from CLAW96: Proceedings of the First International Workshop on Controlled Language Applications, Katholieke Universiteit Leuven, Belgium, March 1996, pages 233-242.

Controlled Language for Knowledge Representation

Stephen G. Pulman
SRI International Cambridge Computer Science Research Centre,
Suite 23, Millers Yard, Mill Lane,
Cambridge CB2 1RQ, UK.
and
University of Cambridge Computer Laboratory
sgp@cam.sri.com

February 1996

Introduction

Let us take expert systems as the paradigm example of something using a knowledge representation language, although there are of course many other types of application that also use such languages. Expert systems are widely used in research, commerce, medicine, the law, and in government and are one of the success stories of artificial intelligence, as everybody knows (Winston 1987).

Current expert systems represent knowledge in several different forms. In most cases, there is an object or inheritance hierarchy representing static facts about the objects, actions, or events in the domain of application, and the important relationships between them. Other types of knowledge are represented in the form of rules, usually conditionals saying what action should be performed if such and such a condition is encountered, or what conclusion can be drawn if such and such other conditions obtain. A third type of knowledge is explicit control information, either in the form of a 'conflict resolution' strategy, saying how to decide which of several alternatives to pursue, or (increasingly common) in the form of a statistically driven component also deciding among several competing hypotheses on the basis of prior experience. We shall not be concerned with this type of knowledge in the following.

Although widely used and effective, everyone agrees that there are at least three areas in which the current generation of expert systems could be significantly improved. Firstly, the area of **knowledge acquisition** remains a serious bottleneck in the rapid and effective deployment of expert systems. The typical process

of knowledge acquisition involves a team consisting of at least one knowledge engineer and one domain expert, who work together often over an extended period of weeks or even months. The engineer and the expert interact together to find a satisfactory explicit encoding of the expert's implicit knowledge in the knowledge representation formalism required by the expert system in question. Although there have been many developments concerning practical methodologies for guiding this process, and there also exist some tools (e.g. Schreiber et al. 1993), it remains the case that the knowledge acquisition process is a time-consuming and expensive part of the task of deploying an expert system in a new domain.

The second problem area is that of **transparency**. In most applications the route by which the conclusions are reached by an expert system must be capable of being made transparent to the user, if they are to be treated with confidence. This is especially so in many legal, medical, or military applications, but applies in some degree to all kinds of expert system, even if only in a debugging mode. In the case where a system is recommending a course of action that is in some way unexpected to the user, particularly where the user is a domain expert, then the system must be able to justify its decision if it is to be taken seriously.

The third major problem area is that of **maintainability** or **knowledge refinement**. Knowledge does not remain static and for most systems, it is necessary to change, augment, or update the rule base at regular intervals. Efficient and low-cost maintainability presupposes at least a partial solution to the other two problems, because it is not economical to re-do the acquisition process any more than is necessary, and in cases where the rule base needs adaptation or correction, easy diagnosis of the existing problem is required.

The remainder of this paper describes a programme of research that we are undertaking into the use of a variety of controlled English as a tool to help solve these various problems. The basic idea is to see to what extent the kind of knowledge used in these systems can be expressed in a restricted form of English which is then systematically related to the underlying knowledge representation language. Our hypothesis is that if this proves to be possible, we could partly avoid the necessity for having a knowledge engineer involved, and also increase the effective use of domain experts, who are usually the most valuable resource in developing a knowledge based application.

Computer Processable English

Usually when people talk about 'controlled English' they mean some variety or sublanguage of English which is restricted as to vocabulary (or more strictly, word

senses) and syntactic form, with a view to minimising ambiguity and maximising clarity. Frequently, computational tools are employed to encourage writers to stay within this variety of English. These tools may carry out partial or even complete parsing, and some vocabulary checking.

By 'Computer Processable English' (CPE) we mean a dialect of English that has all the properties of controlled English, and more: a dialect that is restricted so as to be capable of being completely syntactically and semantically analysed by a natural language processing system. At present, these dialects are likely to be somewhat restricted in expressiveness by comparison with other types of controlled language, but even at the current state of the art, a natural language system like the Core Language Engine (Alshawi (ed) 1992) is capable of accurate syntactic and semantic processing of pretty well all the basic context-independent syntactic and semantic constructs of English, and in a suitably restricted setting, can also interpret many of the contextually dependent elements like pronouns and ellipsis that are frequently needed for sequences of sentences to sound natural.

