5th AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop
Banff, Canada, November, 1990.

AN ARCHITECTURE FOR INTEGRATED KNOWLEDGE ACQUISITION SYSTEMS

Brian R Gaines

Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4
gaines@cpsc.ucalgary.ca

ABSTRACT

An architecture for knowledge acquisition systems is proposed based upon the integration of existing methodologies, techniques and tools developed within the knowledge acquisition, machine learning, expert systems, hypermedia and knowledge representation research communities. Existing tools are analyzed within a common framework to show that their integration can be achieved in a natural and principled fashion. A detailed architecture for integrated knowledge acquisition systems is proposed that also derives from parallel cognitive and theoretical studies. An implementation of this architecture around a knowledge representation server is described.

INTRODUCTION

The past five years has seen an explosion in research on, and application of, knowledge acquisition methodologies, techniques and tools (Boose, 1989). The knowledge acquisition community world-wide has grown in numbers and scope of projects. There are significant international collaborative developments involving the sharing of ideas and software. However, the comments with which I began my paper at this meeting three years ago still apply:

"The problem now is not so much to access research and experience in knowledge acquisition but to make sense of the diverse and wide ranging material available and, in particular, to apply the results to improve effectiveness in the development of knowledge-based systems. There are a number of major impediments to such understanding and application:

- A diversity of techniques and tools that overlap in their applications and where it is not clear whether they are competitive alternatives or complementary partners.
- Lack of variety, detail and evaluation in the case histories of applications of the techniques and tools.
- Lack of access to the techniques and tools outside the narrow research communities originating them.
- Lack of standardization in the knowledge representation resulting from the techniques and tools making it difficult to integrate them and to interface them to existing knowledge based systems.
- Lack of standardization in the forms of data required by the techniques and tools making it difficult to apply them in the same situation and compare them.
- Lack of standardization in the user interfaces to the interactive tools making it difficult to integrate them in an effective environment for human-computer interaction.
- Lack of portability in the run-time environments required by the tools making it difficult to integrate them with other systems." (Gaines, 1987)

In 1987 I went on to say, "These are not unreasonable problems at this stage of development of knowledge based systems." In 1990 I would like to propose to this meeting that these problems are becoming highly unreasonable. We should accept them as the major research challenges for the 1990s, and develop an action plan to overcome them. They are challenges not only for us as individual researchers, research groups and organizations, but also for the knowledge acquisition community as a whole. Indeed many of the problems are essentially ones for the community not the individual—we can all contribute but no one of us alone can solve them.

This paper draws on the research and results of many researchers in the knowledge acquisition and knowledge representation communities to propose a practical architecture for future knowledge-based systems. It synthesizes a system design from what already exists, putting a diversity of well-founded and widely used approaches to knowledge acquisition within an integrative framework. The design is intended to be clean and simple, easy to understand, and easy to implement. It is very open in its architecture, providing a frame within which to experiment with new approaches to knowledge acquisition rather than a cage that locks us in to static paradigms. It is also intended to be practically applicable by being based on what we already know how to do well.

The open, international collaboration in the knowledge acquisition community is by far the most important phenomenon in current research. Ideas and technologies need to circulate freely in order to be in the right place at the right time—the place where the appropriate researcher can give them new impetus, and the time when he or she has the creative opportunity to do so. This paper is not about anybody's system or research product. It is a distillation of ideas from the whole community put forward as a contribution to a next generation of system designs. It encapsulates the objectives of many projects world-wide. I hope it will encourage the continuation and expansion of collaborative research.

KNOWLEDGE ACQUISITION PARADIGMS

This paper will adopt a techniques and tools perspective on knowledge acquisition. A companion paper (Gaines & Shaw, 1990) provides a more foundational cognitive science and logical perspectives, and appropriate links will be made to it. There are other significant perspectives that focus on methodological issues (Shaw & Woodward, 1989; Wetter & Woodward, 1990). If the tools are well designed, the foundational perspectives should provide theoretical justification for them, and the methodological perspectives should provide meta-knowledge guiding their use. Hopefully, the systematic design and presentation of the tools architecture will expedite the analysis of these deeper issues.

Four major paradigms underlie the most widely applied knowledge acquisition tools:

1 Direct editing of knowledge in a semantic network, frame, rule, representation

Graphic editors providing direct access to semantic network representations allowing knowledge to be encoded in frames and rules provide the most common development environment for knowledge-based systems. They are part of the application programming support environment of most expert system shells, and the widespread availability of modern graphic workstations has made it possible to provide excellent knowledge visualization environments.

A wide range of knowledge acquisition tools have been developed that structure and improve the graphic editing environment, often taking advantage of domain knowledge to provide a more specific, meaningful and familiar knowledge framework to the expert.

Examples are MOLE (Eshelman, Ehret., McDermott & Tan, 1987), KNACK (Klinker, Bentolila, Genetet, Grimes, & McDermott, 1987), SALT (Marcus, 1987), ONTOS (Monarch & Nirenburg, 1987), Cognosys (Woodward, 1990), KEATS (Motta, Eisenstadt, Pitman & West, 1988), CODE (Skuce, Shenkang & Beauville, 1989), CAMEO (Jones, 1989), and techniques directly supporting domain ontologies such as KADS (Breuker & Wielinga, 1987).

Figure 1 characterizes the major features of these direct editing systems. The expert interacts through a graphic interface with an underlying semantic network knowledge representation schema which may already contain pre-encoded domain knowledge. What is elicited are:

- The distinctions that the expert makes about domain entities (attributes and relations).
- The way in which these distinctions are grouped and constrained to form major concepts.
- The entailments between concepts that constitute procedural, decision-making rules in the domain.

Older systems have less well-structured knowledge representations but Figure 1 captures the essence of recent developments in knowledge representation that are moving towards very clean, and theoretically well-founded schema.

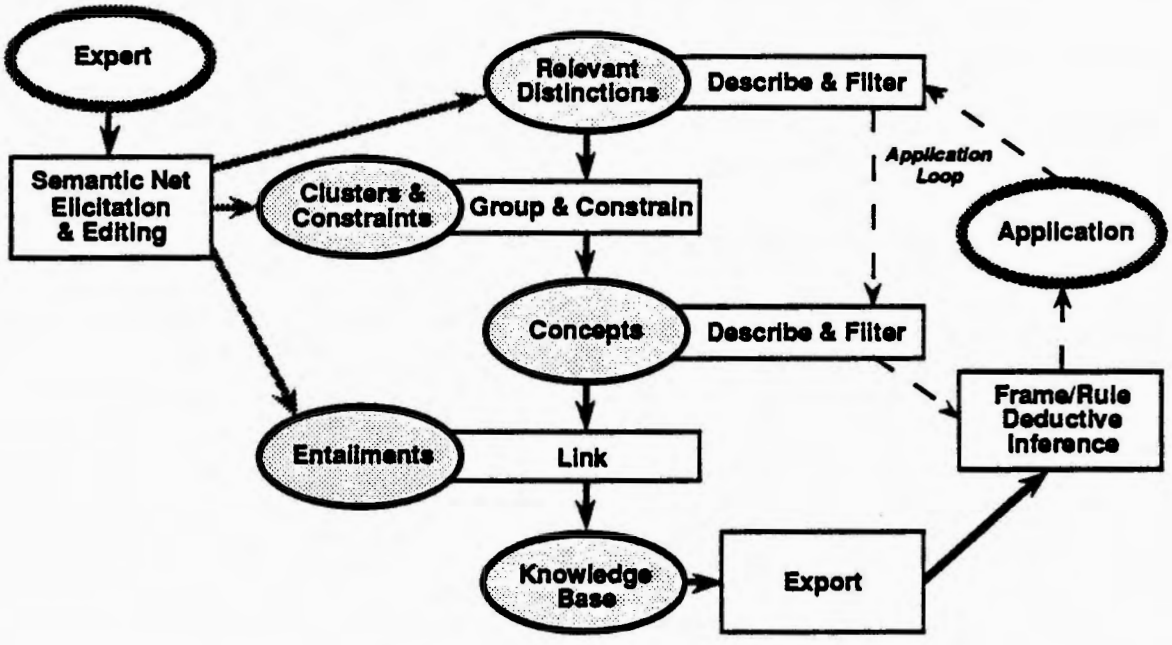


Fig.1 Semantic net architecture

The knowledge base of frames and rules developed in this way is then usually exported to a performance tool and validated against test case data. This generates the application loop shown on the right in which the expert's distinctions lead to a description of the problem which is structured through the concepts leading to the application of the inference rules that link them. Many acquisition tools also incorporate some form of performance tool so that this validation can be made part of the elicitation process.

The advantage of knowledge editing tools is that, if the knowledge is overtly available, they provide a fast and effective rapid prototyping environment for its direct capture. Other techniques assume that the nature of expertise is such that much knowledge is not so directly available. However, effective editing tools may be seen as essential bedrock to any integrated knowledge acquisition architecture. There will usually be significant

sub-domains where direct elicitation is possible, and provides the fastest, most effective means of knowledge capture. There is always a need to present and make available for validation and editing the knowledge structures captured by indirect methods.

Many indirect knowledge acquisition tools leave the knowledge presentation and editing to the associated performance tools since these often have excellent facilities. However, in an integrated architecture it is important to incorporate editors in the knowledge acquisition tool that interact effectively with all the different forms of knowledge captured. One of the major problems to be overcome is that once the knowledge has been exported and edited in the performance system it has lost its relation to the acquisition system. Most current knowledge acquisition tools do not support long-term development and knowledge base maintenance largely because of this lack of integration.

