Knowledge Representation with KnowLang

The marXbot Case Study

Emil Vassev
Lero–the Irish Software Engineering Research Centre
University of Limerick
Limerick, Ireland
emil.vassev@lero.ie

Mike Hinchey
Lero–the Irish Software Engineering Research Centre
University of Limerick
Limerick, Ireland
mike.hinchey@lero.ie

***Abstract* — Intelligent systems are capable of AI exhibited via knowledge representation and reasoning, which helps to connect abstract knowledge symbols to real-world meanings. This paper presents a formal language for knowledge representation called KnowLang. The language implies a multi-tier specification model emphasizing knowledge corpuses, knowledge base operators and inference primitives. The approach allows for efficient and comprehensive knowledge structuring where ontologies are integrated with rules and Bayesian networks. The paper presents the KnowLang specification constructs formally along with a case study based on a mobile robotics platform.**

Keywords - knowledge representation; reasoning; robotics.

I. INTRODUCTION

When it comes to intelligent systems, one of the major problems we are facing is related to the knowledge we must transfer to the computerized machines and have them use that knowledge, so they exhibit intelligence. Modern intelligent systems have intrinsic knowledge that helps them reason about situations where autonomous decision making is required. In this regard, one of the first questions we need to answer is on the notion of knowledge. So, what is knowledge? To answer this question we should consider two facts: 1) it is known that knowledge is related to intelligence; and 2) the definition of knowledge should be given with terms from the computing domain. Scientists agree that the concept of intelligence is built upon four fundamental elements: data, information, knowledge, and wisdom. In general, data takes the form of measures and representations of the world—for example, raw facts and numbers. Information is obtained from data by assigning relevant meaning, usually by putting data in a specific context. Knowledge is a specific interpretation of information. And wisdom is the ability to apply relevant knowledge to a particular problem.

Intelligent system engineers use knowledge representation to give computerized systems large amounts of knowledge that helps them understand the problem domain. Still today computers "talk" in a "binary" language, which is simple, logical, and sound, and has no sense of ambiguity typical for a human language. Therefore, computers cannot be simply given textbooks, which they understand and use, just like human do. Instead, the knowledge given to computers must be structured in well-founded computational structures that computer programs may translate to the binary computer language. Knowledge representation structures may be primitives such as *rules*, *frames*, *semantic networks*, *concept maps*, *ontologies*, and *logic expressions*. These primitives might be combined into more complex knowledge elements. Whatever elements they use, engineers must structure the knowledge so that the system can effectively process it and humans can easily perceive the results.

Computer intelligence mainly excels at *formal logic*, which allows it, for example, to find the right chess move from hundreds of previous games. Intelligent systems might employ appropriately structured knowledge that is used by embedded inferential engines. The knowledge is integrated in such systems to build a computational model of the operational domain in which symbols serve as *knowledge surrogates* for real world artifacts, such as robot's components and functions, task details, environment objects, etc. The domain of interest can cover any part of the real world or any hypothetical system about which one desires to represent knowledge for computational purposes.

In this paper, we present a formal language called KnowLang, developed for the purpose of employing *knowledge representation and reasoning* (KR&R) in intelligent systems, e.g., cognitive robotic systems. The language is explained via a case study from the robotics domain.

The rest of this paper is organized as follows. Section II covers some related work and provides a brief overview of KnowLang. Section III presents a KR case study based on a mobile robotics platform and finally, Section IV provides brief concluding remarks and a summary of our future goals.

II. BACKGROUND

The application of KR&R to robotic systems has been an increasingly interesting topic for intelligent systems. Examples are found in semantic mapping [1], improving planning and control aspects [2], and most notably HRI systems [3, 4]. Many conventional developers doubt the utility of KR. The fact is that KR&R can significantly slow a system down when it has to decide what actions to take, and

it looks up facts in a knowledge base to reason with them at runtime. This is one of the main arguments against knowledge representation. Why not simply "compile out" the entire knowledge as "procedural knowledge", which makes the system relatively faster and more efficient. However, this strategy will work for a fixed set of tasks, i.e., procedural knowledge will give the system the entire knowledge the system needs to know. However, AI deals with an open set of tasks and those cannot be determined in advance (at least not all of them). This is the big advantage of using knowledge representation – AI needs it to solve complex problems where the operational environment is non-deterministic and a system needs to reason at runtime to find missing answers.

