How AIPS Uses KL-ONE

AIPS uses KL-ONE primarily as a vehicle for object-oriented programming in LISP. KL-ONE's Attached Data and Attached IData escape mechanisms let us use Generic Concepts to define classes of program objects which can receive and respond to messages. Individual Concepts can inherit message response "methods" (tagged LISP procedures) from their generic ancestors, and thus can serve as instances of these classes. Using KL-ONE in this manner brings to LISP many of the advantages of object-oriented languages such as SIMULA and SMALLTALK-80.

Of course, KL-ONE's generality confers additional benefits not found in those languages. For example, role inheritance wires afford us greater power over the inheritance of object parts or attributes. Meta-description allows us to build interpretable commentaries on our "classes" and "instances". Finally, the resulting thorough declarative expression of program structure provides a solid point of departure when arranging for inferencing about program constructs and capabilities, or for semantic interpretation of natural language input from the user.

The AIPS Group
What Could It Be?: Reasoning about context, equality and Compatability

James Allen and Alan Frisch
The University of Rochester

Our work in Natural Language Processing has brought up some issues that must be faced by any Knowledge Representation. Many of these are considered in detail in KL-ONE, however, others seem to have been neglected. A few that we would like to discuss are:

- the indexing of propositions by contexts, i.e. by beliefs and wants of agents, by time, by microworlds,...

- the explicit representation and reasoning about equality between concepts;

- The precise specification of inexact matching of concepts.

These issues are not independent of course. We would like to describe how they arise and interact, what we have thought of doing about them, and would like to find out what KL-ONE could do in this respect.
KL-ONE WORKSHOP

I was first attracted by Ron Brachman's knowledge representation ideas at his 1976 COLING presentation, and was fortunate to be able to have some discussions with him about his developing ideas and to make some small contributions. Recently I have been merely following with interest at a distance - away, and behind.

I have an implementation in LISP 1.6 which preceeded the BBN KL-ONE implementation, but which is, however, far less complete. A major problem that I had to face was that I was working with an old DEC 10, with an old and rather undeveloped LISP, with a relatively small user space. These facts, along with the facts that I was working alone and in an environment with little or no LISP support software, made my knowledge representation implementation progress very slowly. It is difficult to not let that experience color my view of the theory.

In my use of Structured Inheritance Networks (SIN) it appeared to be necessary to group some hierarchies together and have inter-group associations. I used specially labelled pointers for this, for speed and general hygiene, despite the fact that that is against the spirit of SIN.

I'm not sure of the current KL-ONE mechanisms for defaults values, but I found that one has to be very precise about exactly what is being defaulted and how that is expressed. In addition, it is important to note where the default value came from when it has been put into an individual concept.

In addition, the importance of the Structural Description (SD) in SIN has been underplayed. In most cases, the structural information is much more extensive (and important) than the rest of the concept. I worry too about the purpose of the SD. It is for checking to see whether an individual is in fact an example of a Generic; it serves as something which can be examined to see how elements of the concept are to be related; and (possibly) it could be used to 'drive' the building of an individual given some details and a Generic. Why should each of these be a single piece of representation? Is this assuming a factorization of knowledge to support different purposes, without any investigation to see whether this is really the case?
Prospective user: Jos de Bruin (University of Amsterdam)

Project title: Learning of problem-solving skills in physics.

Project outline:
The hypothesis that forms the basis of the project is that "learning by doing" is a process through which problem-directed knowledge structures emerge, which allow an improved performance of the information processing system. This seems to be particularly the case in problem domains which require a large amount of knowledge in finding the solution, e.g. physics, medical diagnosis, chess. The specific questions to be considered are:

1) what is the precise nature of the problem-directed knowledge structures in the successive phases of the learning process,
2) what are the conditions and activities in the learning system that allow their information.

A preliminary theory of problem-solving as a learning process will be further developed on the basis of the analysis of think-aloud protocols of subjects solving a series of the thermodynamics problems. A computer program will be developed that is able to code the protocols in terms of the theory.

The main object of this research is the explicit representation of the procedural and strategic (meta-) knowledge that people with differing degrees of expertise bring to bear when solving problems in thermodynamics.

The first stage involves the construction of a problem solving program, later to be differentiated according to the amount of knowledge available to it.

Solving physics problems:

In such a problem solving program we can distinguish several tasks, each placing its own demands on the knowledge representation supporting it:

1. Formation of a conceptual representation of the problem, from natural language or some intermediate representation (natural language understanding is more a personal interest than a first concern of the project). On a knowledge (or definitional) level this requires a good representation of processes, states and state-dependent attributes, on a meaning (or assertional) level this asks for a way to represent propositions.

* The project described just started. Four people will be working on it for the next four years. I am only indirectly involved, as a student and future assistant of one of these four.
2. Transformation to canonical concepts from the physics domain (e.g. "thermodynamic system", "container-with-piston", "expansion", etc.).