2 Inductive derivation of knowledge from data sets of varying quality

When experts cannot directly enter a knowledge structure emulating their expertise, they may still be able to point the knowledge engineer towards case histories that incorporate that expertise in being described in terms of relevant attributes and correct decisions. Empirical induction techniques may then be used to derive knowledge structures underlying the decisions made in these cases (Michalski & Chilausky, 1980).

The best known empirical induction methodology is that of Quinlan's ID3 which has been refined in many ways, particularly to tolerate noise (incorrect decisions), resulting in the latest implementation, C4.5 (Quinlan, 1987). The original decision tree structure of ID3 based on a subsumption hierarchy of concepts with rules at the leaf nodes only is unnecessarily large in many situations, and extensions to ID3 have been developed that generate modular production rules directly, such as Cendrowska's (1987) Prism. This also can be extended to deal with noisy data as in Induct (Gaines, 1989).

The extensions that deal with noisy data also make it possible to combine the decision tree and modular rule methodologies to generate default reasoning in which rules are placed at non-leaf nodes in the subsumption structure, and more specialized rules override more generalized ones. Such default rule structures are more compact than either decision trees or modular rules—they are generated by both C4.5 and Induct.

Figure 2 shows the minimal default rule structure generated by Induct for Cendrowska's (1987) contact lens problem—the notation is that from Gaines & Shaw (1990)—ellipses denote concepts, black arrows denote the subsumption relation, and gray arrows rules. What is significant about this diagram is that it illustrates the way in which empirical induction techniques can be used to generate conceptual structures for a domain. The left hand and right hand sides of a rule each defines a concept. With default rules in particular these give rise to a natural subsumption ordering. Thus the distinctions, concepts and entailments shown directly entered in Figure 1 can also be indirectly generated through induction as shown in Figure 3.

Figure 3 characterizes the major features of conceptual induction systems. The expert indicates a database whose designer has supplied distinctions to categorize the world and cases described in terms of these distinctions to represent it. What is derived are:

- That subset of the distinctions that are relevant to the decisions.
- The way in which these distinctions are grouped and constrained to form concepts.
- The entailments between concepts that regenerate the decisions in the database.

Classic empirical induction tools do not generate the conceptual structure but this is a fairly simple extension.

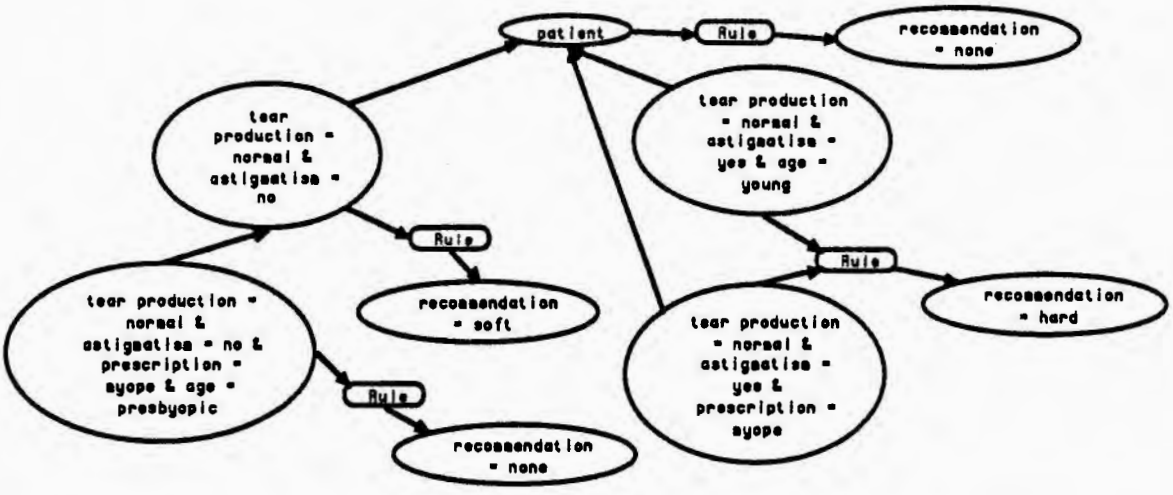


Fig.2 Conceptual structure generated through empirical induction

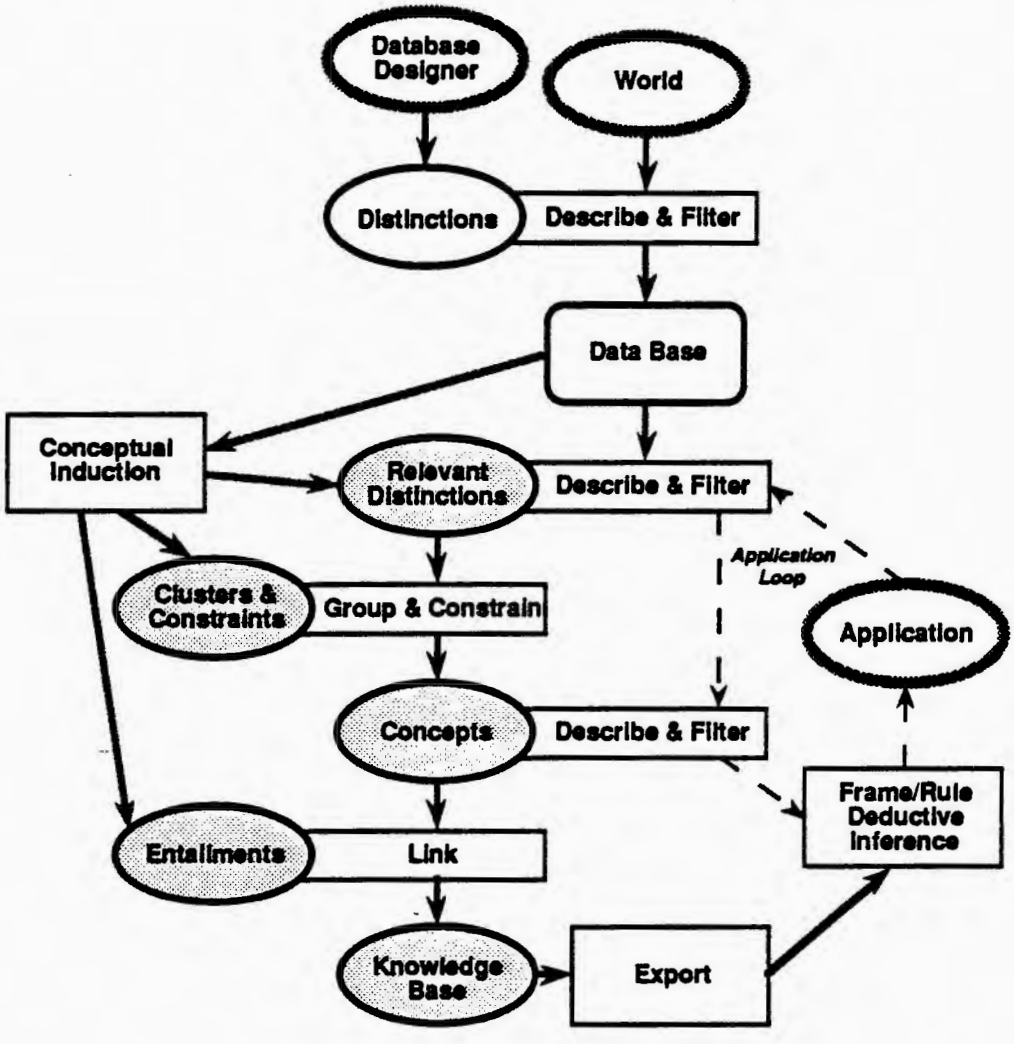


Fig.3 Conceptual induction architecture

Figures 1 and 3 indicate strong similarities between the outcomes of direct knowledge elicitation and conceptual induction. This is as it should be since both techniques are building a complete knowledge base. However, what is not apparent is the relative efficiencies of the two methodologies—that is, how large a database is required to generate the knowledge required to solve a problem? Figure 4 shows the results of one study to investigate the relationship between empirical induction and expertise transfer as knowledge acquisition methodologies (Gaines, 1989). Cendrowska's contact lens data was subjected to random distortion with known statistics to generate large datasets with a certain number of irrelevant binary attributes and a certain percentage of incorrect decisions. Induct was then run on the dataset with 5,000 items, 4,999 items, and so on, until the dataset failed to generate rules giving correct performance. This was done ten times for different datasets of the same type to give estimates of the mean and standard deviation of the size of dataset required to generate correct performance for different forms and levels of distortion.

