Ontology utilization in MARABU – a support system for modeling, simulation and control design

Ivana Budinská
Institute of Informatics SAS
Dúbravská cesta 9
845 07 Bratislava
Slovakia
utrrbudi@savba.sk

T.- Tung Dang
Institute of Informatics SAS
Dúbravská cesta 9
845 07 Bratislava
Slovakia
utrrtung@savba.sk

Abstract – **The paper describes a support system for modeling, simulation, and control system design, based on the user's requirements. The system integrates some modeling, simulation, and control design tools. It enables users to model a real system, design control system and then to simulate results. The user's requirements are grabbed via so called questionnaire. The user is lead through a system of questions, similarly to an expert system, with the aim to specify a real system as in depth as it is possible. In order to manage user's requirements and to match the best tools to the requirements, ontology of integrated tools, methods and theories is created. Each tool, or method is characterized by a set of attributes – properties, which are captured in ontology. The reasons of ontology creation for system MARABU are discussed.**

I. INTRODUCTION

The system MARABU is based on Multi Agents technology and it is intended to serve experts and technical staff from manufacturing enterprises, and students of technical schools, as well, to find a solution (a model, a control algorithm, etc.) on the basis of user's requirements. User's requirements are entered to the system through an interactive questionnaire.

The system consists of three layers:

1. Interaction layer
2. Process layer
3. Knowledge base layer

An interaction layer: provides an intelligent user's interface. The core of the interaction layer is a User agent (UA). Interaction layer is crucial for capturing user's requirement. User's requirements describe a current system, which the user wants to model, control or simulate. To avoid complicated transformation of user's requirements into the form that is comprehensible for other agents, users are lead through questionnaire and are asked to enter attributes' values. These values correspond to attributes, by which tools, methods, and algorithms in Modelling, Control and Simulation Blocks are characterized and determined. The questionnaire is built similarly to an expert system.

A process layer: is o core of the support system. It consists of three blocks: Modelling, Control System Design, and Simulation. Each block is composed from a corresponding agent (Monitoring agent MA, Control system design agent CA, Simulation agent SA) with a communication facility, search engine, and a reasoner. A Generic block (BG) is a multi agent system, which is responsible for managing activities of all other agents, and provides interfaces among the system, users, and external applications. There is one more agent in the system – Monitoring agent. The role of this agent is to monitor activities of other agents and accordingly update database. This is the only agent that can write into database. MA, CA, and SA can only read in the database and search for solutions that respond to user's requirements.

A knowledge base layer: The system works in two regimes:

1. finding an appropriate tool, method, or algorithm
2. searching in a case-base.

Therefore the database is organised as a structured database. The database for tools, algorithms, and methods, stores metadata for each item. There are two types of metadata associated to each item in that part of the database. There exist data describing the item – attributes, and data that enable to locate the item – link, address, etc. The other part of the database is called a case-base library. It contains a list of some ready-to-use cases and associated solutions.

Section 2 describes in more details architecture of the system MARABU, agents that compose the system, and their behavior. Section 3 is addresses to a knowledge base, especially to ontology creation. Section 4 deals with an inference engine and describes some methods to discover the required solution. Also utilization of a Case-based reasoning (CBR) is introduced there. Section 5 contains an example of otology for MARABU system. Future work and implementation aspects are discussed in the conclusion.

II. MAS ARCHITECTURE OF MARABU

The core of the system is created by a Generic block (see Fig.1), which consists of the following agents:

Monitoring Agent (MoA) - follows the system behaviour after applying the recommended method for designing the control. If all the requirements are satisfied, then MoA updates the database by newly achieved results. That means the system stores all solutions for the next reuse and application. Otherwise, the MA and CA have to repeat their calculations.

User Agent (UA) – interposes communication between a user and the system. UA transfers user's requirements to the system's agents and on the other hand return solution from the system's agent to the user in a human readable form. There is one UA per user in the system.

Modelling Agent (MA) – after receiving information about the process, search for appropriate modelling tools and algorithms on the basis of user's requirements. It returns suggested algorithms and ask for more precise information according to a chosen algorithm. The final decision on which algorithm and/or tool has to be chosen is up to the user. Finally, Modelling agent returns a model of described production process.