KnowLang is an initiative undertaken by Lero – the Irish Software Engineering Research Center within Lero's mandate in the ASCENS project. Autonomic Service-Component ENSembles (ASCENS) [5] is an FP7 (Seventh Framework Program) [6] project targeting the development of a coherent and integrated set of methods and tools providing a comprehensive development approach to developing *ensembles* (or swarms) of intelligent, self-aware and adaptive *service components*. One of the main scientific contributions that we expect to achieve with ASCENS is related to KR&R. A key feature of KnowLang is a multi-tier specification model (see Figure 1) allowing for integration of ontologies together with rules and Bayesian networks [7]. The language aims at efficient and comprehensive knowledge structuring and awareness based on logical and statistical reasoning. It helps us tackle 1) explicit representation of domain concepts and relationships; 2) explicit representation of particular and general factual knowledge, in terms of predicates, names, connectives, quantifiers and identity; and 3) uncertain knowledge in which additive probabilities are used to represent degrees of belief. Other remarkable features are related to knowledge cleaning (allowing for efficient reasoning) and knowledge representation for autonomic robotic behavior.

KnowLang imposes a multi-tier specification model (see Figure 1), where we specify *knowledge corpuses*, *KB (knowledge base) operators* and *inference primitives* at different hierarchically organized tiers. As shown in Figure 1, knowledge is organized in a special Knowledge Base (KB) at three main tiers: 1) Knowledge Corpuses; 2) KB Operators; and 3) Inference Primitives. The tier of Knowledge Corpuses is used to specify KR structures. The tier of KB Operators provide access to Knowledge Corpuses via special class of ASK and TELL operators where ASK operators are dedicated to knowledge querying and retrieval and TELL operators allow for knowledge update. Moreover, this tier provides for special *inter-ontology operators* intended to work on one or more ontologies. Note that all the KB Operators may imply the use of *Inference Primitives*, i.e., new knowledge might be inferred and eventually stored in the KB. The tier of Inference Primitives is intended to specify algorithms for reasoning and knowledge inference. In this paper, we do not present the language itself, but rather how it can be used to specify knowledge in robotic systems. The interested reader is advised to refer to [8] for more information on the KnowLang's specification model.

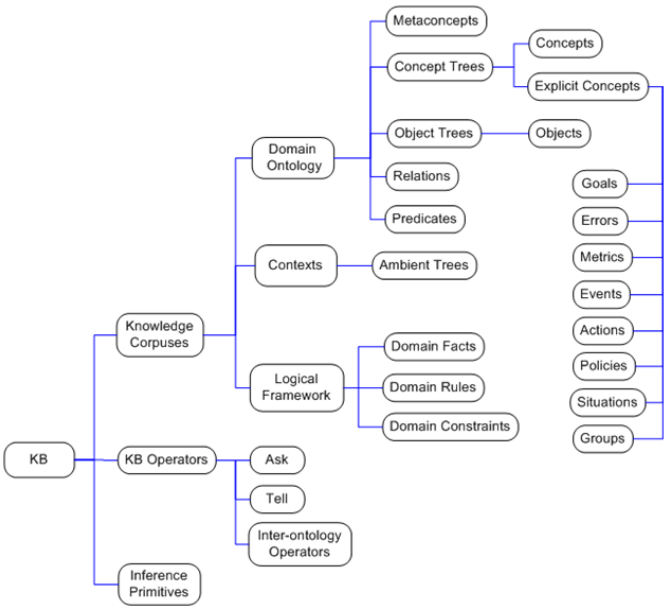

Fig. 1. KnowLang Multi-tier Specification Model

III. KR Case Study

KnowLang has been applied to derive an initial KR structures for the marXbot mobile robotics platform [9, 10].

A. The marXbot Robot

The marXbot [9, 10] is a modular research robot equipped with a set of devices that help the robot interact with other robots or the robotic environment. The environment is defined as an arena where special cuboid-shaped obstacles are present in arbitrary positions and orientations. Moreover, the environment may contain a number of light sources, usually placed behind the goal area, which act as environmental cues used as shared reference frames among all robots.

