Intelligent Architectures Stewart Sims stewart@ssims.co.uk

Knowledge Management Environment Specification V1.0

This specification asserts the necessary attributes required for a Knowledge Management Environment to be Specification compliant (KMES compliance)

In the context of this document a Knowledge Management Environment (KME) is a system architecture designed to assist development and utilisation of artificially intelligent entities in computing.

Through a series of simple case studies, it may be possible to demonstrate the effectiveness of a KME implementation and the viability of the technology in practical scenarios.

The key components a KME consists of are:

- Knowledge Parser – a module which interprets meaningful statements and processes them to alter the known information of a particular Knowledge Agent via the Logic Engine. The Query Bus carries out a similar function for two-way query based communication with existing knowledge stored a Knowledge Agent. The Knowledge Parser may accept a variety of inputs and interpret them accordingly. For example it may accept text and interpret it using Natural Language Processing techniques.
- Logic Engine – a component acting as a service ready to interpret logical statements passed to it in order to update a Knowledge Agent's knowledge database and to respond to queries directed at a Knowledge Agent. It carries out the following:
    - Asserts information, rules and predicates from the Knowledge Parser
    - Manipulates the known information of the Knowledge Agents
    - Responds to queries from the Query Bus directed at a particular Knowledge Agent

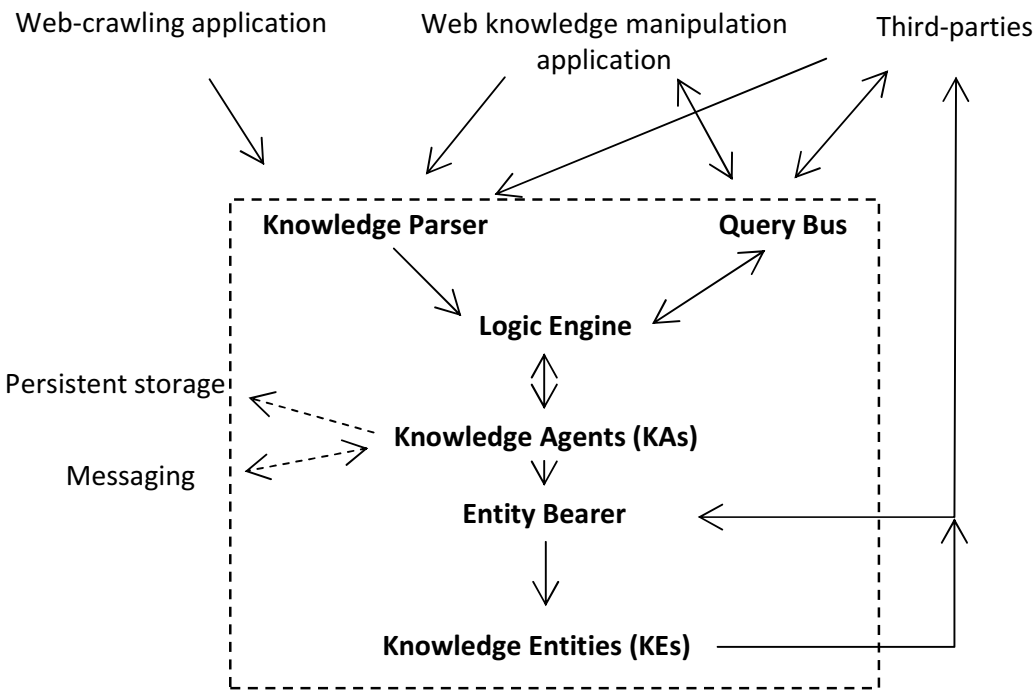
  Existing candidates for logic engines include most Prolog implementations as they are ready to use for this purpose and many have libraries which allow their features to be exposed through the internet.
- Knowledge Agents (KAs) – knowledge containing structures which live in the logic engine. They each hold a modifiable database of facts, relations and predicates. From this knowledge important queries can be resolved and inferences made.
- Entity Bearer – a module which prepares a Knowledge Agent to become detached from the KME in the form of a Knowledge Entity.
- Knowledge Entities (KEs) – an exported form of the Knowledge Agent with which it is associated. The potential exists for them to be semi-intelligent entities that can be readily integrated into other software.

Additional components a Knowledge Management Environment may consist of are; persistent storage and backup for the database of KAs, messaging module for KAs to autonomously communicate with external systems and potentially each other.

In order to maximise the scope of benefits that may be gained from such an environment, all components are intended for deployment on the internet by use of appropriate web technologies. This will allow for an interoperable interface between a KME and other software.

Intelligent Architectures Stewart Sims stewart@ssims.co.uk

Architecture Diagram



- Bounded box represents core KME components, interfaces to which are exposed through web services

This diagram represents the architecture of a KMES compliant system. It is not drawn in UML or any other modelling style because it represents the abstract components of the KME Specification. Actual implementation of the KMES standard is likely to have a more complex structure depending on the manner in which it is implemented.

The above diagram and abstract nature of the KME Specification is important as it defines the components in a way in which the technical, business and end-user audience of a KME can all relate to.

Due to the important interoperability feature of web services none of the components in the KME Specification are programming language or technology specific, although they do lend themselves to implementation with the use of many modern development technologies.

Component Specifications

Knowledge Parser

This component is the entry-point for all new knowledge to be integrated into the environment.

The knowledge parser must carry out two functions in the form of:

- *Unit Scanning* – providing an interface that can be used to send knowledge information to apply to a knowledge agent in whichever formats are desirable. It splits up data into chunks which can be consumed and interpreted by unit processing
- *Unit Processing* – the processing of the units by extracting semantic meaning and asserting this meaning as logical facts, relations and predicates into the logic engine. This may for example be carrying out natural language processing on a unit such as a phrase of text.

