifted, from performance to generality--how one problem solving mechanism can solve a wide variety of problems. The most well known AI efforts of the time were the general problem solvers, both heuristic programs (GPS) and theorem provers. While these programs exhibited considerable generality, the actual problems these were able to solve were very simple, essentially toy, problems: i.e. high generality, low power.

In 1965, Feigenbaum, Lederberg, and Buchanan at Stanford University initiated a project in modeling scientific reasoning, for which high performance as a goal was once again given prominence. The task of this program, called DENDRAL, was to interpret the mass spectrum of organic molecules in terms of an hypothesis of the structure of the organic molecule that was present in the instrument [2].

The intent of DENDRAL's designers was that the program was to perform the difficult mass spectral analysis task at the level of competence of specialists (Ph.D.s) in that area. As it turned out, AI's problem solving methods (primarily search-based methods) were useful, but not sufficient. Most important in achieving the goal of expert-level competence was knowledge of chemistry and mass spectroscopy. The key empirical result of DENDRAL experiments became known as the knowledge-is-power hypothesis (later called the Knowledge Principle), stating that *knowledge of the specific task domain in which the program is to do its problem solving was more important as a source of power for competent problem solving than the reasoning method employed* [3].

The knowledge that DENDRAL needed was provided by scientists of the Stanford Mass Spectrometry Laboratory, in an intense collaboration with the AI scientists. Such efforts at codifying the knowledge of specialists for use in expert systems later came to be called Knowledge Engineering. The first use of the term "expert" in connection with such programs was made in an article analyzing the generality vs power issue in the light of the results of DENDRAL computational experiments [4].

In short, the DENDRAL program was the progenitor of the class of programs subsequently called expert systems, and the development of DENDRAL illustrated the major principles, issues, and limitations of expert systems that will be discussed later. A case has been made [5] that an MIT project of the same era, MACSYMA, that built a program to assist people in doing complex symbolic mathematics, shared with DENDRAL the same underlying view of the primacy of domain-specific knowledge and the same motivation to achieve high levels of competence in performance. However, the MACSYMA papers and extended work were focused on the domain of symbolic mathematics, whereas the DENDRAL papers and extended work actually provided the foundation for most of the subsequent ES technology.

Extensions of DENDRAL, done in the 1970s, were of two types. First, many pioneering expert systems were built by AI researchers to explore and extend the new technology and, by the sheer weight of accumulating evidence of capability, to lend credibility to the ES technology and to knowledge engineering. The earliest and most famous of these was the MYCIN system, which showed the first integrated architecture for interactive consultation between an expert user and an ES, including explanation of the line of reasoning [6] and was the first of a fruitful line of applications of ES to medical diagnosis [7]. Second, the underlying programming systems and languages of these ESs, that embodied the reasoning procedures and the framework for representing knowledge, were generalized so that they were no longer domain-specific and therefore could be used as programming systems and languages for the construction of new, albeit similar, ESs. Such software came to be known as ES "development environments" or "ES tools" or "ES shells". The pioneering tool/shell, derived from MYCIN, was EMYCIN [8], the prototype for literally dozens of commercially available ES shells.

Transfer of ES Technology into Practice

As the 1970s came to a close, so did the decade of laboratory exploration of the first-generation ES ideas and techniques. A period of transfer of the technology to industrial use began and is still underway (1991). The two best known early industrial applications were XCON from Digital Equipment Corporation [1] and the Dipmeter Adviser from Schlumberger Ltd. [9].

XCON's task was the configuration under constraints of a DEC minicomputer from a large number of component subassemblies. The configuration task was done so fast and so accurately by XCON that DEC saved tens of millions of dollars per year in manufacturing and sales operations. XCON configures minicomputers about 300 times faster than human engineers and does it virtually error-free. XCON has been extended to many different families of DEC computers and other equipment; and was generalized into an ES called XSEL to assist DEC salespeople to correctly configure and price equipment sales at the time of customer contact. XCON had an immediate effect outside of DEC: most other computer manufacturers copied the idea for their own operations. More important, the success of XCON opened the way to a broader generalization: most devices that are engineered and manufactured are built out of component subassemblies; hence the XCON idea could be used to realize a new generation of CAD, or "intelligent CAD" as it is called [10].