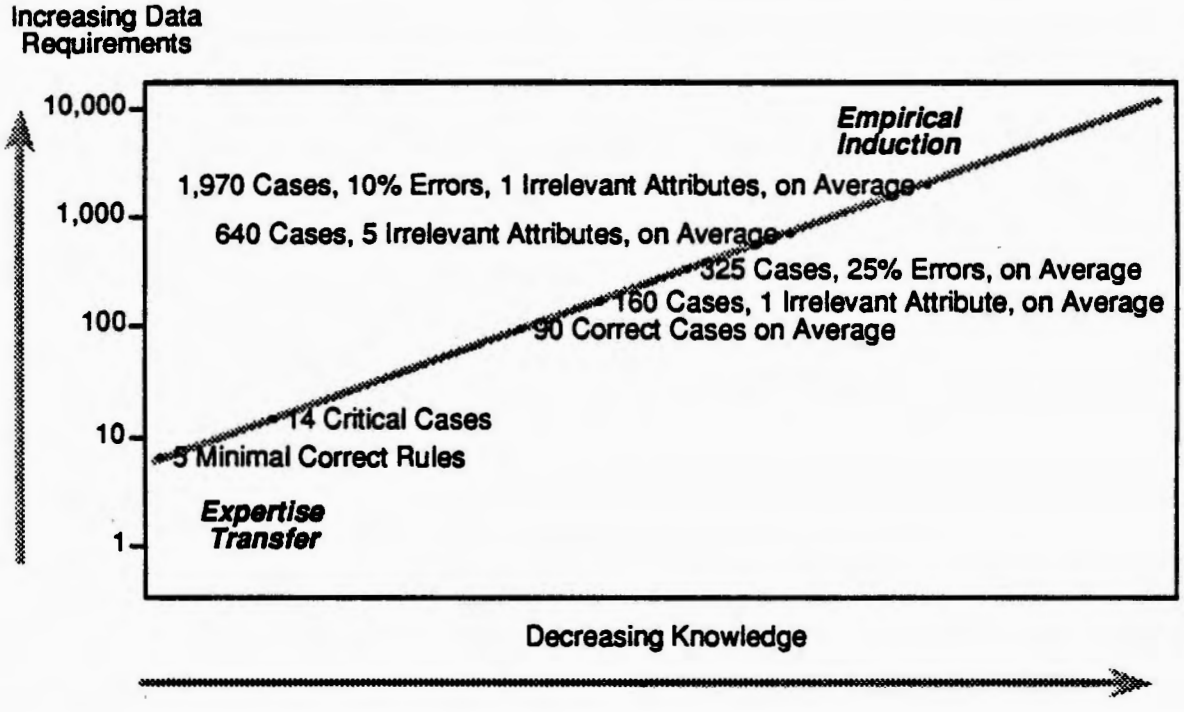


Fig.4 Data/knowledge tradeoff relating expertise transfer and empirical induction

The results shown in Figure 4 indicate the very wide range of the tradeoff between data and knowledge: from direct entry of the minimal knowledge structure of 5 default rules; through entry of 14 critical cases; through an average of 90 randomly selected correct cases; to 325 cases with 25% errors; 640 with 5 irrelevant binary noisy attributes; to 1,970 when a single irrelevant attribute interacts with a 10% error rate.

The moral from Figure 4 is not that expertise transfer is better than empirical induction, although the direct entry of overt knowledge is clearly highly ergonomic if it is available. It is rather that both techniques are capable of producing equivalent quality knowledge, and there is a continuum between them in which knowledge is traded for data. The knowledge of the database designer is implicit in the distinctions made and data selected, and can be made explicit through induction. In practice, it is rare to be working at either extreme of this curve in system development—the repertory grid techniques described in the next section focus on the middle region, attempting to elicit relevant distinctions and critical cases, and building conceptual structures and rules by analyzing these.

3 Indirect elicitation through critical cases described in relevant attributes

Repertory grids are another technique for knowledge elicitation when experts cannot directly enter a knowledge structure. They stand between direct editing and empirical induction, and prompt the expert for distinctions relevant to the problem domain and for critical cases that exhibit significant phenomena in the domain. The prompting is done through online analysis of the data being entered leading to feedback to the expert suggesting missing distinctions and cases. This highly focused feedback aids the expert in developing his or her mental model of the domain. It also reduces the inefficiencies of duplication and the mental blocks of psychological set, supporting rapid prototyping.

A wide range of knowledge acquisition tools have been developed that incorporate repertory grid elicitation and analysis as their major interface to the expert. Examples are PLANET (Shaw & Gaines, 1983), ETS (Boose, 1984), AQUINAS (Boose & Bradshaw, 1987), KRITON (Diederich, Ruhmann & May, 1987), KITTEN (Shaw & Gaines, 1987), and KSS0 (Gaines & Shaw, 1987).

Figure 5 characterizes the major features of repertory grid elicitation systems. The expert interacts through a graphic interface to enter individuals in the domain (elements) and bipolar distinctions (constructs). Conceptual clustering techniques are used to feedback the elicited domain structure in an easily assimilated form for validation. Rule induction is used to generate entailments, or, more recently, conceptual induction as in Figure 3 to generate a default rule structure. What is elicited are:

- The distinctions that the expert makes about domain entities.
- Critical cases exhibiting the major phenomena affecting decision-making in the domain.
- The way in which distinctions are grouped and constrained to form concepts.
- The entailments between concepts as induced from the critical cases.

Older systems did not have explicit conceptual induction but left grouping into concepts or frames as a task for the export module.

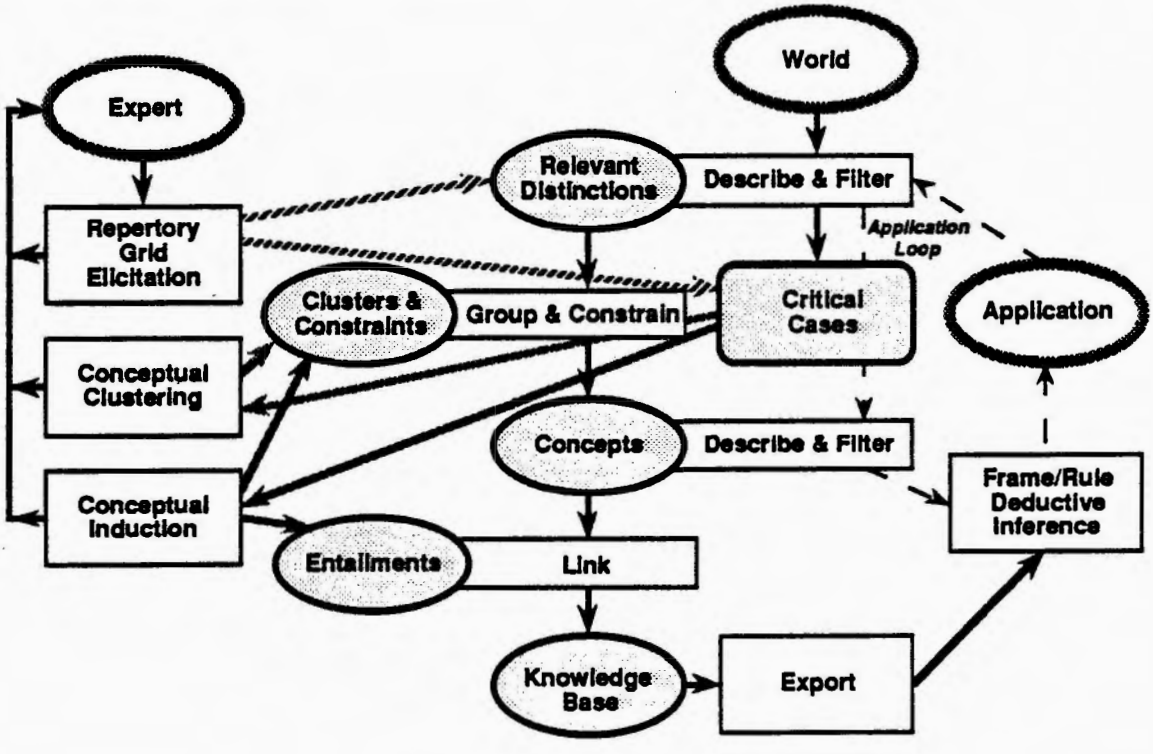


Fig.5 Repertory grid architecture

The clustering and induction modules in Figure 5 are extensions of the basic repertory grid technique incorporated in PLANET and KSS0, and other major extensions have been incorporated in other tools. In particular, AQUINAS makes provision for a wider range of data types than the rating scales of the basic grid, and also allows hierarchies of cases and attributes to be specified that are related to those of semantic nets. Both AQUINAS and KSS0 also incorporate tools supporting multiple sources of expertise and analyzing the relationships between different sources.

There are also other indirect methods of knowledge elicitation that have been used to enhance interaction with direct editing tools. For example, Cognosys analyzes protocols in terms of Graesser and Clark's (1985) linguistically derived "general knowledge structures", and implementations of the KADS methodologies also provide general ontological frameworks directing the expert and knowledge engineer's model building.

4 Semi-formal elicitation and structuring through hypertext and hypermedia

The three techniques so far described focus on the elicitation, derivation and structuring of computational knowledge, that is, knowledge that can be operationalized within the computer as an active emulation of expertise. However, much knowledge is informal yet still valuable in an advisory expert system—text and pictures can encode and impart usable expertise, supplementing computational knowledge. Thus, the parallel development of hypertext and hypermedia is having a substantial impact on expert system architectures and knowledge acquisition tools.

Figure 6 shows some of the features of modern document processing systems that impinge on knowledge acquisition. Documents may be acquired from many sources, displayed, parts re-used in other documents, and parts linked for hypertext navigation. The text in documents may also be analyzed for associative cluster and a person may group these clusters to indicate significant concepts—such text analysis has been incorporated as a preliminary acquisition stage in repertory grid tools, for example the TEXAN module in KITTEN and KSS0.