The Knowledge Parser must accept logical statements native to the logic engine to be compliant.

The Knowledge Parser must be exposed through web-based means, such as a web-service.

Important – the Knowledge Parser must direct units of knowledge to the relevant Knowledge Agent which will be denoted by a KA Identifier provided by the caller of the web service.

Query Bus

The query bus interprets queries, processes them and returns a verdict. It must accept logical queries native to the logic engine. The verdict can be given in either natural language, a form of data digestible by another application (e.g. XML or JSON), or a custom format dependent on the KME implementation. When processing a query in which there are any internally unknown values which are externally known there are two methods of operation:

- *Active verdict* – the query is carried out using communication with an external system to obtain the value and returns the verdict having received the required information (this could be done through KA messaging)
- *Passive verdict* – the query bus embeds a variable in the relevant section of the response which is recognisable by a user or existing system in order for them to then lookup the value

The Query Bus must be exposed through web-based means such as a web-service.

Important – the Query Bus must direct queries to the relevant Knowledge Agent which will be denoted by a KA Identifier provided by the caller of the web service.

Knowledge Agents (KAs)

KAs are data structures acting as a modifiable collection of facts, relations and predicates (knowledge). KAs are contained within the logic engine's database in its native knowledge format.

Knowledge Entities (KEs)

An exported KA ready to be utilised by another knowledge / logic based system, as well as ready to be used by the KMES implementation which generates the KE itself. There is no

standard format for such an entity, therefore there is no constraint on the format in which KEs should appear under the KMES.

Entity Bearer

The entity bearer prepares KAs for exportation in a KE format. This format must be able to be processed by the KME itself and ideally other knowledge / logic based systems.

Persistence

Provision should be provided to maintain KAs as persistent data structures within the logic engine. The KAs must be subject to storage at the relevant intervals, and able to be restored on a KME reboot scenario. Additionally important events and errors that occur as a result of usage of the KME should be caught and written out to a persistent log file.

Messaging

Optionally KAs may have the facility to send and receive messages to/from web applications, other KAs or potentially any other entity or service which can digest their data.

Scalability and Load

Scalability and load management are an important consideration when building a system which may potentially receive a large volume of requests, therefore the KME should be designed with this in mind considering its particular purpose.

Contextual Information

This section outlines the potential use cases and context for the KMES architecture. References to specific usage examples are not included, as the specification is for an abstract system. Intelligent Architectures are currently working on an implementation of the KMES called the Knowledge Management System. The aim of this system is to demonstrate the capabilities of the architecture as well as providing production quality functionality. For more information contact Intelligent Architectures.

Areas of application

The acquisition, storage and utilisation of logic based knowledge have a broad range of applications. There are many practical scenarios that can benefit from use of a knowledge based system that combines an architecture designed for careful organisation of knowledge with an easily accessible interface.

The KMES architecture allows for the select cultivation of knowledge from sources such as natural language on the web, which can then be used to provide more intelligently derived results from queries. It has the potential for implementation of a dynamic expert system comprising of multiple expert knowledge bases. It can also be used to process business logic, and act as a rules engine service. In fact a KMES implementation has the advantage of being an entirely separate system from core applications and as such can carry out multiple knowledge based services in a distributed environment.

One usage example is employing the semantic analysis of web pages to provide a web summary service which returns a customised selection of desired information from a number of pages.

An example of an expert business system could take the form of a service which assesses investment opportunities based on constantly streamed data which is persisted to the Knowledge Agents.

In addition features such as messaging capabilities of a Knowledge Agent and their exportation into Knowledge Entity format allow for some very interesting possibilities.

The knowledge paradigm

Artificial Intelligence and computational logic has advanced tremendously, however it remains a field that seems to appeal largely to an academic audience. Nevertheless many areas that have been studied have important practical applications. Acquisition and inquisition of knowledge is important to businesses and to end-users. There are many forms of knowledge representation and ‘logic’ has been selected as a suitable candidate for the KME Specification, intended to refer to ‘first order logic’ such as that interpreted by the Prolog language. Logic of this form is easy for developers to work with in its native state, lends itself to artificially intelligent procedures, and is suitable for representing a wide range of data. Additionally this allows a KMES implementation to leverage the existing development in the field. Put simply, Prolog implementations are ready to be used for knowledge oriented applications, whereas other knowledge representation techniques have much less ‘out-of-the-box’ appeal.

Future of the KMES

The problem of generic knowledge

This specification provides a detailed approach to tackling the problem of bringing logic/knowledge based programming and modern development techniques together in a generic manner.

There is however an intrinsic complexity when moving forward with this specification. This complexity comes from the fact that there are currently no universal methods under which logic based information can stored, transmitted and utilised in the world of modern web technology.

There are many attempts to standardise such information so that it can be used for a wide variety of purposes. Unfortunately no single technology exists that allows for the unification of such powerful information.

However, this does not mean that a KMES implementation is currently an unviable system in terms of practical usage.

The way forward

After consultation with an experienced software business, the potential for the architecture looks promising. The resolution to this problem of 'generic knowledge' is to examine the scenarios under which the implemented system itself may be used and to tailor the resulting system to the service which it is to provide.

The key in successful implementation is to focus on the 'customer', whom in this context is a user of the system. Whether they own the system themselves or utilise it remotely is dependant upon their requirements.

Designing a KMES implementation with a clear customer oriented goal will help to narrow down the required end-point I/O so that it can provide a valuable service in the manner in which a potential customer wishes.

KMES compliance

Future development carried out by Intelligent Architectures will include a system based on the KMES architecture. We anticipate that although this system will closely follow the KMES, there may be some differences in design. As a result of the complexities of implementation, it would seem more apt to label such an implementation as 'based on KMES'.