KRF

**Knowledge Representation Framework Project**

Department of Computer and Information Science, Linköping University,
and Unit for Scientific Information and Learning, KTH, Stockholm

Erik Sandewall

# Knowledge Representation Framework: Introduction and Overview

## 1 The KRF and its Uses

The Knowledge Representation Framework (KRF) is a consistently designed set of conventions and notations for knowledge representation, software development, intelligent autonomous agents and the organization of documents. In this case, 'documents' includes not only articles and reports, but also static and dynamic webpages. We believe that much can be gained by having a coherent design for this range of topics. The present short note describes the structure of the KRF in brief overview, as well as its motivation and its major uses.

The Knowledge Representation Framework is used as the notational and conceptual framework for the following three primary purposes:

- For *texts on Knowledge Representation,* including both lecture notes and scientific articles.
- For *the Leonardo Software System,* where in particular it provides the notation for scripts and for structured information in textual or "serialized" form.
- For *work on Knowledge Acquisition and Information Analysis,* that is, those activities that can lead to reliable and repeatedly useable knowledge modules and knowledgebases of high quality.

Both the KR Framework itself and each of these three primary purposes is represented by a set of reports and of webpages that will be described below. In addition the KRF is used in an expository fashion for lecture notes on the following two topics:

- For an *introduction to Formal Logic*
- For an *introduction to Programming Paradigms,* in particular for list processing, functional programming and pattern and content driven invocation, as well as for the Lisp programming language

These tutorial materials have been developed for use in my university course on "Artificial Intelligence and Lisp," where in particular the lecture notes on logic serve as an introduction to the presentation of Knowledge Representation in the course.

One specific result from the work on Knowledge Acquisition and Information Analysis is the *Common Knowledge Library* (CKL), [1] which is a freely available, module organized knowledgebase of commonly known facts. The CKL currently contains more than 60,000 entities each of which is described by a number of attributes.

If the Knowledge Representation Framework is considered as a first layer of design, and the three primary purposes as a second layer, then there is also a third layer which at present consists of one topic area:

- For work on *Analysis and Development of Electronic Publishing Technologies,* such as novel peer-review procedures, novel ways of communicating research results, and IT support for open access and other author-side publishing.

[1] http://piex.publ.kth.se/ckl/

The systems design side of this work makes use of all three areas in the second layer as well as the Knowledge Representation Framework itself.

We are working actively in the areas that have now been described. These activities are jointly called *the CAISOR Research Agenda,* where the CAISOR acronym presently stands for the *Combined Agenda for Information Analysis, Software Systems, Open-Access Publishing and Knowledge Representation.* The CAISOR Webpage [[2]] describes this agenda including its earlier history.

Much of the work under the CAISOR Agenda can be characterized as *work towards a large and complex design* using design iteration. The methodology for such work is an important topic in itself, and we have formulated the MORADOR methodology [[3]] for this work. This acronym stands for *Methodology Of Research And Dissemination Of its Results.* We consider that research methodology and research publication styles are interdependent in important ways.

The current documents for the Knowledge Representation Framework are further described in the next section, and in the KRF webpage [[4]] . Research articles and tutorial texts on Knowledge Representation are found in the KRF Courseware page [[5]] , which also contains links to the KRF-based tutorial on logic, Lisp and programming paradigms. There are similar webpages for the other main areas: for Leonardo [[6]] , for information analysis [[7]] , and for electronic publishing technologies [[8]] . Each of these contains an overview of the respective area and links to publications within it.

## 2 Documents Defining the KRF

Besides the present Introduction and Overview, the following documents describe and define the Knowledge Representation Framework.

- *Knowledge Representation Framework: Overview of Languages and Mechanisms* [[9]]
- *The KRF Type System and Ontology* [[10]]
- *Leonardo Document Preparation Facility* [[11]]
- *The KRF Agent Messaging Framework* (forthcoming)

The first one of these should be read before the others, but the documents in items 2 to 4 can be read independently of each other. The following is a brief summary of their contents.

[2]http://www.ida.liu.se/ext/caisor/
[3]http://www.ida.liu.se/ext/morador/
[4]http://www.ida.liu.se/ext/krf/
[5]http://www.ida.liu.se/ext/kr-courseware/general/overview/page.html
[6]http://www.ida.liu.se/ext/leonardo/
[7]http://piex.publ.kth.se/ext/inka/
[8]http://piex.publ.kth.se/adept/
[9]http://www.ida.liu.se/ext/caisor/pm-archive/krf/009/
[10]http://www.ida.liu.se/ext/caisor/pm-archive/krf/004/
[11]http://www.ida.liu.se/ext/caisor/pm-archive/leonardo/010/

## 2.1 Languages, Aggregates and Mechanisms

The Overview of Languages and Mechanisms introduces the syntax of *Knowledge Representation Expressions* as the first step. This syntax is a *syntactic style* in the same sense as S-expressions and XML syntax, since it provides a framework within which one can define a number of specific languages, just like Lisp, KIF, FIPA-ACL, PDDL and other languages are expressed as S-expressions. The report continues to defining the *Common Expression Language* (CEL) as a language for expressing terms that can be evaluated and commands that can be executed. It also defines the representation for larger sets of concepts: *entity-descriptions* where an entity is assigned values for a set of *attributes,* each value being expressed in the Common Expression Language, *entityfiles* consisting of a set of entity-descriptions, and so forth for larger aggregates.