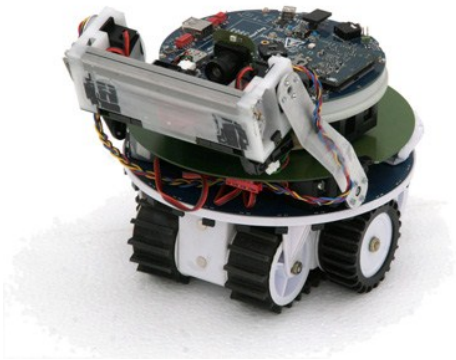

Fig. 2. A marXbot Robot [10]

Figure 2 shows a marXbot robot [10]. Such robot is equipped with a set of devices to interact with the environment and with other robots of the swarm [9]:

- a light sensor, that is able to perceive a noisy light gradient around the robot in the 2D plane;
- a distance scanner that is used to obtain noisy distances and angular values from the robot to other objects in the environment. Its range is 1.5 meters.
- a range and bearing communication system [11], with which a robot can communicate with other robots that are in line of sight. Its range is 4 meters.
- a gripper, that is used to physically connect to the transported object;
- two wheels independently controlled to set the speed of the robot.

Currently, the marXbots robots are able to work in teams where they coordinate based on simple interactions on group tasks. For example, a group of marXbots robots may collectively move a relatively heavy object from point ***A*** to point ***B*** by using their grippers.

B. KR for marXbot Robot

Figure 3 depicts a concept tree with a tree root "Thing". The concept "Thing" is determined by the meta-concept "Robot Thing", which carries information about the interpretation of the root concept "Thing" such as "Thing is anything that can be related to the robot". According to this concept tree there are two categories of things in a robot: *entities* (physical entities) and *virtual entities*, where both are used to organize the vocabulary in the internal robot domain. Note that all the explicit concepts (see Figure 1) are presented as concepts in this concept tree – qualified path "Thing→Virtual Entity→Phenomenon", i.e., in this ontology tree, the explicit concepts inherit the concepts "Phenomenon", "Function" and "State". Note that within the scope of any concept tree (or object tree) the concept names (or object names, respectively) must be unique.

The following KnowLang code presents the actual specification of the *Locomotion_System* concept. Due to space limitations, we do not present the language syntax, which can be easily grasped from the example below.

```
CONCEPT Locomotion_System {
 CHILDREN {}
 PARENTS { SC.Thing..System }
 STATES { STATE operational {} STATE on {} STATE off {} }
 PROPS {
  PROP engine { TYPE {SC.Thing..Engine} CARDINALITY {1} }
  PROP wheel {  TYPE {SC.Thing..Wheel}  CARDINALITY {5}  }
  PROP locomotion_soft { TYPE {SC.Thing..Locomotion_Soft }  CARDINALITY {1} }
  PROP battery {  TYPE {SC.Thing..Battery }  CARDINALITY {1} }
 }
 FUNCS {
  FUNC move {
   TYPE {SC.Action.Move }
   PRE_CON {}
   POST_CON {}
   PARAMS { SC.Thing..Direction }
   RETURN {}
   BODY { IMPL }
   ERRORS { }
  }
  FUNC stop {
   TYPE {SC.Action.Stop }
   PRE_CON {}
   POST_CON {}
   BODY { IMPL }
   ERRORS { }
  }
  FUNC turn {
   TYPE {SC.Action.Turn }
   PRE_CON {}
   POST_CON {}
   PARAMS { SC.Thing..Direction, SC.Thing..Angle}
   BODY { IMPL }
   ERRORS { }
  }
}
```

