KRS - A Hybrid System For Representing Knowledge in Knowledge-Based Help Systems

Rolf Adams
Institut fuer Programmstrukturen und Datenorganisation
University of Karlsruhe, D-7500 Karlsruhe, FRG

Abstract

This paper describes the hybrid knowledge representation system KRS. It has been designed for representing knowledge in CASE tools, especially help systems. It supports easy encoding of the knowledge by a variety of constructs and performs efficient hybrid inferences. Knowledge acquisition is also supported by a classification procedure. Compared with other approaches for representing and retrieving information, such as hypertext or library science techniques, it supports semantic based information retrieval, as well as a natural integration of a natural language processor. Its convenience for domain modelling in a help system for the LaTeX formatter is shown.

Keywords: hybrid systems, knowledge representation, help systems, domain modelling

1 Introduction

Knowledge-based CASE tools such as software information systems or help systems become more and more important due to the increase of occasional users of software with functionally complex interfaces, the fast change of software according to the development of improved methods or more efficient hardware, and finally due to the increasing complexity of software. On-line help systems give the user advice about the functionality of software. Software information systems may be looked at as help systems which assist the software developer in finding a piece of software, or information about software. They may even give help information about the used programming tools. This paper describes a knowledge representation formalism, KRS, for representing knowledge in help systems. It shows how the formalism may be used for domain modelling in a help system for the LaTeX formatter [8].

Previous methods to represent knowledge in help systems include library science techniques [3], hypertext [9], databases [5] and AI techniques [4] [7]. Each of these has advantages and disadvantages. Problems with library science techniques include the finding of a suitable vocabulary or attribute set, the large size of the vocabulary, and the syntactic rather than semantic based retrieval. Using hypertext seems promising at first but shows some problems too. Using keywords for retrieval results in the same problems as the library science techniques above. Using a browser to navigate through the network for certain information puts the burden of search onto the user. Furthermore, the effort for knowledge acquisition is high. Special schema-like representations [7] are not as general as for example languages like BACK. This makes it difficult to use them for systems which provide a slightly different structure, that is some knowledge cannot be represented at all or rather unnatural.

Problems using databases include the weak expressiveness, the lack of inferences, and the lack of a suitable query language. Newer semantic data models or knowledge representation languages such as CLASSIC [2] or CANDIDE [1] try to overcome these problems. They are based on well defined semantics and support specific inferences due to subsumption and realization. One such language, KANDOR [10], has been used for representing knowledge in the software information system Lassie [4]. Using such a language has some benefits compared to the other approaches: it allows semantic retrieval and the coupling or integration with a natural language processor. Problems include the effort for knowledge acquisition, as well as the expressiveness of the language and the efficiency of inference procedures.

The rest of the paper is organized as follows. The following section contains an overview of the most important constructs in the KRS formalism. Section 3 provides an outline of what kind of knowledge should or can be represented in help systems. Furthermore it contains a description how KRS can be used for domain modelling in a LaTeX help system and for answering queries about the stored knowledge. The paper finishes
with the conclusions and the plans for future work.

2 The KRS knowledge representation language

KRS (Knowledge Representation for Software) is a hybrid formalism for representing knowledge. It has a well-defined model theoretic semantics and is based on newer hybrid formalisms like BACK [13] or CLASSIC [2]. For each notation the representation system includes a part, called a box which maintains the knowledge represented in the notation. The system consists of three boxes: T-Box, A-Box, and R-Box. Furthermore there is a query language to retrieve the represented information. The boxes and the associated notations are described in the following.

The T-Box is responsible for representing terminological knowledge and allows the specification of primitive and defined concepts or roles. The constructors AND, OR, ALL, AT-LEAST-ONE (or SOME), EXACTLY-ONE (or ONE), AT-MOST-ONE (or L-ONE), NO-ONE, VALUES, S-VALUES, PAR, S-PAR, and finally concept calls may be used to build composite concept descriptions. AND specifies concept conjunction and OR concept disjunction. The definition of defined concepts with the OR construct has not been useful in the considered applications. Hence it is not allowed to use OR as the outermost constructor in the specification of a defined concept. The ALL constructor restricts the range of a role to a given concept. AT-LEAST-ONE, AT-MOST-ONE, EXACTLY-ONE, and NO-ONE specify for a role that there exists at least one, at most one, exactly one or no filler of a certain concept and all role fillers belong to this concept. VALUES or S(ingle)-VALUES specify certain individuals as fillers or as the only fillers of a role. These constructors could be inbedded in the SOME or ALL constructors by allowing the second argument to be a list of instance values, but to me a distinction increases the readability. The system supports predefined concepts for integer numbers (INTEGER, POSITIVE) and strings (STRING). Special constructs (DEF-ATTRIBUTE, DEF-INTERVAL) may be used to define concepts with a fixed extension. DEF-ATTRIBUTE allows the definition of concepts all of whose instances are known. For the easy specification of concepts representing integer intervals interval expressions and set constructors may be specified with the DEF-INTERVAL construct.