Since the output of analysis of CPE sentences is a meaning representation in some formal language (in the case of the Core Language Engine, a logical language consisting of first order logic augmented with some higher order constructs), it is natural to think of somehow linking this formal representation with those provided by various knowledge representation languages. The link might be analogous to that provided by, say, transfer-based machine translation systems. If this were achievable in the knowledge representation language to CPE direction, then it would be possible to make the contents of a knowledge base, or some parts of the operation of the expert system, accessible to a domain expert without the intervention of a knowledge engineer. Since CPE is just a dialect of ordinary English, then no special competence is required to understand it, other than perhaps the understanding of technical terminology characteristic of the domain of application. If it were to be achievable in the CPE to knowledge representation language direction, then this would open up the possibility that at least some of the knowledge acquisition and knowledge refinement processes could be simplified in that the domain expert could input the necessary knowledge without having to understand the formal or computational properties of the underlying knowledge representation language. Of course, it would be necessary to ensure that the CPE input always had a valid translation into the underlying knowledge representation language, and there are many other computational linguistic and human interface problems to be solved before this is a practical possibility. Before continuing, let me say what is NOT being proposed here. Many current commercial expert systems offer a 'quasi-English' interface, allowing the programmer to enter facts and rules using templates like:

```
A <subtype> is_a <type>
A <thing1> has_as_subpart a <thing2>
If cardinality of <things> is_greater_than <number> then ....
```

Although useful, these macro-style notations do not have the flexibility of CPE: in particular, as we will see later, they will not support the type of interactive knowledge refinement that some types of application require.

Also, many systems allow rules or facts to be associated with pieces of canned text, possibly containing slots to be filled in with run-time values for variables (Moore 1995:24-27). This is a different solution to the problem of transparency from what is being proposed here. While it has the virtue of simplicity, it suffers from several disadvantages. Firstly, there is no guarantee of fidelity: since the canned text bears little systematic relation to the underlying rules, there is no guarantee that it correctly relates what is happening. This will be a particular risk if the system is being updated or changed frequently. Secondly, the solution lacks flexibility: only one kind or level of explanation can be given, and if the user has a query that was not anticipated by the author of the canned text, there is no way to satisfy it.

CPE for knowledge representation

Unfortunately, it is not realistic to expect to translate sentences of CPE directly into representation languages that are used directly for computation.

To illustrate this consider a rule like the following. It is from an expert system used as part of a military 'war-game' simulation environment (Lankester and Robinson, 1994): an application domain that we are using. The rule is displayed in a template-like form that is actually generated from the underlying Lisp code. The rule describes the conditions under which an RPV ('remotely piloted vehicle') should be dispatched for reconnaissance purposes when the intended route of the agent is blocked.

```
...
 IF
     cardinality of <AVAILABLE-RPVS> > 0
  and cardinality of <ACTIVE-RPVS> < 3
 WHERE
  <DIV-ASSESSMENT> is assessment of my own
  <AVAILABLE-RPVS> is available RPVs of <DIV-ASSESSMENT>
  <ACTIVE-RPVS> is active RPVs of <DIV-ASSESSMENT>
```

```
THEN
...
ACTION
 strategy: each
 ALWAYS
  WHERE
    <DIV-HQ-LOCATION> is location of battlefield entity of myself
    <RECCE-LOCATION> is parameter <LOCATION>
     ...
    <RPV> is first of available RPVs of <DIV-ASSESSMENT>
  THEN
   set RPV of <MISSION> to <RPV>
   ...
   remove <RPV> from beginning of available RPVs of <DIV-ASSESSMENT>
   add <RPV> to end of active RPVs of <DIV-ASSESSMENT>
   deploy RPV with RPV <RPV>;
          mission route <MISSION-ROUTE>;
          mission-type SINGLE-MISSION
   ...
```

The rule contains several sections which are really not relevant to an understanding of its operation. For example, all the 'WHERE' clauses are only there to effect the setting of local variables like DIV-ASSESSMENT or RECCE-LOCATION, variables which have scope only within that rule and simply serve to allow access to components of a database object. Furthermore, the lines:

```
remove <RPV> from beginning of available RPVs of <DIV-ASSESSMENT>
add <RPV> to end of active RPVs of <DIV-ASSESSMENT>
```

are simply a verbal description of the Lisp operations of taking an element off one list-valued variable and putting it on another. These book-keeping clauses are a way of saying that the currently available RPV is to be regarded as no longer available. If no distinction is made between the important components of a rule and those that are merely an artefact of the implementation technique, then having a CPE version of the rule is probably not going to increase its clarity.

