PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT. © ICOT, 1992

The CLASSIC Knowledge Representation System or, KL-ONE: The Next Generation

Ronald J. Brachman   Alexander Borgida*   Deborah L. McGuinness
Peter F. Patel-Schneider   Lori Alperin Resnick†
AT&T Bell Laboratories, 600 Mountain Ave.,
Murray Hill, NJ 07974-0636, U. S. A.

Abstract

CLASSIC is a recently developed knowledge representation (KR) system, based on a view of frames as structured descriptions, with several important inferable relationships, including description classification. While much about CLASSIC is novel and important in its own right, it is especially interesting to consider the system in light of its unusual (for Artificial Intelligence) intellectual history: it is the result of over a decade of research and evolution in representation systems that trace their origins back to work on KL-ONE, arguably one of the most long-lived and influential approaches to KR in the history of AI. We outline some of the novel contributions of CLASSIC, but pay special attention to its roots, illustrating the maturation of some of the original features of KL-ONE, and the decline and fall of others. A number of key ideas are analyzed—including the interpretation of frames as descriptions, the classification inference, and the role of a knowledge representation system in a knowledge-based application. The rare traceable relationship between CLASSIC and its ancestor gives us an opportunity to assess progress in a generation of knowledge representation research.

1 Introduction

An unfortunately large fraction of work in Artificial Intelligence is ephemeral, accompanied by much sound and fury, but, in the end, signifying virtually nothing. Work on systems with significant longevity to the basic ideas, such as STRIPS, appears to be the exception rather than the rule in AI.

In the area of knowledge representation (KR), there are ideas that have lived on for years, but very few systems or approaches have seen more than a minimal number of users for a minimal number of years.[1] The KL-ONE system [7, 11] is different: it was "born" over a dozen years ago, and has had continuous evolution and influence ever since. Its offspring now number at least twenty significant projects worldwide, all based directly on its key ideas of classification and structured inheritance. With well more than a decade behind us, this rich history bears closer examination, especially with the advent of the CLASSIC Knowledge Representation System, a recent development that clarifies and amplifies many of the central ideas that were more crudely approximated in the KL-ONE of 1978. CLASSIC goes substantially beyond KL-ONE in its treatment of individuals and rules, its clarification of subsumption and classification, its integration with its host language, and its concrete stand on the role of a KR system as a limited deductive database management system.

While a description of the CLASSIC system would be interesting in its own right, its motivation and contribution are more easily understood by placing it in the proper context. Thus, rather than describe the system in isolation, we here briefly explore some of its key properties in light of their intellectual debt to KL-ONE and its children. Besides making the case for CLASSIC, this will also provide us an opportunity to assess in retrospect the impact of some of the original ideas introduced by KL-ONE. This is a chance to see how far we have come in a "generation" of knowledge representation research.

2 KL-ONE: The First Generation

KL-ONE was the first implementation (*ca.* 1978) of a representation system developed in Brachman's thesis [7]. It was influenced in part by the contemporary Zeitgeist of "frames" (e.g., see [20]), with emphasis on structured objects and complex inheritance relationships. But KL-ONE's roots were really in semantic networks, and it had a network notation of labeled nodes and links.

Despite its appearance, in some key respects KL-ONE was quite different from both the semantic network systems that preceded it, and the frame systems that grew up as its contemporaries. Following papers by Woods [33] and Brachman [6], KL-ONE rejected the prevailing idea of an open-ended variety of (domain-specific) link- and node-names, and instead embraced a small, fixed set of (non-domain-specific) "epistemological primitives" [8] for constructing complex structured objects. These constructs—which represented basic general relationships like "defines-an-attribute-of" and "is-a-specialization-of," rather than domain-specific relationships like "owns" or "has-employee"—were considered to be at a higher level of representation than the data-structuring primitives used to implement them. They could be used as a foundation for building application-dependent conceptual models in a semantically meaningful way (rather than in the ad hoc fashion typical of semantic nets).

*Also with the Department of Computer Science, Rutgers University, New Brunswick, NJ.

†Electronic mail addresses: rjb@research.att.com, borgida@cs.rutgers.edu, dlm@research.att.com, pfps@research.att.com, resnick@research.att.com.

[1] SNePS and Conceptual Graphs are among the few exceptions.

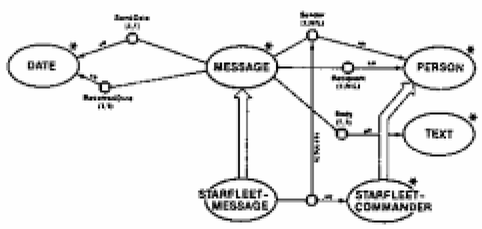

Figure 1: A KL-ONE Concept.

