# Some Guidelines for Knowledge Acquisition Strategies

**Caroline Hayes, Michael Fu, Dan Gaines**
University of Illinois at Champaign-Urbana
hayes@cs.uiuc.edu, mfu@cs.uiuc.edu, dmgaines@cs.uiuc.edu

## Abstract

Knowledge acquisition is a large bottleneck in the construction of complex intelligent problem solvers. Unfortunately, the construction of effective KA tools to ease this process is also a bottleneck. In this paper we attempt to present some guidelines that can be used to determine whether one would do best to adopt a manual KA strategy, a computer-assisted KA strategy, or a completely automated KA strategy. Three knowledge properties that are relevant to determining the applicable KA strategy are the *generality* of the knowledge, the *ease* with which the expert can express this knowledge, and the degree of *control* that the expert wishes to exert over that part of the knowledge base. We will examine knowledge in an automated planner, $P^3$, and a designer's assistant, SEDAR. The KA strategies that are most likely to be the advantageous for aquiring the various types of knowledge will be identified. [1]

## 1 Introduction

Knowledge acquisition is difficult, regardless of whether it is done by 1) a knowledge engineer using completely manual techniques, 2) a knowledge engineer using the guidance of an intelligent computer assistant, or 3) completely automated machine learning techniques. For this reason the knowledge acquisition process is a huge bottle neck in the construction of complex automated problem solvers and computer assistants for tasks such as design, planning and diagnosis. The task of implementing a complex problem solving engine using manual knowledge aquistion strategies can require years of effort. Once the system is complete, the task of maintaining and updating it can be just as daunting. For these reasons it is very appealing to think about developing computer assistants that can aid or automate the task of knowledge acquisition and greatly reduce the burden of system development and maintenance. However, the burden of developing good knowledge acquisition tools is also very great. It is critical to focus the development of KA tools on those areas that will have the most impact.

In this paper we will describe two systems which operate in the planning and design domains respectively, then summarize some of the different types of knowledge that occur in these problem solvers and offer some recommendations on which knowledge acquisition strategy is best matched to which knowledge type.

## 2 Background

*Knowledge acquisition* (KA) refers to any and all methods for gathering knowledge about how to solve problems. This knowledge is collected for the purpose of constructing automated problem solvers, and intelligent-assistants to aid in the solution of complex, "knowledge-rich" tasks such as architectural design, medical diagnosis, and manufacturing planning. Such systems are often referred to *knowledge-based systems.* A knowledge-rich system that assists a user in a

[1]This work is supported in part by NASA grant NASA NAG-1-613, and by Core of Army Engineers

problem solving task (in contrast to systems that automate tasks) is called an intelligent computer assistant.

Problem solving knowledge could be collected either from a practitioner who is an expert in solving a particular class of problems, or it could be collected from a set of problem solving examples. The practitioner, who's problem solving methods are studied, is referred to as the *domain expert.* A person who collects knowledge from a domain expert is known as a *knowledge engineer.*

Sometimes, the *user* of a knowledge-based tool is also a domain expert and possibly a knowledge engineer as well. In such cases, the tool is used to magnify or multiply the abilities of the expert-user. In other cases, the user is relatively inexperienced in the problem area and the knowledge-based tool is used to enable the novice-user to solve problems which they would otherwise lack the knowledge to solve.

**Knowledge Aquistion Strategies.** Knowledge acquisition can happen through a wide variety of strategies that vary continuously from completely manual to totally automated. In *manual* strategies, knowledge engineers interview domain experts in order to understand their problem solving methods. The knowledge engineers summarize the information they learn in the form of rules or program statements. This is a very flexible strategy for knowledge acquisition. It was also the only strategy available for constructing early knowledge-based systems. However, for many tasks it is also quite tedious and time consuming.