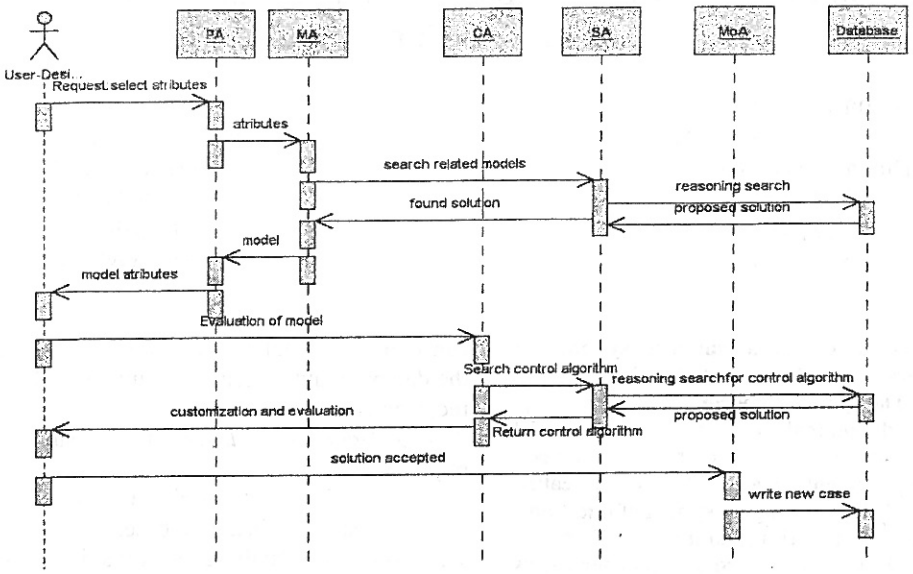

Fig. 1 - Schema of communication among agents.

Control Agent (CA)- receives a finally chosen model with all necessary attributes defined. On the basis of the model, CA searches for an appropriate control algorithms in the database. Through UA it negotiates with the user and finally chooses appropriate control algorithms. The user has to choose an algorithm from the suggested ones and specify all needed values for that. CA returns control algorithm for the process according user's specifications.
Simulation Agent (SA) – is responsible for simulation of control for the chosen model and control algorithm with the aim to help the designer to assess the proposed solution.
Agents in a proposed system cooperate to find a solution to satisfy user's requirements. should be tested and when the proposed adapted solution shows to be good, it is entered to a database as a new case related to a new solution.
An algorithm for assisting a user in modelling, designing the control system and simulation is as follows:

Initialisation: filling database with default data. There are two types of data in the database: description of cases – basic attributes, set of default solutions. There are relations among default cases and default solutions.

Input: User's requirements are given through PA. Users are supervised to fill in a questionnaire, where basic attributes for the current situation (description of production process) are described. After user's requirements are recorded, MASS begins search for appropriate modelling tool.

1. MA asks SeA to find a similar situation from database of cases. Requirements: search and data mining algorithms.
2. After finding similar case, related solutions – a method for modelling, are sent to the user. Otherwise a new search is executed as long as a solution is found.
3. CA receives a (mathematical) model identified by MA and numerical parameters entered by the user. On the basis of numerical parameters of the model, CA adapts the past solution to the current model and designs the control system.
4. MoA follows the system behaviour after applying the designed control system.
5. SA receives mathematical model and design of control system. Its goal is to simulate proposed control system and verify it.

A basic schema of communication among agents is depicted in Fig.1.

A multi-agent support system development is based on Java technology. In order to develop a system in compliance with FIPA standards and specifications, JADE – Java Agent Development framework was chosen for the project. JADE is also used as an agent middleware that implements an agent platform. Agents are implemented as one thread per agent. Advantage of MAS is in parallel processing of tasks, which is enabled using JADE. The platform provides a GUI for the remote management, monitoring, and controlling of the status of agents, allowing e.g. to stop and restart agents. It enables also to monitor messages and communication among agents. JADE also supports scheduling of cooperative behaviour, it schedule tasks in effective way. The important feature of JADE is integration with JESS, where JADE provides the shell of the agents and JESS provides the inference engine that enables necessary reasoning. Although there exists some predefined rules for reasoning in JADE and JESS it is required for the MARABU system to develop new rules to execute valuable reasoning based on the user's requirements. The problem is, that inference engines offers tools for rule-based reasoning. The system MARABU reasoning is based on cases, so a case-based reasoning