In addition to its clear stand on the semantics of semantic networks, the original KL-ONE introduced a number of important ideas, including these:

- rather than manipulating "slots"—which are in reality low-level data structures—KL-ONE looked at relationships as *roles* to be played; roles get their meanings from their interrelations—just like the roles in a drama—and they are not just meaningless labeled fields of records or indistinguishable empty bins into which values are dropped;
- a *role taxonomy*, which allowed roles to be subdivided into more specific roles; e.g., if `child` is a more specific role than `relative`, then being a child entails something more constrained than being a relative, but includes everything that being a relative in general does;
- *structural descriptions*, which served to define the relationships between role players; e.g., the difference between a `buyer` and a `seller` in a `PURCHASE` event would be specified by reference to other concepts that specified in which direction money and goods would flow. These concepts would give substance to the roles, rather than leaving their meanings open and subject only to human interpretation of strings like "buyer."
- *structured inheritance*, which reflected the fact that concepts (KL-ONE's name for frames/classes) were complex structured constructs and their parts were not independent items to be manipulated arbitrarily.

The KL-ONE language showed its semantic-network heritage rather directly, in that KL-ONE structures were drawn in diagrams, with different link-types being indicated with different pictorial realizations. For example, Figure 1 illustrates a typical KL-ONE concept: the "`STARFLEET-MESSAGE`" concept uses its parent, "`MESSAGE`," to create the description corresponding to "a `MESSAGE` whose `Sender` is a `STARFLEET-COMMANDER`." In general, a user built a KL-ONE net like this by calling rather low-level LISP functions, whose actions might be to "create a role node" or "add a superconcept link."

After a number of years of use and reimplementation, it gradually became clear that KL-ONE's approach to structured objects was substantially different than that of virtually all of its contemporary systems. The primary realization was that those objects had previously been used for (at least) two purposes [6, 9]: (1) to represent *statements*, usually of some typical properties (e.g., "elephants are gray"), and (2) to act as *structured descriptions*, somewhat like complex mathematical types (e.g., "a black telephone," rather than "all telephones are black"). In the KL-ONE community, the structured-description aspect came to be emphasized over the assertional one.

Viewing frames as descriptional, rather than assertional, emphasized the *intensional* aspects of knowledge representation. This had one primary benefit: it yielded the idea that the central inference to be drawn was *subsumption*—whether or not one description is necessarily more general than another. Subsumption in turn led to the idea of description *classification*—taking a description and finding its proper place in a partial order of other descriptions, by finding all subsuming (more general) descriptions and all subsumed (more specific) descriptions. KL-ONE-based classification systems were subsequently used in a number of interesting applications, including natural language understanding [11], information retrieval [27], expert systems [22], and more. Because of this view of frames, the research foci in the KL-ONE family gradually diverged somewhat from those of other frame projects, which continued to emphasize typicality and defaults.

Another key issue in the KL-ONE community has been the tension between the need for expressiveness in the language and the desire to keep implementations computationally reasonable. Two somewhat different approaches can be seen: NIKL [17], and subsequently LOOM [19], added expressive power to the original KL-ONE language, and admitted the possibility of *incomplete* classification. KRYPTON [12], and subsequently KANDOR [26], on the other hand, emphasized computational tractability and completeness. While neither of these approaches is right for every situation, they provide an interesting contrast and highlight a significant current issue in knowledge representation. This topic is still under active exploration (see Sections 4.5–4.6).

Over the last decade, systems based on the ideas in KL-ONE have proliferated in the United States and Europe (with significant ESPRIT funding), with at least twenty related efforts currently underway (see [34]). The work has also inspired seven workshops, two recently being held in Germany (in 1991) and one coming soon in the US (1992). These workshops have attracted both theoretical and practical scientists from several countries, and made it clear that the class of "KL-ONE-like" representation systems has both important theoretical substance and practical impact.

3 The CLASSIC System

The CLASSIC Knowledge Representation System[2] represents a new generation of KL-ONE-like systems, emphasizing simplicity of the description language, a formal approach, and tractability of its inference algorithms. In this regard, it is most like KANDOR (and also BACK [32]), which, while setting important directions for limited subsumption-based reasoning, had a number of inadequacies. However, the CLASSIC system goes significantly

[2] CLASSIC stands for "CLASSification of Individuals and Concepts." It has a complete, fully documented implementation in Common Lisp, and runs on SUN workstations, Apple Macintoshes, Symbolics Machines, etc. It has been distributed to numerous (> 40) universities for research use.

beyond previous description-based KR systems in many important respects, including its language, integration with the host system, treatment of individuals, and clarity on the role of a KR system.

In CLASSIC's language, there are three types of objects:

- *concepts*, which are descriptions with potentially complex structure, formed by composing a limited set of *description-forming constructors*; concepts correspond to one-place predicates;
- *roles*, which are simple formal terms for properties; roles correspond to two-place predicates; within this class, CLASSIC distinguishes *attributes*, which are functional, from *multi-roles*, which can have multiple fillers;
- *individuals*, which are simple formal constructs intended to directly represent objects in the domain of interest; individuals are given properties by asserting that they are described by concepts (e.g., "Chardonnay is a GRAPE") and that their roles are filled by other individuals (e.g., "Bell-Labs' parent-company is AT&T").