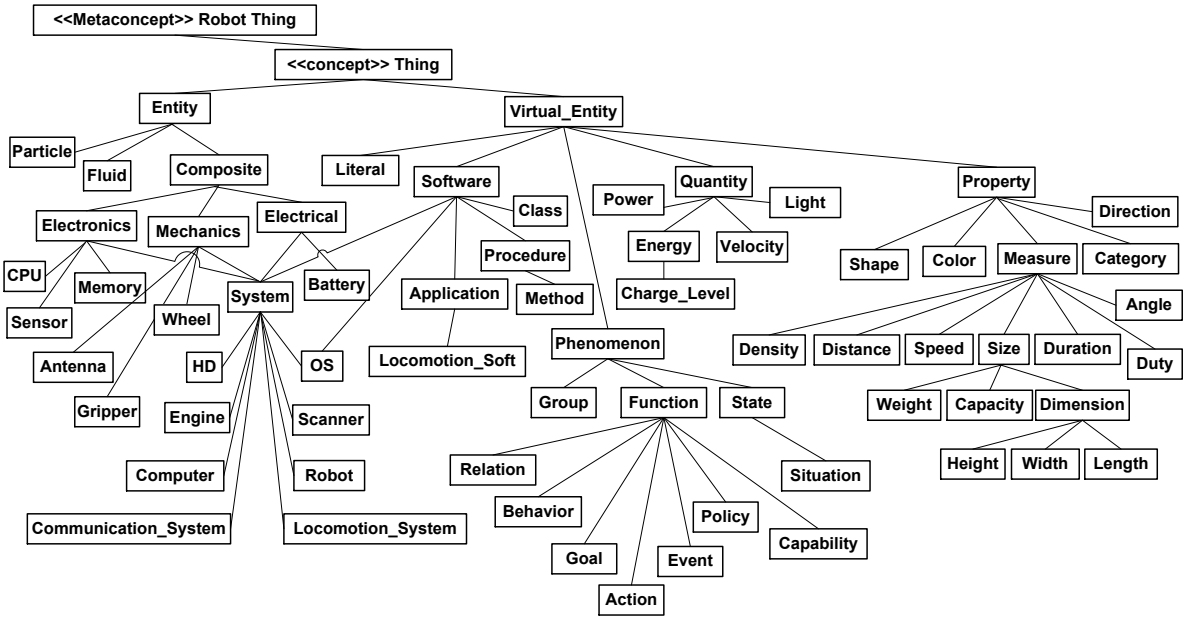

Fig. 3. Concept Tree: "Robot Thing"

Note that, every concept specified with KnowLang has an intrinsic attribute *STATE* that may be associated with a set of possible *state values* the concept instances may be in. The *STATE* attribute is a concept descending from the *State* concept (see Figure 3). A system may occupy a new state when values of concept properties have been changed or some events or actions have occurred in the system or the environment [8]. Therefore, a state can be determined by values held by concept properties, events or actions. Thus, a state of a complex concept might be the product of the states of its properties. Only significant states should be specified and evaluated by using predicates. For example, the predicate *Is_Operational* evaluates whether a concept instance is in *operational* state. For example,

```
Predicate.Is_Operational(THIS.locomotion_system)
```

For example, we may consider the states of the following concept instances: **robot[1]**, **robot[1].locomotion_system**. The possible sets of state values associated with these states could be:

```
robot[1].STATES := { moving_forward, pursuing_goal_B, pursuing_goal_A,
                     operational, on, off }
robot[1].locomotion_system.STATES := { operational, on, off }
```

Further, we specify the concept *Capability* (see Figure 4), which descends from the *Function* concept (see Figure 3) and couples a Function with elements that increase its depth, scope, productivity, etc. Capability may carry information about possible range, limits, etc..

The concept *Action* (see Figure 5) descends from the *Function concept* (see Figure 3) and defines the entire robot's functionality as possible actions. Moreover, actions are used to specify the functions of a concept.

The concept *Relation* (see Figure 6) descends from the *Function* concept (see Figure 3) and defines the entire robot's set of relation terms used to build the Ontology's relations (see Figure 1). As shown in Figure 6, we introduce a special Relation concept termed *Means*. This makes it possible to express that a certain word refers to a certain entity, like in the following example:

```
RELATION { Relation.Means  ("robot one", Object..robot[1]) }
```

This allows us to deal with synonymy and ambiguity. Other possible Relation specifications are as follows:

```
RELATION { Relation.Instance_Of  (object.robot[1].locomotion_system,
           Thing..Locomotion_System) }
RELATION { Relation.Part_Of  (object.locomotion_system, object.robot[1]) }
RELATION { Relation.Engrouped  (object.robot[1], object.robot[2], 1) }
```