In *computer-assisted* knowledge acquisition, an intelligent computer assistant aides in the process of collecting rules and data and forming them into program statements. Such knowledge acquisition tools may guide a knowledge engineer through the process, or they may allow a domain expert to interact directly with the system, reducing the need for a knowledge engineer. Such systems, where they have been successfully applied, have greatly reduced the time and effort required to construct knowledge-based tools [MM89], [Esh88].

In totally *automated* knowledge acquisition systems, the KA system (rather then the knowledge engineer or domain expert) automatically formulates rules which encode problem solving knowledge, and some times domain concepts, as well. Such a system may work from examples provided by an expert, or it may passively observe the expert solving problems. Such systems could potentially reduce the burden on knowledge engineers and domain experts even further, but they are difficult to construct effectively.

## 3 Knowledge Properties

There are a number of properties that influence the appropriateness of utilizing a given knowledge acquisition strategy to acquire a given knowledge type. The following list of properties represents the ones we have observed to be important in the problem domains we have studied. These properties include the:

- **Generality of knowledge** to a broad or narrow family of problem domains.
- **Ease in expressing knowledge concepts** experienced by domain experts.
- **Degree of control desired** by expert/user over knowledge in a part of the system.

**Generality of knowledge** refers to the size of the family of problems to which the knowledge applies. Knowledge may be specific to a broad family of related domains, a single domain, or even to a single user. One can view problem solvers as belonging to a taxonomy of problem solving families. Within one problem solving system one may find some knowledge that may be very specific to one narrow domain such as roofing design, some that is shared by a broader problem solving family such as the family of mechanical design problems, and some that is general to almost all problem solvers. It may be desirable to customize a system not only to a specific problem domain, but also to a specific

user group or even to a single user. User knowledge includes items such as user preferences in task ordering and preferences in operator selection.

Contrary to what some people may presume, the more general the knowledge, the less utility there may be in developing an specialized tool to acquire that knowledge. If the knowledge will only be codified once and will be reused many times, then manual knowledge acquisition strategies may be most appropriate. Automated and computer-assisted KA tools are probably most useful in specializing general frameworks developed for families of problem solvers into high performance, domain specific problem solvers. A well designed KA tool can be reused many times to specialize the framework into several specific domains.

The ability to specialize a system to meet the preference of a particular user may be important in achieving user acceptance of a system. Users feel comfortable and confident in using a tool if they can feel that they "own" it, in the sense that they can customize it to meet their specific needs. Since it is the users (who are often also experts), and not the system, who is ultimately responsible for the correctness and quality of the solutions it is important that the user be able to tune the system to the specific company practices, properties of their own facility, or personal preferences.

**Ease in expressing concepts.** Another factor influencing the applicability of automated and computer-assisted knowledge acquisition tools is the ease or difficulty that experts experience in expressing concepts relevant to particular knowledge types.

If experts can easily articulate concepts accurately, then the most appropriate strategies may be either either manual or computer-assisted. These strategies allow the user to interact directly with the knowledge base.

If concepts are hard for the expert to articulate directly, a computer-assisted strategy may not be appropriate. However, it may be possible for a knowledge engineer to tease these concepts out by numerous examples. This is a highly tedious task for human knowledge engineers to perform manually. In this case, a totally automated strategy may be very useful that can generalize the solutions given by an expert in response to examples.

It is important to note, however, that sometimes a concept may be difficult for the expert to express because the computer formulation of the problem is not well suited to task. We found this to be the case in the planning case-study that will be described later. Reformulation of the problem turned a hard-to-elicit concept into an easy-to-elicit one.

**Degree of control desired** by expert/user over knowledge in system.

The degree of control desired by experts and users may also influence what type of KA technique is appropriate. There are some types of knowledge that users have no great desire to examine provided that system performance appears to be correct. However, frequently, experts and users want to be able to inspect much of the knowledge in the system, and to correct it if it is not to their standards. Users may loose some control over what knowledge is acquired when completely automated KA techniques are used. However, is such strategies can also provide a way for users to offer corrective inputs then automated techniques may still be applicable even when users desire a high degree of control.