The CLASSIC description language is uniform and compositional—the meaning of a complex description is a simple combination of the meanings of its parts.[3] The complete description language grammar in Figure 2 illustrates its simplicity. Besides the description language, the interface to CLASSIC has a small number of operators on knowledge bases for the creation of new concepts (and the assignment of names to them), which include *defined concepts*, with full necessary and sufficient conditions; *primitive concepts*, which have only necessary conditions (see [9]); and *disjoint (primitive) concepts*, which cannot share instances (e.g., MALE and FEMALE). There is also an operator to explicitly "close" a role; this makes the assertion that there can be no more fillers for the role (see below).

It is important to emphasize that the description constructors and knowledge base operators were chosen only after careful study and extensive experience with numerous KR systems. For example, virtually every object-centered representation system has a way to restrict the type of an attribute; this yields our **ALL** constructor. All KR languages need to assert that a role is filled by an object; this corresponds to **FILLS**. CLASSIC's set captures the central core of virtually all KL-ONE-like systems in an elegant way: the constructors are minimal, in that one can not be reduced to a combination of others; and they have a uniform, prefix notation syntax, which allows them to be composed in a simple and powerful way. Rules (see Sec. 4.4), procedural tests, numeric ranges (**MAX, MIN**) and host language values expand the scope of KL-ONE-like concepts; these were included after clear user need was demonstrated. Certain more complex operators were excluded because they would have clearly made inference intractable or undecidable. Thus, CLASSIC's language is arguably the cleanest structured description language that tempers expressiveness of descriptions with tractability of inference (but see Section 4.5), elegantly balancing representational needs and inferential constraints in a uniform, simple, compositional framework.

[3] CLASSIC has a formal semantics, but we will not be able to elaborate on it here. See [4].

CLASSIC has many novel features, and improves on its predecessors in a number of ways, one of the most telling of which is its treatment of individuals. Anything that can be said about a concept can be said about an individual; thus, *partial knowledge* about individuals is maintained and used for inference. For example, we can assert that a person has at least three children ((**AT-LEAST** 3 child)) without identifying them, or that all of the children—whoever they are—are female ((**ALL** child FEMALE)). Individuals from the host language (e.g., LISP), such as strings and numbers, can be freely used where CLASSIC-supported individuals can, with consistent treatment. When any individual is added or augmented, or when a new concept is defined, complete propagation of properties is carried out, so that all individuals are continuously classified properly, and monotonic updates are treated completely. The role-fillers of an individual are not considered under the usual closed-world assumption; this better supports the accumulation of partial knowledge about individuals. Roles can be "closed" explicitly when all of their fillers are known. Most crucially, an individual cannot be proven to satisfy an **ALL** restriction or an **AT-MOST** restriction by looking at its fillers for the role unless all of those fillers are known. Previous systems either treated this aspect of assertions incompletely or incorrectly.

Rather than delve further into CLASSIC's individual features, we will attempt to better articulate its more general contributions by examining its relation to the issues that started this whole line of thinking over a decade ago. In that respect we can not only appreciate gains made in CLASSIC, but understand the strengths and weaknesses of the original KL-ONE proposals.

4 Key Intellectual Developments

CLASSIC is innovative in a number of ways, and bears little surface resemblance to KL-ONE. But it is also very much a descendant of that system, which introduced a number of key ideas to the knowledge representation scene. While we will not have an opportunity here to delve into all of these ideas, we will examine a few of the more important issues raised by the original system and its successors.

4.1 Subsumption as a Central Inference

In KL-ONE, as in all semantic networks that preceded it (and most systems to follow), the backbone of a domain representation was an "IS-A" hierarchy. The IS-A ("superc" in KL-ONE) link served to establish that one concept was a subconcept of another, and thus deserved to inherit all of the features of the superconcept. Virtually all of these systems forced the user to state directly that such a link should be placed between two explicitly named concepts. This type of user responsibility is still common in virtually all frame-based systems and expert system shells.

In the early 1980's we discovered that in a classification-based system this was the wrong way around. In the KL-ONE-descendant languages of KRYPTON and KANDOR, where the meaning of a concept could be determined simply and directly from its structure (be-

```
<concept-expression> ::=    THING | CLASSIC-THING | HOST-THING | <concept-name> |                 built-in names
                            (AND <concept-expression>+) |                                          conjunction
                            (ALL <role-expression><concept-expression>) |           universal value restriction
                            (AT-LEAST <positive-integer><role-expression>) |               minimum cardinality
                            (AT-MOST <non-negative-integer><role-expression>) |            maximum cardinality
                            (FILLS <role-expression> <individual-name>+) |                        role-filling
                            (SAME-AS <attribute-path><attribute-path>) |                     role-filler equality
                            (TEST-C <fn><argument>*) |                                   test (CLASSIC concept)
                            (TEST-H <fn><argument>*) |                                      test (HOST concept)
                            (ONE-OF <individual-name>+) |                                   set of individuals
                            (MIN <number>) | (MAX <number>)                                numeric range limits
<individual-expression> ::= <concept-expression> | <individual-name>
<concept-name> ::=          <symbol>
<individual-name> ::=       <symbol> | <string> | <number> | '<CommonLISP-expression>
<role-expression> ::=       <multi-role-name> | <attribute-name>
<attribute-path> ::=        (<attribute-name>+)
<fn> ::= a function in the host language (COMMON LISP) with three-valued logical return type
```

Figure 2: The CLASSIC Description Language (comments in italics).

cause the logic had a compositional semantics and necessary and sufficient definitions), it became clear that IS-A relations were *purely derivative* from the structure of the concepts. In other words, the subsumption relation[4] between two descriptions was determined without any need for a complete explicit hierarchy of IS-A connections.