## 2.2 Type System and Ontological Structure

The KRF Type System and Ontological Structure uses three structures: a *type structure,* a *taxonomy,* and a *subsumption structure.* The type structure is based on the convention that each entity in a knowledgebase must have a type, which is again an entity. For example, the entity `doe.john` representing a particular person may have the type `person,` which has the type `thingtype` which has itself as its type. (The actual structure is not exactly like this, but the example illustrates the idea). Entities that are used as types of other entities are collectively called *types,* other entities are called *objects.* The type structure is used for characterizing what entities can have what attributes, and what should be the type and syntactic structure of the values of those attributes. For example, the type `person` may specify that members of that type shall have, among others, the attributes `father, mother,` and `siblings,` that the value of the latter attribute shall be a set of entities of type `person,` and so forth.

The *taxonomy* introduces an additional structure on types, but not on objects. Each type has an optional value for the attribute `included-in` and the value must be single other type; it can not be e.g. a set of other types. If a type $a$ is included in type $b$ in this sense, then the type of $a$ may be included in the type of $b,$ and in general, the type of $b$ shall be a member of the path that begins with the type of $a$ and that contains the entities that are referenced from it by using the `included-in` attribute iteratively. The taxonomy is used for intrinsic relationships between types, for example for expressing that mammals are animals and animals are living things.

Objects are divided into a few kinds, in particular *things* and *qualities,* but also some others. Things includes for example persons, vehicles, or cities; qualities include for example `red` and `angry.` The *subsumption structure* applies to types and to qualities, and it differs from the taxonomy by allowing an entity to be directly subsumed by more than one other entity, and by allowing these subsumption relationships to be defeasible. For example, the type `whale` may be subsumed by both `mammal` and `sea-animal,` the type `mammal` may be subsumed by `land-animal,` and there is a mechanism whereby `whale` is not implicitly subsumed by `land-animal.`

The subsumption structure also allows to express relations between qualities, for example, that `pink` is subsumed by `warm-color.` It is natural that it does not apply to relations between things.

One may propose that the taxonomy is redundant since its statements can also be expressed using the subsumption structure. There is however a performance argument in favor of using the separate taxonomy, since the inference of subsumption relationships is potentially complex and resource-consuming whereas purely taxonomical subsumption can be identified rapidly and without any risk of ambiguity.

## 2.3 Document Representation Framework

The Document Representation Framework will be discussed in the context of Example 2, below.

## 2.4 Agent Messaging Framework

The purpose of the Agent Messaging Framework is to provide a means for agents to exchange messages for the purpose of communicating facts, sending queries, and making commands, as well as for auxiliary functions that are needed for support of the messaging facility. It consists of two major parts, the *Agent Network Ontology* and the *Language for Agent Query and Response* (LAQR).

The Agent Network Ontology is expressed in terms of the type system and taxonomy. It introduces a reportoire of types along the following lines:

- Types for agents and for "containers" for agents, which are called *individuals* in this ontology.
- Types for electronic devices where agents and individuals may be located, such as desktop and laptop computers, detachable memory units, and the like.
- Types for the constructs that are used for addressing agents and devices, such as IP addresses and port numbers.
- A type for a set of agents that can be distributed across a network, called *societies.*
- Types for human users of these facilities

Each agent contains a *network map* containing entity descriptions for entities of these kinds. This is the resource whereby an agent will know about the existence of, the location of, and the properties of other agents. Each agent belongs to exactly one agent *society;* each agent society has a *registrar* agent, and the registrar agent is in charge of maintaining up-to-date information about the agents in that society, as well as information about other societies that may be accessible by, and of use for the society in question.

The *Language for Agent Query and Response* is still under development, and an ad-hoc solution is used provisionally at the present time.

## 3 Example 1: Tutorial Use

We shall quote two examples for how the Knowledge Representation Framework is being used. The first example is from the course TDDC65, "Artificial Intelligence and Lisp" which I have taught regularly as a full autumn semester course at Linköping University. This course has been reorganized so that in years 2009 and 2010 it has been based on the materials described here.

The course combines theoretical and practical aspects. For the latter, each course participant obtains his or her software agent which is an instance of the Leonardo system. The student retains this agent for the duration of the course and uses it for most of the assignments, except for two which are calculation exercises done on paper. Each of these assignments is done by downloading a software package from the course's lab server, doing the work using the agent, and using the agent for uploading the results to the lab server. Returned solutions are checked manually by myself or the teaching assistants, but using software tools in the lab server that inspect and display the uploaded solution in order to facilitate the work.

