# DEVELOPMENT OF SECOND GENERATION KNOWLEDGE ACQUISITION SYSTEMS

Brian Gaines

Knowledge Science Institute
University of Calgary
2500 University Drive, NW
Calgary, Canada T2N 1N4

Marc Linster

Expert System Research Group
GMD
PO Box 1240
D-5205 St. Augustin 1, FRG

Design and implementation studies are presented of the development of second generation knowledge acquisition systems through the heterogeneous integration of existing tools using inter-application protocols. The overall aim of the studies is to explore the possibility of creating highly integrated *knowledge support systems* through the loose coupling of software tools developed independently at unrelated sites for different purposes. Three tools were integrated to form a combined knowledge acquisition and performance system. The hypermedia tool *HyperCard* was used as a general-purpose knowledge acquisition tool for unstructured material in the form of text and diagrams. The knowledge acquisition tool *KSS0* was used to elicit knowledge more formally and structure it as a computable knowledge base. The expert system shell *BABYLON* was used to represent the elicited knowledge and perform inferences with it to generate advice in a performance situation. Strong synergy was created between the tools in that the annotation and explanation captured in the hypermedium system were available as context-sensitive help to the user of the expert system, and the expert system was used to validate the knowledge base generated by the knowledge acquisition tool and feed back anomalous cases as additional data for induction. An important feature of the overall system is that it is complete in its own right yet retains programmability allowing end user communities to customize it for particular requirements and domains.

## INTRODUCTION

Second generation knowledge acquisition systems targeted on applications in the 1990s have to [7]:

- Integrate a wide variety of weak methods encompassing the diversity of forms and sources of knowledge
- Integrate conflicting sources of knowledge to cope with essential contradictions that cannot be replaced by an artificial and misleading 'consensus'
- Integrate knowledge acquisition with system operation to facilitate a heavy activity of continuing system maintenance and upgrade as knowledge changes
- Integrate knowledge representation in the acquisition and performance systems, retaining the flexibility for new representations to be used that more accurately carry the knowledge

Thus, second generation developments have two objectives that may come into conflict:

- To integrate closely a variety of techniques and technologies to cope with the diversity of knowledge
- To provide flexibility within the integration framework for new techniques and technologies to be incorporated to cope with the dynamics of knowledge

It seems unlikely that monolithic approaches to the development of second generation knowledge acquisition systems will be successful or that the results will be sustainable. Knowledge based system technology is developing very rapidly and involves a very wide range of computational and psychological approaches [2, 10]. This paper reports preliminary studies of building systems through the heterogeneous integration of different tools. The overall aim of the studies is to explore the possibility of creating highly integrated *knowledge support systems* through the loose coupling of software tools developed independently at unrelated sites for different purposes. The criterion for high integration is that the systems should operate as a single knowledge-based system environment from a user perspective. The underlying rationale of this research is that large-scale system development can be expedited through the integration of heterogeneous systems, each representing the best available technology in a particular application area [11].

## SYSTEM ARCHITECTURE

"Knowledge" in the artificial intelligence literature has become associated with formal structures from which inferences can be made, and knowledge acquisition and expert system tools reflect this. However, much human knowledge is not well-structured for formal representation and computer processing [6]. In the 1980s computer-based systems have been developed that are targeted on the representation of very loosely structured knowledge with less emphasis on formal structures from which inferences can be made. *Hypertext* and *hypermedia* systems approach knowledge from a different direction than that of expert systems, accepting it as having limited structure and processibility, and adding a limited structure through links that are intended to aid the user rather than allow computer-based inferencing [5, 13]. They extend knowledge representation to include any media whereby human knowledge can be transferred even if there is no means for analyzing that knowledge or its transfer.

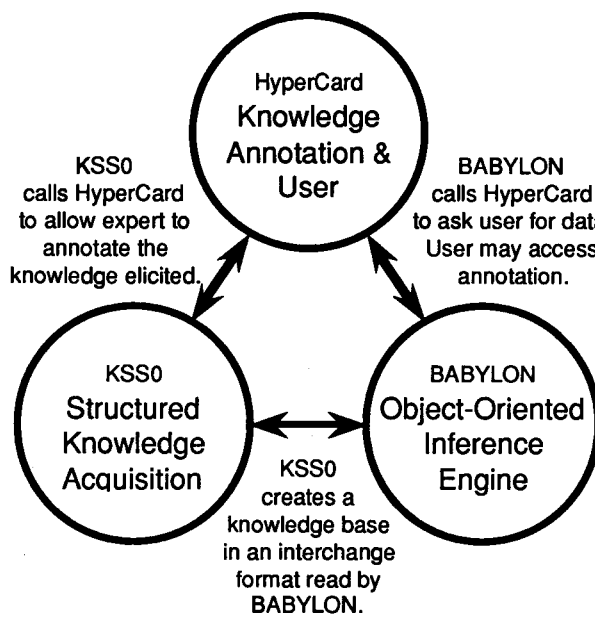


**Fig.1 Prototype knowledge support system design**

Our overall system design called for a hypermedium system to play a central role in providing for both the acquisition of informal knowledge and a flexible user interface. Figure 1 shows our initial system concept with three components:

- The hypermedia tool *HyperCard* to be used as a general-purpose knowledge acquisition tool for unstructured material in the form of text and diagrams.
- The knowledge acquisition tool, *KSS0*, to be used to elicit knowledge more formally and structure it as a computable knowledge base.
- The expert system shell, *BABYLON*, to be used to represent the elicited knowledge and perform inferences with it to generate advice in a performance situation.

It was important that the three components operate together to present the user with the appearance of a single system, and that they mutually support one another's operation. For example, the hypermedium system should provide annotation of the formal knowledge structures and the inference engine should provide validation of elicited knowledge.

The systems being integrated each have complex functionality coupled through heterogeneous integration in their own right. Figure 2 shows the overall architecture of BABYLON [4, 19] which represents knowledge in six ways, as: lisp code; free text; objects or frames; production rules; Horn clauses with the full power of prolog; and as a system of constraints [14, 15].