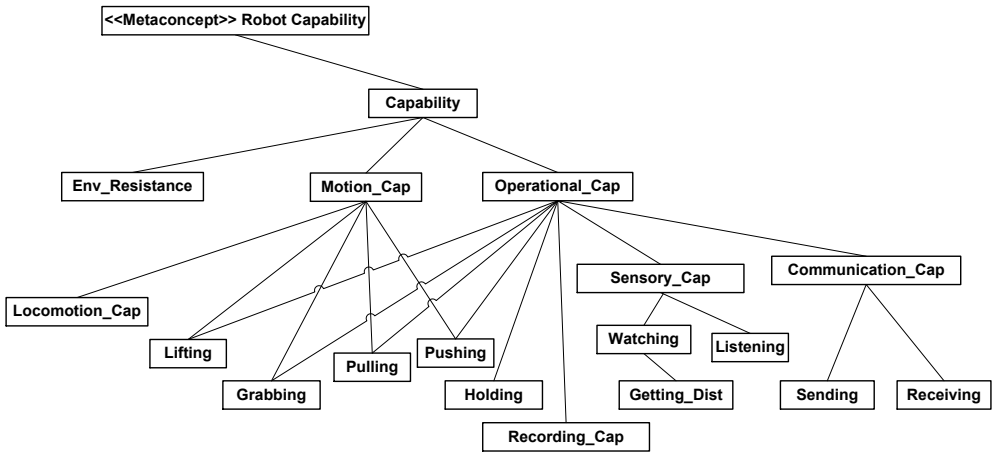

Fig. 4. Concept Tree: "Robot Capability"

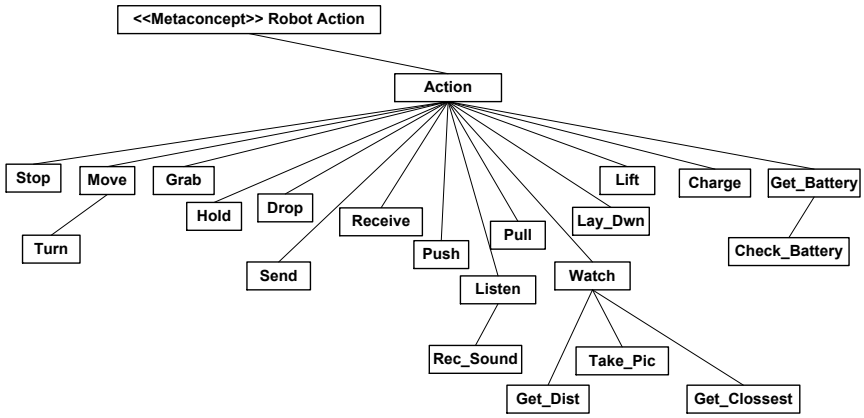

Fig. 5. Concept Tree: "Robot Action"

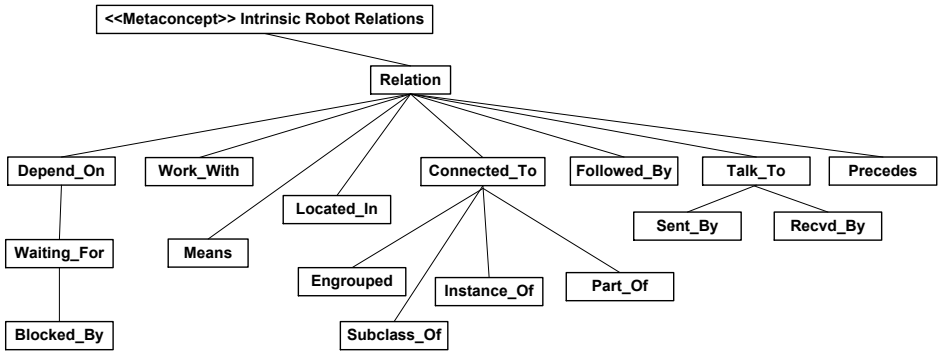

Fig. 6. Concept Tree: "Relation"