Schlumberger’s Dipmeter Adviser (DA) is typical of programs that analyze streams of data (with or without real-time considerations) and propose hypotheses to account for and explain the data. (DA) interprets the data taken from instruments lowered into bore holes during the search for oil and gas. It offers hypotheses about the tilt, or so-called “dip”, of the rock layers far beneath the earth’s surface. Knowing the dip of each of the hundreds of rock layers in a bore hole is valuable in oil exploration.

Another example of a program that interprets signal data to produce hypotheses is the Charley program of General Motors, for analyzing vibrations of machinery as a means of troubleshooting mechanical equipment, particularly automotive and manufacturing equipment [11].

This ES is a model of the expertise of a senior GM specialist in vibrational analysis whose retirement was imminent, and is an instance of a very important class of ESs that are done to capture and preserve rare corporate expertise.

With similar motive, an ES was built by DEC for Lend Lease, the largest construction firm in Australia, for the task of estimating the time to completion of high rise buildings with an accuracy of plus/minus ten percent from just the first few hours of preliminary discussions with the customer [1]. Similarly, also, NKK Steel Co. in Japan modeled the expertise of one of their senior specialists in blast furnace operations (he is called at NKK “the furnace god”) in an ES that predicts in near-real time the probability of two different types of catastrophic failures of blast furnace operations [1]. This ES was built with the intention of selling it to buyers of NKK blast furnaces (NKK makes equipment as well as steel).

The use of an ES to improve the quality of human decisions was the motive behind the well known Authorizers’ Assistant (AA) of American Express. To assist a human authorizer with the decision to allow or disallow a customer charge, AA analyzes a large amount of customer data from the database files that AmEx keeps on the customer; then AA issues a recommended decision; offers the explanation or rationale for its recommendation; and gives the human authorizer a screenful of the data that supports its recommendation [1]. The payoff in terms of avoiding bad debt and fraud amounts to millions of dollars per year.

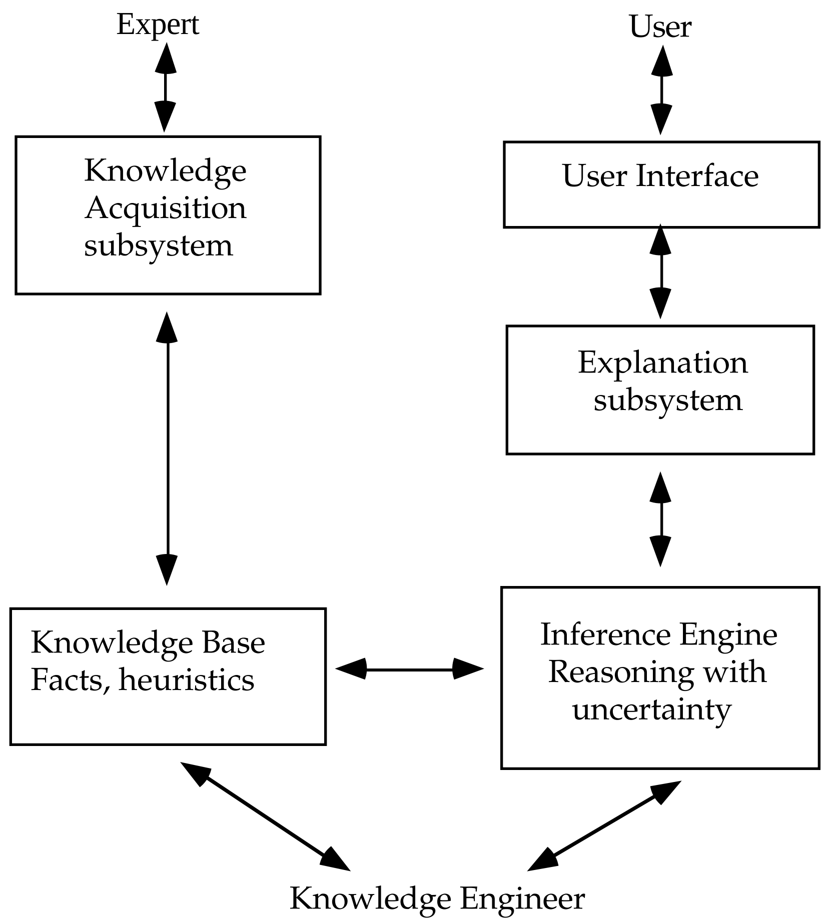
ESs have been built to assist people with the accurate and timely processing of “cases” in the context of very complex systems of bureaucratic laws, rules, and regulations. A system done for the British Social Security Administration assists clerks in answering the written queries of citizens concerning their pensions [1]. The Taxpayers' Assistant Expert System of the US Internal Revenue Service helps IRS personnel to give accurate tax information in response to telephoned queries from taxpayers (usually in the few months before tax returns are due). And in both Fresno and Tulare Counties of California, ESs assist social welfare case workers with the decision about whether a person applying for welfare qualifies under the complex rules.