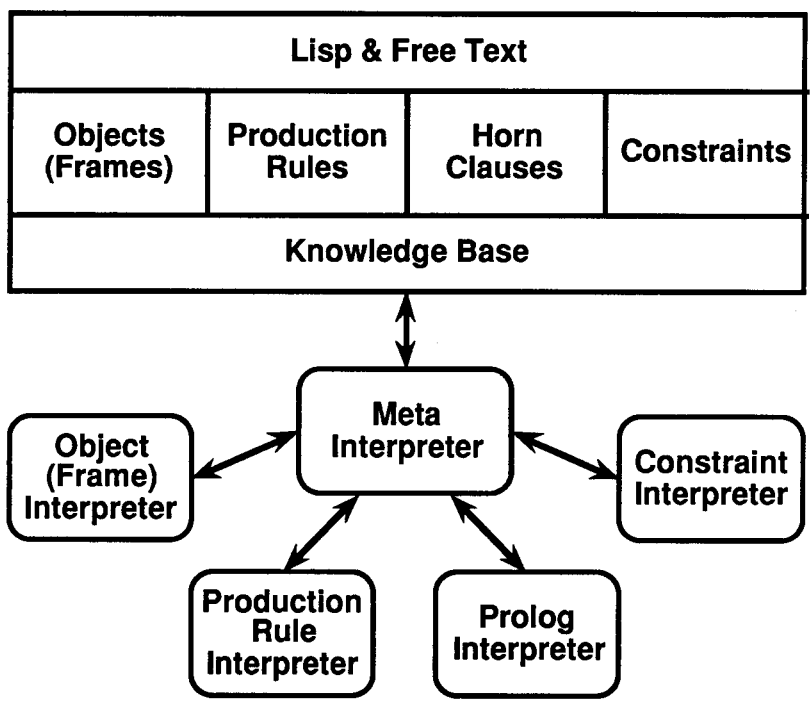


**Fig.2 The overall architecture of BABYLON**

The different representations are coupled through a meta interpreter providing a blackboard architecture through which the object, rule, prolog and constraint interpreters interact. Programmers see a uniformly available system within which they may freely mix paradigms. Table 1 shows the primary interactions possible between knowledge represented in the different forms available within BABYLON.

| Access from to | Lisp | Objects | Rules | Prolog | Constraints |
|---|---|---|---|---|---|
| **Lisp** | Direct | As slot value, in behaviors | As condition, as action | Lisp-expression as premise | Lisp-expression as value specification, activation condition, in patterns |
| **Objects** | By using messages to objects, functions for objects | Direct | Slot and behavior references serving as condition and action | Slot and behavior reference, meta predicate | Object reference as variable, value specification, activation condition, in patterns |
| **Rules** | Evaluation of rule sets | Not possible | Direct | Meta predicate | Not possible |
| **Prolog** | Proving of hypotheses | Not possible | In conditions and actions | Direct | Prolog expression as activation condition |
| **Constraints** | Definition and satisfaction of constraints | By using messages | Satisfied p-expression as premise | Satisfied p-expression as premise or in candidate | Direct |
| **Free Text** | Not possible | Not possible | As condition and action | Meta predicate | As activation condition and in patterns |

**Table 1 Interactions between knowledge in different paradigms in BABYLON**

KSSO [8] is an integration knowledge acquisition system based on an extendible set of techniques encompassing elicitation tools, visualization tools, group comparison tools, inductive tools, and export tools as shown in Figure 3.

The elicitation tools have their foundations in the personal construct psychology of Kelly [17], subsequent work on the computer elicitation of conceptual structures through repertory grids [20], and applications of these techniques to knowledge elicitation [1, 18, 21]. The objective is to gather domain knowledge from prototypical entities represented along certain dimensions:

- *Elicit* accepts specifications of entities within a sub-domain and provides an interactive graphical elicitation environment within which the experts can distinguish entities to derive their attributes. The resultant class is continuously analyzed to provide feedback prompting the expert to enter further entities and attributes.

The visualization tools consist of an interactive interface to represent the abstractions derived from those entities in terms of hierarchical clusters using Shaw's FOCUS algorithm [20], and relational diagrams such as a non-hierarchical conceptual maps derived through principal components analysis [22]. The objectives are to validate the raw domain knowledge and suggest further structure at a higher level:

- *FOCUS* hierarchically clusters entities and attributes within a sub-domain prompting the experts to add higher-level entities structuring the domain.
- *PrinCom* spatially clusters entities and attributes within a sub-domain prompting the experts to add higher-level entities structuring the domain.

The group comparison tools consist of an interactive interface to represent the relations between the terminologies and conceptual systems of different experts, or experts and clients. The objectives are to determine the consensus, conflict, correspondence and contrast between different conceptual systems [12]:

- *Socio* compares the structures for the same sub-domain generated by different experts, or the same expert at different times or from varying perspectives.

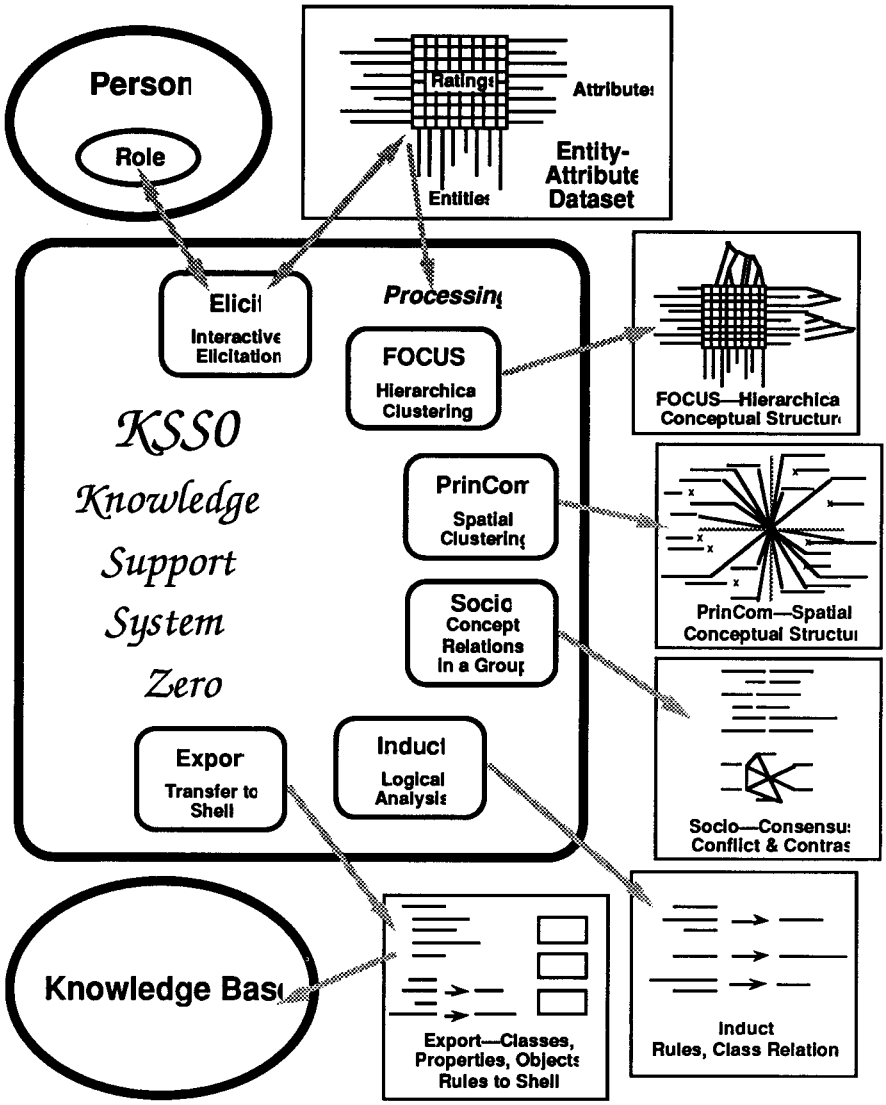


**Fig.3 The overall architecture of KSS0**

The inductive tools support the derivation of constraints within the conceptual structures through empirical induction analyses [3, 9]:

- *Induct* induces logical entailments enabling the attributes of an entity, or the evaluations of a decision-making situation in a domain, to be derived from other attributes.