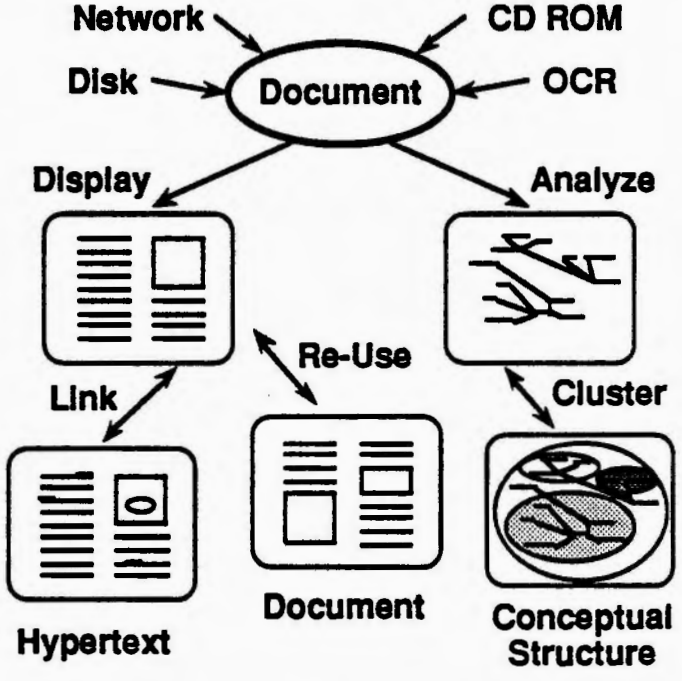


Fig.6 Hypermedia and text processing architecture

Hypertext systems have been coupled to knowledge acquisition tools to provide annotation of the distinctions made and cases described which can then be used to provide explanation facilities in the final performance system (Gaines, Rappaport & Shaw, 1989; Gaines & Linster, 1990). Apple's HyperCard system also provides a powerful programming language and user interface tools which has made it an attractive environment in which to build knowledge acquisition tools such as Cognosys and CAMEO. These then naturally incorporate many hypertext techniques. Other knowledge acquisition tools such as KEATS have been built around a hypertext environment specifically designed for knowledge acquisition.

There is also knowledge acquisition and linguistics research targeted on the direct transfer of knowledge expressed in text to structures of frames and rules (Hausser, 1989; Gomez, 1990). Since so much knowledge is already overtly encoded in text and diagrams, in the long term this will become an essential knowledge acquisition technology. However, it is still at an early stage of development that does not satisfy the "what we already know how to do well" criterion, and is not yet suitable for incorporation in integrated systems.

AN INTEGRATED SYSTEM ARCHITECTURE

The previous section has analyzed the current major knowledge acquisition techniques and tools within a common framework to show their similarities and differences with a view to designing an integrated system in which they operative effectively together. A companion paper analyzes the cognitive psychology and algebraic structure of knowledge representation to develop foundations for knowledge acquisition systems (Gaines & Shaw, 1990). Practice and theory join together in the knowledge acquisition architecture shown in Figure 7 which is an extension of Figure 14 in the other paper to incorporate the acquisition of informal knowledge from interviews, protocols and media as a stage ⓪.

Stage ① consists of using the informal knowledge to elicit the major coherent sub-domains that together encompass the significant phenomena in the overall domain. Stage ② consists of using the repertory grid methodology in each sub-domain to elicit relevant attributes and critical cases. Stage ③ consists of using the conceptual induction methodology to derive concepts and rules from the grids. Stage ④ consists of linking the sub-domains together, generally by specifying as a constraint in one domain that some role in that domain has as value an individual in another. Stage ⑤ consists of testing the overall knowledge base and iterating back through any of the earlier stages in order to develop and refine it further.

The architecture of Figure 7 is intended to be illustrative of the way the various knowledge acquisition techniques described naturally combine than a rigid framework for an integrated tool. For example, some sub-domain structures may be developed by direct editing, others by induction from databases, others by text analysis, and others by tools not yet defined or developed. The point is that that the techniques and tools that we have already come together naturally in this framework, and it also provides an open architecture for incorporating further techniques and tools. The companion paper also gives strong theoretical foundations for what is shown. In the longer term, the architecture shown may need major extension, but currently it seems to encompass what is available to us in knowledge acquisition.

Figure 8 combines and extends Figures 1, 3, 5 and 6 to show the modules necessary to support the architecture of Figure 7. Nearly all the modules shown have been described already. At the upper left, text and picture entry and analysis are shown as prompting the expert and developing a hypermedia base that also has links to the computational knowledge structures.

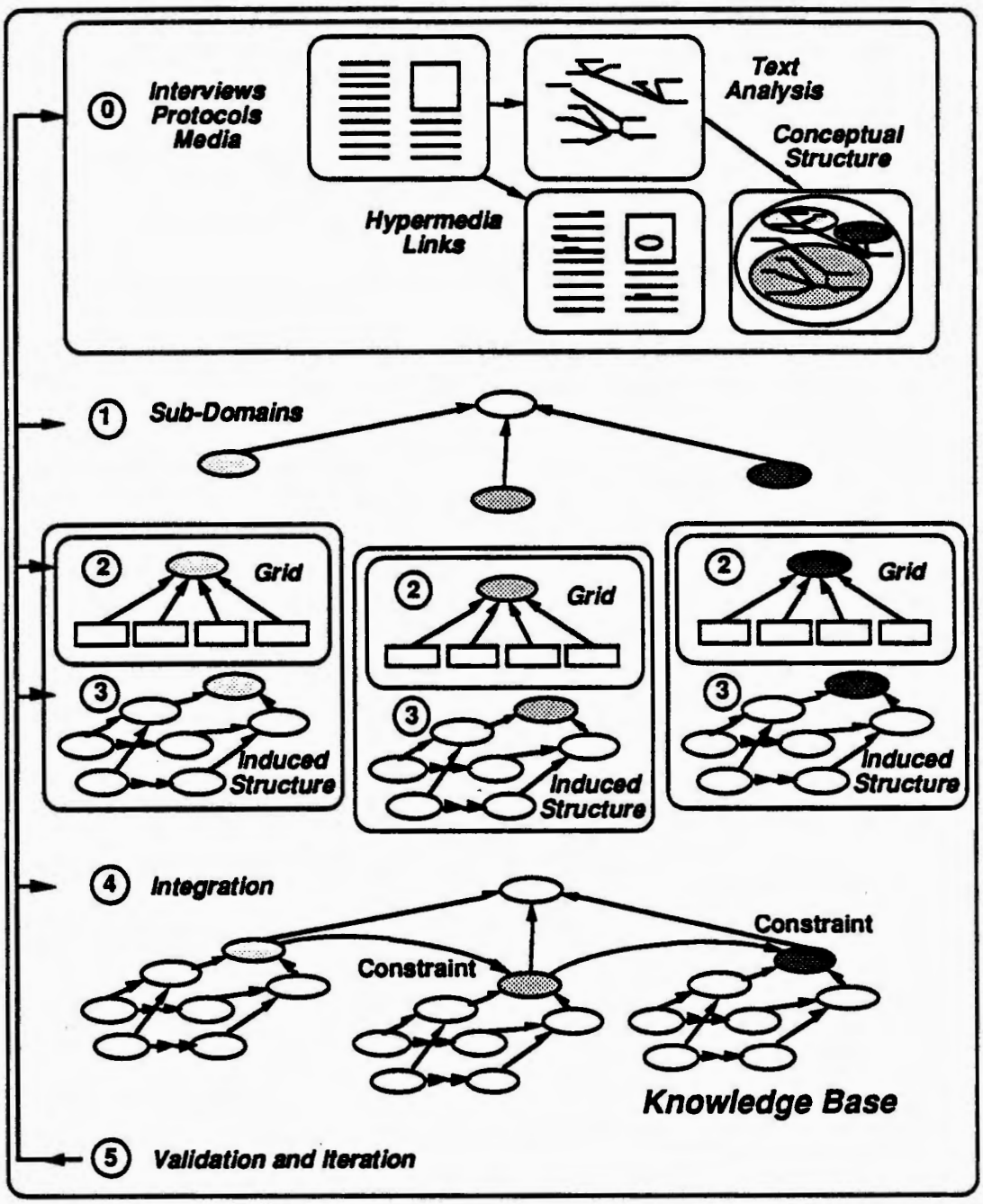


Fig.7 Integration of knowledge acquisition techniques

At the lower right of Figure 8 the knowledge base has been separated into a procedural one based on the rules and a declarative one based on the cases, and two additional inference paradigms have been added. The center one combines frame/rule inference with *case-based* inference, as, for example in PROTOS (Bareiss, 1989) The lower one provides *analogical* inference (Leishman, 1989) based on the cases in the declarative knowledge base, as, for example, in AQUINAS' performance module. These mixed paradigms based on both abstract knowledge and concrete cases are seen as the target inference modes. Note that the inference modules are now shown as an integrated part of the knowledge acquisition system for validation purposes, but that the need to export to specialist performance systems is still recognized.