An important step in solving problems is finding the right perspective from which to view the objects and processes described in the problem-statement, i.e. establishing their relevant functional role in this particular context, while abstracting away from irrelevant details. For instance, the problem solver must "see" the process described in the problem-statement as a number of changes in the attributes of a fixed amount of matter. In general, these perspectives will not be represented with the objects or events themselves. Therefore a hierarchical taxonomy will not be enough. To match the concepts found with canonical concepts, we need an inference mechanism on the basis of common sense knowledge and functional descriptions of the canonical concepts.

KLONE offers SD's to represent functional properties. This led to the following search principle: given a proposition, expressed as a network of individuated concepts, find a SD (i.e. a para-individual) that partially matches. If so, assert the existence of the corresponding concept by individuating it with the values given in the first proposition. For instance, to find the thermodynamic "system", the important thing to discover is a certain "contain" relation.

What we need to continue in this direction, are ways to represent defaults, defining vs discriminating attributes and control information, to allow partial matches.

We also need some causal reasoning à la Rieger and wonder to what extent this means introducing new link- and/or modeltypes.

3) Solving the problem on the basis of a canonized representation. This requires an explicit (declarative) representation of rules (physical principles) and strategies to search the ones to be used. Such a representation should allow for the propagation of accumulating knowledge, which again points at the control problem.

Why KLone?

We choose KLONE as the basis from which to attack (some of) these problems because we consider clean semantics and methodological reluctance to introduce new links or node types of great importance.

Furthermore, we think the notion of structural descriptions offers a new perspective on recognition and therefore inference.
Although procedural attachment by itself is another advantage, especially in the context of problem-solving, we have reservations about unlimited use of it. Firstly, because our aim is to make procedural knowledge declarative, i.e. "understandable" for other knowledge, secondly, because unrestricted use of "black box" procedures threatens to ruin the clean semantics again.

Some final remarks

We think it's very important at this stage to really start the proof of the pudding. It seems premature and impossible to give detailed opinions on KLONE before it has actually been used and compared with other formalisms. The ultimate criterion must be: "What is made possible (or more efficient) by KLONE that was impossible or more difficult with other formalisms?"

Relying on our own (very powerful!), mental "interpreters" to judge the usefulness of new representational devices seems dangerous.

Even more urgent is the need to come to some sort of agreement on how to control the introduction of new links or nodes.
Position Paper
from a category 4.17 participant,
an interested party "semi" familiar with KL-ONE

Nick Cercone
Computing Science Department
Simon Fraser University
Burnaby, British Columbia, CANADA V5A 1S6

Skillful exposition and delineation of conceptual entities can engender much progress for the continuing activity of knowledge representation and knowledge organisation. Well considered theoretical foundations emphasized in the design of KL-ONE set it apart from other, disparate approaches. Programming invariably uncovers many pragmatic problems with the designs of various representations, nonetheless, methodology, lacking serious commitment to thoughtful, systematic theory will fail. Therefore, I cannot view approaches where programming precedes theoretic development sympathetically. KL-ONE, happily, is not one of these.

In our earlier work, summarized in Schubert et al (1979), we described our computer-oriented logic representation and introduced some organisational strategies. Programming was indeed minimal and experimental in nature. Since that time Schubert has proceeded with a very good mini-implementation extending the earlier work; this research is reported in Covington's (1979) thesis. Cercone is currently supervising work to extend the topic organisation by developing the notion of an "adaptive" topic predicate. Progress on this research has been very slow to date.

Initial intrigue with Brachman (1979) led to my considerable interest in KL-ONE. Brachman's taxonomy of representation levels was clear and precise. Subsequent development of KL-ONE was also clear and precise. I agree in principle that it would be advantageous to use a prepackaged knowledge representation language, one which captures my concerns for expressive power, clarity, ease of use, and predictive power; I am not sure my present understanding is adequate to make such a judgment about KL-ONE. Hopefully, this workshop will provide some answers (and perhaps new questions).

Lingering doubts can always ultimately trace to foundations. Let us examine KL-ONE's epistemological primitives - objects. I cannot accept Brachman's disclaimer regarding the history of network representations fraught with imprecision on the meanings of nodes and links. Further, I cannot accept this disclaimer to justify "objects" which may indeed be justifiable. The logical machinery for interpretation and inference should also be (formally) outlined. A KL based on propositions can command considerable respect regarding elegance, manipulation, and interpretation. As well have learned, however, this does not ensure success as a KRL. KL-ONE appears to have captured
important distinctions logicians warn of—generic vs individual, descriptions, etc. I reserve the right as a type 4.17 participant to become skeptical as the pragmatic issues evolve and force the ultimate foundational introspection.