The export tools support the transformation of the knowledge elicited and analyzed by tools into a formalism understandable by the shell representation and inference mechanisms:

- *Export* formats the specifications of sub-domains as classes, of entities as objects, of attributes as properties, and of entailments as methods, and transfers them to the performance tool.

## KNOWLEDGE BASE EXCHANGE

Detailed examples of the individual operation of KSS0, BABYLON and HyperCard will not be given in this paper as the focus is on the integration rather than the individual programs. One requirement for integration is that KSS0 and BABYLON be able to share a knowledge base. The basic dataset developed in KSS0 to represent an expert's conceptual structure and inference procedures consists of four parts:

a) Domain Definition: the domain, the context, the interview, the term for entities in the domain, and the term for attributes of these entities, plus the original source of these definitions if they are not original to the current dataset.

b) Attribute Definitions: the name, type and possible values of attributes, plus information about their use as inputs and outputs, and the order in which they should be requested in consultations.

c) Entity Definitions: the name and values on the defined attributes of the entities.

d) Implication Definitions: The name, test conditions and inferences of the implications.

The domain and attribute definitions are used to define a frame with the attributes as slots, together with a general instance of it in BABYLON. The entity definitions are used to define specific instances of the frame. The implication definitions are used to define rule packages in BABYLON, one for each slot whose value might be inferred.

Figure 4 shows a KSS0 dataset for Cendrowska's contact lens problem [3] in graphical form and Figure 5 shows implications derived from this dataset by Induct.

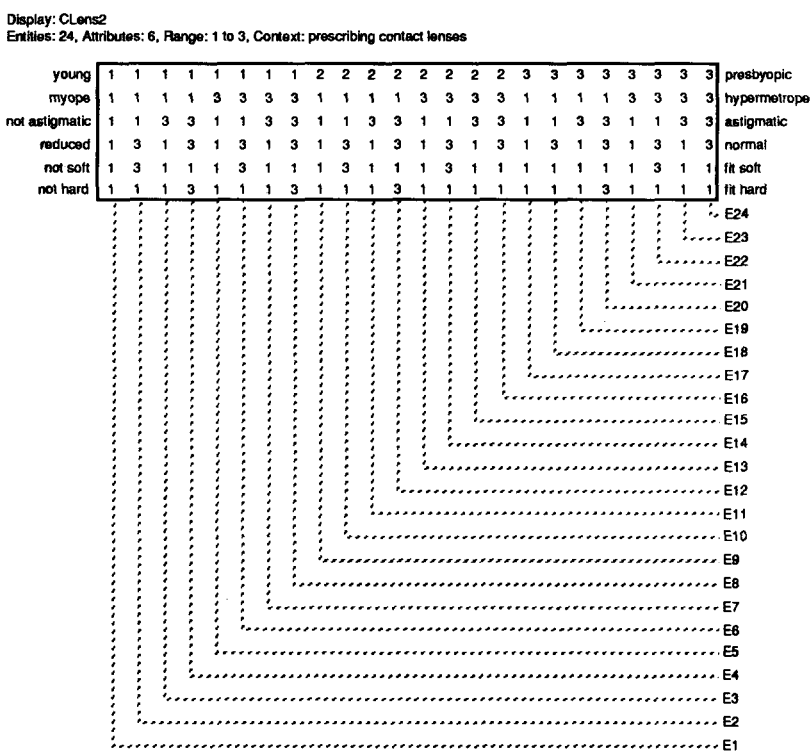


**Fig.4 KSS0 dataset**

astigmatism=astigmatic -> soft contact lenses=not soft 100% 6.06%
tear production=reduced -> soft contact lenses=not soft 100% 6.06%
age=presbyopic & prescription=myope -> soft contact lenses=not soft 100% 39.3%

prescription=hypermetrope & astigmatism=not astigmatic & tear production=normal -> soft contact lenses=fit soft 100% 0.904%
age=young & astigmatism=not astigmatic & tear production=normal -> soft contact lenses=fit soft 100% 4.34%
age=pre-presbyopic & astigmatism=not astigmatic & tear production=normal -> soft contact lenses=fit soft 100% 4.34%

astigmatism=not astigmatic -> hard contact lenses=not hard 100% 11.2%
tear production=reduced -> hard contact lenses=not hard 100% 11.2%
age=pre-presbyopic & prescription=hypermetrope -> hard contact lenses=not hard 100% 48.2%
age=presbyopic & prescription=hypermetrope -> hard contact lenses=not hard 100% 48.2%

prescription=myope & astigmatism=astigmatic & tear production=normal -> hard contact lenses=fit hard 100% 0.463%
age=young & astigmatism=astigmatic & tear production=normal -> hard contact lenses=fit hard 100% 2.78%

**Fig.5 Implications derived from KSS0 dataset by Induct**

Figure 6 shows part of this dataset and its implications exported from KSS0. Figure 7 shows part of the dataset imported into BABYLON. Transforming the KSS0 knowledge base to the BABYLON knowledge base is automatic and involves no post-editing.

```
(KB CLens2
 (DOMAIN contact_lenses "prescribing contact lenses")

 (ATTS
  (age (CAT young pre_presbyopic presbyopic) (IO I) (WGTS 99))
  (prescription (CAT myope hypermetrope) (IO I) (WGTS 99))
  (astigmatism (CAT not_astigmatic astigmatic) (IO I) (WGTS 99))
  (tear_production (CAT reduced normal) (IO I) (WGTS 98))
  (soft_contact_lenses (CAT not_soft fit_soft) (IO O) (WGTS 99))
  (hard_contact_lenses (CAT not_hard fit_hard) (IO O) (WGTS 99)))

 (ENTS
  (E1 (ATTS young myope not_astigmatic reduced not_soft not_hard))
  (E2 (ATTS young myope not_astigmatic normal fit_soft not_hard)))

 (IMPS
  (Ind_1 (PROB 100) (ATTS ? ? astigmatic ? (not_soft) ?))
  (Ind_2 (PROB 100) (ATTS ? ? ? reduced (not_soft) ?))))
```

