Knowledge Interchange Format: The KIF of Death

Matthew L. Ginsberg

Computer Science Department
Stanford University
Stanford, California 94305
ginsberg@cs.stanford.edu

Knowledge Interchange Format:
The KIF of Death

Abstract

There has been a good deal of discussion recently about the possibility of standardizing knowledge representation efforts, including the development of an interlingua, or KIF (knowledge interchange format) that would allow developers of declarative knowledge to share their results with other AI researchers.

In this paper, we examine the practicality of this idea. We present some philosophical arguments against it, describe a straw man KIF, and suggest specific experiments that would help explore these issues.

1 Introduction

There has been a flurry of interest recently in the possibility of standardizing existing work on knowledge representation, supported by DARPA and other funding agencies. One focus of this effort has been the development of an interlingua, or KIF (Knowledge Interchange Format) by which knowledge could be moved from one declarative system to another.

This paper will touch on a variety of issues. In Section 2, we discuss general concerns about both the practicality of developing a KIF and about the effect that a partially successful development might have on the knowledge representation community generally. Section 3 attempts to address the technical concerns raised, presenting a straw man KIF proposal that may provide a way around these difficulties. In Section 4, we discuss a specific translation experiment that will, we hope, settle some of the questions raised by Sections 2 and 3. Finally, Section 5 discusses the question of whether the development of a KIF should be viewed as a standardization or as a research effort, and the consequences of the natural conclusion.

2 Philosophical arguments

An examination of recent work on knowledge representation makes it clear that there are deep differences between the approaches taken by one researcher and those taken by others, and it is these differences that underlie our expectation that attempts to translate automatically between one knowledge representation scheme and another are premature. [1]

[1] These differences make the interlingua effort quite different from typical standardization efforts in computer science (e.g., standardization of programming languages). DARPA is aware of this and has discouraged any analogy between the development of KIF and the development of a language such as Common Lisp.

The interlingua group is attempting to address this difficulty by creating a single language in which all of the knowledge representation schemes can be expressed [2], but this seems impossible given the current state of the field. Perhaps it is possible to construct a single language that can capture all *existing* KR schemes (although we expect that such a language would be an unmanageable Hydra); it is surely not possible to construct a language that will also incorporate all *future* KR work, other than in the trivial sense guaranteed by the universality of some specific method such as either first-order logic or a general-purpose programming language. Furthermore, attempts in this direction will inevitably constrain future KR efforts; even very gentle constraints may have a stifling impact on future KR work. This is especially true given the facts that it is to our funders' advantage to encourage the emergence of a standard (even a premature one, sad to say), and that increasingly tight AI funding will compel researchers to conform to any perceived standard that does exist.

But let me return to my remarks regarding the universality of some specific methods, such as first-order logic or a general-purpose programming language. Why wouldn't one of these methods serve as a KIF?

With regard to the use of a programming language as a KIF, this is little more than the suggestion that researchers exchange knowledge by swapping both the knowledge bases that they use and the programs that manipulate these knowledge bases. This sort of "exchange" is too limited to be of interest; the aim of the standardization effort is to allow the inference tools used by one group direct access to the knowledge developed by another group.

The argument against using first-order logic as a KIF is rather different. The reason is not that it will be impossible to translate from any specific KR formalism *into* a universal language such as first-order logic, but that connections to special-purpose inference mechanisms (semantic networks or relevance logic, for example) are likely to be lost by so doing. As an example, a translation into first-order logic of a probabilistic database will of necessity include an axiomatization of probability itself. Many KR schemes involve other extensions to first-order logic such as nonmonotonic or limited-resource reasoning, and so on. Again, since the intent of the KIF is to allow users of one language (and the associated inference mechanism) access to the knowledge encoded by others, this problem needs to be addressed.

Some of these extensions are treated in [2] and some are not. The point, however, is that the treatments that do exist are – and inevitably will be – hopelessly idiosyncratic. The presentation in [2] deals with nonmonotonic reasoning using ideas that are less than a year old and deals with quotation using techniques that have never even appeared in the literature. I am not saying that these ideas are without merit, only that their inclusion in a standard is obviously premature.

We will return to this issue in Section 5, where we discuss the question of whether the development of a KIF is a research or standardization venture. In the meantime, our own approach to the idiosyncracies of [2] will be to describe a language that refuses to commit on any topic that is the focus of current KR research.