KRS allows concept calls to create anonymous instances or subconcepts of a called concept. These are only referenced in the calling concept or instance. In concept calls actual parameters may be bound to roles of the called concept. An actual parameter may be a concept to restrict the range of a role, an individual as a role filler, or a closed role to assert that the role fillers are identical.

role of an instance or concept. Finally the PAR or S-PAR construct enables the easy, fine-grained specification of the range of a role for a concept or an individual. They allow to specify that specific fillers of a certain type for a role exist but the concrete fillers are unknown. Useful applications are the specification of parameter profiles or object fields whose values are unknown.

Currently KRS supports a role hierarchy with the constructors ANDR, DR, and RR, and INV. Analogously to concepts, primitive and defined roles may be specified. ANDR defines role intersection, DR domain restriction RR range restriction, and INV the inverse role. Currently KRS don't support more sophisticated role constructors, like for example ORR in the T-Box. They weren't important in specifying concept descriptions and seem also not important for other applications. Furthermore it requires considerable effort to completely realize them.

There are further operators which may be used in concept descriptions but which are not important for the concept taxonomy. So it is possible to specify default values which are assumed when the associated role is closed and no values are given as role fillers. Similar role value maps may be specified.

The A-Box allows the specification of knowledge about individuals. In KRS this knowledge is specified with the same constructors like knowledge about concepts.

In the planned applications the closed world assumption seems to be mostly appropriate. However, to be more flexible the system assumes an open world assumption and a CLOSE construct allows the user to specify that a role or concept should be closed. Closing a role means that all fillers of the role are known. Closing a concept means that all individuals belonging to the concept are known.

The R-Box allows the specification of so called derived roles. These roles may not be used in the specifications given in the T- or A-Box. Currently the following constructors are supported: ANDR, DR, RR, ORR, INV, TRANS, COMP, D-CLOSED, and R-CLOSED. ANDR defines role intersection, DR domain restriction, RR range restriction, ORR role unica, INV inverse roles, TRANS the transitive closure, and COMP role composition. R-CLOSED and D-CLOSED build the closure of a role according
to another role. They can be simulated by the
TRANS and COMP constructors (R-CLOSED(r1,r2)
= COMP(r1,TRANS(r2)), and D-CLOSED(r1,r2)
= COMP(TRANS(r1),r2)), but are explicitly included
due to their frequent usage in modelling. Note that
derived roles may not be used in concept descriptions.

For the individuals we assume a unique name assump-
tion. Furthermore for concepts and roles certain meta
information may be specified in predefined facets.
Currently the facets Has-Name and Has-Def are sup-
ported. The fact Has-Name is filled for all entities
with the associated unique name. Has-Def can contain
an informal definition or description of the knowledge
represented by the entity.

There exists a query language for retrieving the infor-
mation stored in the T-, R- and A-Box. This language
allows the retrieval of individuals or pairs of individu-
als by concept or role definitions. Note that in queries
also roles belonging to the R-Box may be used. Further
queries ask for role and concept relationships like sub-
sumption or disjointness. A more detailed description
is excluded here.

Compared with the language CLASSIC the KRS lan-
guage provides an extended expressiveness, except that
we do not allow general number restrictions. CLAS-
SIC does not provide as specific constructs in the T-
Box, nor does it provide role constructors. The main
reason why a role hierarchy is not supported is the
worst-case complexity of the subsumption algorithm.
Allowing the constructs present in CLASSIC with gen-
eral number restrictions but no role hierarchy results
in a complete subsumption algorithm whose worst-
case complexity is polynomial. Adding furthermore
a role hierarchy, as in CANDIDE, would result in a
subsumption algorithm whose worst-case complexity is
PSPACE complete [6]. Due to the SOME construc-
tor and the role hierarchy the worst-case complexity
in KRS is also non polynomial. However, the current
experience in modelling shows that the subsumption
relationships to be determined are simple. The exponen-
tial worst-case complexity is not important for the
considered applications, since the number restrictions
are small and the role hierarchy is flat.

3 Using KRS for Domain
Modelling

When building information systems first a domain
analysis is necessary to determine what kind of knowl-
edge should be represented. This analysis results in
the following general scenario for the considered help
systems:

The application domain contains a number of agents
(commands, functions, modules, processes, ...), whose
purpose or main effect is performing actions on ob-
jects of the application domain. The main effect often
comprises a number of subeffects which are performed
by accessing (reading, changing, or activating) other
objects. Some of the subactions or accesses to other
objects are conditional. Dependent on the applica-
tion such conditions may be complex or rather simple.
Complex conditions including boolean operators and
function calls may for example be found in an IF- or
WHILE-statement of a C program.