Figure 1 summarizes the relationship of knowledge properties to the three knowledge acquisition strategies. Each of the three properties is shown on one axis. The shaded zones indicate which KA strategy is most applicable for a given combination of properties. For example, suppose one wishes to add domain specific knowledge to a general problem solving framework in order to specialize it. Also suppose that it is easy for the expert to articulate this knowledge, and the system developer wants to repeat this process for several domains. Then it may appropriate to develop a computer-assisted KA

tool that will guide an expert in entering knowledge.

## 4 Knowledge Domains – Case Studies

Two systems are described in this section: $P^3$ [Hay96b] an automated manufacturing planner, and SEDAR [FHE96], an intelligent support tool to aid in the task of roofing design. The initial implementation of these systems has been done using completely manual strategies, however it is one of our goals to use automated strategies, where possible, to adapt these frameworks to other closely related planning and design domains. These systems will be described in terms of the the types of knowledge they contain, the properties of that knowledge (general, user specific, easy to express, hard to express, etc.) and the knowledge acquisition techniques that are likely to be applicable.

### 4.1 $P^3$: An Automated Manufacturing Planner.

$P^3$ (Positional tolerance Process Planner) is an automated planner which produces manufacturing plans for fabrication of metal parts. Its intended use is either as a manufacturing engineer's assistant, or as a desinger's assistant. As a designer's assistant, $P^3$ can provide the designer with immediate feedback during the design process on their design's manufacturing feasibility and cost [HS95], [Hay96a].

This planner takes a specification of a mechanical part, and automatically produces a manufacturing plan to fabricate it. All planning tasks in this domain involve a problem solving framework in which the over-all goal (the part) is subdivided into a number of manufacturing subgoals (holes, slots, pockets, etc), a number of candidate manufacturing methods are generated for each sub-goal, interactions between methods are determined, and then lastly methods are selected and sequenced in a near optimal way by a given set of criteria. Knowledge used in implementing this domain included knowledge about the general problem solving architecture, how to generate manufacturing operators, how to determine interactions between operations, criteria for a desirable plan, and preferences between operator choices.

In particular, for this domain we found that much of the knowledge in embodied in the basic problem solving architecture was relatively stable across closely related domains. Development of the problem solving architectures that apply to families of domains may be best developed by manual KA techniques.

However, it became apparent that there was a need to provide some way of modifying the system to adapt to changes in manufacturing technology. When the manufacturing technology changed, or when we wanted to adapt the planner to slightly different manufacturing tasks, we found it was necessary to change many of the operator descriptions. This is a task that is tedious for both experts and knowledge engineers. So tedious, in fact, that industrial practitioners express reluctance to adopt generative planners because of the difficulty of updating and maintaining them. However, the solution we devised was not to build a KA tool to make it easier to enter operators, instead we changed the representations used by the planner. Instead of representing domain properties in operators, we constructed an operator generator that takes descriptions of the manufacturing equipment, and uses that information to generate operators. It is much easier for the domain expert to enter a description of the equipment and its capabilities than it is to enter operator descriptions. By changing the representation we turned a difficult KA task into an easy one.

Additionally, there was a need to allow users to input their own criteria for desirable plans, often in the form of preference rules to determine which operators and trade-offs are preferable in a given situation. However, preferences are a type of knowledge that is very specific to particular users. Furthermore, specifying preferences is a task that users find relatively easy to do. They also desire a great deal of control over this task. However, it is very tedious to enter these preferences by manual KA strategies. This is