We will do this by acknowledging that different development groups will use different KR schemes, and that if they are to exchange their information with other researchers, they will need some way to strip off their conventional operators or methods. Ideally, however, backward pointers will be retained so that groups that share ideas can recover one another's original formulations. Our proposed interlingua is therefore a minimalistic language that is nearly equivalent to first-order logic, with specific notational conventions introduced to allow users of special-purpose methods to include them.

Note also that adoption of this *sort* of a language – one with a facility that allows individual users to extend it as necessary – will address our concerns regarding an interlingua's negative impact on the field as a whole, since novel ideas can be developed and included freely in the overall language. All sorts of approaches to KR problems are encouraged, not just the specific ones that happened to be selected by the designers of the KIF. The minimality of the core language prevents its being used as an overall "standard" and it is clear from the outset that no lofty goals are being set.

3 An interlingua proposal

As we have discussed, there are two approaches that one might take to the problem of anticipating in a KIF all of the possible methods that might be used by developers of special-purpose knowledge engineering techniques. On the one hand, one might try to design a KIF that anticipated special-purpose techniques by including specific facilities to describe existing approaches; on the other hand, one might accept special-purpose methods as currently outside the scope of information that can be legitimately exchanged, and resign oneself to the ensuing inaccuracies in exchanged information.

This paper embodies a strong commitment to the second of these approaches. As already argued, the reason is simply that the state of the art in KR work does not support any other option.[2] But given that the KIF cannot contain a declarative description of special-purpose techniques, what *should* it contain? We suggest the following:

1. The basic KIF language should be one accepted by the entire KR community. This is to ensure that any recipient of a KIF database can do something useful with the knowledge he receives.

2. It should be possible to extend the language as the KR community comes to accept more things as "standard." In addition, if some subgroup comes to share specific understandings, they should be able to include these understandings in their KIF databases, although this inclusion should be transparent to any users that do not share them.

The way in which we propose to do this is as follows:

1. The basic KIF language will be first-order predicate calculus, with specific notational conventions to handle variables, quantification, and so on. This basic language is described in Section 3.2. Note that we are subscribing to the *syntax* of first-order logic only, and not necessarily to its semantics.

[2]For what it's worth, this is not only my opinion as author, but was the opinion expressed by the attendees at the 1991 International Conference on Knowledge Representation and Reasoning. We will return to this issue in Section 5.

2. All KIF documents will include information indicating the KR commitments made by the underlying system. Any recipient who can support all of the commitments made by the sender is assured that no information has been lost in the transmission of the system; recipients who cannot support all of the commitments may receive approximate information in some sense. See Section 3.3.

3. The additional information passed by KIF documents making specific KR assumptions is clearly separated from the declarative information that is also being passed. This is to ensure that a recipient not sharing these commitments retains partial access to the knowledge being sent. The syntax used for the additional information is described in Section 3.3.

3.1 Overall syntax

A KIF document is a sequence of ASCII characters that can be processed by a LISP reader; we will concern ourselves with the results returned by the reader instead of the character stream itself, viewing a KIF document as a stream of s-expressions. The following s-expressions have a special meaning in KIF:

1. Any keyword (i.e., any top-level s-expression that is an atom beginning with a `:`) *other than* `:label`, `:cancel` and `:nonsemantic` is the name of a KR commitment made by the system. The next s-expression should be used to convey additional information as appropriate (examples are in Section 3.3). These keyword/s-expression pairs will typically contain system-dependent information concerning subsequent sentences.

2. `:label` should be followed by an atom that labels the last keyword encountered, and also indicates that this keyword is "sticky" in that it applies to all subsequent sentences until cancelled. `:cancel` should also be followed by an atom and indicates that the keyword information so labelled should not be applied to the following sentences. Once again, examples are in Section 3.3.

3. `:nonsemantic` indicates that the most recent keyword has no semantic content (e.g., control information; see Section 3.3 for an example). This is an indication that a system that doesn't understand the keyword is still safe in using the information.

All other s-expressions are interpreted as basic KIF sentences with a declarative semantics; these are described in the next section.

3.2 Basic KIF

The legal sentences in basic KIF are precisely those of first-order logic, expressed in prenex normal form. In order to make this precise, we need to specify the fashion in which variables are distinguished from constants, and to specify alphanumeric replacements for symbols such as $\neg$, $\forall$ and $\exists$.

Variables Any atom beginning with the character `?` is a variable. We also support the use of *sequence* variables, which are variables that can match arbitrary sequences of atoms instead of just single ones; there is substantial evidence from the PROLOG community that such variables are useful in the construction of declarative databases.