**Fig.6 Partial export format for dataset from KSS0 to BABYLON**

```
( Defframe People
  ( Slots
    ( Hard_Contact_Lenses - :Possible-Values
      ( :One-Of Not_Hard Fit_Hard) :Io O :Wgts 99)
    ( Soft_Contact_Lenses - :Possible-Values
      ( :One-Of Not_Soft Fit_Soft) :Io O :Wgts 99)
    ( Tear_Production - :Possible-Values
      ( :One-Of Reduced Normal) :Io I :Wgts 98)
    ( Astigmatism - :Possible-Values
      ( :One-Of Not_Astigmatic Astigmatic) :Io I :Wgts 99)
    ( Prescription - :Possible-Values
      ( :One-Of Myope Hypermetrope) :Io I :Wgts 99)
    ( Age - :Possible-Values
      ( :One-Of Young Pre_Presbyopic Presbyopic) :Io I :Wgts 99)))

( Definstance Person Of People)

( Defrule-Set :Conclude_Hard_Contact_Lenses
  ( Ind_7
    ( ?And
      ( Person Astigmatism = Not_Astigmatic))
    ( $Execute
      ( Person Hard_Contact_Lenses = Not_Hard)))

  ( Ind_6
    ( ?And
      ( Person Tear_Production = Normal)
      ( Person Astigmatism = Not_Astigmatic)
      ( Person Age = Pre_Presbyopic))
    ( $Execute
      ( Person Soft_Contact_Lenses = Fit_Soft))))
```

**Fig.7 Partial representation of dataset in BABYLON**

The BABYLON knowledge bases as shown in Figure 7 implement a heuristic association problem-solving model [16], i.e. the values of output attributes are heuristically defined on the basis of input attributes using the implication rules defined by *Induct*.

For each output attribute a rule-set is defined. A control structure fires the rule sets if their corresponding output attributes are still undefined. The firing sequence for the rules is defined by the weights of the attributes.

The operational knowledge bases are generated automatically and can be used as an expert advisory system having exactly the behavior required by Cendrowska in her original description of the dataset. In particular, BABYLON does not request information about the time-consuming measurement of "tear production" unnecessarily.

## INTER-APPLICATION COMMUNICATION

The Macintosh currently has no inter-program communication protocol and it was necessary to write code that would enable KSS0, BABYLON and HyperCard to communicate. The requirements are:

a) that KSS0 and BABYLON be able to issue a function call sending a structured message to HyperCard, and making HyperCard the active program running normally for as long as the user requires.

b) that HyperCard eventually returns to its caller, sending it a structured message in reply, and reactivating the calling program continuing as if it had returned from a function call.

c) that the three programs are able to interact asynchronously without loss of control.

An inter-application protocol was written taking commands of the form:

```
Procedure ipccall(Var toProg,fromProg,indat,outdat: Chars);
```

which accepts strings as arguments and can be called by a foreign function call from Pascal, Lisp and HyperCard. This procedure activates the program "toProg", sending it the data "indat" and asking it to return to the program "fromProg" with a reply in "outdat".

The structure of the data in "indat" and "outdat" is specific to the use of the inter-program communication. For example, to use HyperCard as a user interface for data queries from BABYLON the following structure for "indat" was defined:

Query-1,<knowledge-base>,<frame>,<instance>,<slot>,<desired-value>,<possible-values>

This information is used in HyperCard to construct a data request card as shown in Figure 8. The value on which the user clicks is returned to BABYLON in "outdat". When the consultation is complete, BABYLON may send the inferences made to HyperCard for display, or may show them within BABYLON in the usual fashion as shown in Figure 9.

The interface conforms to the normal non-modal operation of the Macintosh, and the client may browse within HyperCard, or switch to other programs using MultiFinder. In this way it is possible for the client to bring the BABYLON screen to the front without there being any return message from HyperCard.

BABYLON operates modally in a consultation and hence its screens will be inactive under these conditions. The ipccall protocol checks for any key depression and returns control to BABYLON with a null message returned. The BABYLON interface detects this message and defaults to the normal BABYLON consultation dialog.

Alternatively the user may bring the HyperCard screen to the front again and continue the dialog through HyperCard. Thus, the interface allows the maximum user freedom and never locks him or her out.