At this point another desideratum should enter the picture: that of portability. For many types of application, it would be eminently desirable that the knowledge base developed should be reusable, both within other implementations carrying out the same tasks (e.g. to avoid dependency on one particular commercial expert system framework) or for other types of task (e.g. the same knowledge that is used for solving real problems can also be used for training or simulation

purposes). One recent solution to the portability and reusability problem is the KIF (Knowledge Interchange Format) representation language (Patil et al. 1992) within which declarative knowledge bases can be formulated.

KIF is a knowledge representation language based on first order logic, with some higher order and non-monotonic extensions. It has a fully specified syntax and semantics (Genesereth and Fikes, 1992). KIF contains constructs which correspond to most of those one would find in commercial expert systems as well as some required for more sophisticated reasoning and problem solving systems. The knowledge thus formulated can then be translated into a variety of proprietary or other formats suitable for different types of processing. Existing knowledge bases can be translated into KIF to the extent that this is possible: various systems have been developed for the semi-automatic translation of such knowledge bases between KIF and some of the well-known knowledge representation formalisms (Gruber 1993, van Baalen and Fikes 1993).

Using a language like KIF as the canonical representation language for knowledge bases avoids one of the difficulties of going directly from CPE to an executable knowledge representation language. We no longer have to worry about the precise syntactic form of the KIF representation: in principle, any logically equivalent formulation will be adequate. The fact that not all such formulations would support efficient inference is no longer a problem for we can safely assume a further translation process from KIF to a proprietary knowledge representation formalism which will be precisely concerned with matters of that kind. We are therefore (reasonably) free to concentrate on finding a CPE version of the rules in the knowledge base which is easily understandable and only constrained by the requirement that it should be translatable into KIF.

Here now is a possible CPE equivalent of (the full version of) the rule above. We may assume that the user is presented with a template consisting of the upper case items, so that all that is entered are individual CPE phrases and sentences:

```
TASK: dispatch an RPV
INPUTS: a blocked route, B; a location, L.
PRECONDITIONS:
 there must be at least 1 available RPV
 there must be less than 3 active RPVs
ACTION:
 make a mission record, M,
 which has mission route, R,
 and which has blocked route, B,
 and which has blocked location L.
```

```
  R's launch location is divisional HQ.
  R's recce route is L.
  R's search radius is 3000.

deploy an available RPV, X,
  which has mission route R,
  and which has mission type 'single mission'.

change status of X from 'available' to 'active'.
```

Notice that the use of pronouns is avoided by using letters like 'M', 'L' etc, in a way reminiscent of mathematics textbooks: a device used in Macias and Pulman (1995: 313). The syntax of this construct is simply that of an appositive NP: the semantics is approximately that of a 'discourse referent' in Discourse Representation Theory. Sentences and phrases like this are well within semantic coverage of a system like the Core Language Engine.

Does this solve our problems?

With such a picture, the knowledge acquisition task now becomes somewhat redefined. One of the main tasks of the knowledge engineer is now to collaborate with the domain expert to define a dialect of CPE which can express the required knowledge in a declarative form. It is possible that in some applications, the domain expert, guided by a few example rules, can then describe at least some parts of the knowledge base with only minimal guidance. Even if this proves not to be the case, the process of verification or validation of the resulting knowledge base should be easier, since the domain expert has only to interpret CPE rather than some knowledge representation formalism.

Transparency of operation of the expert system should also be enhanced, although the use of CPE does not solve the problem completely. As many people have pointed out (e.g. McKeown and Swartout, 1987), it often needs more than what is contained in the relevant rule to understand what is going on when a particular decision is made. The piece of information on which the decision is based may bear only a tangential relation to the underlying causal structure: a well known example of this is the question posed by a medical expert system 'is the patient aged 17 or over?' as a test for whether alcoholism is a factor in the diagnosis (Davis and Lenat, 1982). Nevertheless, when attempting to understand at a detailed level, having access to the relevant rules in CPE form is likely to be considerably more useful to the domain expert than trying to understand the Lisp/English hybrid which would otherwise be offered in our application.

As far as maintenance and knowledge refinement is concerned, again the use of CPE provides no magic solution, but still a substantial incremental improvement. In the particular application that we are using, it is quite plausible to think of the domain expert interacting with the expert system during a re-run of a simulation, to change individual phrases or sentences in a CPE rule, either because they are inacccurate, or in order to try out 'what if' alternatives. Of course, there will be the same human factors problem as for any other controlled language application, that of making sure the user stays within the right dialect. But is is no worse in our application than in many others (and in some ways it is much better, since military personnel are used to being trained to use specialised sublanguages).