Unlike PROLOG, we do not assume either that sequence variables appear only at the end of argument lists or that only one sequence variable appears in any particular expression. Although this makes unification more difficult than it typically is, it is possible to construct a unifier that handles this more general problem and only incurs significant computational overhead when attempting to unify expressions that actually *do* contain multiple sequence variables. This unifier is described in [3] and is available via anonymous ftp from t.stanford.edu.

Any variable whose second character is `*` is a sequence variable; all other variables are not. Thus `?*` and `?*x` are examples of sequence variables; `?` and `?x` are nonsequence variables. The character `*` was chosen because of the obvious analogy with BNF expressions; we have not used a distinct initial character for sequence variables because it is important that all variables have a uniform and easily identified representation.

Connectives The replacement of connectives with alphanumeric expressions is summarized by the following table of standard logical sentences and their KIF equivalents:

Standard	KIF
$\neg p$	`(not p)`
$p_1 \wedge \ldots \wedge p_n$	`(and p1 ... pn)`
$p_1 \vee \ldots \vee p_n$	`(or p1 ... pn)`
$p \supset q$	`(if p q)`
$p \leftrightarrow q$	`(iff p q)`
$\forall xy.p(x,y)$	`(forall (?x ?y) (p ?x ?y))`
$\exists xy.p(x,y)$	`(exists (?x ?y) (p ?x ?y))`

The first argument to **`forall`** and **`exists`** can be an atom if only a single variable appears under the scope of the quantifier.

3.3 Semantics

The syntax of KIF is as given in the preceding two sections; s-expressions following keywords have no restrictions (hence the inherent flexibility of this proposal), while the syntax of other sentences is dictated by the rules of first-order logic.

In the absence of information to the contrary, the semantics of a KIF database are also given by the rules of first-order logic; the extension facility describes situations in which first-order logic is inappropriate for some system-dependent reason.

The form of the additional information used by a KIF database is

```
:extension s-exp
```

where `:extension` is the name of a particular extension to first-order logic and `s-exp` is additional information that is to be used when considering the subsequent sentence or sentences.

It cannot be within the scope of any specific KIF effort to delimit the list of possible extensions; although we are about to give a variety of examples showing the flexibility of the scheme, it should not even be in the scope of this effort to *propose* such extensions. Rather, we expect that these extensions will be developed by the designers of the various knowledge bases themselves; as common agreements are reached by these designers, they will settle on specific extensions that sanction them.

In the remainder of this section, we will consider the use of our extension facility to treat four separate semantic extensions: definitions, probabilities, procedural attachments and relevance logic [1]. We have attached `mlg` to the extension names in order to indicate that they are mere trial balloons that might be supported by only a single researcher. In some cases, we would hope that the community interested in the ideas (probabilities, for example) would rapidly settle on an agreed language that would describe their shared semantic commitments.

Definitions A KIF document that contained definitions might look like this:

```
:definition-mlg  predicate
:nonsemantic
:label  d001

(forall ?x (iff (bachelor ?x) (and (single ?x) (male ?x))))
(forall ?x (iff (old-maid ?x)
                (and (single ?x) (female ?x) (old ?x))))

:cancel d001
```

The `definition-mlg` keyword indicates that the following sentences are to be interpreted according to a particular set of rules that are agreed upon by all users of the definition-mlg "package." In this particular case, the following atom, `predicate`, indicates that the rules are being used to define predicates – `bachelor` and `old-maid` in the example. The label `d001` is used to delimit the scope of the definitional declaration.

Any system that recognizes the `definition-mlg` commitments is now free to interpret the above two rules as definitions, presumably obtaining some computational benefit as a result. A system that doesn't recognize these commitments will still be able to make sense of the rules, since the `nonsemantic` keyword indicates that the keyword doesn't affect the *meaning* of the following information. Other control information (indicating that a certain rule is to be used for backward-chaining, for example) can be handled similarly.

Probabilities The interesting extensions are those that actually change the semantics of the knowledge in some way. Here is one:

```
:probability-mlg  .75
(flies Fred)

:probability-mlg  .80
(forall ?x (if (bird ?x) (flies ?x)))

(forall ?x (if (ostrich ?x) (bird ?x)))
```

This database tells us that Fred flies with probability 0.75, that birds fly with probability 0.8, and that ostriches are birds. The interpretation of the sentence, "Birds fly with probability 0.8" is up to the conventions of `probability-mlg`; this particular system might assume that the probability actually labels the entire sentence, or might strip off the leading quantifier and interpret the result as a conditional probability.