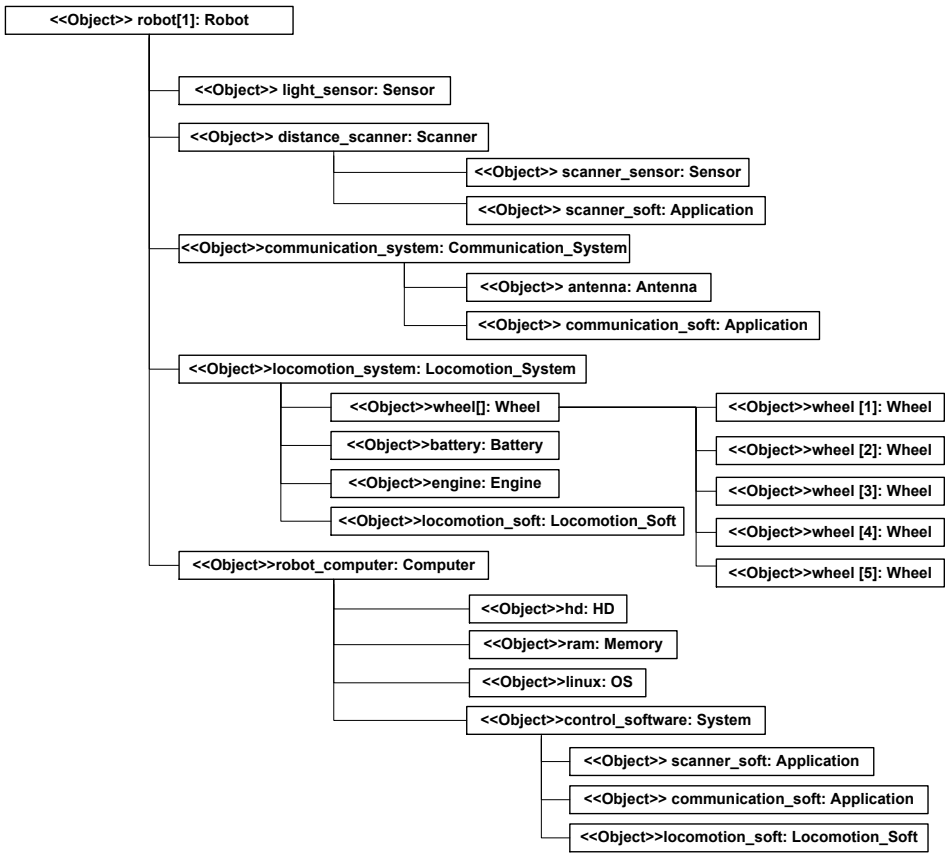

Fig. 7. Object Tree: "robot[1]"

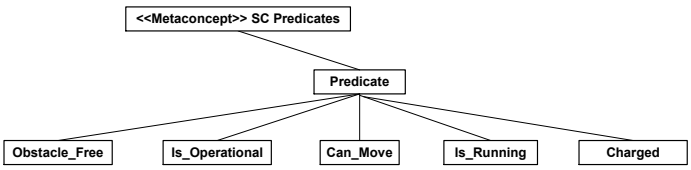

Fig. 8. Concept Tree: "Predicate"

Figure 7 depicts a possible *object tree* of the marXbot Robot Ontology. As shown, the *Robot[1] Object Tree* shows the *object properties* of the *robot[1]* object.

Predicates (e.g. Is_Operational) are specified by another ontology tree (*SC.Predicate*) as shown by Figure 8. Note

that predicates must be specified both syntactically and semantically.

Facts by definition specify true statements in ontology, e.g., implication. The following examples present some facts.

```
FACT {
  Predicate.Work_With(object.robot[1], object.robot[2]) =>
    Predicate.Engrouped(object.robot[1], object.robot[2]) }

FACT {
  Predicate.Is_Operational(THIS.locomotion_system) AND
    Predicate.Obstacle_Free(THIS) => Predicate.Can_Move(THIS)  }
```

Rules:

- can be used to specify simple behavior, e.g.:

```
RULE {
  IF  NOT Predicate.Can_Move(THIS) THEN  {
    DO {Action.Check_Battery(THIS..battery);}
  }
}
RULE {
  IF  NOT Predicate.Can_Move(THIS) AND Action.Get_Battery(THIS..battery) > 0.5
  THEN {
    DO {Action.Get_Dist(THIS, Action.Get_Closest(THIS, ENV.Thing..Obstacle)); }
  }
}
```

- can be used to imply predicates, e.g.:

```
RULE {
  IF Action.Get_Battery(THIS..battery) > 0.9  THEN  {
    Predicate.Charged(THIS..battery)
  } ELSE {
    NOT Predicate.Charged(THIS..battery)
  }
}
```

Constraints:

- may constraint the behavior, e.g.:

```
CONSTRAINT {
  IF Action.Get_Battery(THIS..battery) < 0.1 THEN { NOT Action.Move(THIS) }
}
```