References


Prospective user: Jeff Conklin
Computer and Information Science Department
University of Massachusetts at Amherst

We are developing an A.I. system to write English descriptions of suburban outdoor scenes. We take as our input not an image, but the output of a visual perception system (the VISIONS system at U. Mass./Amherst). Since the systems' high-level routines and representation are still under development, we will be simulating their perceptual representation in a hand-coded semantic network. This network will contain general knowledge about objects, attributes, and relationships (both spatial and structural) found in suburban neighborhoods, as well as the specific objects, attributes, and relationships that are found in the specific image whose perception is being simulated. We are investigating the pros and cons of KL-ONE as a formalism both for our perceptual representation simulation and for its potential use within the VISIONS system itself.

The system under development will select concepts in the perceptual representation, "dress them up" with the appropriate pragmatic and rhetorical details, and pass them, as KL-ONE networks, to McDonald's MUMBLE system for realization into English. Here KL-ONE offers the convenience of being a formalism that MUMBLE already knows how to interpret.

The appeal of KL-ONE, besides being a really nifty formalism, is that it could serve to represent both the perceptual input, the linguistic surface structure, and any stages in between without, we hope, straining its links.
The notion of aggregation has been fundamental to all associative network formalisms since the beginning. A complete formal semantics for this notion has still not been established, however. KL-ONE has attempted to formalize, via its Structural Description construct, means for specifying the relevant interrelationships that exist among elements of an aggregation, but it has not yet provided us with a useful framework for specifying inheritance constraints along the generalization/specialization (SUPERC) hierarchy. It has also left the introduction of additional attributes into an aggregation along this hierarchy somewhat of a semantic mystery. And it seems to assume, along with most other associative network formalisms, that attributes can be treated as single-argument functions (i.e., given an individual concept and an associated attribute, this is sufficient for the latter to yield up an unequivocal description of its value). These issues, as it turns out, are intimately connected with other problems for KL-ONE, in particular suitable means for introducing notions of time, change and context of interpretation into the modelling of conceptual entities.

We define a context of interpretation as the constraint space in which particular conceptual entities must remain for them to be recognized as themselves. In some cases, the spaces are bounded temporally, but severely constrained with respect to any change, e.g., 'being the N'th root of M'. In other cases, just the opposite obtains, e.g., 'being employed by Z'. In all cases, however, the space serves to delimit the possible relationships in which entities can be with respect to one another. By viewing an entity in such a space, one in essence conceives of it in particular roles vis à vis other entities, thereby associating with it attributes (not just values) that it did not have before. For instance, '4' has the attribute 'INDEX' and 'Radicand' associated with it only with respect to its being the N'th root of some other number M. But note that such attributes cannot be treated as single-argument functions; neither one can return an unequivocal description of its value without reference to the other. Similarly, 'EMPLOYER' or 'SALARY' of 'ABBIE HOFFMAN' is meaningful only insofar as it can be relativized with respect to a particular employment arrangement.

This has led us to introduce a new construct into KL-ONE, which we refer to as the QUA cable (after the use of the word in such expressions as "Quine, qua lay physicist, believes in physical objects and not in Homer's gods"). In addition to providing a way to conveniently represent the role-dependent relationships between entities and their defining contexts over time, the QUA cable also serves to establish a new type of inheritance hierarchy with a natural partitioning of inherited attributes into vistas (represented by V12 sub-links or bundles within a given QUA cable). Furthermore, by viewing descent down
the QUA-hierarchy in terms of the embedding of one defining context within another, one in essence is modelling the type of presupposition involved in a 'loaned object' normally being considered an 'owned (or possessed) object'. Along these same lines, one can also view constraint spaces as a natural way to model event-dependent presuppositions (e.g., for X to 'fire' Y, Y must be in the employ of X at the time). Although there are many ways to represent constraint spaces, we have found Augmented Event Transition Nets (AETNs) to be an especially good one, both for purposes of giving one an overall view of the possible event sequences that may transpire, as well as for providing a convenient framework in which to specify formal constraints on the specialization of such spaces. This latter advantage addresses a major aspect of the inheritance problem associated with Structural Descriptions in KL-ONE.

Viewed as a grammar which specifies all (and only all) legitimate event sequences, the AETN enables the association of relevant 'event logs' with individuations of 'mutable' conceptual entities. These event logs can record the particular sequence of events that has or eventually will affect the individual, oftentimes with respect to some 'dynamic attribute', such as 'MARITAL STATUS'. In this way, the AETN construct creates the possibility of automatically generating 'derived' attribute values, based on different event sequences. In many cases, this may involve the generation of a whole set of related attribute values, depending, say, on how far back in an event sequence one looked in order to compute a resulting state value. For example, an individual's marital status may simultaneously be characterized by the 'values' single, widower, divorced, and (twice) previously married. In addition to simply referencing particular events, however, the event log can also serve to monitor for events yet to occur in some hypothetical or possible world. The natural association of AETN 'monitors' and 'alerters' provides an extremely flexible basis for maintaining database integrity and consistency, determining historical perspectives, and signalling possible irregularities in the attempted recording of new events which violate the legal event sequence defined by the net (e.g., the report of a marriage between two people, one of whom is believed to be still married to someone).