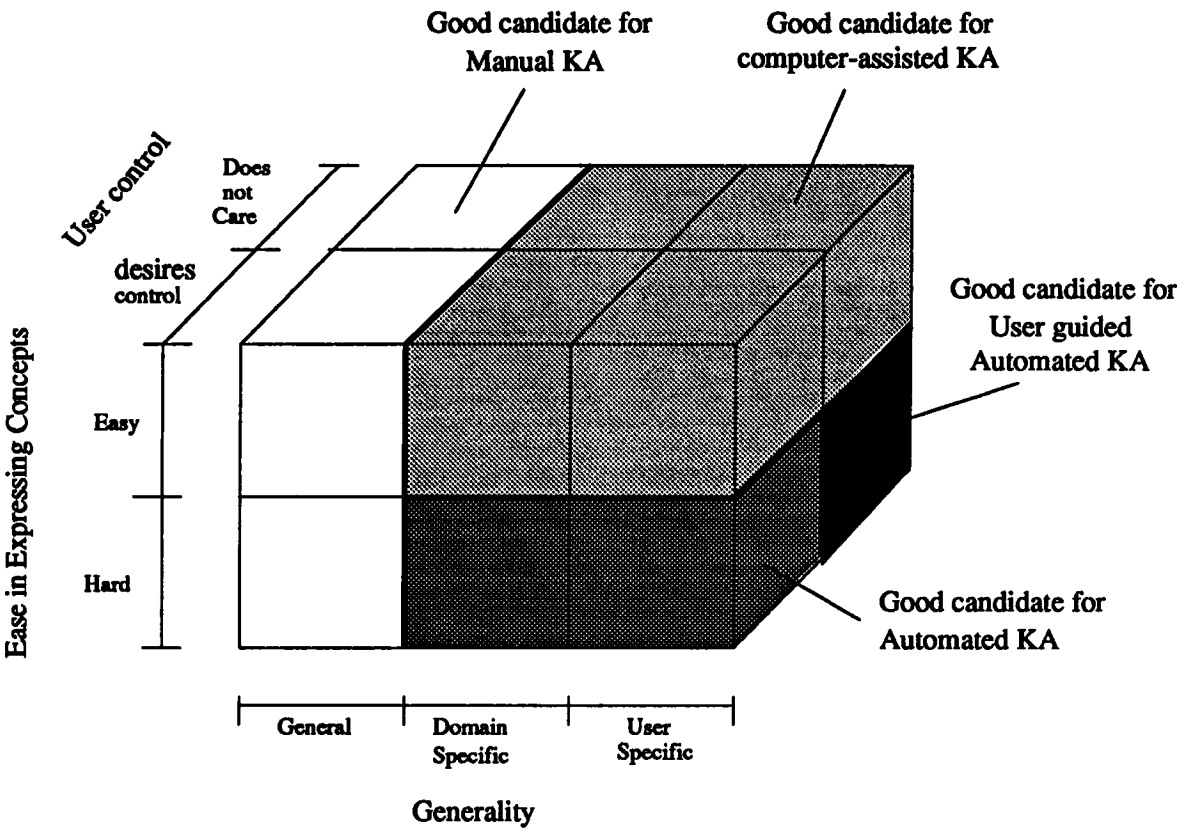


Figure 1: Relationship of knowledge properties to knowledge acquisition strategies

a KA task that is probably very well suited to computer-assisted KA tools.

### 4.2 SEDAR: A Designer's Assistant.

SEDAR is a desinger's assistant which provides critiques and intelligent support during the design process. This system is very different than P3 in that P3 automatically generates complete solutions while SEDAR's function is to guide human designers in generating solutions, and to provide design evaluations or partial solutions when asked. SEDAR allows designers to develop solutions while it tracks their behavior, and provides critiques of the partially developed solution. Thus, SEDAR does not need to know how to generate complete solutions, but it does need to know enough about the design task in order to track the user's progression through the design problem.

By modeling the design sub-task structure and tracking the user's focus within this structure, the system can provide relevant critiques. *Relevant* critiques are ones that pertain to the user's current focus of attention. As a counter-example, an irrelevant critique would be one in which the system flags required structures as missing from the design if the designer simply has not gotten to those structures yet.