Of course, it might make a difference to the efficiency of the system if all subsumption relationships that had been calculated were cached in some kind of structure that obviated the need to compute them a second time, and this is now common practice. But in a system like CLASSIC, it is clear that this is strictly an efficiency issue.

In essence, systems that force a user to think only in terms of direct IS-A links place the entire burden of knowledge structuring on that user. Since every IS-A assertion is taken at its word, the system can provide no feedback that the correct relationship has been represented; all responsibility is the user's. On the other hand, the CLASSIC system (and others like it) can reliably decide under which concepts a new concept or individual must fit, since it has a compositional interpretation of the parts of any concept. This provides valuable help to the user in structuring large knowledge bases, because it is all too easy for us to assume that just because *we* know something that a term (e.g., a complex concept, like RED-WINE) implies, the system will know it as well. This advantage has been documented in the LASSIE system [14], which uses classification to support a software information system. Systems that do not do classification do not have defined concepts, and therefore treat everything as primitive [9]. Thus we can be falsely lulled into assuming that when we assert that a particular WINE has color = Red, the system will know that it is a RED-WINE; but a non-classification system will *not* make that inference.[5]

[4]Subsumption is defined formally in [18] and [4]. Concept a subsumes concept b iff instances of b are instances of a in all possible interpretations.

[5]Note that CLASSIC and its cousins all do normal inheritance of properties. Most of these systems are strictly monotonic for simplicity, but LOOM [19] has a default component.

4.2 From LISP Functions to Languages

The realization that the structure of a concept is the only source of its meaning, and that any IS-A hierarchy is *induced* by such structures, leads to another significant point of departure for the CLASSIC system. CLASSIC has a true knowledge representation *language*—a grammar of expressions. KL-ONE and even many of its successors treated a knowledge base as a set of data structures to be more or less directly manipulated by a user, and thus the user interface was strictly in terms of node- and link-managing functions. Instead (following KRYPTON) CLASSIC is really based on a formal *logic*, with a formal syntax, rules of inference, and a formal interpretation of the syntax (see [4]).

Of all of the KL-ONE-like systems, the CLASSIC system has the cleanest language. As shown in Figure 2, the language is simple, uniform, and compositional. Figure 3 illustrates the difference in style between KL-ONE structures and the lexical language of the CLASSIC system.[6] The advantages of a true logic over a set of data-structure-manipulating programs should be obvious: one can write parsers and syntax checkers for the language, formal semantics can be specified, inference mechanisms can be verified to adhere to the semantics, etc.

4.3 Attached Procedures

One of the more popular features of the early frame systems was the ability to "attach" programs to pieces of the data structures. The ultimate incarnation of this idea was probably KRL [3], which had an elaborate process framework, including "servants," "demons," "traps," and "triggers." The program fragments could be invoked at various times, and cause arbitrary computations to occur. KL-ONE had its own elaborate procedure attachment and invocation framework. However, arbitrary access to LISP meant that KR systems with this feature ceded control completely to the user—an at-

[6]The symbols $\sqsubseteq$ and $\doteq$ indicate a *primitive concept specification* and a *defined concept specification*, respectively. The KL-ONE community has developed an algebraic notation that includes operators like these for all constructs in CLASSIC and related languages.

```
MESSAGE ⊑ (AND (AT-LEAST 1 sender)
               (ALL sender PERSON)
               (AT-LEAST 1 recipient)
               (ALL recipient PERSON)
               (AT-LEAST 1 body)
               (AT-MOST 1 body)
               (ALL body TEXT))
PRIVATE-MESSAGE ≐ (AND MESSAGE
                       (AT-MOST 1 recipient))
```

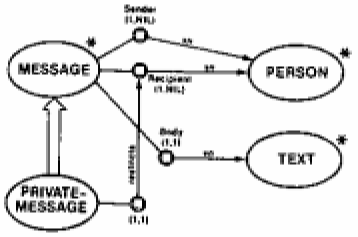

Figure 3: CLASSIC Expressions and KL-ONE Diagrams (adapted from [11]).

tached procedure could alter any data structure in any way at any time. The semantics of KL-ONE networks and other frame systems thus became very hazy once attached procedures were utilized.