Finally, the effective partitioning of associated attributes among the various viewpoints facilitated by the AETN also provides a basis for very fine-grained access control, and for reducing the search space through the associative net during the resolution of high-level (e.g., natural language) role-dependent references to particular entities.
ROLE PROPAGATION IN KLONE

Bill Mann
ISI

There is a highly recurrent situation in Klone representations for which I would like to propose special treatment. By example, consider the plan for making a move in chess. We could start to represent this with five concepts: CHESSMOVEPLAN, CHOOSEMOVE, GRASP, MOVE-HAND, RELEASE. The intention is that the concept of the plan for making a move in chess will have four steps, choosing the move to make, grasping the piece to be moved, moving the grasping hand (and thereby the piece), and releasing the piece. These will be fleshed out with various roles, including the ACTOR of all the steps and the plan, the PIECE chosen, the SOURCE-SQUARE and DESTINATION-SQUARE, etc.

Notice how many reuses of role fillers there will be in this concept:

- The ACTOR appears 5 times, once on the plan and once on each step.

- The HAND appears on the GRASP, MOVE-HAND and RELEASE steps, and presumably on the ACTOR, 4 times.

- The SOURCE-SQUARE and DESTINATION-SQUARE each appear 2 times, on the MOVE-HAND and either the GRASP or RELEASE.

- The PIECE appears 3 times.

In each case there is only one object of the conceived kind in a non-defective move, and if the uniqueness of that one object in the plan is not assured, then the concept covers too much because it covers cases with multiple objects of some kind.

Note: This memo was written during an effort to represent planful action in Klone. The action representation utilized was partly inspired by A.I. Goldman's action theory, and so the example is an attempt to formalize one of his.
PROBLEM: REPRESENTATION OF REOCURRENCE IS TOO COMPLEX

All of these same-object constraints must be represented in the structural condition of the CHESSMOVEPLAN. Formation of the condition will require parainstanceion of all of the steps, and will require (for N items to be equated as one, N-1 coref relations) at least 10 coref relations, and the job is by no means complete. Expression of a single equating of a role-filler of the plan and a role-filler of a step of the plan requires 4 new nodes in the SC and 6 arcs.

There are several reasons to seek a different representation:

1. Creating the actual representation (manually) is tedious and error-prone (and very prone to omissions);
2. The resulting diagrams are cluttered and hard to use;
3. There is an underlying phenomenon to be recognized and given representational status.

The easy way to work out an alternate representation is by characterizing the underlying phenomenon.

ROLE PROPAGATION

Notice that many of these role-filler equalities arise out of the necessity (for the plan's effect) to achieve a combined effect of several operators or operations. The piece that you choose must be the piece that you move; the destination square that you choose must be the one of the release; the hand used for
grasping must be the same hand that is moved, or the piece won’t move, etc.

Notice also that recurrent roles on the plan steps are mostly those which are naturally of interest when representing the plan (or its use) as a whole.

WHITE (Fischer)                           BLACK (Mann)
1. P-K4                                   2. P-QB3

In a full chess transcript the actor, piece, destination square are explicit, the source square is implicit and the hand is not represented. This suggests making recurrent roles in the subparts of the plan be roles in the plan, and then representing the recurrent equalities by relating these to the others. As an example, consider representing just the recurrent roles of the PIECE. Figure BEFORE shows the structural condition needed to represent these recurrences in Brachman’s 1978 thesis notation, and figure AFTER shows a corresponding rerepresentation.

FILE BROKEN AT THIS POINT.

I had a solution to the problem here. It was in an obsolescent and somewhat irregular notation, and was only lightly tested. Right now I have no confidence in that solution, so I won’t put it forth. However, the problem is real and recurrent when representing actions, and I suspect that it is as widely recurrent as the problem of representation of sets.
Just A KL-ONE Call Away

Bill Mark

(on behalf of the Consul Group: David Wilczynski, Bob Lingard, Tom Lipkis)
USC/Information Sciences Institute
3 October 1980

KL-ONE On Its Own

The philosophy of KL-ONE provides a useful discipline for modelling what we need to model: the actions, objects, and events of interactive computing. That is, KL-ONE serves as a framework for clarifying our thinking and ensuring that our representation decisions are implemented in the knowledge base in a clear and consistent manner. On the other hand, KL-ONE does not provide much guidance in the solution of specific knowledge representation problems (at the level of, say, how do you represent actions affecting objects?). Furthermore, even as a philosophical framework, KL-ONE is incomplete for our purposes. Nonetheless, we think it's the best thing around right now.

Compared To Who?

KL-ONE does offer certain unique advantages as a knowledge representation system. The most basic one is coherence of approach. We feel that in many cases KL-ONE makes a real effort to do things "right" where other systems simply do things (e.g., the relationship of characteristics and inheritance). Unfortunately, this seems to mean that KL-ONE does fewer things than some other systems, leaving more work for us (when we require short term solutions). KL-ONE also makes distinctions that are not made in other representation systems (e.g., the differences between generics, individuals, nexuses, and paraindividuals). So far, we have found these distinctions to be important and useful—in fact we required all of them to build our knowledge base.