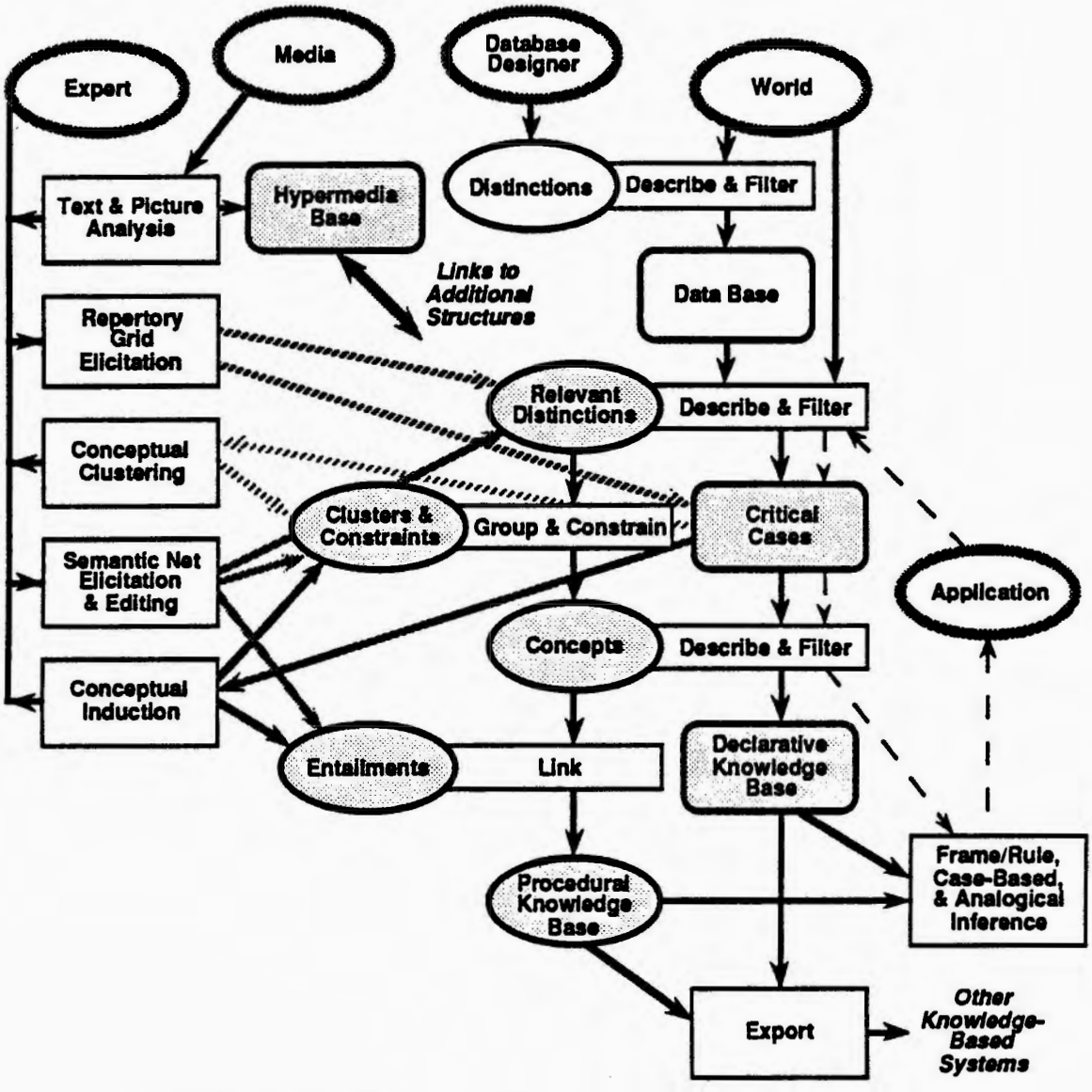


Fig.8 Integrated knowledge acquisition system architecture

Even though inference is integrated, at the bottom right of Figure 8 an export module is shown since transfer of knowledge bases to specialist performance systems is still seen as an important requirement. We need to integrate performance tools with knowledge acquisition tools for testing and validation, and for long-term support and updating. However, there will continue to be major requirements for stand-alone knowledge acquisition tools and stand-alone performance tools optimized for their primary tasks.

This presentation has emphasized that the methodologies, techniques and tools necessary to implement the integrated knowledge acquisition system architecture shown in Figure 8 already exist in the knowledge acquisition, machine learning, expert system, and hypertext communities. There is one basic technology underlying Figures 7 and 8 that is significant to those communities but has been sufficiently important and self-contained to be the focus of a major community in its own right, and that is knowledge representation. One can view Figure 8 as a set of related applications modules around a very comprehensive *knowledge representation module.* This is proposed as an implementation paradigm in the following sections.

IMPLEMENTATION—A KNOWLEDGE REPRESENTATION SERVER

In recent years there have been major advances in the the theory and practice of knowledge representation systems originating from semantic nets. In particular the series of knowledge representation schemas and tools commencing with KL-ONE (Brachman & Schmolze, 1985), developing through KRYPTON (Brachman, Gilbert & Levesque, 1985) and currently culminating in systems such as CLASSIC (Brachman, Borgida, McGuinness & Resnick, 1989; Borgida, Brachman, McGuinness & Resnick, 1989) and LOOM (MacGregor, 1988) has reached a maturity of technology which offers the promise of knowledge representation 'utilities' or 'services' in Levesque's (1984) terminology. The logical foundations of the subsumption relation, techniques for its correct and complete calculation, and the interaction between representation power and the tractability of subsumption algorithms have been widely studied (Brachman & Levesque, 1984; Nebel, 1988; Schmidt-Schauss, 1989) and are becoming reasonably well-defined.

It is now feasible to develop knowledge representation servers on a par with floating-point arithmetic units and numeric libraries, as software (and perhaps ultimately hardware) modules with well-defined functionality and fast, reliable performance. As with arithmetic units, such servers by no means solve all the problems of a particular application domain, but they do greatly reduce the burden of system development, allowing effort to be focused on the specifics of particular systems rather than on what should be general utilities.

In view of these advances in knowledge representation, and the central role played by knowledge representation in knowledge acquisition, it seems reasonable to implement the system shown in Figure 8 around a knowledge representation server satisfying the following requirements:

- A representation schema that is theoretically well-founded, adequate to support current knowledge-based system applications, and simply extensible to cope with future requirements.
- A knowledge base definition language mapping unambiguously into the representation schema, supporting data interchange between knowledge acquisition systems, expert system shells, databases, and other forms of knowledge support systems.
- Implementations that are light-weight in their space and time requirements, portable across platforms, cleanly structured so that functionality, data structures, and algorithms are visible and easily checked, and open architecture with well-defined interfaces to external modules.
- Offering a range of basic service modules such as parsers, graphers and network communications.

Figure 9 shows the structure of an integrated knowledge acquisition system implemented as a set of modules communicating with a knowledge representation server. It is designed to manage multiple knowledge bases and to interface to other servers either of the same or differing architectures. The various sub-modules attached to the server are options offering specific additional functionality. The modules are (reading clockwise from the top left):

- Interface modules to other formal knowledge bases and KR servers, including databases.
- A hypermedia module allowing informal knowledge structures in text and images to be captured, accessed and linked. The linkage structure itself is held as a knowledge base.
- A text analysis module allowing documents to be analyzed in terms of word usage, and associations between significant words to be graphed—based on TEXAN in KSS0

(Gaines, 1988; Gaines & Shaw, 1989). This enables protocols and technical documents to be used to initiate knowledge acquisition.

- A repertory grid expertise transfer module allowing graphic definition of concepts and graphic creation and editing of individuals—based on the elicitation screens of KSS0.
- A conceptual clustering module allowing interactive definition of new concepts—based on the hierarchical and spatial clustering from KSS0.
- A knowledge editing module allowing the interactive development and editing of knowledge structures through a visual language.
- A conceptual induction module creating rules about specified subsets of individuals and transforming them to a minimal set of concepts and default rules—based on the Induct algorithm (Gaines, 1989).
- A problem solving module supporting frame, rule and case-based inference from the knowledge structures.
- A grapher laying out specified parts of the concept subsumption graph, concept structures and individual structures—based on an incremental layout algorithm by Watanabe (1989) that can be used interactively to support the production of clear visual knowledge structures.
- A language interface accepting and generating definitions and assertions in formal knowledge representation languages, both textual and visual.