Current state of project

We have begun to work on some of these issues in the context of the simulation system described earlier. We have focussed on a subset of the knowledge base (which is encoded in a proprietary knowledge representation formalism) concerned with 'unit movements', that is, planning the movements of a collection of military units according to an 'order of march'. This is more complicated than it might sound, since the relative positions and orientations of the units may have to be preserved even while avoiding natural or military obstacles, and on arrival at the destination the convoy may have to be re-arranged in a particular defensive or offensive posture.

In the first phase of the project we have developed tools for translation of the knowledge base into KIF. We are currently engaged in producing CPE paraphrases for the resulting rule base. However, since the form of the KIF expressions is related more to the properties of the original knowledge representation formalism than to the CPE equivalent we are not as yet developing a complete bidirectional translation between them. Instead, the CPE and KIF are partially linked in that a CPE template will be associated with a KIF rule, and only phrases describing the values of various slots will be genuinely translated. However, this will allow us to experiment in a preliminary way with assessing the effectiveness of CPE as a partial solution both to the transparency and the maintainability problem.

In order to be able to do this we have developed a KIF interpreter which will 'replay' the operation of the original expert system, but using the KIF version of the rules. We are then able to generate CPE explanations of the factors leading to a decision, as well as being able to 'rewind' or step backward or forward through an execution cycle. This mechanism will also allow a domain expert to intervene in the cycle by changing data or rules on the fly. Eventually our aim

is to allow this to happen via a spoken language interface, but that is some way in the future. Nevertheless, the ability of CPE to support interaction between the domain expert and the knowledge base is an important part of our long term strategy, for it makes possible the most effective use of domain expertise in the maintenance and refinement process, namely when that expertise is actually being deployed for real.

Conclusion

We have described the possible role of a variety of controlled language in overcoming some problems in the effective deployment amd maintenance of knowledge based systems. We have begun to experiment with the use of Computer Processable English in formulating the knowledge needed by a particular expert system in a military simulation application. This work is not yet complete, and is in any case only the first stage of a longer term research programme, be we hope shortly to be in a position to assess whether this is an approach which fulfils the promise we believe it to have.

Acknowledgements

The work reported here was supported at SRI Cambridge by the Centre for Defence Analysis (Land), DERA, Fort Halstead, under contract CDA/H/150. I am grateful to Janusz Adamson and Nick Roberts of DERA, and John Herbert at SRI for their contributions.

Bibliography

H. Alshawi (ed.), 1992, 'The Core Language Engine', Cambridge, Mass: MIT Press.

J. van Baalen and R. Fikes, 1993, The Role of Reversible Grammars in Translating between Representation Languages, Stanford University Knowledge Systems Lab, Technical report no. KSL-93-67.

R. Davis and D. Lenat 1982 Knowledge Based Systems in Artificial Intelligence, New York: McGraw Hill.

M. Genesereth and R. Fikes, 1992, 'Knowledge Interchange Format Version 3.0 Reference Manual', Computer Science Department, Stanford University.

T. Gruber, 1993, A translation approach to portable ontology specifications, Knowledge Acquisition, Vol 5, 199-220.

H. Lankester and P. Robinson, 1994, 'GENKNOFLEXE: A Generic, Flexible Model of C3I', in Proc. of Fourth Conference on Computer Generated Forces and Behavioural Representation, Orlando, Florida: pp 79-89.

B. Macias and S. G. Pulman, 1995, *A Method for Controlling the Production of Specifications in Natural Language*, The Computer Journal, Vol 38, No 4, 310-318.

K. McKeown and W. Swartout 1987 Language Generation and Explanation, in Annual Review of Computer Science, Vol 2, 401-449.

J. Moore 1995 Participating in Explanatory Dialogues, Boston, Mass: MIT Press.

R. S. Patil, R. E. Fikes, Peter F. Patel-Schneider, D. McKay, T. Finin, T. Gruber, and R. Neches, 1992, 'The DARPA Knowledge Sharing Effort: Progress Report', in KR92 (Proceedings of the Third International Conference on Knowledge Representation and Reasoning, Palo Alto: Morgan Kaufmann).

G. Schreiber, B. Wielinga and J. Breuker, 1993, 'KADS: A Principled Approach to Knowledge-Based System Development', London and New York: Academic Press. J. R. Quinlan (ed.), 1987, 'Applications of Expert Systems', London and New York: Addison Wesley.

P. H. Winston, 1987, 'The Commercial Debut of AI', in Quinlan (ed.), pp 3-22.