There are major applications of ES across the complete spectrum of human professional and semi-professional work: in medicine, law, manufacturing, sales, maintenance, engineering, architectural design, finance, insurance, and so on. Presently (1991) there are tens of thousands of operational ESs, in the USA, Japan, and Europe primarily; though increasing numbers are seen in Australia, Singapore, and India (which has a National Center for Expert Systems in Hyderabad).

Expert System Technology

The word "expert" in expert system refers to the intention of the ES designer to have the system achieve a level of competence of problem solving in some domain of work that rivals the performance of human specialists (experts) in that domain. To accomplish this, the ES must be given the knowledge that such human experts have that distinguishes experts from novices and enables experts to perform well. To acquire and represent that knowledge is the job of the knowledge engineer. Increasingly, with the advance of ES development tools, experts are able to be their own knowledge engineers. ESs are almost always used as <u>interactive intellectual aids</u> for human decision makers, almost never as autonomous unsupervised agents.

Every expert system consists of two principal parts: the <u>knowledge base</u> and the <u>reasoning, or inference, engine.</u> (See Figure 1.) The knowledge base contains both factual and heuristic knowledge. The factual knowledge is that knowledge widely shared in the domain and commonly agreed upon by experts. The heuristic knowledge is the non-rigorous, experiential knowledge, the rules-of-thumb, the knowledge of good judgment. Heuristics constitute the "art of good guessing" in the domain.



Basic Structure of an Expert System

Figure 1.

Knowledge representation formalizes and organizes the knowledge for use by the inference engine. One widely used representational form is the production rule, or simply the rule. A rule consists of an IF part and a THEN part (also called a condition and an action). The IF part lists a set of conditions in some logical combination. The piece of knowledge represented by the rule is relevant to the line of reasoning being developed if the IF part of the rule is satisfied; consequently the THEN part can be concluded, or its action taken. Expert systems whose knowledge is represented in rule form are called rule-based systems. This kind of representation is action-oriented.

Another widely used representational form, called the unit (or frame, or schema) is based upon a more passive object-oriented view of knowledge. Systems of units (sometimes called frame-based systems) are siblings of the object-oriented systems common in Computer Science. Typically, a unit consists of a symbolic name, a list of attributes of some entity and the values associated with the attributes. That is, the unit is a complex symbolic description of an entity that the ES needs to know about. There is a knowledge base management system, akin to a database management system, associated with the units. One of its important functions is to handle automatically some routine inference functions for knowledge updating and knowledge propagation. The automatic handling is called inheritance.

Another general and powerful formalism for representing knowledge is the "standard" mathematical way, given by the symbols and formulas of mathematical logic, particularly first order predicate logic and some higher-order logics [12].

In addition to naturalness and expressiveness of the representational form, a representation needs to be very modular and flexible. The process of building a knowledge base is an iterative one that has been called "an incremental approach to competence." The knowledge is teased out of the expert and the problem domain little by little on each iteration. The "module size" needs to match these little pieces, and the knowledge "modules" need to be easily integrated into the existing, growing knowledge base with virtually no incremental reprogramming of the knowledge base.