KL-ONing Ourselves

Our experience with KL-ONE, then, is basically positive. However, as we said, many necessary constructs, facilities, and conventions are not yet part of KL-ONE. We therefore find ourselves making up and using our own conventions within the KL-ONE formalism—that is, using our own language within KL-ONE. We consider this to be the result of a trade-off: we like KL-ONE's attempts to be precise and consistent, but this means that we must await results. Since we are in the midst of building a working system, though, we must develop our own solutions to many problems, at least for the purpose of experimentation. In our "language" we try hard not to circumvent KL-ONE restrictions and stay in the spirit of KL-ONE. But we feel that there is a growing danger of real divergence as we rely more and more on our own conventions; we may soon be using a KL-ONE derivative, rather than KL-ONE.
Structured-Inheritance Networks in Model Management

The idea of model management is based on the recognition that models are in many respects like data, both from the point of view of the user, and from the point of view of the programmer. A Model Management System (MMS) would provide the kinds of services in regard to models that a Data Base Management System (DBMS) provides in regard to data. However, in addition to the familiar DBMS functions, such as simplified storage and retrieval procedures, the MMS would include at least one important component for which there is no obvious analogy in the standard DBMS. This additional function, which might be called an "automated consultant", would make the MMS accessible to users not familiar with the abstract conceptual framework of modeling. The automated consultant would interact with the user to help him or her structure the situation to be modeled so that the automated consultant could determine what set of models supported by the MMS would be most appropriate. For example, a user who knows nothing about network models, but a great deal about a certain food processing operation, might describe the situation to be modeled in terms of "shipping tomatoes", "canning tomatoes", and "shipping cans". Having elicited a complete description of the situation by prompting the user for information concerning the places to and from which goods are shipped, the location of the canning plant, etc., the automated consultant might then infer that the user's situation would best be modeled by a general transshipment model.

From just this brief characterization, it should be clear that such an automated consultant must be fairly intelligent if it is to be of any help at all. It is this requirement which led to the choice of S-I nets as the representational form for the MMS knowledge base. In particular, the S-I net approach meets our needs by providing representation at a sufficiently abstract level to facilitate the kind of inference that the automated consultant must be able to make, while at the same time being sufficiently structured to aid consistent extension of the knowledge base. Because the MMS knowledge base is a single S-I net which contains both technical concepts such as "transshipment model" and non-technical concepts such as "production process", as well as certain context-independent abstractions such as "process" and "state", the automated consultant is able to draw conclusions in technical terms from information received from the user in non-technical terms.

Although the knowledge base is integral to all aspects of the MMS, our attention is currently focused on development of a primitive version of the automated consultant to be implemented soon in PASCAL, using a CODASYL DBMS (SEED) to handle the S-I net itself. In this version, the user does not interact directly with the knowledge base. Instead, the user interacts with an "interface manager" which (at least in this primitive version) knows a fair amount about both the structure and the contents of the knowledge base, and which more or less controls the interaction.

The interface manager uses the S-I net in three importantly different ways:

(1) It "reads" definitions of concepts from the S-I net representation to verify that the user's understanding of important terms corresponds to that represented in the knowledge base.
(2) It constructs a representation of the situation described by the user by constructing a new set of S-I net concepts with appropriate links to the fixed MMS knowledge base.

(3) It makes use of abstract features derived from the S-I net representation of the situation to be modeled as the basis for an inference as to which of the models represented in the knowledge base is most appropriate.

Much of the difficulty encountered in developing the primitive automated consultant stems from a need to severely limit the range of concepts represented in the S-I net. To a large extent, this constraint is dictated by our interest in implementing a first stage prototype as quickly as possible. We hope to learn a great deal about the strengths and weaknesses of our overall design by actually implementing a modest system. At the same time, however, we are proceeding on the assumption that a reasonably sophisticated automated consultant can be developed which does not include a full-blown natural language understander. In connection with this assumption, we are very much interested in evolving techniques to construct S-I net representations of specific areas of knowledge which are complete enough for purposes of the automated consultant, without including all possible connections to other areas of knowledge.

The current primitive automated consultant will be simple-minded in its processing of the S-I net as well as in its paucity of concepts. We hope that increased familiarity with S-I nets will lead to greater insight into how to process these structures intelligently.