Now imagine that this knowledge base is sent to a system that has no probabilistic facility. The recipient, upon seeing the `probability-mlg` keyword, will presumably realize that the database he is receiving uses probabilistic information. He can then do one of three things:

1. Accept the probabilistic information as valid, concluding above that Fred flies and that birds do. The resulting database is likely to contain information not intended by the sender, but may nevertheless be the most attractive option available.

2. Ignore the probabilistic information, retaining only the statement that ostriches are birds. Only knowledge thought by the sender to be valid is used, but the overall value of the information received may be reduced.

3. "Subscribe" to `probability-mlg` in some sense. This does *not* imply the use of a full probabilistic reasoning facility, only that some sense is made of the additional information being received. Perhaps any sentence stated with probability in excess of some threshold should be accepted, and the rest ignored. Perhaps a default reasoning facility will be used instead, and so on.

The point, of course, is that the recipient of the system is able to interpret the incoming information in a convenient and rational fashion. Note, incidentally, that the first two options described above (accept or ignore suspect sentences) can be done automatically; the third requires some sort of human intervention.

Many other existing declarative schemes can be handled similarly. ATMS's, for example, need to label sentences as assumptions or otherwise; nonmonotonic reasoning schemes typically need to label sentences as defaults.

Procedural attachment and relevance logic Suppose that we want to evaluate the truth of the predicate `subsetp` using LISP as opposed to inferentially. We might write this as:

```
:attach-mlg  subsetp
:label a001
```

This presumably means that `subsetp` is procedurally attached. Since the label is never cancelled, all of the subsequent information might be affected. Or, we might have

```
:attach-mlg  lisp!
:label a002
```

indicating that any LISP predicate without side effects is procedurally attached in this way.

Once again, other examples are similar. If we intend that the database be interpreted using Belnap's four-valued relevance logic [1] instead of conventional methods, we might write

```
:relevance-ndb  nil
:label b001
```

A modal operator of knowledge with an $S5$ semantics might lead us to write

```
:modal-sak  (L S5)
:label s001
```

while if L is described truth-functionally instead, we would have

```
:modal-mlg  (L lisp-fn)
:label g001
```

where `lisp-fn` is the function that actually does the computation. If mlg and sak manage to resolve their differences, they might settle on a single package `:modal`.

3.4 Summary

Standards are appropriate only where there is consensus. The aim of the interlingua that we have proposed is to standardize on what really is standard – the fact that existing declarative databases are largely, but not wholly equivalent to collections of first-order sentences. Our inability to transmit declarative information between large systems may reflect more the fact that there is no common notation than the existence of fundamental differences between the systems being used, and we have attempted to provide a way around this difficulty.

We have *not* tried settle the differences that do exist, since these are not yet ripe for standardization. Instead, we have provided a general way for users to include their nonsemantic knowledge when transmitting declarative databases. We have shown examples of the translation of several non-standard KR mechanisms using our methods, and hope that this approach will encourage the emergence of common understandings in these areas from the knowledge representation and engineering community itself.

4 An experiment

But will it work? Is it reasonable to expect it to be possible to transmit *useful* knowledge from one group to another using our interlingua or any other, or are there simply too many special-purpose idiosyncrasies to which any practical system is committed?

The developers of the hypothetical KIF standard [2] have proposed answering this questioning by using their standard to translate a declarative knowledge base from one special-purpose language to another. This "experiment" proves nothing, however – the same group is translating both into and out of the KIF!

To see why this assumption invalidates the KIF experiment, suppose that we take a semantic network language, translate it into an interlingua, and then translate back into semantic networks. Presumably the first translation will produce sentences such as

```
(subclass ostrich bird)
```

that will need to be retranslated into a semantic network.

But how is an automatic translation mechanism to recognize the `subclass` predicate? What if `is-a` is used instead? Or is the translator into the interlingua expected to provide a declarative *description* of the predicates used (e.g., that `subclass` is transitive and so on), and the translator out is supposed to recognize *that*? There is no obvious solution to any of these difficulties; they are only obscured by an experimental "validation" of the KIF idea that fails to isolate the users of the special-purpose languages involved.[3]

In order to experimentally validate the KIF idea, we would suggest the following: Two interlingua developers should be found and should each write translators to or from not a special-purpose language, but to or from the other interlingua. Then for a variety of special-purpose languages L, one of the interlingua groups should write a translator from L into the interlingua, and the *other* group should write a translator from their interlingua back into L. Declarative knowledge bases written in L should be mapped into one interlingua, then to the other interlingua and finally back to L. If the result is equivalent to the original knowledge base (in the sense of the range of conclusions drawn by the special-purpose system), then the interlinguas will have performed successfully. This experiment avoids the problems of the earlier one because the translations to and from L are to be developed by isolated researchers.