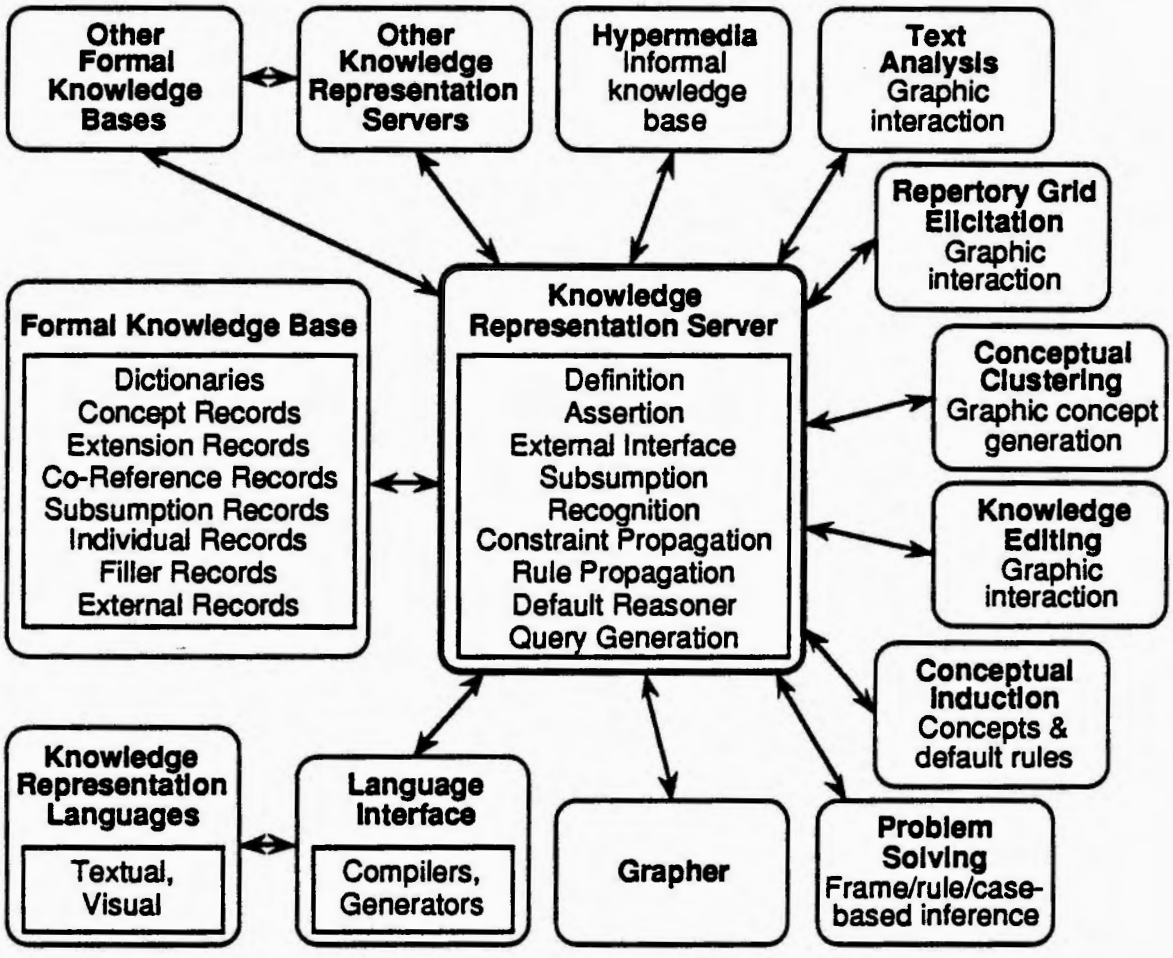


Fig.9 Knowledge representation server and associated modules

The data structures of the knowledge base and the functionality of the server are discussed in more detail below. The functionality of most of the modules on the right has already been described. The graphic interfaces and algorithms for eliciting, clustering, editing and induction modules already exist in tools such as KSS0 and AQUINAS—they have to be extended to a wider range of data types. The hypermedia informal knowledge base uses the server to maintain its linkage structures. The links to other servers and knowledge bases reflect the open architecture requirement. The problem solving module using frame, rule and case-based inference is intimately connected with knowledge processing functions within the server and is discussed below. The grapher is a general layout system to show the concept subsumption graph, concept and individual structures, and to provide interactive access to these structures. The language interface to knowledge representation languages reflect the knowledge base definition languages requirement.

There are a number of contenders for a knowledge representation server already within the knowledge acquisition community such as ONTOS (Monarch & Nirenburg, 1987) written in Lisp and CODE (Skuce, Shenkang & Beauville, 1989) written in Smalltalk. We have developed an implementation of a CLASSIC-like (Borgida, Brachman, McGuinness & Resnick, 1989) system in a dialect of C++—choosing the schema because it is the latest incarnation of KL-ONE, designed to be computationally tractable, and extremely well-defined—and choosing object-oriented C for speed, portability and the existence of very good class libraries. The purpose of the implementation is to investigate feasibility rather than to create a definitive server. The initial CLASSIC implementation took some six person-weeks and the extension of the schema to include correct default reasoning correctly tracking assertions and retractions about the same time again. Modern programming technology makes it straightforward to create open architecture knowledge representation servers in a range of languages running on a variety of platforms.

Figure 10 shows the objects used in the server implementation and their data structures. Our basic class library provides variable-length record objects which are sub-classed to provide dictionaries, sets and record objects for concept and individual records. The objects are:

- *Concept records* hold the constraint defining a concept: its extensions as the subsets of individuals that may be, and must be, included in any subset recognized; its cardinality and co-reference constraints on any subset recognized; its rule concept, if any, which will be asserted of any individual recognized; and, for each role constrained in the concept, the concept constraining it.
- *Individual records* hold the templet concept defining intensional constraints upon an individual and, for each role constrained in the templet or filled in the individual, a specification of the role fillers.
- *Filler records* hold subsets of individuals that fill slots.
- *Dictionaries* recognize concept, role and individual names and generate an accession number used to index other structures. Syntactically there is a clear separation between concepts, roles and individuals and a lexical item can occur in more than one dictionary with different meanings.
- *Extension records* hold subsets of individuals, typically those that may occur in a role filling set and those that must occur in it.
- *Data records* hold subsets of external individuals that are explicitly recognized by the server, notably integers, reals, dates and strings.
- *Co-reference records* hold equivalence subsets of role chains expressing co-reference constraints.
- *Subsumption records* hold computed subsumption relations on a three-valued basis, that one concept subsumes another, does not, or cannot.

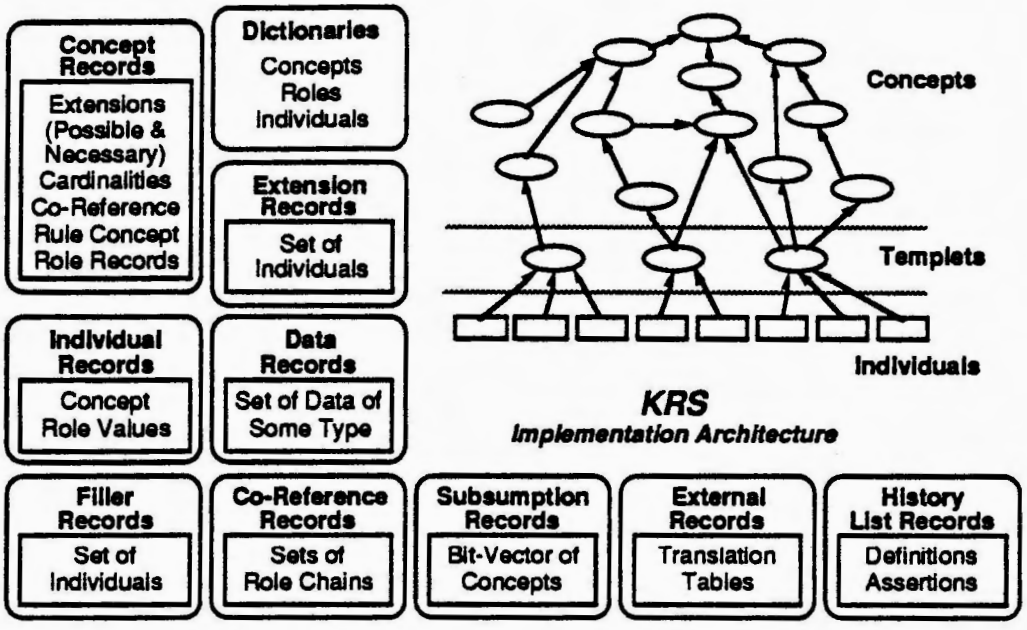


Fig.10 Knowledge representation server data structures

- *External records* hold tables translating between internal and external concepts and individuals.
- *History list records* keep track of explicit and implicit definitions and assertions in such a way that the knowledge base may be rolled back to any previous state.

A number of optimizations are applied in the implementation to minimize storage and maximize speed. For example, special codes are used for the empty, universal and singleton sets such that the set objects never store them, and subsumption is calculated in such a way that only a fast lookup is needed to check subsumption between role concepts rather than a recursive call. The object implementation allows these optimizations to be put in override methods in sub-classes so that the kernel implementation is small and easily checked but potentially slow, and the full implementation can be checked against the kernel for correctness and optimized for speed.

The server satisfies the requirements for knowledge representation for knowledge acquisition so far defined—the co-reference constraints go beyond them but are necessary to support analogical reasoning (Leishman, 1989), and, in particular, Russell's (1989) *determinations*. It provides a number of functions:

- A space- and time-efficient post-relational database for variable-length records of variable-length data whose types include: other records (relations); strings; integers; reals; dates and times; and sets of these.
- A constraint-specification system that allows intensional constraints on slot-fillers in records to be specified (that is, concept definitions).
- Automatic tracking of relations between systems of constraints—the *subsumption* relation of strict inheritance, and *disjoint* relation of attribute formation.
- Intensional constraints may be specified not only in general but also for individual records—this enables 'abstract objects' to be created which shadow concrete instances.
- Automatic propagation of intensional constraints on individual records as the individuals are specified as slot fillers in other records.
- Automatic tracking of whether an individual record satisfies the constraints defining a concept—this enables individual records to be 'recognized' as falling under a concept even if they have not been specified to do so.

- Sufficient conditions for an individual to be recognized by a concept may be associated with necessary conditions that then hold—this enables 'rules' to be specified.
- The failure of an individual to be recognized by a concept is classified as 'factually necessary' (due to slot fillers), or 'open' (further data could lead to recognition)—this allows principled default reasoning.
- The detailed basis for the failure of an individual to be recognized by a concept is available—this allows TEIRESIAS-like debugging of the knowledge base.
- A distinction between intensional constraints and assertions (facts)—the propagation of constraints from slots, or through rules, never changes individual records, only the abstract objects constraining them—this allows facts to be distinguished from inferences based on them and supports proper updating of inferences when assertions are retracted.

The algorithms providing these functions within the server are logically complete and correct—there are no heuristics. This is important in testing and validating the knowledge structures—inferences made with test cases can be traced precisely to the underlying conceptual structures. When exporting the knowledge base to shells that use heuristics such as 'first rule fired' or 'rule priorities' to simulate default reasoning the test cases must be re-validated.

A number of the modules shown in Figure 9 have already been implemented around the server. A recursive descent compiler provides a knowledge representation language similar to that of CLASSIC. A grapher has been implemented that supports interaction through popup menus, including calls to HyperCard. KSS0 has been interfaced to the server through an inter-application protocol so that its graphic elicitation, clustering and induction modules may be used to generate knowledge structures in the server. The problem-solving module currently fully supports inferences based on the procedural knowledge, the concepts and rules. We are investigating some interesting case-based decision-making models that use only operations such as specialization and generalization available within the server. It is interesting to discover how much can be achieved within the representation schema without adding other calculations and heuristics in external modules.