In every expert system, the inference engine (embodying the problem solving method or procedure) uses the knowledge in the knowledge base to construct the line of reasoning leading to the solution of the problem. The most common method (the method of choice in rule based systems) involves chaining of IF-THEN rules. If the chaining starts from a set of conditions and moves toward some (possibly remote) conclusion, the method is called forward chaining. If the conclusion is known (e.g. it is a goal to be achieved) but the path to that conclusion is not known, then backward chaining with rules is employed, seeking conditions under which a particular line of reasoning will be true.

Sometimes an inflexible commitment to either forward or backward chaining is not optimal, especially if new data is arriving needing interpretation, and if a changing situation demands that goals change. In these cases, an opportunistic strategy that allows the flexible mixing of some forward chaining with some backward chaining is used. Opportunistic problem solving strategies are the hallmark of blackboard systems [13].

Other procedures commonly found in expert systems are procedures for reasoning with uncertain information and knowledge and procedures for explaining the line of reasoning to the user.

Knowledge of a domain and of a particular problem is almost always incomplete and uncertain. To deal with uncertainty, a rule may have associated with it a confidence factor (CF), or weight. A standard calculus for using CFs to construct and evaluate lines of reasoning is available. In an alternative method, called fuzzy logic, uncertainty is represented by a distribution of values, and another standard calculus is available for handling these distributions. Finally, where sufficient statistical data is available, Bayes Theorem and its associated calculations have been used [9].

Since ESs explicitly build lines of reasoning, there is little difficulty in responding to users' questions about the line of reasoning: how and why did it take the form that it has? Typical questions that can be answered by the explanation procedure are: "Why are you asking me that (particular) question?; or "How did you conclude (something)? Also "Show me your line of reasoning (main conclusions, intermediate and final)." The explanation procedure answers these questions by looking back at what rules were examined, which were selected and why, and what conclusions were drawn.

From the applications viewpoint, the AI science offers only a small number of practical techniques for representing knowledge, for making inferences (certain and uncertain) and for generating explanations. Commercial software systems are offered that carefully integrate various selections from this small menu to assist knowledge engineers who do not want to program the techniques themselves to build their expert systems. These are the hybrid commercial tools or shells of the ES software industry, with representative names like KEE (Knowledge Engineering Environment) and ART (Advanced Reasoning Tool). The use of these tools/shells is very widespread--indeed, the method of choice in building expert systems. Powerful tools/shells are available on all platforms from PCs to mainframes [14].

Note, however, that although tools/shells simplify programming ESs, they do not usually help with the crucial bottleneck problem of knowledge acquisition. As the Knowledge Principle informs us the choice of reasoning method is important, but not nearly as important for the ES's ultimate performance as the accumulation of high-quality knowledge. It might be asked why automatic methods of extracting knowledge from experts, from textbooks, and from real world data have not destroyed this bottleneck? Stating the question in another way, where do we stand with machine learning for ES applications? Although the field of machine learning research is quite vigorous and exciting, virtually no techniques have emerged to help with ES knowledge acquisition (with the exception of a few simple inductive algorithms that allow the inductive formation of some kinds of rules from data sets [15].

In the early 1990s, the size of a typical ES was 10^2 - 10^3 rules and/or objects. Many ESs were of order 10^4. Only a few (perhaps even just one or two) were of order 10^5.

Research Aimed at Removing Limitations

The techniques that made first generation ESs work at all were also responsible for the key limitations of the technology. These are narrowness (or overspecialization) and its sibling brittleness. The Knowledge Principle tells us that an ES has little competence outside of the domain of specialization for which its knowledge base has been carefully and systematically built. Since the ES knows only **that**, and has no ability to generalize, analogize, or in any other way extend knowledge, it can only solve problems of **that** kind--thereby exhibiting narrowness. If a problem posed by a user is simply beyond the boundary of what the ES knows about, the ES's performance degrades ungracefully (with brittleness), falling from expert levels of competence to complete incompetence. The boundaries of an ES's knowledge, hence the margins of its competence, are almost never represented explicitly so that this knowledge can inform the user and also be available to the reasoning process.