David Oppenheim
Hassan Ait-Kaci
Operational Decision Aiding Project
Decision Sciences Department
The Wharton School
The University of Pennsylvania
Philadelphia, PA 19104
Applying KL-ONE to Physical Systems

Martha Palmer
University of Pennsylvania

In working with simple physics problems, there is one aspect of knowledge representation for which KL-ONE is ideally suited. The standard relationships - inheritance of properties, division into parts and subparts, etc., are all necessary. But it is in the structuring of the internal relationships of frame-like physical systems, such as pulley systems and scaffold systems, that the KL-ONE structural description facility would be especially useful. A physical system is a standard collection of everyday objects that act together to perform a particular function. A pulley system has three components - or ROLESETS: the pulley, the string, and one or two solid objects. In solving pulley problems, it is helpful to recognize when a collection of objects described in the problem statement are in fact an instance of a pulley system. But it is not sufficient to merely list the set or objects. The internal relationships between the objects must also be specified, similarly to the "ARCH" example. The pulley is in "moveable contact" with the string, and the string has a "fixed contact" to the solid objects involved. It is immediately obvious when looking at the following problem statement why these relationships must be specified.

"Two pulleys of weights 12 lb and 8 lb are connected by a fine string hanging over a smooth fixed pulley. Over the former is hung a fine string with weights 3 lb and 6 lb at its ends, and over the latter a fine string with weights 4 lb and x lb..."

The real benefit of KL-ONE would be realized, however, after the pulley system has been recognized when other internal relationships would become available. The acceleration of one side of the string is equal to the negative acceleration of the other side of the string, if there is no friction the tension in the string is constant, and so on. It is not clear that at some stage these physical law relationships might also be used for recognition purposes, and the structural relationships then assumed. At any rate, it is a powerful representation language that can accommodate such diverse information.
Implementing KL-ONE in Prolog

As a Prospective KL-ONE User, I thought it would be interesting to investigate implementing KL-ONE in Prolog. Such an implementation, if feasible, would provide a logical description of KL-ONE mechanisms that could make KL-ONE more widely accessible. It is also possible that Prolog's unification facilities might make more transparent some of the KL-ONE facilities, such as COREF and ROLEVALUEMAP. But Prolog would probably receive the greatest benefit as such an implementation would illustrate that Prolog can handle complicated "real-world" database problems as well as isolated theoretical problems.

Without having access to details of KL-ONE implementation, the following types of inferences rules are suggested as a first step towards such an implementation. For the most part they refer to the RoleValueMap illustrated in Fig. 7 of Report No. 4274, An Introduction to KL-ONE.

"THE NAME OF AN ARCH IS THE SAME AS THE LAST OF THE NAME OF ITS DEDICATEE"

\[
\text{llname(Concept,Name) } \leftarrow \text{roleset(Concept,Rs), name(Rs,name)}. \\
\text{rvm(X) } \leftarrow \text{roleset(arch,R1), name(R1,Name), instance(R1,X), roleset(arch,R2), name(R2,dedicatee), relate(dedicatee,name), roleset(name,R3), name(R3, last), instance(R3,X)}. \\
\text{relate(X,Y) } \leftarrow \text{concept(C,X), roleset(C,R1), (name(R1,Y); vr(R1,Y)).} \\
\text{relate(X,Y) } \leftarrow \text{roleset(R,X), vr(X,Y)}. \\
\text{relate(X,Y) } \leftarrow \text{roleset(R,X), vr(X,Z), relate(Z,Y)}. 
\]
Prospective User: Rachel Reichman (Harvard University/BBN)

Project: A Process model of discourse that simulates ‘coherent’ discourse generation (and/or interpretation).

The theoretical basis underlying the grammar is its analysis of a conversation into a hierarchical network of formally related context spaces; and its recognition that at any given point in the conversation, the relevant discourse context for the production/interpretation of succeeding utterances are the set of context spaces whose ‘state’ is either ‘active’ or ‘controlling’.

Each succeeding utterance in a discourse is seen as either continuing a preceding ‘coversational move’ or beginning a new one. Among others, the grammar recognizes the following types of coversational moves: Supporting the claim of a preceding context space; attacking the claim (or its supports) of a preceding context space; further-developing the subject of a preceding context space; interrupting the subject matter of a context space; or returning to the subject matter of an initiating context space of an analogy.

The start of a new conversational move usually results in the creation of a new and distinct context space; it always entails the reassignment of the ‘states’ of preceding context spaces. Actual reassignment is dependent upon the type of conversational move being executed. For example, while support of a context space is being given, the context space being supported is in a ‘controlling’ state.

A context space having a ‘controlling’ state, reflects that its elements are "visible" to the new context space being developed. Elements of a "closed" context space, for example, are not visible to a context space being currently developed.

Nine different types of context spaces have been defined, they themselves being specified in terms of a "systemic" grammar (Halliday), or a network incorporating both inheritance and discrimination net features (KL-One, Brachman). Besides the "state" slot, all context spaces have a "Claim"/"Support" slot in which is noted a representation of the discourse utterances said to lie in this context space.