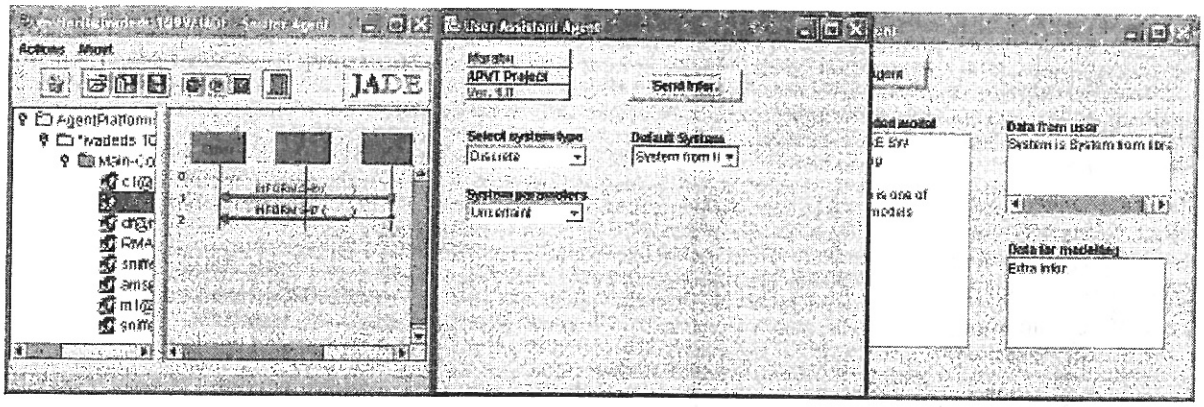

Fig. 2 - A screenshot of the system MARABU

has to be integrated into the system. A screenshot of the system MARABU is in Fig.2.

III. DATABASE VS. ONTOLOGY

Agent-based software integration involves designing ontology for a specific domain and integration of different tools to allow agents communication using that ontology. Ontology presents a shared understanding about a certain specific domain. To build ontology for a system MARABU a cooperation of experts from the area of control theory is essential. The system intends to integrate methods and tools that are known from various field of research in the area of modelling, control and simulation of continuous and discrete events dynamic systems. Also system with distributed parameters are integrated. There are huge amount of known methods, tools and algorithms, but not many of them can be integrated into a distributed support system. Each method, tool, and algorithm have to be described using specific attributes which enables to recognise, which item is applicable for which manufacturing system. Also manufacturing system has to be described using specified attributes, which matches to the attributes of methods, tools and algorithms.
In order to make the system as useful and effective as possible, on ontology approach for knowledge representation was chosen.

In comparison to a database, ontology allows to handle not only numerical transactional data. It is suited to model unstructured informal knowledge. However the advent of object oriented databases, improved logics and faster inference is making the distinction between DBs and ontologies more fuzzy, i.e. there is (multiple) inheritance, strong encapsulation, fuzzy set algorithms, meta-data standards, neural networks to train, discover and disambiguate meaning, and increased computing power so we can cast a larger window to get at context. Also ontology enables processing knowledge and data, the most important role of ontology is in defining sharing meaning, emergence and discovery of gaps and for improving tacit knowledge transfer. If an organization or group can leverage their language, e.g. by having shared patterns, they can learn, become more aware and respond faster. Besides ontology, a knowledge base may contain information specified in a declarative language such as logic or expert-system rules, but it may also include unstructured or unformalized information expressed in natural language or procedural code.

Knowledge representation is the application of logic and ontology to the task of constructing computable models for some application domain. Each of the three basic fields: logic, ontology, and computation; presents a different class of problems for knowledge sharing:

- *Logic*. Different implementations support different subsets and variations of logic. Sharing information between them can usually be done automatically if the information can be expressed in the common subset. Other kinds of transfers may be possible, but some of the information may be lost or modified.

- *Ontology*. Different systems may use different names for the same kinds of entities; even worse, they may use the same names for different kinds. Sometimes, two entities with different definitions are intended to be the same, but the task of proving that they are indeed the same may be difficult or impossible.

- *Computation*. Even when the names and definitions are identical, computational or implementational side effects may cause the same knowledge to behave differently in different systems. In some implementations, the order of entering rules and data may have an effect on the possible inferences and the results of computations. Sometimes, the side effects may cause a simple inference on one system to get hung up in an endless loop on another system.

In the system MARABU knowledge is modelled using an OWL – Web Ontology Language. The application of the OWL format for ontology for the agent system is relatively new. One advantage of owl ontology is the availability of tools that can reason about it. Tools provide generic support that is not specific to the particular subject domain. Building useful and reliable reasoning system is not a simple effort. Constructing ontology is easier. Constructing ontology in owl enables to benefit from third party tools based on the formal properties of the OWL language.