EDITING KNOWLEDGE STRUCTURES

KL-ONE-style knowledge structures are notoriously difficult to edit because they are based on the 'top-down' definition of general concepts and their stepwise refinement to more specific concepts. This is reasonable in well-established domains where the ontology has been developed as part of a professional discipline, but it is rarely a suitable basis for knowledge acquisition in novel domains. Experts often develop their knowledge structures 'bottom-up' from cases through commonality to general ontologies, and find a 'top-down' approach very unnatural. There are a number of ways of overcoming this problem—repertory grid, conceptual clustering and induction tools develop concept hierarchies 'bottom-up' from cases—but general concept editing services are difficult to define.

The problem is to design an interface allowing the person editing to adequately express his or her intentions. For example, if a concept is edited it is simple enough to maintain referential integrity by disallowing use of concepts defined after it in the definition, and even this constraint can be relaxed in a more intelligent editor that places a new or edited concept in an appropriate place in the referential order if one exists. However, it is difficult to determine whether the user intends, when editing, that subsumption relations be preserved and hence that edits should be propagated, and, if so, whether the user really wants full propagation or the option of checking what is occurring.

The BLIP knowledge acquisition tool developed at the Technical University of Berlin to support 'sloppy modeling' (Morik, 1987) provides a conceptual system development and editing tool that computes all consequences of definitions dynamically, including induction from cases. For the initial stages of knowledge acquisition even this may be too strong in that experts often generate inconsistent knowledge structures but are not ready to resolve the inconsistencies until they have a developed a fairly large-scale model giving them an overall perspective on the task and domain. The graphic knowledge editing tool shown on the center right of Figure 11 supports this relaxed approach. Knowledge structures can be developed and edited using drawing tools providing a natural visual language. The structures can extend across multiple screens supporting modularity and ontology 'libraries.' Structures can be freely duplicated and definitions may be fragmented across screens. The consistency of the definitions is determined only when they are compiled into a formal knowledge structure and inconsistencies are graphically highlighted.

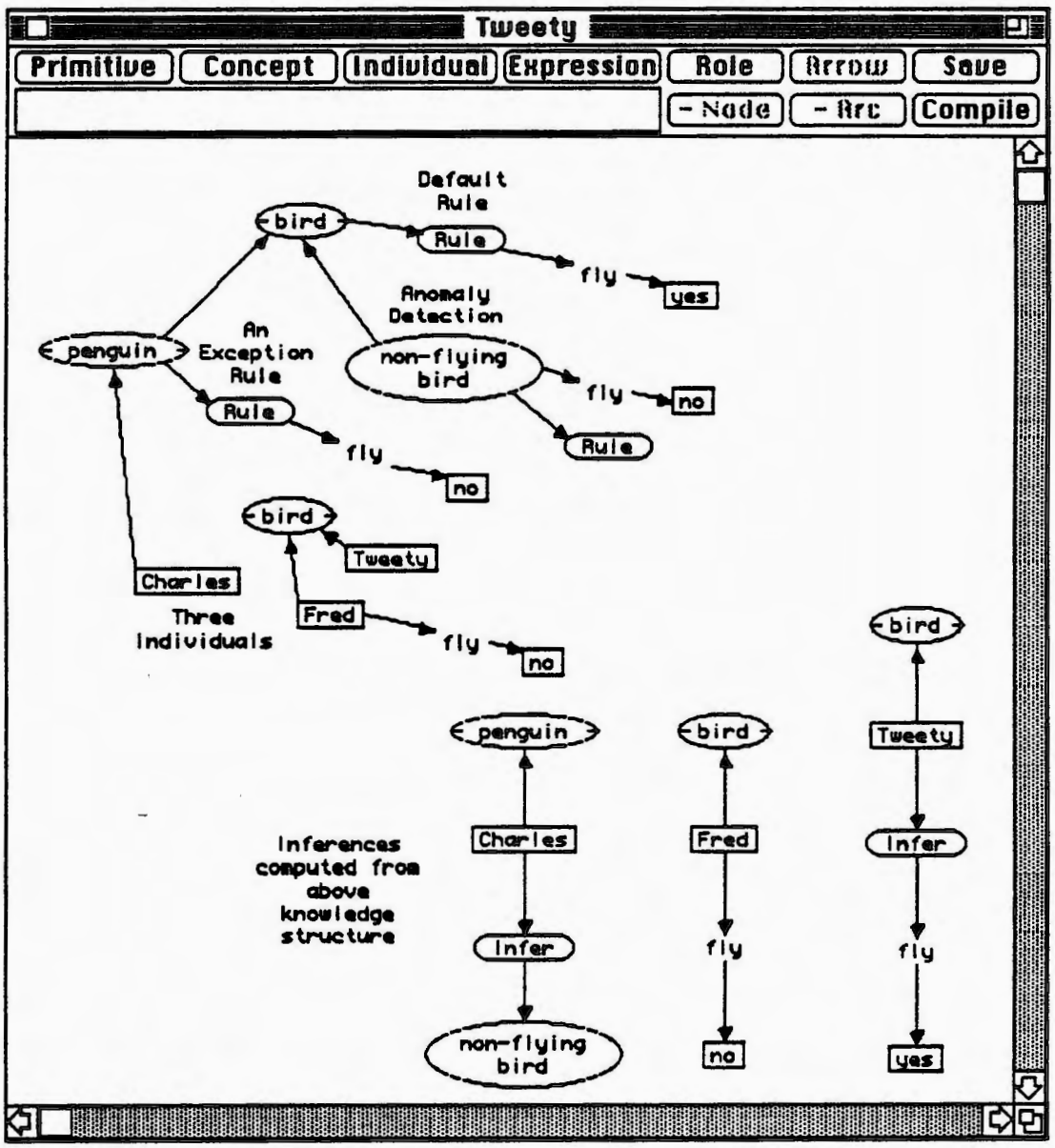


Fig.11 Graphic knowledge editor for the knowledge representation server

The knowledge structure at the top left of Figure 11 is an excessively classic one in artificial intelligence, but it still exemplifies significant features of knowledge acquisition and reasoning. A three rule default reasoning structure is shown in which birds by default can fly, but penguins are an exception, and birds that factually cannot fly are an anomaly. Three individual instances of birds are graphed at the middle left. When this structure is compiled for KRS it generates the knowledge base shown in Figure 12. When this is run for inferences and the individuals re-graphed the result is the output on the lower right.

The visual language used in Figure 11 is precisely defined. Concepts are ovals, individuals rectangles, roles (and annotation) unboxed text, and expressions rounded-corner boxes. The interpretation of the arrows is well-defined by the types of the objects at their head and tail. Arrows with open heads denote rules. Lines without arrows connecting concepts would denote that the concepts are disjoint. The compiler traces concepts and individuals through their outgoing arrows, builds a dependency table for concepts, highlighting circular links that it will not use, and generates concept definitions and individual assertions. Role objects are only noted when on paths and hence may also be used for annotation. The text analysis module at the top of Figure 9 generates such 'annotation' which may then be developed into a formal graphic specification by specifying objects as concepts and individuals.

```
Primitive(bird)
Concept(non-flying bird, bird, (All fly,  (Include no)))
Primitive(penguin, bird)
Rule(bird, (All fly,  (Include    yes)))
Rule(non-flying bird)
Rule(penguin, (All fly,  (Include no)))
Individual(Tweety, bird)
Individual(Fred, bird, (Fills fly, no))
Individual(Charles, penguin)
```

Fig.12 Visual knowledge base icompiled for knowledge representation server

The accompanying paper (Gaines & Shaw, 1990) gives a more substantial example in terms of the minimal five rule default reasoning solution of Cendrowska's (1987) contact lens prescription problem.

DEFAULT REASONING

One important application of the server in knowledge acquisition is to support the restructuring of rules inductively derived from elicited datasets as a minimal set of concepts and rules that can be presented to the expert in an easily understood form for checking and validation. This involves default reasoning since the best understood presentation is usually the most commonly occurring situations as the major rules with the rarer situations as exceptions. CLASSIC in its current definition supports rules but not default reasoning, and there are known problems to simply incorporating exceptions as rules triggered by concepts lower in the subsumption hierarchy (Ginsberg, 1987).

KRS fully supports default reasoning in that rules are checked by tracing the subsumption hierarchy and giving precedence to inferences made at lower levels. Thus, in the knowledge structure defined in Figures 11 and 12, the top level default rule is that an individual recognized as a 'bird' has 'fly' set to 'yes.' This is superseded by 'no if the individual is recognized by the subsumed concept, 'penguin.'

This reasoning is logically correct if all the facts about the individual are known. It only becomes default reasoning if the exception rule cannot be tested because some facts are unknown. KRS can indicate whether an inference based on partial information is plausible or exact because it computes a three-valued subsumption relation which allows the system to recognize when no further assertions could make an exception rule trigger. This is termed a 'release' and may be viewed as a rule which triggers when it is impossible for an individual to satisfy a concept except through retraction. Releases may be used to manage the frame problem by discriminating between plausible reasoning based on defaults and exact reasoning based on adequate information.

The reasoning algorithm in KRS is efficiently implemented and operates rapidly even with large knowledge bases; for example, the Garvan Institute of Medical Research thyroid diagnosis knowledge base, GARVAN-ES1, consists of 650 rules (Compton & Jansen, 1989). The 'ripple-down rules' version of this was moved to KRS and checked on the cornerstone case database of some 650 cases, taking less than one second a case on an Apple Mac IIci. Nebel's (1990) 'worst-case' example of the combinatorial explosion of anonymous concepts runs without problems—for example, his 20 concept example generates a total of 1,701 concepts and a total data structure of 460 Kbytes.