I would envisage having such representations in terms of KL-One structures, where, for example, a speaker’s uttering of "Mark ate the whole cake", would point to an instantiation of an "eating food" activity. Based on the type of discourse analysis performed by the grammar, its use of a KL-One representation schematization for the discourse will necessarily entail heavy emphasis on a methodological development of procedures that would add the representations to the data base in such a way as to reflect the visibility or non visibility of these representations to others.
In addition, some of my current research seems to indicate that not only is access to entire representations of discourse elements dependent upon the discourse 'state' of the context space in which they lie, but that, in addition, parts of these representations may be visible while others are not. Specifically, while certain attributes of an entity seem to be visible anytime the entity is referenced, the visibility of many of an element's attributes seem to be more highly linked to the discourse context space in which the element is referenced and the connection between this context space and preceding context spaces of the discourse in which these attributes of the element may have been introduced.

As a brief example, two children were first introduced in context space C1 of a discourse as 'twins'. In a later context space, C3, these same entities were referenced as 'two brothers'. In yet a later context space, C12, they were implicitly referenced as being members of the class of 'two kids brought up in a same home'. In response to this simple classification and resulting assertions about them as members of this class, a second speaker stated, "But these kids were twins though". This disclaimer, which involved the close deictic reference 'these', clearly reflects that these entities were visible in the current context. Simultaneously, however, the disclaimer seems to indicate that the property of their being twins was not immediately visible. There are three possible causes for this non-visibility: preceding implicit reference; intervening paraphrase as 'two brothers', or the simple fact that this new reference was in a distinct context space from the other references. Whatever the cause, our KL-One representations will have to facilitate such differentiation between access to some properties while non-access to others.

Since it seems that we might be able to predict visibility of properties based on a context spaces's relation to a preceding space, or intervening paraphrase, it would be nice if such visibility features were automatically accommodated by KL-One. For example, let's say we had the rule that a later reference to an entity in a succeeding context space did not have access to earlier more specific descriptions of this entity if in the intervening a more general paraphrase for the description were given (e.g., 'two brothers' in place of 'two twins'). Then, whenever such a state-of-affairs occurs we would want KL-One to automatically restructure the description of the entity in its associated context space so that the data base would reflect that the earlier description is no longer visible to references to the entity outside of this context space.
Perspective: A Prospective User

Harry Tennant
Texas Instruments Inc.

We are working on natural language database query, and we want to be able to describe the domain of discourse in a knowledge representation formalism such as KL-one. This would enable us to answer questions about the domain (questions that exceed the scope of the formal database), allow us to respond "I understand you but can't answer the question", and be able to plan search strategies and negotiate queries to a very large database.

Our goals are to build a transportable system, meaning one in which the parts that must change between domains is clearly marked. Thus we are interested in a knowledge representation language that can express concepts from a variety of domains. We feel that the attention given to epistemological issues in KL-ONE may make it better suited than most. We are concerned, however, about getting commited to a very large system that requires us to pay for capabilities that are not necessary for our applications. In short, we are still considering.
Aravind and I are members of a group of potential KL-one users here at Penn. The technical problems we are concerned with (in planning, alerting, and understanding texts involving changing objects and/or situations) all touch upon the representation of possible or actual changes over time in the descriptions of objects and situations. The reason for wanting to use KL-one in these areas is that its structured descriptions (along with SDs, RVMs, RSRs) provide a clear and systematic way to describe complex objects and situations and the things that can happen to them. Moreover, KL-one provides a mechanism for linking up these descriptions to the individuals in a context to which they apply. What it does not provide are handles onto change.

For planning (as well as for understanding texts about "time-varying" objects and/or situations - e.g., narratives, building instructions, etc.), important handles onto change involve a way of linking contexts (here representing states of the world) or, more precisely, a way to have this context-linking fall out automatically from the assertion (i.e., nexus-wiring) of an inherent context-changing operator/concept like "build", "move", "die", "stack", "delete", etc. That is, asserting that one has (or plans to) build a raft from some logs and twine or carve a moose from a block of ice should establish at least two contexts which differ in how object descriptions are wired to nexuses - a BEFORE and an AFTER. (Other verb-type concepts like "see", "say", etc. do not necessarily entail multiple contexts of this sort.)

In the context of dynamic (i.e., updatable) data bases, the following handles onto change are important:

1. Whether or not an attribute is updatable (i.e., its value can be updated)
2. Whether or not a relation is updatable (i.e., its domain or range can be updated)
3. Whether or not there are any constraints on these updates (e.g., constraints that hold between different attributes or different relations)
4. When the dynamic character of the data base is due more to some underlying process than to ad-hoc user-made changes (e.g., a student starting as undergraduate, then becoming a graduate student, then a PhD candidate after passing the prelims, etc.)
5. How volatile the updates are (e.g., faculty office numbers are likely to change more often than their phone numbers).
Are there other people interested in these or similar problems? I seem to recall a hypothetical harbor master dialogue that Phil Cohen wrote last year that involved changing situations and alerting - i.e., the hypothetical harbor master saying, "Tell me when that tug passes Hells Gate".