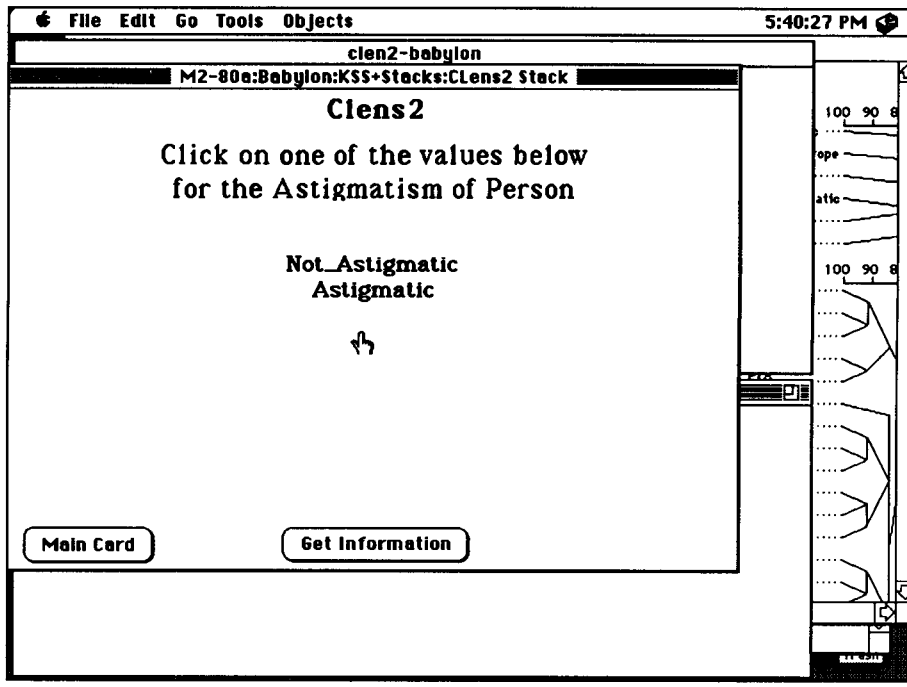


**Fig.8 HyperCard used from BABYLON to request data from client**

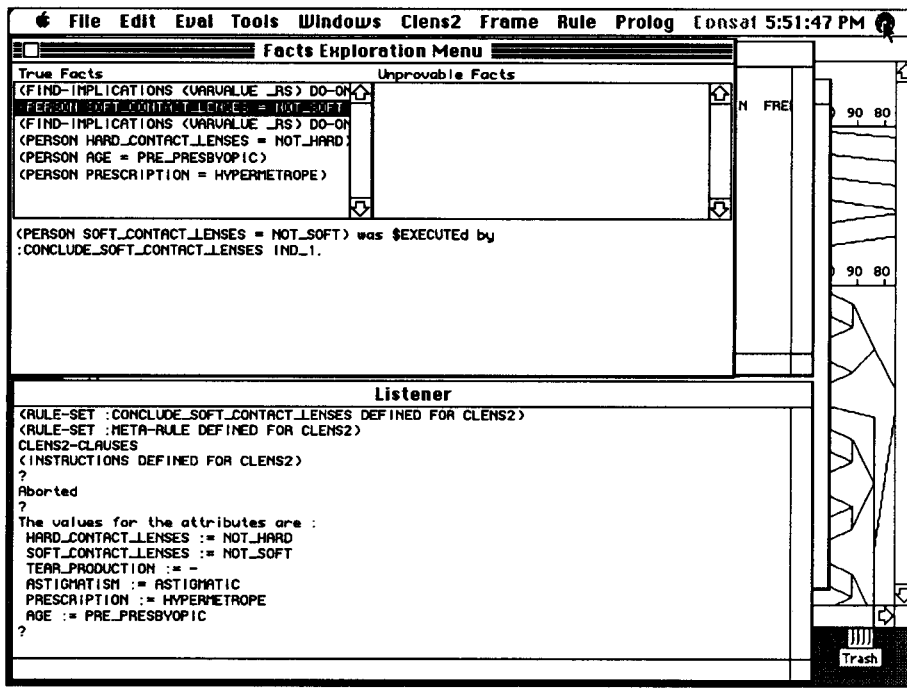


**Fig.9 Results of consultation being traced in BABYLON**

## PROTOTYPE KNOWLEDGE SUPPORT SYSTEM

KSS0 already supports interaction with HyperCard such that during knowledge acquisition an expert can transfer to HyperCard and annotate any of the entities or attribute values entered with textual notes and graphics. The stack supporting this feature automatically generates new annotation cards for each new entity and attribute value. The stack can also be used as the primary controller of KSS0, starting KSS0 if it not already running, loading datasets and sending KSS0 commands to analyze them.

The availability of the three links, HyperCard-KSS0 communication, HyperCard-BABYLON communication, and KSS0-BABYLON knowledge exchange, made it feasible to construct a prototype knowledge support system having the structure shown in Figure 1. A more detailed system design corresponding to our implementation is shown in Figure 10.

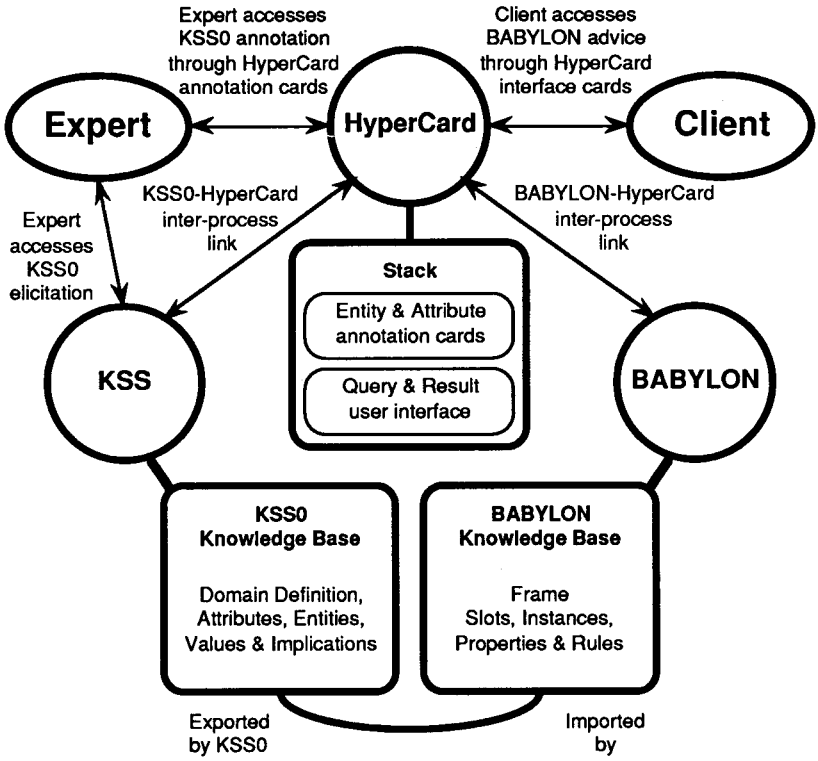


**Fig.10 Knowledge support system developed**

A generic HyperCard stack was developed with three basic cards:

1 An initial control card as shown in Figure 11 holding the dataset name and giving access to KSS0.

2 An annotation card as shown in Figure 12 used to hold notes on entities and attribute values elicited from the expert.

3 A BABYLON data request card as shown in Figure 8 used as a query interface to BABYLON during a consultation.

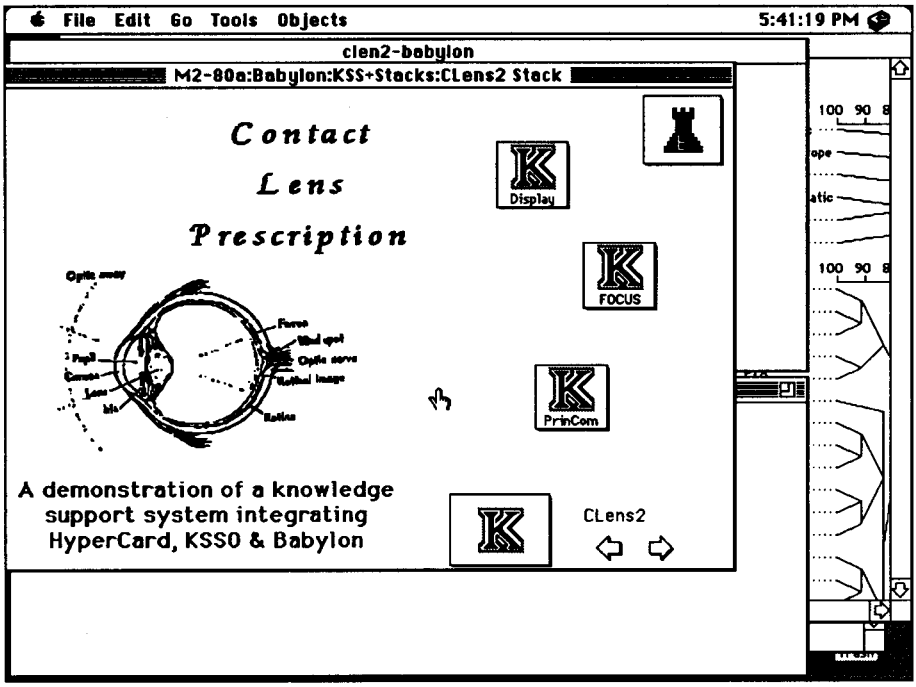


Fig.11 Initial control card

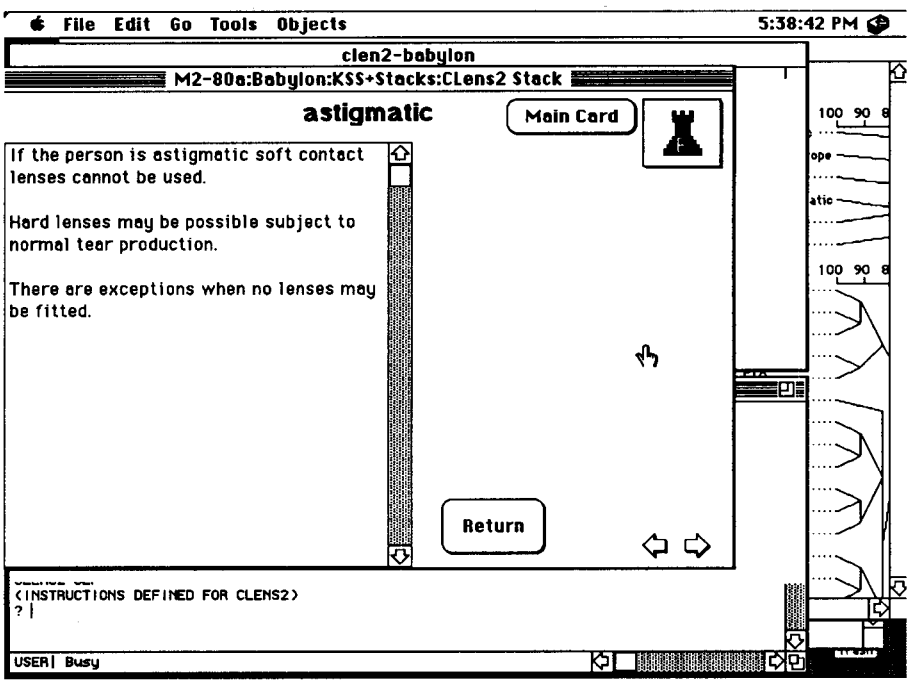


Fig.12 Annotation card

A HyperCard script was written that supports communication with both KSS0 and BABYLON. In particular it supports the "Main Card" and "Get Information" buttons shown in Figure 8. When "Main Card" is clicked control is transferred to the card shown in Figure 11. When "Get Information" is clicked the card shown in Figure 8 changes to the information request card shown in Figure 13. Clicking on one of the values now transfers control to the associated annotation card such as that shown in Figure 12. Clicking on the "Return" button returns to the BABYLON data request card (if this card had been accessed from KSS0 the "Return" button would return control to KSS0.

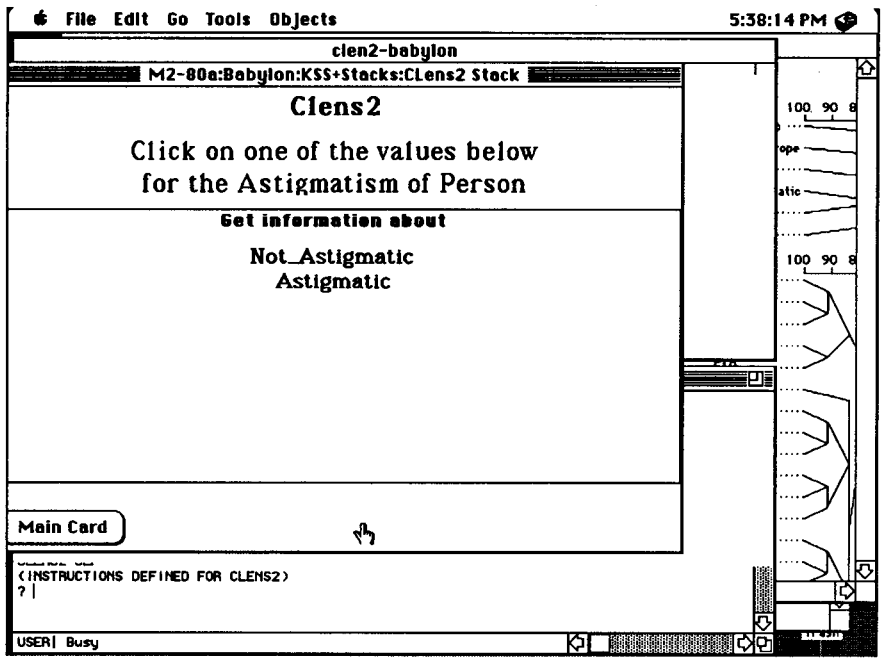


**Fig.13 Information request card**

Communication with KSS0 operates in a similar fashion. Clicking on one of the KSS0 buttons on the initial control card of Figure 11 starts KSS0 if it is not already running, loads the specified dataset into KSS0, and runs the requested analysis program if specified. Figure 14 shows a KSS0 entity match screen operating in conjunction with HyperCard and BABYLON. During knowledge acquisition with KSS0, if HyperCard is running, KSS0 makes transfer to HyperCard an option in its pop-up menus as shown in Figure 15. The availability of a pop-up menu is shown by the cursor changing to a menu symbol as it is moved over entity names and attribute value names on the screen. Depressing the mouse then brings up the pop-up menu allowing one of a number of actions to be taken, including transfer to HyperCard.

If the mouse were released over the attribute value "astigmatic" and the menu option "HyperCard" as shown in Figure 15 then control would be transferred to the HyperCard annotation card shown in Figure 12. The expert can then annotate the intended use of the term "astigmatic" with text and diagrams, and return control to the KSS0 screen of Figure 15 by clicking "Return". The expert can also browse the HyperCard stack at will, reading and modifying other annotation, before returning control to KSS0. All KSS0 screens support the pop-up menu communication with HyperCard giving access to the annotation from, for example, the PrinCom conceptual cluster screen as shown in Figure 16.