OWL ontology is a sequence of axioms and facts, plus inclusion references to other ontologies, which are considered to be included in the ontology.

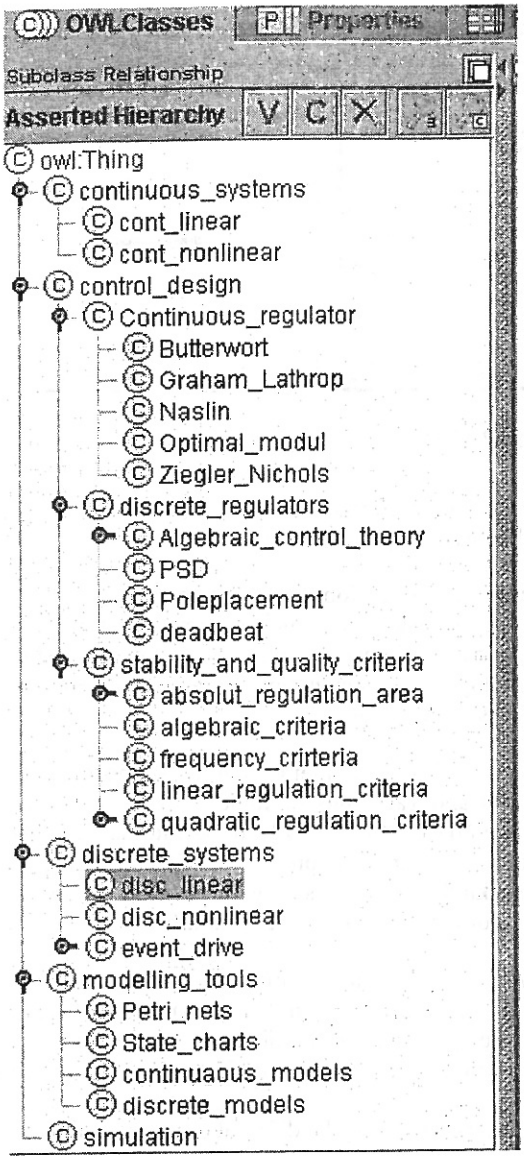

Fig. 3 – Ontology for MARABU using PROTÉGÉ, an ontology editor

IV. EXAMPLE OF ONTOLOGY FOR MARABU

Ontology for MARABU contains two basic parts:

1. Ontology of processes
2. Ontology of modeling, control, and simulation tools, methods, and algorithms.

Ontology of processes:

- Continuous
  - a. Stochastic
    - i. Linear
      - . time variant
      - . time invariant
    - ii. Nonlinear
      - . time variant
      - . time invariant
  - b. deterministic
    - i. Linear
      - . time variant
      - . time invariant
    - ii. Nonlinear
      - . time variant
      - . time invariant
- Discrete
  - a. Linear
    - i. time variant
    - ii. time invariant
  - b. Nonlinear
    - i. time variant
    - ii. time invariant
  - c. Nonlinear fuzzy
  - d. Nonlinear fuzzy-neuron
  - e. Nonlinear neuron
- Hybrid
  - a. Linear
    - i. time variant
    - ii. time invariant
  - b. Nonlinear
    - i. time variant
    - ii. time invariant

Ontology of methods, tools, and algorithms is built on the basis its attributes. All methods, algorithms, and tools that are integrated into the system is characterized thorough common attributes. When an attribute is not applicable for the specific tool, method, or algorithm, it is set to zero. Attribute value representation is the basis for establishing reasoning mechanism within the system.

V. USER'S INTERFACE – QUESTIONAIRE

In the system MARABU a specific user's interface is designed. Users access the system through an User agent. The user is led through a system of questions, which is called multi-level questionnaire. The questionnaire is built similarly to expert systems and users follow the steps in the questionnaire. The user enters his/her requirements by answering questions. On the basis of answers, the next level of the questionnaire is opened for him/she.

An example of user's questionnaire:

Q: What do you want to do in the system?

- Modeling
- Control
- Simulation

(Users can sequentially model the system, design a control algorithm, and simulate the designed solution).

On the basis of choice, the next level of questionnaire is opened:

1. Modeling:

   Q: What kind of system do you want to model?

   - Continuous
   - Discrete
   - Hybrid

(The second level is similar also for control and simulation).