Also as members of the group of potential KL-one users, we want to bring up the topic of more general accessibility ("Klone-heads, out of the Closet!"). What is being done (or can be done) to make KL-one more readily available as a research tool? We have here at Penn the student support to bring KL-one up on the VAX in Franz, but we need much more of a sense of the effort involved and perhaps some shared experiences in transferring (re-implementing?) all or some parts of the system.
What attracts me to the class of languages like KL-ONE is their ability to handle natural objects, organize information, and summarize facts hierarchically. What attracts me to KL-ONE over other formalisms having those properties are the following aspects:

1. the epistemological foundations
2. the explicit use of structural conditions
3. the emphasis on distinguishing between criterially defined objects and natural-kind objects, and
4. the attempt at a natural-like language (JARGON).

What I think might be done differently:

1. the structural conditions (Representing these conditions as graphs seems to obscure them to me, as opposed to using expressions, such as in logic.)
2. a second kind of inheritance (Conveying information by analogy is a very useful concept in natural reasoning. A second kind of inheritance cable might be defined to capture the semantics of analogy.)
PROSPECTIVE USE OF KL-ONE*

Two different projects might use KL-ONE in a related way.

Specifications of abstract data types (or modules) are very difficult to create and understand at present. I hypothesize at least two reasons for this:

1. there is no body of defined concepts to draw on as there is in natural language specifications
2. the information in formal specifications can be organized quite differently than it is in natural language.

The ability to organize and summarize information in KL-ONE may provide a more natural way to capture the information necessary in formal specification of abstract data types. Similarly, the same features may allow for the development of a library of defined concepts in formal specification, not only to draw on the previously stored definition, but also to modify parts of it to tailor the defined concept to a new need.

The second project is a joint effort with Norman Sondheimer at Sperry Univac in responding intelligently to input that is ill-formed, either because a native speaker would consider it ill-formed or because it does not correspond to the system's expectations. One type of ill-formedness in a natural language interface is exceeding the functional capabilities of the underlying system. An example is "What is the weight of sample S23?" If the underlying system does have weights of certain entities (e.g. the atomic elements) but not of samples, the question exceeds the underlying data base's capabilities. Brachman's dissertation sketches how to represent the functional capabilities of a system. We believe we can develop algorithms to use such a representation to detect overshooting the underlying capabilities of a system.

*Research sponsored by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under contract no. F49620-79-C-0131 and grant no. AFOSR-80-0190 and by the National Science Foundation, Division of Information Science & Technology, under grant no. IST-8009673. The United States Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.
A User's View of KL-ONE

The attraction of KL-ONE is that it is a general-purpose knowledge representation language written in Interlisp, whose core primitives (links and nodes) have well defined semantics. In most cases, writing a sufficient KL-ONE description for some purpose is a straightforward process which does not involve straining against the language in order to achieve the desired effect. The most notable awkwardness (and it is a minor point) is the lack of a mechanism for disjunctive value restriction. In any case, KL-ONE's meta-description features and escapes into LISP provide necessary flexibility for necessary hacking.

The second version of KL-ONE is a much more complicated beast than its predecessor. The set of primitive entities has been considerably expanded (e.g.: nexuses, contexts, RVM's, and RSR's). Some of these serve rather specialized needs. Some depart from the original purely descriptive spirit to enable the expression of Predicate Calculus-like assertions. There is some reason to wonder whether the rush to cover functional territory with language primitives and their features may not lead to KL-ONE becoming a sort of PL-1 of knowledge representation languages. No doubt these extensions are convenient for their users, but they are not without their costs. Perhaps rather than be extended, KL-ONE should be extensible. This means implementing the essential KL-ONE and its environment so as to facilitate user extensions to the language and permit a "package library" approach to satisfying needs that are less than universal. Apart from considerations of cleanliness and elegance, defining a nucleus KL-ONE would provide a delimited point of departure for building up new language features (e.g.: meta-level pattern grammar, pattern matcher, when-added network consistency checks).

KL-ONE's most critical difficulty is the lack of tools or support environment for making serious use of the language. For example, there is no perfected facility for dumping KL-ONE networks onto files in human-readable format. Similarly, none of the three available network writing systems is totally satisfactory: CKLONE lacks an interpretive reader, JARGON can hide network topology and interposes too much "English", and the BRACKETREADPRINT syntax is difficult to work with. We need an easily read symbolic equivalent for KL-ONE networks which maps fairly directly onto actual network structure, and facilities for writing and reading it to and from files as an Interlisp file type.

KL-ONE also needs an editor. The above shortfalls, combined with the lack of a KL-ONE structure editor, force users into cumbersome regimes of over-frequent loadup and/or double editing. Of course, the chief prerequisite for such an editor is a consistent rationale for structure deletion.
Finally, there is a critical problem when using KL-ONE under Interlisp-10: there isn't enough room for "real" networks. Considering that some potential KL-ONE users can anticipate being Interlisp-10 bound for two or more years into the future, it may be worthwhile to spend some energy on the problem of getting more KL-ONE into Interlisp-10.

Frank Zdybel
Martin Yonke
Norton Greenfeld
Jeff Gibbons