In CLASSIC, we have invented an important way to control the use of such "escape hatches." Through the notion of the **TEST-C** and **TEST-H** constructors, we have isolated the use of procedures in the host language to testing predicates. As one can see from the grammar, such concepts are treated syntactically uniformly with other concepts. The procedure simply provides a *primitive sufficiency condition* for the concept—it will be invoked only when trying to recognize an instance. These test functions are particularly useful when trying to relate individuals from the host language, such as when two roles are filled with numbers, and one should be a multiple of another. In their use, the user agrees to avoid side-effects and to use only monotonic procedures (i.e., those whose value never changes from true to false or *vice versa* in the presence of purely monotonic updates). While under arbitrary circumstances, resorting to program code for tests renders the semantics of the language useless, in CLASSIC, if the user abides by this "contract," the semantics of concepts with tests is manageable, and the inferences that the system draws are still guaranteed to be sound. Indeed, tests work just like other restrictions on concepts as far as classification of individuals goes, but since the procedures are inscrutable they have the flavor of primitive concepts. While primitive concepts allow primitive necessary conditions, tests give us primitive sufficient conditions.

Another innovation in CLASSIC is the requirement that the test functions must be 3-valued. If a system like CLASSIC says that an individual does not satisfy a concept, then that means only that it cannot be currently *proven* to do so. A complementary question can still be asked—whether it can be proven that the individual could *never* satisfy the description (i.e., that it is disjoint from the concept). For example, if `Fred` has exactly one child (i.e., (**AND** (**AT-LEAST** `1 child`) (**AT-MOST** `1 child`))), but nothing is known about it yet, then he cannot be proven to satisfy the description (**ALL** `child FEMALE`). But it is possible that at a later time he could be, if he were stated to have a known female child. On the other hand, if it were asserted that his child was `Barney`, who was known to be a `MALE`, and `MALE` and `FEMALE` were disjoint concepts, then it would be provable that `Fred` could never satisfy the description. Thus, in order to fit into the classification framework, procedural tests must provide the same facility—to differentiate between a guarantee never to satisfy a description and lack of ability to prove it given the current knowledge base.

4.4 Definitions, Assertions, Individuals

As mentioned, KL-ONE ultimately distinguished itself from other frame languages by its emphasis on structured descriptions and their relationships, rather than on contingent and typical facts. At one point in its development, the system was in a strange state: there were facilities for building complex concepts, but none for actually using them to describe individual objects in the domain. "Individual concepts" were KL-ONE's initial attempt to distinguish between generic class descriptions and descriptions that could apply only to single individuals. As it turned out, these were typically misused: an individual concept with two parent concepts could only really mean a conjunctive description. One example that was used often was the conjunction of `DRIVING-IN-MASSACHUSETTS` and `HAZARDOUS-ACTIVITY`, intended to express the fact that driving in Massachusetts is hazardous. However, in truth the concept including them both was just a compound concept with no assertional force at all.

While KL-ONE initially correctly distinguished between the import of different links between concepts, it failed to distinguish between those and a link that would make a contingent assertion about some individual. Eventually an alternative mechanism was proposed—the "nexus," to stand for an individual—but this was never really used. In the end, it took the work on KRYPTON to get this right. In KRYPTON, it was proposed that *terminological* knowledge (knowledge about the structure of descriptions) and *assertional* knowledge (facts) are two complementary aspects of knowledge representation competence, and that they should be maintained by distinct components, with an appropriate logical connection between them. From this distinction arose the terms "TBox" and "ABox," which are used extensively in the KL-ONE community to refer to the two components.

But KRYPTON went too far in another direction, integrating an entire first-order logic theorem-prover as its assertional component. The CLASSIC system makes what we think is a better compromise: it has a limited object-centered logic that properly relates descriptions and individuals. As is apparent from the grammar, CLASSIC treats assertions about individuals in a parallel and uniform manner with its treatment of the formation of subconcepts; but it also carefully distinguishes the logical meaning of the different relationships. Thus, for example, while individuals can be used in concept value restrictions (i.e., in a **ONE-OF** expression, e.g., (**ALL** `wine-color` (**ONE-OF** `Red White Blush`))), no contingent property of an individual can be used in determining subsumption between two concepts (e.g., if `White` just happens to be my favorite color for a wine, that fact cannot be used in any subsumption inference).

As mentioned, CLASSIC also supports the propagation of information between individuals. If we assert that some individual is described by a complex description (e.g., that Rebecca is a PERSON whose mother is a DOCTOR), then that may imply some new properties about other related individuals (e.g., we should assert that Rebecca's mother, if known, is a DOCTOR). Such propagated properties can in turn cause other properties to propagate (e.g., that Rebecca's mother's office is a DOCTOR'S-OFFICE).[7] This type of inference was never handled in KL-ONE, and only partially handled in some of its successors. Note that as soon as a property propagates from one individual to another, the latter individual might now fall under some new descriptions. CLASSIC takes care of this *re-classification* inference as well (as well as any further propagations that result, etc.).

The CLASSIC system has two other features along these lines that distinguish it from its predecessors. First, the previously mentioned apparatus does not allow the expression of general contingent rules about individuals. Thus, given only what is in the CLASSIC concept grammar, while we could form the concept of, for example, a LATE-HARVEST-WINE, we could not assert that all LATE-HARVEST-WINEs are SWEET-WINEs. The sweetness is a derivative property—it is not part of the *meaning* of LATE-HARVEST-WINE, but rather a simple contingent property of such wines. In CLASSIC, one can also express general rules of a simple form. A rule has a named concept as the left-hand side, with an applicability condition (filter) that limits the rule's firing to the desired subcases (i.e., if x is a $<concept_a>$ with property $<filter>$, then x is a $<concept_b>$). These rules are used only in reasoning about individuals, and do not affect subsumption relationships.[8]

Most KL-ONE-like systems were unclear about the status of individuals that could easily be expressed in the host implementation language (i.e., numbers and strings in LISP). CLASSIC integrates such individuals in a simple and uniform way, and makes it virtually transparent whether an individual is implemented directly in the host language, or in the normal complex structure for CLASSIC individuals. This aspect of CLASSIC has proven critical in applications that deal with real data (for example, from a database), as in [29].