The agents and other objects of the application do-
main have certain properties and relationships to other
objects or themselves, especially Is-Generalization-Of,
Is-Specialization-Of, Is-Part-Of, and Has-Purpose or
Has-Main-Effect. The purpose or main effect of an
agent can be described by an action. The roles in these
actions correspond to the usual semantic cases of case
theory, found in sentences that contain the action verb
as a main verb. The actions often result in a change of
the world description by affecting or changing parts of
other objects.

Based on this scenario there are in general various
possibilities for modelling certain aspects, but usually
one seems to be most natural. Furthermore, modelling
some aspects are more important or require much more
computation resources than others. Based on this sce-
nario a natural modelling in KRS is as follows: Con-
cepts of the domain, including the actions performed
by agents, with their relationships are modelled as a
terminology in the T-Box, relationships between roles
which are not important in the description of concepts
or are not describable due to the limited expressiveness
are modelled in the R-Box, and objects are modelled
as individuals in the A-Box.

We have used KRS for domain modelling in a help
system for the text preparation system LaTeX. This
modelling mainly comprises a classification of com-
mands and actions. Figure 1 shows an excerpt of the
terminology in the usual graphical notation. Primitive
concepts are marked with a star while defined concepts
aren’t. Role relationships are excluded. Below the root
there are three subconcepts: actions for the class of
actions, latex-objects for the class of objects in the
application domain, and finally conditions for the class
of conditions. The corresponding textual specification
for some of the concepts is given in figure 2. The fol-
lowing lexical conventions are used: Concept names and
instance names are completely written in lower case.
Concept names are given as nouns in the plural while
instance names are specified in the singular form. Role
names are written in lower case except that the first
letter of all subnames is written in upper case.

The examples show that the concept descriptions are rather simple and the constructs allow an adequate description of terminological knowledge. Other constructors, such as general number restrictions or a general OR construct are not necessary. Furthermore role hierarchies are useful in modelling. The R-Box is especially useful for specifying role relationships such as transitive closure or role composition. Note that classification and realization eases the knowledge acquisition process considerably. Defined concepts and instances are properly classified or realized by the representation system. This is especially important in application domains where the knowledge base increases over time, as for example in software information systems or in help systems where user defined commands are modelled. Maybe that in these applications the acquisition process is performed by different people. In this case they have to understand only the most specialized primitive concepts and the associated roles in order to properly specify the new knowledge.

It is not possible to model arbitrary conditions with KRS. But certain equality conditions, which state preconditions for calling a command, can be modelled easily. For example in the LaTeX domain some commands may be used only in a certain mode or only in the context of another command. This may be easily modelled by creating a role Has-Predicate and assign the corresponding fillers for the individual commands. However, it is not possible to model conditions including boolean operators others than And or function calls as arguments. Furthermore it cannot be modelled that a command calls another command only when some condition holds, for example if the command is given in a certain mode. This would require to be able to specify a kind of conditional role filler. However, in this application the modelling of conditional command calls, or more complex conditions in general, does not provide useful help information. Furthermore, it requires relatively large computation resources.
4 Conclusions and Future Work

The paper presents the hybrid knowledge representation system KRS and demonstrates its convenience for domain modelling in help systems. We have modelled a rather small excerpt of the LaTex manual. However modelling the rest would be much easier, since the main problem was to work out what kind of knowledge should be represented and which constructs allow easy encoding of this knowledge. The implementation of KRS has not finished. Hence we are not able to test the given specifications. However, to be build a help system which may really be used in practical situations much more modelling work is necessary.

Using the query language of the representation system is a major burden for a user, even if an editor like ARGON [11] is available. An interface much more easier to use would be natural language. Therefore an existing caseframe parser for German is being integrated and an intelligent answer component developed. We are designing a formalism to specify the syntactic knowledge associated with the elements of KRS.

Another tool to support knowledge acquisition would be a graphical editor for editing or displaying taxonomies. Therefore it is planned to integrate the customizable graph editor EDGE [12]. The overall result will be an environment which enables the development of knowledge-based tools that offer a natural language interface in a reasonable time. Knowledge acquisition is supported by a convenient representation formalism, a classification procedure and a graphical interface for editing and displaying taxonomies.

We are currently using KRS for the development of a software information system. This should enable a user to find a piece of software, or provide information about software. The system will be integrated in the UNIX environment with existing tools, like for example the revision control system RCS [14]. When the programmer checks in a software object, the associated information can be translated in corresponding A-Box assignments. Tools which scan the software object may extract further information, like for example cross references, and translate it into corresponding A-Box assignments.

Currently KRS is being extended based on the modelling experiences. For example in software information systems it would be useful to represent dates adequately. Hence, a special predefined concept type TIME together with some special operators will be introduced. Another extension is the possibility to arrange and access the fillers of a role as an ordered sequence.

References