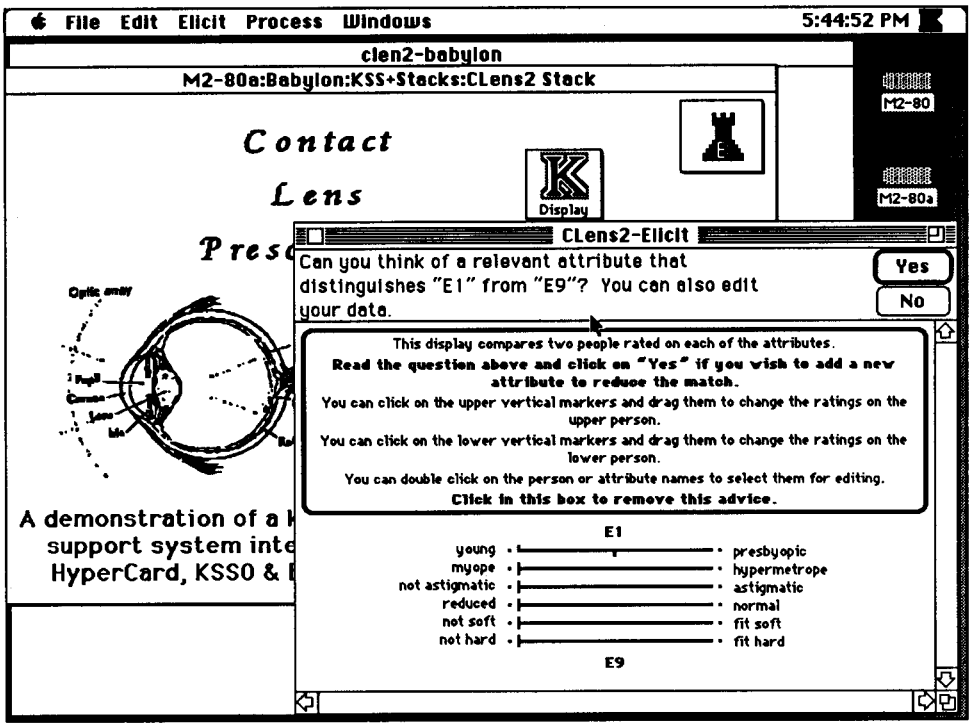


Fig.14 KSS0 operating in conjunction with HyperCard and BABYLON

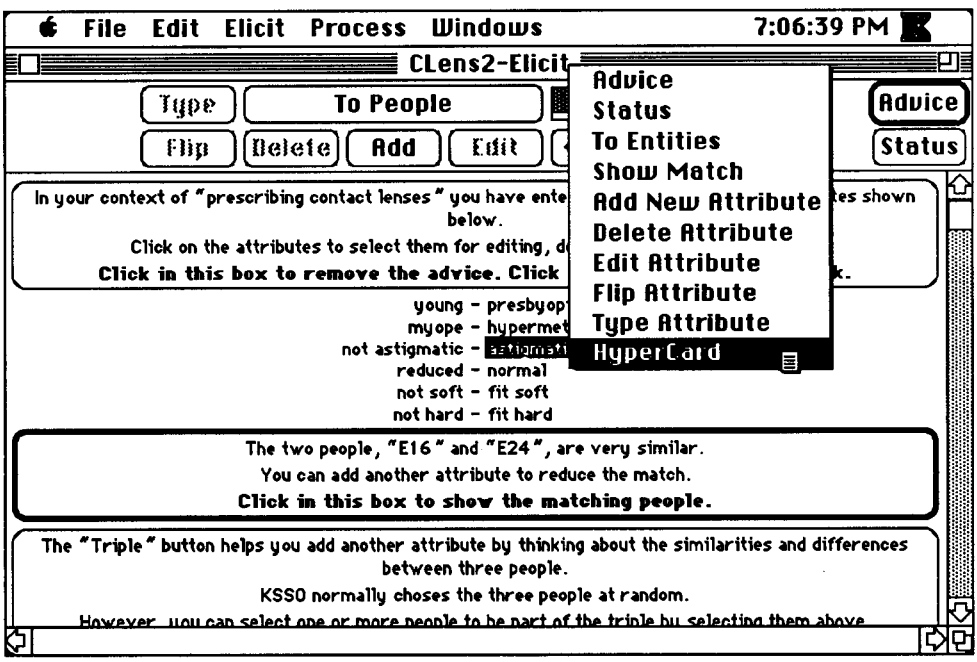


Fig.15 Pop-up menu in KSS0 allowing transfer to HyperCard

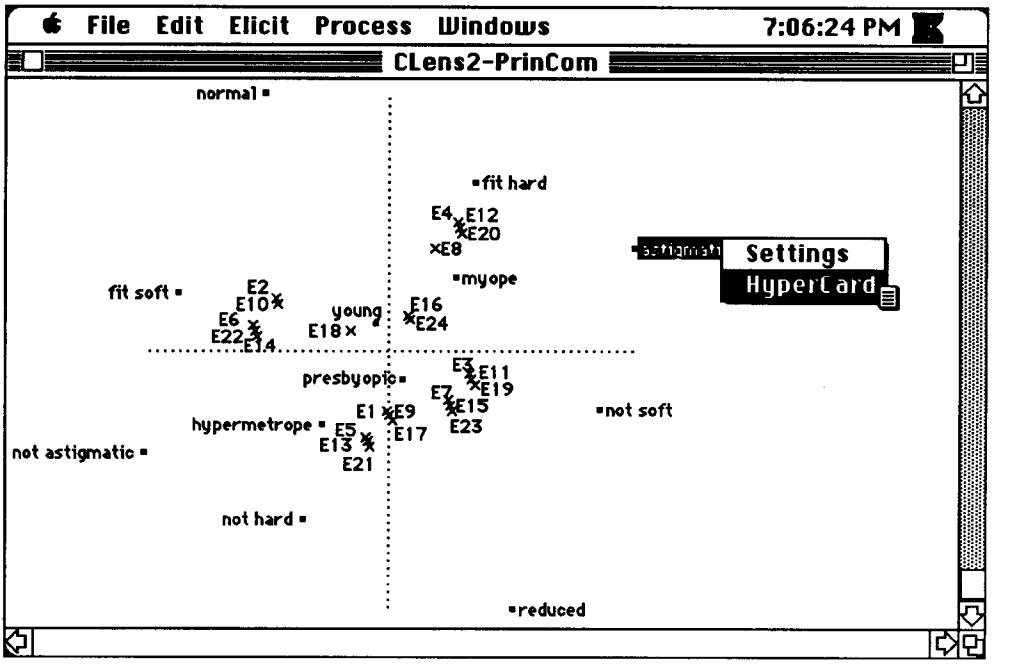


**Fig.16 Data clustering in KSS0 with access to HyperCard**

The close coupling between KSS0 and BABYLON enables the expert system shell to be used as a performance tool validating the knowledge base created in KSS0 against test cases entered through BABYLON. If the decisions recommended for the test cases are valid then no action need be taken, but if they are not the expert can transmit them from BABYLON to KSS0 as new data. This will affect the results of conceptual clustering and induction, and hence the knowledge base that is exported to BABYLON. Dependent on the amount of data and the difficulty of the induction, either the new knowledge base will operate correctly or further data may be posted back to KSS0 until it does so. Thus the the coupling of the acquisition tool and the expert system shell results in new modes of acquisition.