4.5 KR and Computational Complexity

Once it was apparent that the clearly defined logical relationship of subsumption was central to the KL-ONE family, a new factor could be introduced to the analysis of frame-based knowledge representation systems. In 1984, Brachman and Levesque gave a formal analysis of the complexity of computing subsumption in some frame languages [10]. That analysis showed that the apparent simplicity of some frame languages could be deceptive, and that the crucial subsumption inference was co-NP-hard. The original paper initiated a sequence of results on the complexity of computing in the KL-ONE family, culminating most recently in two that show that the original language is in fact *undecidable* [24, 28].

This line of analysis has caused some major rethinking of the knowledge representation enterprise. No longer can we view language features as simply providing more expressiveness (which was the common view in the early years of knowledge representation). Rather, as in other areas of computer science, we must consider how expensive it will be to add a feature to a language. The addition of new features may demand the excision of some others in order to maintain computational manageability, or the system must be clear on where it is incomplete. In CLASSIC, subsumption is complete and tractable, but with respect to a slightly non-standard semantics; that is, it is clear what CLASSIC computes, and how fast it can compute it, but it does not compute all the standard logical consequences of a knowledge base. In this regard, we have opted for a less conservative approach than in KANDOR, but a more limited and disciplined approach than in LOOM. The consequences of this are explored briefly in the next section. We should point out that the viability of our approach has been proven in practice: CLASSIC is the first KL-ONE-derived system to be deployed in a fielded (AT&T proprietary) product, used every day in critical business operations. It was expressive enough to do the job.

4.6 The Role of a KR System

The above developments in the KL-ONE saga give rise to an important general question that usually goes unasked in AI: what role is a knowledge representation system expected to play? There are clearly different approaches here. On one extreme we have the large commercial systems, or expert system shells, which include substantial knowledge representation apparatus. The philosophy of those systems seems to be that a KR system should provide whatever apparatus is necessary to support virtually *any* AI application. In that regard, such systems are like very powerful programming languages, with complex data-structuring facilities.

But this is definitely not the only approach, and in many respects its requirements are overly demanding. Given the kind of complexity results mentioned above, users of such powerful systems must be very careful in "programming" their KR tools: predicting when a computation will return is difficult or impossible in a very expressive logic.

In many contexts (but not all, of course), it may be appropriate for a knowledge representation system to act in a more constrained fashion, rather like the database component of an application system. This is the point of view explicitly espoused in CLASSIC. Users cannot expect to program arbitrary computations in CLASSIC, but in return they get predictable response time and clear semantics. The burden of programming an application, such as a medical diagnostician, must be placed on some other component of the overall system. Since most KR systems attempt to be application-independent, it is ap-

[7] In order to keep the complexity down, CLASSIC only propagates properties to known individuals. Thus, if Rebecca's mother were unknown, the system would not attempt to create an individual about which to assert the DOCTOR description. If it did, it would then have to do very complex reasoning about existentials.

[8] Some of the newer KL-ONE-derivatives, such as LOOM, have developed similar rule mechanisms.

propriate for them not to be asked to provide general diagnostic, planning, or natural language-specific support. What is gained in return for certain limitations (and this in part accounts for the appeal of databases) is a system that is both complete with respect to an intuitive and simple semantic model and efficient to use.

Failure to acknowledge this general issue has been a source of difficulty with knowledge representation systems in AI. KL-ONE, uniformly with its contemporary KR systems (and subsequently NIKL), never really took a stand as to the role it should play. This has resulted, for example, in a pair of recent critiques of NIKL [15, 30], for failing to live up to a promise it perhaps was never intended to make. With CLASSIC, on the other hand, we expect to provide a powerful database service, but with limited deductive and programming support. This is a unique kind of database service, as it is both deductive and object-oriented (see [5]). But nevertheless it is firmly limited. To use the CLASSIC system in the context of an expert system, for example, it would be appropriate to use it as a substitute for working memory in a rule-based programming system like OPS5, not for all computation to be done by the overall system. Several recent applications ([14], [29], [23], and others) have shown convincingly that this approach, while not satisfying all needs for all applications, is quite successful in important cases.