Issues the system needs to address are 1) it must allow each user to *flexibly* determine the order in which they will address design sub-tasks, and 2) it must deliver relevant critiques in an unobtrusive way which would not interfere or constrain the user's problem solving. The types of knowledge that SEDAR uses to address these goals include: critiquing strategies, design sub-tasks and and their relationships with each other, and knowledge of building codes.

The critiquing strategies used by SEDAR turned out to be a fairly general type of knowledge. The same strategies can be applied to many problem domains. Once acquired by a knowledge engineer, can be re-used for a variety of related domains. Manual KA techniques are appropriate for understanding and devising these critiquing strategies.

Knowledge about design subtasks and their inter-relations, and knowledge about building codes are both very specific to the domain of roofing design. However, by installing a different problem sub-task structure, and a different set of codes, we believe the system could be easily adapted to provide critiquing support in other closely related configuration design tasks. Since both of these knowledge types are easy to express, it may be appropriate to develop a computer-assisted KA tool to enable experts to quickly and effectively specify this information.

## 5 Discussion

Devising a good problem solving architecture is an important part of a system's problem solving knowledge, and is still largely an art. Manual KA strategies may still be the best methods for devising the very general and reusable parts of a system.

In the manufacturing planning and roof design domains, there turned out to be very little need for completely automated KA strategies. However, in other domains, such speach in tasks such as correctly pronounce vowel sounds, one could imagine that automated KA/learning techniques, such as neural nets, may be quite applicable.

For elicitation of domain and user specific knowledge, computer-assisted KA strategies may be most appropriate, especially if the knowledge is also easy to express, such as equipment properties, sub-task hierarchies, and building codes.

## 6 Conclusions

Knowledge acquisition is a large bottleneck in the construction of complex intelligent problem solvers. KA tools can greatly reduce system development time, but unfortunately construction of effective KA tools to ease this process is also a bottleneck. In this paper we attempt to present some guidelines that can be used to determine for a given knowledge-type whether one would do best to adopt a manual KA strategy, a computer-assisted KA strategy, or a completely

automated KA strategy.

Three knowledge properties that are relevant to determining the applicable KA strategy are the *generality* of the knowledge, the *ease* with which the expert can express this knowledge, and the degree of *control* that the expert wishes to exert over that part of the knowledge base. Computer assisted KA strategies lend themselves well to acquiring knowledge that is both easy for the expert to express, and will be used to specialize a general framework to a variety of domains. Completely automated KA strategies are appropriate for similar specializing KA tasks, but which are difficult for the expert to express unless it is through examples. Old fashioned but flexible manual KA strategies are probably best for obtaining general knowledge like problem solving architectures and general critiquing strategies.

Lastly we examined an automated planner, $P^3$, and a designer's assistant, SEDAR. We identified knowledge types used in these programs, the properties of those knowledge types, and the type of KA strategy that is likely to be the most advantageous in adapting these frameworks to other closely related domains.

## References


[Esh88] L. Eshelman. *Mole: A Knowledge-aquistion tool for cover-and-differentiate systems.* Kluwer, 1988.

[FHE96] M. C. Fu, C. C. Hayes, and E. W. East. Sedar: An expert critiquing system for flat and low-slope roof design and review. *submitted to Journal of Computing in Civil Engineering*, 1996.

[Hay96a] C. C. Hayes. $p^3$: A planner that reasons about cost effective plans to achive positional tolerances. *Journal of Applications of Engineering Systems.*, 1996.

[Hay96b] C. C. Hayes. $p^3$: A process planner for manufacturability analysis. *IEEE Robotics and Automation, special issue on Assembly and Task Planning*, 1996.

[HS95] C. C. Hayes and H. C. Sun. Using a manufacturing constraint network to identify cost-critical areas of designs. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AIEDAM): Special Issue on Innovative Approaches to Concurrent Engineering*, 9(2), May 1995.

[MM89] S. Markus and J. McDermott. Salt: A knowledge aquisition language for propose-and-revise systems. *Artificial Intelligence*, 39(1), 1989.