CONCLUSIONS

An architecture for knowledge acquisition systems has been proposed that is based upon the integration of existing methodologies, techniques and tools developed within the knowledge acquisition, machine learning, expert systems, hypermedia and knowledge representation research communities. Existing tools have been analyzed within a common framework to show that their integration can be achieved in a natural and principled fashion. A detailed architecture for integrated knowledge acquisition systems has been proposed that also derives from a parallel cognitive and theoretical studies.

An implementation of this architecture around a knowledge representation server has been described, and the server architecture has been detailed to the level of record structures and implementation experience.

As noted in the introduction, this paper presents a distillation of ideas from the whole community put forward as a contribution to a next generation of system designs. It now needs to be critically tested against other researchers' experiences and requirements.

ACKNOWLEDGEMENTS

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. I am grateful to many colleagues for discussions over the years that have influenced the research described in this paper, in particular John Boose, Jeff Bradshaw, Bill Clancey, Debbie Leishman, Marc Linster, Alain Rappaport, Doug Skuce, Brian Woodward and other colleagues at the Knowledge Acquisition Workshops world-wide. I am particularly grateful to Mildred Shaw for research collaboration that has shaped many of the ideas reported in this paper.

REFERENCES

Bareiss, R. (1989). **Exemplar-Based Knowledge Acquisition**. New York: Academic Press.

Boose, J.H. (1984). Personal construct theory and the transfer of human expertise. **Proceedings AAAI-84**, 27-33. California: American Association for Artificial Intelligence.

Boose, J H. (1989) A survey of knowledge acquisition techniques and tools. **Knowledge Acquisition 1** (1), 39-58 (March).

Boose, J.H. & Bradshaw, J.M. (1987). Expertise transfer and complex problems: Using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. **International Journal of Man-Machine Studies, 26,** 3-28.

Borgida, A., Brachman, R.J., McGuiness, D.L. & Resnick, L.A. (1989). CLASSIC: a structural data model for objects. Clifford, J., Lindsay, B. & Maier, D., Eds. **Proceedings of 1989 ACM SIGMOD International Conference on the Management of Data.** pp.58-67. New York: ACM Press.

Brachman, R. J., Borgida, A., McGuinness, D. L., and Resnick, L. A. (1989). The CLASSIC knowledge representation system, or, KL-ONE: The next generation. **Workshop on Formal Aspects of Semantic Networks.** Santa Catalina Island, California (February).

Brachman, R. J., Gilbert, V.P. & Levesque, H. J. (1985). An essential hybrid reasoning system: knowledge and symbol level accounts of KRYPTON. **Proceedings of IJCAI85.** pp.547-551. Los Altos: Morgan Kaufmann.

Brachman, R. J., & Levesque, H. J. (1984). The tractability of subsumption in frame-based description languages. **Proceedings of AAAI-84.** pp.34-37. San Mateo, California: Morgan Kaufmann.

Brachman, R.J. & Schmolze, J. (1985). An overview of the KL-ONE knowledge representation system. **Cognitive Science, 9**(2) 171-216 (April-June).

Breuker, J. & Wielinga, B. (1987) Use of models in the interpretation of verbal data. Kidd, A. (Ed) **Knowledge Elicitation for Expert Systems: A Practical Handbook.** Plenum.

Cendrowska, J. (1987) An algorithm for inducing modular rules. **International Journal of Man-Machine Studies 27** (4), 349-370 (October).

Compton, P. & Jansen, R. (1989) A philosophical basis for knowledge acquisition. **Proceedings of the Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems (EKAW'89)** pp.75-89. Paris (July).

Diederich, J., Ruhmann, I. & May, M. (1987). KRITON: A knowledge acquisition tool for expert systems. **International Journal of Man-Machine Studies, 26**(1), 29-40.

Eshelman, L., Ehret, D., McDermott, J. & Tan, M. (1987). MOLE: A tenacious knowledge acquisition tool. **International Journal of Man-Machine Studies, 26**(1), 41-54.

Gaines, B.R. (1988) Knowledge acquisition systems for rapid prototyping of expert systems. **INFOR, 26**(4), 256-285 (November).

Gaines, B.R. (1989) An Ounce of Knowledge is Worth a Ton of Data: Quantitative Studies of the Trade-Off between Expertise and Data based on Statistically Well-Founded Empirical Induction. **Proceedings of 6th International Workshop on Machine Learning,** pp.156-159. San Mateo, California: Morgan Kaufmann (June).

Gaines, B.R. (1987). Advanced expert system support environments. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of Second AAAI Workshop on Knowledge Acquisition for Knowledge-Based Systems.** pp.8-0-8-14. Banff (October).

Gaines, B.R. & Shaw, M.L.G. (1987). Knowledge support systems. **ACM MCC-University Research Symposium.** Austin, Texas: MCC. 47-66.

Gaines, B.R. & Shaw, M.L.G. (1989) Comparing the conceptual systems of experts. **Proceedings of the Eleventh International Joint Conference on Artificial Intelligence.** pp.633-638. Los Angeles, California: Morgan Kaufmann (August).

Gaines, B.R. & Shaw, M.L.G. (1990) Cognitive and logical foundations of knowledge acquisition. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fifth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** pp.9-1-9-24. Banff (November).

Ginsberg, M.L. (1987) **Readings in Non-Monotonic Reasoning.** San Mateo, California: Morgan Kaufmann.

Gomez, F. (1990) Knowledge acquisition from natural language for expert systems based on classification problem-solving methods. **Knowledge Acquisition 2** (2), to appear (June).

Graesser A.C. & Clark, L.F. (1985). **Structures and Procedures of Implicit Knowledge.** New Jersey: Ablex.

Hausser, R. (1987). **SComputation of Language.** Berlin: Springer.

Jones, W.P. (1989) Bringing corporate knowledge into focus with CAMEO. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fourth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** pp.17-1-17-23. Banff (October).

Klinker, G., Bentolila, J., Genetet, S., Grimes, M. & McDermott, J. (1987) KNACK—report-driven knowledge acquisition. **International Journal of Man-Machine Studies, 26**(1), 65-79.

Leishman, D. (1989). Analogy as a constrained partial correspondence over conceptual graphs. **Proceedings of KR'89: First International Conference on Principles of Knowledge Representation and Reasoning.** pp.223-234. San Mateo, California: Morgan Kaufmann (May).

Levesque, H. J. (1984). A logic of implicit and explicit belief. **Proceedings of AAAI-84.** pp.198-202. San Mateo, California: Morgan Kaufmann.

MacGregor, R.M. (1988). A deductive pattern matcher. **Proceedings of AAAI88.** pp.403-408. San Mateo: Morgan Kaufmann.

Marcus, S. (1987). Taking backtracking with a grain of SALT. **International Journal of Man-Machine Studies, 26**(4), 383-398.

Michalski, R.S. & Chilausky, R.L. (1980). Knowledge acquisition by encoding expert rules versus computer induction from examples—A case study involving soyabean pathology. **International Journal of Man-Machine Studies 12**, 63-87.

Morik, K. (1987) Sloppy modeling. Morik, K. Ed. **Knowledge Representation and Organization in Machine Learning.** pp.107-134. Berlin: Springer-Verlag.

Monarch, I. & Nirenburg, S. (1987) The role of ontology in concept acquisition for knowledge-based systems. **Proceedings of the First European Workshop on Knowledge Acquisition for Knowledge-Based Systems (EKAW'87).** Reading University, UK (September).

Motta, E., Eisenstadt, M., Pitman, K. & West, M. (1988) Support for knowledge acquisition in the knowledge engineer's assistant (KEATS). **Expert Systems 5**(1), 6-28.

Nebel, B. (1988). Computational complexity of terminological reasoning in BACK. **Artificial Intelligence, 34** 371-383.

Nebel, B. (1990) **Reasoning and Revision in Hybrid Representation Systems.** Berlin: Springer-Verlag.

Quinlan, J.R. (1987) Simplifying decision trees. **International Journal of Man-Machine Studies 27** (3), 221-234 (September).

Russell, S.J. (1989). **The Use of Knowledge in Analogy and Induction.** London: Pitman.

Schmidt-Schauss, M. (1989). Subsumption in KL-ONE is undecidable **Proceedings of KR'89: First International Conference on Principles of Knowledge Representation and Reasoning.** pp.421-431. San Mateo, California: Morgan Kaufmann (May).

Shaw, M.L.G. & Gaines, B.R. (1983). A computer aid to knowledge engineering. **Proceedings of British Computer Society Conference on Expert Systems,** 263-271 (December). Cambridge.

Shaw, M.L.G. & Gaines, B.R. (1987). KITTEN: Knowledge Initiation & Transfer Tools for Experts & Novices. **International Journal of Man-Machine Studies, 27**, 251-280.

Shaw, M.L.G. & Woodward, J.B. (1989) Mental models in the knowledge acquisition process. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fourth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** pp.29-1-29-24. Banff (October).

Skuce, D., Shenkang, W. & Beauville, Y. (1989) A generic Knowledge Acquisition Environment for conceptual and ontological analysis. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fourth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** pp.31-1-31-20. Banff (October).

Watanabe, H. (1989). Heuristic graph displayer for G-BASE. **International Journal of Man-Machine Studies, 30**(3) 287-302 (March).

Wetter, T. & Woodward, J.B. (1989) Towards a theoretical framework for knowledge acquisition. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fifth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** submitted. Banff (November).

Woodward, B. (1990) Knowledge engineering at the front-end: defining the domain. **Knowledge Acquisition 2**(1), 73-94 (March).