5 Perspective

While CLASSIC is a "KL-ONE-like" system, it differs in so many ways from the original that it must be treated in its own right. While KL-ONE began the thinking on numerous key issues, it has taken us until CLASSIC to begin to truly understand many of them. Among its virtues, the CLASSIC Knowledge Representation System

- isolates an important set of language constructs, distilled from many years of use of frame representations, and knits them together in an elegant, straightforward language with a compositional interpretation; novel language features include enumerated sets of individuals treated in a uniform manner with other concepts (**ONE-OF**), and limited generic equalities between role fillers (**SAME-AS**);
- treats individuals in a more complete way than its predecessors, supporting propagation of facts and reclassification of individuals;
- allows contingent universal rules that are automatically applied, with the affected individuals being reclassified and any derived facts being propagated;
- offers tight, uniform integration of individuals from the host language, including numeric range concepts (**MAX**, **MIN**);
- offers a facility for writing procedural 3-valued tests as primitive sufficiency conditions, and integrates such tests into the language and semantics in a clean way.[9]

[9] CLASSIC also allows retraction of any asserted fact, with full dependency maintenance, but we have not had room to discuss this here.

CLASSIC offers these facilities in the context of complete computation of subsumption, while remaining computationally tractable. The CLASSIC system can be thought of as a limited, deductive, object-oriented database management system as well as a knowledge representation system, and has been used to support several real-world applications.[10]

In this discussion, we have limited ourselves to considering the KL-ONE family and its contributions. Related work involving manipulation of types and their relations can be found in programming language research, in some semantic data modeling work, and in feature logics in support of (among other things) natural language processing. We do not have room to draw comparisons with this other work, but in general it is clear that the bulk of that work does not include classification and description-processing of the sort found so prevalently in KL-ONE-like systems. Recent work in some of these areas does bear a strong relationship to ours, but not by accident: work on KL-ONE and its descendants has had direct influence, for example, on LOGIN [1] (a programming language), CANDIDE [2] (a DBMS), and feature logics [21].

There are still, of course, many open questions yet to challenge CLASSIC and its relatives. Technically, the notion of a "structural description," introduced by KL-ONE, has still not been treated adequately (although the **SAME-AS** construct provides a limited form of relationship between roles). And there are important computational questions to be answered so that CLASSIC can handle significant-sized databases, involving persistence of KB's, automatic loading of data from conventional DBMS's, and complex query processing.

But perhaps chief among the remaining research questions is how exactly to cope with the tradeoff we are forced to make between expressive power and computational tractability. Is it even possible to provide the kind of knowledge representation and inference services demanded by AI applications in a computationally manageable way? The CLASSIC Knowledge Representation System has provided convincing evidence that this is possible at least for a limited set of applications, but it is but one point in a large space of possibilities that we are still mapping out, after more than a dozen years of research inspired by KL-ONE.

References

[1] Aït-Kaci, H., and Nasr, R. "LOGIN: A Logic Programming Language with Built-in Inheritance," *Journal of Logic Programming*, **3**:187–215, 1986.

[2] Beck, H. W., Gala, S. K., and Navathe, S. B. "Classification as a Query Processing Technique in the CANDIDE Data Model," *Proc. Fifth Intl. Conf. on Data Engineering*, Los Angeles, 1989, pp. 572–581.

[3] Bobrow, D. G., and Winograd, T. A., "KRL, A Knowledge Representation Language," *Cognitive Science* **1**(1), 1977, pp. 3–46.

[10] One testimony to the success of CLASSIC's clean and simple approach is the fact that a group from the University of Calgary has simply picked up a written description of the system and quickly implemented their own version as a C++ library to support their work in knowledge acquisition [16].

[4] Borgida, A., and Patel-Schneider, P. F., "A Semantics and Complete Algorithm for Subsumption in the CLASSIC Description Logic," unpublished manuscript, AT&T Bell Laboratories, Murray Hill, NJ, 1992. Submitted for publication.

[5] Borgida, A., Brachman, R. J., McGuinness, D. L., and Resnick, L. A., "CLASSIC: A Structural Data Model for Objects," *Proc. 1989 ACM SIGMOD Intl. Conf. on Management of Data*, Portland, Oregon, June, 1989, pp. 59–67.

[6] Brachman, R. J., "What's in a Concept: Structural Foundations for Semantic Networks," *Intl. Journal of Man-Machine Studies*, **9**(2), 1977, pp. 127–152.

[7] Brachman, R. J., "A Structural Paradigm for Representing Knowledge," Ph.D. Thesis, Harvard University, Division of Engineering and Applied Physics, 1977. Revised as *BBN Report No. 3605*, Bolt Beranek and Newman, Inc., Cambridge, MA, May, 1978.

[8] Brachman, R. J., "On the Epistemological Status of Semantic Networks." In *Associative Networks: Representation and Use of Knowledge by Computers*. N. V. Findler (*ed.*). New York: Academic Press, 1979, pp. 3–50.

[9] Brachman, R. J., " 'I Lied about the Trees,' or, Defaults and Definitions in Knowledge Representation," *AI Magazine*, Vol. 6, No. 3, Fall, 1985.

[10] Brachman, R. J., and Levesque, H. J., "The Tractability of Subsumption in Frame-Based Description Languages," *Proc. AAAI-84*, Austin, TX, August, 1984, pp. 34–37.

[11] Brachman, R. J., and Schmolze, J. G., "An Overview of the KL-ONE Knowledge Representation System," *Cognitive Science*, **9**(2), April–June, 1985, pp. 171–216.