First generation ESs usually represent associational or phenomenological knowledge, i.e. knowledge near the surface of events, not the knowledge of what is deep below the surface, causing the events. Such deeper knowledge is called "first principles" knowledge or model-based knowledge. Methods for representing such knowledge, and procedures for the inference engines of model-based reasoners, are important topics being vigorously researched by the AI community. It is readily seen that this kind of knowledge helps to remove the limitations of narrowness and brittleness, since the use of principles and models generalizes much more readily than does the phenomenological description of events (for example on the IF side of a rule).

The breadth of an ES's competence can also be extended with cases, that is "worked examples" that have been experienced in the past. Human experts know a great many particulars not necessarily in the rule form but in the case form. A major branch of second-generation ES research is concerned with the representation of case libraries and methods for reasoning from cases that are fundamentally different from the earlier methods based on logical chaining, fuzzy logic, or Bayes' Theorem.

Finally there is the engineering economics of ES construction. Each ES is built from scratch. In first generation ESs, there is almost no reuse of knowledge and there is no systematic way in which a community of knowledge engineers cooperate to allow their ESs to share knowledge. The conceptual infrastructure of knowledge sharing and reuse is presently a major topic of research, and will be heard from during the 1990s under such names as knowledge interchange formats (KIF), ontolingua, shared ontologies, national engineering knowledge bases, and national "common sense" knowledge bases [16].

References

[1] Feigenbaum, E., McCorduck, P., and Nii, H. P., *The Rise of the Expert Company,* New York, 1988: Times Books.

[2] Lindsay, R.K, Buchanan, B.G., Feigenbaum, E.A. and Lederberg, J., *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project*, 1980, McGraw-Hill.

[3] Feigenbaum, E.A., "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering," *Proceedings of the International Joint Conference on Artificial Intelligence*, Cambridge, MA, 1977.

[4] Feigenbaum, E.A., Buchanan, B.G. and Lederberg, J., "On Generality and Problem Solving: A Case Study Using the DENDRAL Program," in *Machine Intelligence 6*, B. Meltzer and D. Michie (eds), 1971: Edinburgh University Press.

[5] McCorduck, P., *Machines Who Think*, San Francisco, 1979: Freeman.

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

[7] Clancey, W.J. and E.H. Shortliffe, *Readings in Medical Artificial Intelligence: The First Decade*, Reading, MA, 1984: Addison-Wesley.

[8] van Melle, W., Shortliffe, E.H. and Buchanan, B.G., "EMYCIN: A Knowledge engineer's Tool for Constructing Rule-Based Expert Systems," in *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*," B.G. Buchanan and E.H. Shortliffe (eds), 1984: Addison-Wesley.

[9] Buchanan, B.G. and R.G. Smith, "Fundamentals of Expert Systems," in *The Handbook of Artificial Intelligence, Vol. IV*, A. Barr, P.R. Cohen and E.A. Feigenbaum (eds), 1989: Addison-Wesley.

[10] Katajamaki, M., "Knowledge-Based CAD," in *Expert Systems with Applications,* Vol. 3, 1991: Pergamon.

[11] Bajpai, A. and R. Marczewski, "Charley: An Expert System for Diagnostics of Manufacturing Equipment," in *Innovative Applications*

of Artificial Intelligence, H. Schorr and A. Rappaport (eds), 1989: AAAI Press/MIT Press.

[12] Genesereth, M. and N. Nilsson, *Logical Foundations of Artificial Intelligence*, San Mateo, CA, 1987: Morgan Kaufmann.

[13] Nii, H. Penny, "Blackboard Systems," in *The Handbook of Artificial Intelligence, Vol. IV*, A. Barr, P.R. Cohen and E.A. Feigenbaum (eds), 1989: Addison-Wesley.

[14] Harmon, P. and B. Sawyer, *Creating Expert Systems for Business and Industry*, New York, 1990: Wiley.

[15] Firebaugh, M.W., *Artificial Intelligence: A Knowledge-Based Approach*, Boston, MA, 1988: Boyd and Fraser.

[16] Neches, R., et al, "Enabling Technology for Knowledge Sharing," AI Magazine, Vol. 12, No. 3, Fall 1991.