The lecture part of the course includes artificial intelligence, with an emphasis on knowledge representation and KR-based AI, but it also covers languages and systems that are used in AI, such as KIF, FIPA-ACL, PDDL, OWL, and so forth, as well as AI platform architectures such as the BDI and OAA architectures. The notational and conceptual framework of KRF is used and offers several advantages. For the theoretical and logic-based parts, it provides a notation that is essentially the same in the lecture notes, the Leonardo system for the agents, and for the lab assignments and their solutions. The use of a uniform notation from theory to systems facilitates the study.

Furthermore, the fact that the course starts by providing one coherent platform with respect to concepts and notation is an advantage when dealing with the variety of actual languages and systems, such as those mentioned above. Rather than describing one language and one architecture after the other and expecting the students to be able to relate them, we are able to first identify important design issues in terms of the chosen platform, and then describe each language and each system in terms of what design choices it has made with respect to those repeatedly used design issues.

## 4 Example 2: Website Management

The documentation of the Knowledge Representation Framework as well as the other parts of the CAISOR agenda is a combination of webpages and conventional reports in pdf format. To be precise, the webpage part of the documentation consists of a set of *hyperpages,* each of which is a set of specific HTML pages *(webnotes)* that are held together by a common menu, and of course with cross-links between the hyperpages as well as between hyperpages and reports. The webnotes may change over time, but successive versions of a webnote are retained and each webnote contains a link to a page containing its earlier history.

The Document Representation Framework in the KRF is the basis for an actual implementation, the Document Preparation Facility which is based on the Leonardo system. Several parts of the KRF framework are used for this purpose. The KRF notational system is used as the original markup language for both webnotes and documents, but it is also used for bibliographic, versioning and other metadata for those same webnotes and documents. The representation of links and of conventional citation lists is facilitated by the use of a uniform notation without unnecessary duplication of facilities.

The preparation and publication of documents of these kinds can also benefit from the agent-based architecture, where some agents are used for authoring materials for eventual posting, and other agents operate as webservers and document servers.

Besides publishing 'documents' of these kinds, our entire system also includes a modular knowledgebase, the *Common Knowledge Library* (CKL) [12] which presently contains around 65.000 separate entities, each of which carrying a number of attribute assignments. Unlike other open-source knowledgebases such as Freebase and DBpedia, the CKL is organized as a collection of modules which can be downloaded and used independently of each other. The posting of modules in the CKL has therefore many similarities with the publication of articles, and we intend to pursue that analogy further.

The modules in the Common Knowledge Library are expressed using the Knowledge Representation Framework, which means that it is straightforward for Leonardo agents to download CKL modules and integrate them into their local beliefbase. One interesting additional aspect of the CKL is that each module is required to have a supermodule containing type information that the given module shall conform to, and there are automatic routines for validating modules against their supermodules. Validation is performed on each version of a module, when new versions arrive, and the validation results are represented in the same format and are posted in the same knowledgebase. Supermodules are also required to have and to conform to super-supermodules, recursively.

## 5 Other Areas of Usage

The two examples of current use for the Knowledge Representation Framework can easily be generalized. The agent-based methods in the first example may be applicable to other educational situations that use computer-based assignments and projects for the students. In comparison with the use of social facilities on Internet, our approach has the advantage of providing a powerful platform for knowledge sources and knowledge-based processing, which may be relevant both for the work of the students themselves, and for providing some automatic feedback to the students. The addition of messaging facilities would be important and is an easy extension to the present system. The addition of a graphical user interface complementing the present command-line interface will be important in some contexts.

The example concerning document preparation support can also be generalized in a number of directions, and first of all towards an integrated

[12]http://piex.publ.kth.se/ckl/

authoring system for research publications that is able to manage both authored text, research data and bibliographic data in an integrated way, and that can provide both individual and group-level support.

The topic of *Knowledge Acquisition and Information Analysis* is one of our three primary topics, as was mentioned in the Introduction. Information Analysis can be characterized, in our view, as an activity where software tools and an existing knowledgebase are used for importing, restructuring and checking medium-sized collections of structured information, in such a way that the resulting information module is correct and reliable both with respect to its formal properties and its factual correctness. The KRF-based software tools that were used for constructing the present contents of the Common Knowledge Library are a first step towards a well-designed toolset for Knowledge Acquisition and Information Analysis.

Besides these three areas we want to mention two more candidate application areas. The *Semantic Web* is an obvious application area of the KRF and other work in the CAISOR agenda since all the basic requirements for the Semantic Web are there:

- A representation language
- A type system and ontology
- Autonomous agents with a capability for agent-to-agent communication
- Document preparation and management

One other candidate area is for *Dialog Systems* using natural language. We notice the importance of dialog engines for the design of natural-language dialog systems, and observe that most of the characteristics of a dialog engine – except its language-specific facilities – are very similar to what is required for an intelligent autonomous agent. This gives reasons for attempting to extend the approach described here in the direction of natural-language communication.