[12] Brachman, R. J., Fikes, R. E., and Levesque, H. J., "Krypton: A Functional Approach to Knowledge Representation," *IEEE Computer*, Vol. 16, No. 10, October, 1983, pp. 67–73.

[13] Brachman, R. J., McGuinness, D. L., Patel-Schneider, P. F., Resnick, L. Alperin, and Borgida, A. "Living with CLASSIC: How and When to Use a KL-ONE-like Language." In *Principles of Semantic Networks*. J. Sowa (*ed.*). San Mateo, CA: Morgan Kaufmann, 1991, pp. 401–456.

[14] Devanbu, P., Brachman, R. J., Selfridge, P. G. and Ballard, B. W., "LaSSIE: A Knowledge-Based Software Information System," *CACM*, Vol. 34, No. 5, May, 1991, pp. 34–49.

[15] Doyle, J., and Patil, R. S., "Two Theses of Knowledge Representation: Language Restrictions, Taxonomic Classification, and the Utility of Representation Services," *Artificial Intelligence*, Vol. 48, No. 3, April, 1991, pp. 261–297.

[16] Gaines, B. R., "Empirical Investigation of Knowledge Representation Servers: Design Issues and Applications Experience with KRS," *SIGART Bulletin*, Vol. 2, No. 3, pp. 45–56.

[17] Kaczmarek, T. S., Bates, R., and Robins, G., "Recent Developments in NIKL," *Proc. AAAI-86*, Philadelphia, PA, 1986, pp. 978–985.

[18] Levesque, H. J., and Brachman, R. J., "Expressiveness and Tractability in Knowledge Representation and Reasoning," *Computational Intelligence*, Vol. 3, No. 2, Spring, 1987, pp. 78–93.

[19] MacGregor, R. M., "A Deductive Pattern Matcher," *Proc. AAAI-87*, St. Paul, MN, pp. 403–408.

[20] Minsky, M., "A Framework for Representing Knowledge." In *The Psychology of Computer Vision*. P. H. Winston (*ed.*). New York: McGraw-Hill Book Company, 1975, pp. 211–277.

[21] Nebel, B., and Smolka, G., "Attributive Description Formalisms ... and the Rest of the World." In *Text Understanding in LILOG*. O. Herzog and C.-R. Rollinger (*eds.*). Berlin: Springer-Verlag, 1991, pp. 439–452.

[22] Neches, R., Swartout, W. R., and Moore, J., "Enhanced Maintenance and Explanation of Expert Systems Through Explicit Models of Their Development," *Proc. IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, CO, 1984, pp. 173–183.

[23] Nonnenmann, U., and Eddy, J. K., "Knowledge-Based Functional Testing for Large Software Systems," *Proc. FGCS-92, Intl. Conf. on Fifth Generation Computer Systems*, Tokyo, June, 1992.

[24] Patel-Schneider, P. F., "Undecidability of Subsumption in NIKL," *Artificial Intelligence*, Vol. 39, No. 2, June, 1989, pp. 263–272.

[25] Patel-Schneider, P. F., "A Four-Valued Semantics for Terminological Logics," *Artificial Intelligence*, Vol. 38, No. 3, April, 1989, pp. 319–351.

[26] Patel-Schneider, P. F., "Small can be Beautiful in Knowledge Representation," *Proc. IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, CO, December, 1984, pp. 11–16.

[27] Patel-Schneider, P. F., Brachman, R. J., and Levesque, H. J., "ARGON: Knowledge Representation meets Information Retrieval," *Proc. First Conf. on Artificial Intelligence Applications*, Denver, CO, December, 1984, pp. 280–286.

[28] Schmidt-Schauss, M., "Subsumption in KL-ONE is Undecidable," *Proc. KR '89: The First Intl. Conf. on Principles of Knowledge Representation and Reasoning*, Toronto, May, 1989, pp. 421–431.

[29] Selfridge, P. G., "Knowledge Representation Support for a Software Information System," *Proc. Seventh IEEE Conf. on Artificial Intelligence Applications*, Miami Beach, FL, February, 1991, pp. 134–140.

[30] Smoliar, S. W., and Swartout, W., "A Report from the Frontiers of Knowledge Representation," Technical Report, USC Information Sciences Institute, Marina del Rey, CA, 1988.

[31] Vilain, M., "The Restricted Language Architecture of a Hybrid Representation System," *Proc. IJCAI-85*, Los Angeles, 1985, pp. 547–551.

[32] von Luck, K., Nebel, B., Peltason, C., and Schmiedel, A., "The Anatomy of the BACK System," *KIT Report 41*, Technical University of Berlin, January, 1987.

[33] Woods, W. A., "What's in a Link: Foundations for Semantic Networks." In *Representation and Understanding: Studies in Cognitive Science*. D. G. Bobrow and A. M. Collins (*eds.*). New York: Academic Press, 1975, pp. 35–82.

[34] Woods, W. A., and Schmolze, J. G., "The KL-ONE Family," to appear in *Computers and Mathematics with Applications*, Special Issue on Semantic Networks in Artificial Intelligence.