- Continuous:
  1. Q: Is your system stochastic, or deterministic?
     - Stochqstic
     - Deterministic

The system of questions is built on the basis of modeled ontology for integrated tools, methods, and algorithms.

VI. INFERENCE ENGINE AND CBR

Three basic element for the GB (Generic Block) have to be defined:
Attributes – is defined within ontology as properties (slots) for each concept.
Rules – is defined within an inference engine and serves to find solution for the user on the basis of user's requirements
Control strategies – for searching solutions on the basis of predefined rules. The known control strategies used for CBR are:

- Forward chaining
- Backward chaining

A Case Based Reasoning (CBR) is used within the system to find a solution that matches the best to the user's requirements. The structured database is built independently for all three block of the systems; however all these databases are related to each other. The databases contain a case library, and a set of solutions related to these cases. Objects in the databases are represented by attributes. For an attribute-value representation a couple of methods can be used to measure a similarity degree. A method of weighed attributes is used to asses similarity between a current case (that the user has defined) and a case in the case base. The weights allow expressing the importance of all attributes.
Two different methods to evaluate attributes are used: user specific weights, which are given as a part of user's requirement through the user's questionnaire; and case specific weights (weights are specified as a part of structured database). The third option is to use combination of user specific and case specific weights to find the solution that fulfil the users' expectations the most and do not neglect any important attributes given by case specific weights.
In this paper, two methods proposed to extract information from the database are presented. The first method is based on the similarity degree between two arbitrary situations; the second one is built on the inductive reasoning principle.

The first method works on the following principle: the agents calculate the similarity between two arbitrary situations by comparing all their attributes. Relationships among attributes are evaluated by any number (real or fuzzy) that reflects how much a solution of one situation is useful for the second one, with respect to these attributes. The similarity degree is defined by a combination of those partial relationships.

The second possible method is an inductive reasoning that works as follows: starting with the most important attribute, the agents sort and filter all cases that have the same or *to a certain degree* the same attribute like the target one. This cycle continues with less important attributes, until only one candidate remains. The last case remained is considered as the most similar to the target one.

Let $re(el_1, el_2)$: $\{EL \times EL\} \rightarrow [0,1]$ be a relation expressing the dependence between two attributes $(el_1, el_2) \in \boldsymbol{EL}$. There are some important properties of this relation. [4]

- $re(el_1, el_2) \neq re(el_2, el_1)$, resp. $\neq (1 - re(el_2, el_1))$, i.e. this relation is not symmetric or inverse.
- $re(el_1, el_2) = 1$; when the solution proposed for element el_2 could be applicable to element el_1 without changes. For example, el_1 is a linear system with complete information; el_2 is a linear system with parametrical uncertainty.
- $re(el_1, el_2) = 0$; when both the elements have disjoint domains of effects, i.e. the solution proposed for el_1 is useless for element el_2. For example, el_1 is a discrete event system and el_2 is a linear system.
- $\forall\, el \in \boldsymbol{EL}$; $re(\varnothing, el) = 1$ a $re(el, \varnothing) \geq 0$

The similarity degree between two arbitrary situations $Sim(,)$ is defined as follows:

Suppose a set of cases C of one class – manufacturing system with a discrete or continuous production process represents a case. Each case is described by a set of its attributes A.

$$C = \{c_1, c_2, \ldots\} \tag{1}$$

$$c_i = \{a_{i1}, a_{i2}, \ldots a_{in}\}; \qquad a_{ij} \in A \tag{2}$$

A is a set of attributes that can be recognized in any case. If an attribute is not related to the case, a value is set to zero.
Let's define a set of solutions:

$$S = \{s_1, s_2, \ldots\} \tag{3}$$

$$s_i = \{el_{i1}, el_{i2}, \ldots\} \tag{4}$$

A Case Base is defined as a space of case descriptions and case solutions:

$$CB = C \times S \tag{5}$$

User defines a case through attributes and the system starts to search for similar cases in a CB.
Let a current case is defined as follows:

$$cc_i = \{ca_{i1}, \ldots, ca_{in}\} \tag{6}$$

where ca_{ij} is an attribute j of a current state i.

The general problem of similarity assessment is (according to [1]):

$$sim\left((cc_1, \ldots, cc_n)(c_1, \ldots, c_n)\right) = \Phi\left(sim_1(cc_1 c_1), \ldots, sim(cc_n c_n)\right) \tag{7}$$

A traditional similarity measures for individual attributes is measured by a function Φ that is monotone increasing in every argument.