A drawback of this proposal is that it requires the development of two distinct interlinguas; it might seem more natural to ask two independent research groups to write translators into and out of a single interlingua and to then use their results to translate from one language to the other and back again. The reason that we have not suggested this is that the standard developers are all too likely to blame the (to our mind inevitable) failure of this experiment on inadequacies in the translators developed by the independent research groups. Furthermore, if these groups were to turn to the KIF designers for help, it would invalidate the requirement that they work independently. It is for this reason that we are requiring the

[3] We do not mean to imply that this problem will not arise using the language proposed in Section 3, only that we have provided a clear method by which different groups can *attempt* to express their individual representational commitments.

interlingua teams to write the translators, at least initially – and that means that we need at least two interlinguas to work with.

Although we expect this experiment to fail, attempting it is an important activity – the attempt itself amounts to an experimental evaluation of the question of whether or not KR is ripe for standardization. Such an experiment – viewed as research only and not as a standardization effort in and of itself – would doubtless add to our understanding of the current state of the art in declarative systems.

5 Standards effort or research?

Let us consider this last comment in a little more detail: is the KIF work to be viewed as a standards effort, or as research?

I expect that it will be clear from the arguments that have been made throughout this paper that I feel that the current state of KR work makes it impossible to view the development and evaluation of a KIF as anything but research. As remarked in an earlier footnote, this opinion was also expressed by a unanimous vote of the people attending the KR Standards Panel at the 1991 International Conference on Knowledge Representation and Reasoning.

This observation has some important consequences. First, the work on KIF should not be reported as a "standard," lest the funders of the work misperceive what has been accomplished. The work should be reported as research, and funded and evaluated as research as well. In this vein, I would like to applaud the tone taken in [4], where it is made clear that research is being pursued and that standards are not. I am far less sympathetic with [2], which claims on no fewer than seventeen occasions that a standard is being developed.

If we view the KIF effort as research, what exactly is the question being investigated? Clearly, the question is one of knowledge interchange – is it practical for one group to make use of the declarative knowledge that has been obtained and encoded by another?

This is an important question, and one that *should* be investigated. But the development of a wide-ranging KIF such as that pursued in [2] has very little to do with this question! If neither of the groups trying to exchange knowledge use nonmonotonic reasoning, including it in a KIF is completely immaterial to the question of whether these groups can share knowledge effectively.

The argument *for* a wide-ranging KIF is that it can potentially reduce the effort needed for n groups to exchange knowledge from $o(n^2)$ to $o(n)$. In the first case, each group will need to write a translator into each other group's language; given the KIF, translators into and out of the KIF are all that is needed.

But this is not research! The *research* question is whether knowledge sharing is possible at all; reducing the amount of effort needed from $o(n^2)$ to $o(n)$ is a standards issue, not a research one.

This, then, is the fundamental difference between the ideas presented in this paper and those presented in [2]. By proposing a minimalistic language and describing specific experiments, I am trying to further a research investigation of the practicality of knowledge sharing. By proposing the adoption by fiat of a specific maximalistic language, [2] is clearly in the business of forming a standard.

One of these activities is appropriate. And the other is not.

Acknowledgement

I would like to thank Jim Bennett, Ron Brachman, Johan deKleer, Robert Engelmore, David Etherington, Richard Fikes, Ken Forbus, Don Geddis, Pat Hayes, David Israel, Ben Kuipers, Bill Mark, John McCarthy, Karen Myers, Peter Patel-Schneider, Ramesh Patil, Elaine Rich, Rich Washington and the members of Stanford's PRINCIPIA group for discussing these issues with me. The suggestions of several anonymous reviewers were also quite helpful. The opinions expressed are mine, however.

References

[1] N. D. Belnap. A useful four-valued logic. In G. Epstein and J. Dumm, editors, *Modern Uses of Multiple-valued Logic*, pages 8–37. D. Reidel Publishing Company, Boston, 1977.

[2] M. R. Genesereth. A KIF proposal. Technical report, Stanford University, 1990.

[3] M. L. Ginsberg. User's guide to the MVL system. Technical report, Stanford University, 1989.

[4] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12, 1991.