## CONCLUSIONS

The studies and developments reported show the feasibility of creating highly integrated knowledge support systems through the loose coupling of software tools developed independently at unrelated sites for different purposes. This is an important direction for future research and development of knowledge-based systems. The following analysis examines the roles played by each component and suggests directions for further development.

The hypermedium system plays two major roles in the system. First, it acts as an annotation facility allowing background knowledge and explanations to be captured from the expert that could not otherwise be entered in the structured knowledge acquisition tool. Second, it acts as a programmable user interface to the expert system shell. These two roles are synergistic in that the background knowledge and explanations entered during the acquisition phase are available to the user, and he or she may also extend them with further notes as a result of application experience.

The availability of the programmable hypermedium system has practical significance in giving end user groups the capability to customize the system for particular requirements and domains. It also has theoretical significance in allowing the entry of knowledge that does not fit the computational framework of the acquisition system or shell. This increases the possibility that the resultant system will be complete in that it enables information that goes beyond the formal knowledge base to be entered. Completeness cannot be checked within the formal system. It requires analysis of the knowledge base in the light of the doubts and reservations expressed by the expert and users without the constraints of computationally usable knowledge.

The expert system shell plays two major roles in the system. First, it performs its normal role of providing an advisory system based on the knowledge base. Second, it provides an on-line validation capability for the knowledge acquisition system. The knowledge acquisition system plays two major roles in the system. First, it performs its normal role of rapid prototyping in developing the knowledge base. Second, it provides a continuing on-line acquisition capability for the expert system.

The main limitation of the current system is that edits of the knowledge base in the expert system shell cannot necessarily be propagated back to the knowledge acquisition tool. That is, it is possible to create data and control structures that do not correspond to those created by the acquisition tool. Thus, the continuing applicability of the acquisition system during the application of the knowledge base is dependent on its not being edited within the shell. This is similar to the usual problem of editing the machine code generated by high level languages compilers. However, accepting the loss of editing within the shell would place very strong requirements on the knowledge acquisition system to be capable of representing all aspects of the domain and problem-solving knowledge.

The effort involved in creating the system reported in this paper was two person-weeks, part of which went into the inter-application protocol, part into the HyperCard scripts and part into the knowledge base export and import. The comparable effort to create a monolithic system would be many person-years. The heterogeneous integration strategy is an effective way to develop a next generation of knowledge-based systems.

As inter-application protocols become better standardized and more uniformly available, for example in X-Windows and Apple's System 7, it will become advantageous to split systems into modules at an even finer level of granularity than we have described here. BABYLON already consists of a number of distinct interpreters communicating through a blackboard that could well become a light-weight inter-application protocol. Similarly, KSS0 consists of a number of distinct knowledge acquisition tools operating around a common knowledge base.

Heterogeneous integration enables the full power of tools to be placed in the hands of end user groups without their losing the essential programmability of computer systems. The frame does not become a cage! We see future knowledge based systems as a loose amalgam of programmable modules presenting to the end users the feel and appearance of a highly integrated system, yet preserving for the end user communities the programming capabilities to customize the system as much as they require.

## ACKNOWLEDGEMENTS

We are grateful to our colleagues at the GMD, KSI and Knowledge Acquisition Workshops for many stimulating discussions which have affected the research reported in this paper. Research in the Knowledge Science Institute is funded in part by the Natural Sciences and Engineering Research Council of Canada.

## REFERENCES

[1] 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** (1), 3-28 (January).

[2] Boose, J.H. & Gaines, B.R., Eds. (1988). **Knowledge Acquisition Tools for Expert Systems.** London, Academic Press.

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

[4] Christaller, T., di Primio, F. and Voß, A. (1989) **Die KI-Werkbank BABYLON.** Bonn: Addison Wesley.

[5] Conklin, J. (1986). A survey of hypertext. MCC Technical Report STP-356-86. Austin:Microelectronics & Computer Technology Corporation (October).

[6] Gaines, B.R. (1987) An overview of knowledge acquisition and transfer. **International Journal of Man-Machine Studies 26**(4), 453-472 (April).

[7] Gaines, B.R. (1988a). Second generation knowledge acquisition systems. **Proceedings of the Second European Workshop on Knowledge Acquisition for Knowledge-Based Systems (EKAW'88).** GMD-Studien Nr.143. pp.17-1-17-14. Bonn, Germany: Gellsellschaft für Mathematik und Datenverarbeitung (June).

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

[9] Gaines, B.R. (1989a). 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 the Sixth International Workshop on Machine Learning.** pp.156-159. San Mateo, California: Morgan Kaufmann (June).

[10] Gaines, B.R. & Boose, J.H., Eds. (1988). **Knowledge Acquisition for Knowledge-Based Systems.** London, Academic Press.

[11] Gaines, B.R., Rappaport, A.T. & Shaw, M.L.G. (1989) A heterogeneous knowledge support system. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the Fourth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.** pp.13-1-13-20. Banff (October).

[12] 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).

[13] Gaines, B.R. & Vickers, J.N. (1988). Design considerations for hypermedia systems. **Microcomputers for Information Management,** 5(1), 1-27 (March).

[14] Guesgen, H.-W. (1989). **CONSAT: A System for Constraint Satisfaction,** London: Pitman Publishing.

[15] Guesgen, H.-W., Junker, U. & Voß, A. (1987) Constraints in a Hybrid Knowledge Representation System, **Proceedings of the Tenth International Joint Conference on Artificial Intelligence**, pp 30 - 33. Los Altos, Ca.: Morgan Kaufmann.

[16] Karbach, W., Linster, M. & Voß, A. (1990). Model-based Approaches: One Label - One Idea?, introduced to **EKAW90**, Amsterdam

[17] Kelly, G.A. (1955). **The Psychology of Personal Constructs**. New York: Norton.

[18] Linster, M. (1989) Towards a second generation knowledge acquisition tool. **Knowledge Acquisition, 1**(2) 163-183 (June).

[19] di Primio, F. and Wittur, K. (1987) BABYLON, a Meta-Interpretation Model for Handling Mixed Knowledge Representations. **Proceedings of the 7th International Workshop on Expert Systems and Their Applications**. pp. 821 - 833. Avignon.

[20] Shaw, M.L.G. (1980). **On Becoming a Personal Scientist: Interactive Computer Elicitation of Personal Models of the World**. London: Academic Press.

[21] Shaw, M.L.G. & Gaines, B.R. (1987) KITTEN: Knowledge Initiation & Transfer Tools for Experts & Novices. **International Journal of Man-Machine Studies 27**(3), 251-280 (September).

[22] Slater, P., Ed. (1976). **Dimensions of Intrapersonal Space: Volume 1**. London: John Wiley.