- may impose predicates, e.g.:

```
CONSTRAINT {
  IF Predicate.Is_Operational(THIS.locomotion_system) THEN  {
    Action.Get_Battery(THIS..battery) > 0.5  AND
    Predicate.Is_Operational (THIS..wheel[1])  AND
    Predicate.Is_Operational (THIS..wheel[2])  AND
    Predicate.Is_Operational (THIS..wheel[3])  AND
    Predicate.Is_Operational (THIS..wheel[4])  AND
    Predicate.Is_Operational (THIS..wheel[5])  AND
    Predicate.Is_Operational (THIS..engine)  AND
    Predicate.Is_Operational (THIS..locomotion_soft)  AND
    Predicate.Is_Running (THIS..locomotion_soft)
  }
}
```

- may impose data restrictions, e.g., presume we want to two robots to have different first goals:

```
CONSTRAINT { robot[1].goal[1] <> robot[2].goal[1]; }
```

IV. Summary and Future Work

This paper has presented an approach to KR&R (Knowledge Representation and Reasoning) for intelligent systems, e.g., cognitive robotic systems. The problem is tackled by a framework called KnowLang implying a multi-tier specification model that allows for integration of ontologies together with rules and Bayesian networks. The goal is efficient and comprehensive knowledge structuring and awareness based on logical and statistical reasoning. This is provided via 1) explicit representation of domain concepts and relationships; 2) explicit representation of particular and general factual knowledge, in terms of predicates, names, connectives, quantifiers and identity; and 3) handling uncertain knowledge where additive probabilities are used to represent degrees of belief.

The KnowLang approach to KR has been demonstrated with a case study where the language has been applied to specify knowledge in a mobile robotics platform called marX bot. Note that KnowLang is still under development as part of the ASCENS international European project [5]. Our plans for future work are mainly concerned with further and complete development of KnowLang including a toolset for formal validation. Once fully implemented, KnowLang will be used to specify knowledge representation and autonomic behavior for the ASCENS case studies.

Acknowledgement

This work was supported by the EU FP7 Integrated Project Autonomic Service-Component Ensembles (ASCENS) and by Science Foundation Ireland grant 03/CE2/I303_1 to Lero—the Irish Software Engineering Research Centre at University of Limerick, Ireland.

References

[1] C. Galindo, J. Fernandez-Madrigal, J. Gonzalez, and A. Saffiotti, “Robot task planning using semantic maps”, Robotics and Autonomous Systems, Vol. 56 (11), 2008, pp. 955–966.

[2] O. Mozos, P. Jensfelt, H. Zender, G.-J. M. Kruijff, and W. Burgard, “An integrated system for conceptual spatial representations of indoor environments for mobile robots,” Proc. of the IROS 2007 Workshop: From Sensors to Human Spatial Concepts (FS2HSC), San Diego, CA, USA, November 2007, pp. 25–32.

[3] G.-J. M. Kruijff, P. Lison, T. Benjamin, H. Jacobsson, and N. Hawes, “Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction,” Symposium on Language and Robots, Aveiro, Portugal, 2007.

[4] H. Holzapfel, D. Neubig, and A. Waibel, “A dialogue approach to learning object descriptions and semantic categories,” Robotics and Autonomous Systems, vol. 56 (11), 2008, pp. 1004–1013.

[5] ASCENS – Autonomic Service-Component Ensembles, 2010, http://www.ascens-ist.eu/.

[6] European Commission – CORDIS, Seventh Framework Program (FP7), http://cordis.europa.eu/fp7/home_en.html.

[7] R. Neapolitan, Learning Bayesian Networks, Prentice Hall, 2003.

[8] E. Vassev and M. Hinchey, “Knowledge Representation for Cognitive Robotic Systems”, Proceedings of the 15th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing Workshops (ISCORCW 2012), IEEE Computer Society, 2012, pp. 156-163.

[9] M. Bonani , V. Longchamp, S. Magnenat, P. Retornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada, “The MarXbot, a Miniature Mobile Robot Opening new Perspectives for the Collective-robotic Research”, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010), 2010.

[10] M. Bonani, T. Baaboura, P. Retornaz, F. Vaussard, S. Magnenat, D. Burnier, V. Longchamp and F. Mondada, marXbot, Laborotoire de Systemes Robotiques (LSRO), École Polytechnique Fédérale de Lausanne, http://mobots.epfl.ch/marxbot.html.

[11] J. Roberts, T. Stirling, J.-C. Zufferey and D. Floreano, “2.5D infrared range and bearing system for collective robotics”, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), 2009, pp. 3659–3664.