Inductive reasoning method extracts the desired situation by performing a search of a decision tree, which involves all possible historical cases satisfying certain conditions. A decision tree is generated as follows: starting with the most important attribute – propose it is el_1, all cases that satisfy the following condition are added to the decision tree.

$$re(el_1|_{(si)}, el_1|_{(si_{current})}) \geq \alpha \qquad (8)$$

where coefficient α is a low bound that is used to restrict a set of candidate situations. The classification process continues with less important attributes, until only one candidate solution remains.

After finding a similar situation to the current one, the agents extract the solution that was used in this case (a method for modeling, designing the control system and simulation) and retrieve it appropriately to a solution for the current situation. The user accepts the recommendation and applies it into a practice. The last step that the CAS-Decision System has to do is to observe the real behavior of the system after the user has applied all the methods proposed by the agents. The results achieved by an observation are intended to evaluate the selected methods and for later use.

VII. CONCLUSION

The paper describes a multi-agent support system MARABU, which is dedicated to serve in modeling, control system design and simulation of manufacturing system, either continuous or discrete. That is way, the system handle with a lot of knowledge about methods, tools, and algorithms for modeling, control and simulation, and about continuous and discrete systems as well. The system also integrates work from many areas of the control theory and artificial intelligence. Such a way the system is large and also distributed. To handle all relevant knowledge, ontology approach was chosen. Ontology for that system is built using OWL, which enables utilization of third party tools for reasoning, and development of new reasoning methods as well. Protégé as an ontology editor was used to built ontology for the system, because there is a good cooperation between Protégé, JADE and JESS.

VIII. ACKNOWLEDGMENT

This work was supported by a Science and Technology Assistance Agency under the contract No. APVT-51-011602.

IX. REFERENCES

For a paper citation:

[1] Bergmann, R. 1999 Special issue on case-based reasoning. *Engineering Application of AI*, No. 12, 661-759.

[2] Fogel, J. 2001 „Control Synthesis of Discrete Manufacturing Systems using Timed Finite Automata", *Electrical and Computer Engineering Series, Advances in Signal Processing Robotics and Communications Edited by Kluev, Mastorakis, WSEAS Press*, ISBN 960-8052-42-4 pp. 276-280.

For a book citation:

[3] Hrúz, B.; Mrafko, L. 2003 *Modelovanie a riadenie diskrétnych udalostných systémov s využitím Petriho sietí a iných nástrojov.* Bratislava 1. vyd., Vydavateľstvo STU, 297 p., ISBN 80-227-1883-1

For a conference citation:

[4] Dang, T.T.; Frankovič, B.; Budinská, I. 2003 Case-based reasoning applied for CAS-decision system, In *Proc. of 2nd IFAC Conference: Control Systems Design, Bratislava, September 7-10, Š. Kozák, M. Huba (Eds.)*, 6 pages, on CD

[5] Frankovič, B. 2003 New Approach in Computational Cybernetics for Intelligent Adaptive Control of Non-linear Systems, In *Proc. of 1st Slovak Hungarian Joint Symposium on Applied Machine Intelligence SAMI 2003, Herlany, Slovakia, February 12-14, 2003*,ISBN 963 7154 140

[6] Liguš, J.; Laciňák, S.; Gábor, D. 2003 Analyze of adaptive control and design of reference model. In *Proc. of the 1st Slovakian – Hungarian Joint Symposium on Applied Machine Intelligence SAMI 2003. Herľany, Slovakia, February 12-14, 2003*, ISBN 963 7154 140

[7] Végh, P.; Hulkó, G.; Belavý, C. 2003 Robust control of distributed parameter systems. In Proc. of 14th International Conference on Process Control 2003. Štrbské Pleso,June 8-11, 2003, High Tatras, Slovakia

[8] Zeng D., Sycara K.: Using Case Based Reasoning as a Reinforcement Learning Framework for Optimization with Changing Criteria; Proc. of. Seventh International Conference on Tools with Artificial Intelligence; Virginia USA, 1995; http://csdl.computer.org/comp/proceedings/tai/1995/7312/00/73120056.pdf

For other citation:

[9] http://www.w3.org/TR/2002/WD-owl-absyn-20020729/#3

[10] Fridman Noy N., Hafner C.D. The state of the art in ontology design, http://www.aaai.org/Library/Magazine/Vol18/18-03/Papers/AIMag18-03-006.pdf