17th European Symposium on Computer Aided Process Engineering – ESCAPE17
V. Plesu and P.S. Agachi (Editors)
© 2007 Elsevier B.V. All rights reserved.

Merging Functional and Conceptual Ontologies

Manuel Rodríguez[a]

[a]*Autonomous System Laboratory -UPM,C/Jose Gutierrez Abascal, Madrid 28043, Spain; manuel.rodriguezh@upm.es*

Abstract

The purpose of this paper is to introduce a new architecture to help in the aim of developing autonomous systems and (semi)automatic modelling. Modelling is a task that must be goal oriented in order to derive a model that answers the questions the modeler wants with the minimum effort (in its development). This implies that an ontology based on goals or functions is important and thus, must be considered when developing the new architecture. In the other hand, it is important to know what we have modeled, in terms of the components (structure) of the model (the model should be 'aware' of what it can do). So, an ontology based on concepts (structure) is important and must also be considered in the proposed architecture. With a model based on this architecture (both ontologies) questions related to what the system can achieve or what the system is composed of can be formulated.

Keywords: ontology, functional modeling, autonomous systems

1. Introduction. Towards Autonomous Systems.

Autonomous Systems (AS) are systems that operate by themselves without the need of external intervention. We want to develop technology for autonomous systems for the real world, so they will free humans from supervising them once they're up and running.
Action generation in an intelligent system seems to be produced by means of knowledge exploitation. However, some questions appear: What does an agent need to know to achieve its objectives?, How is this knowledge acquired?, How

is it stored?, How is it used?, etc. For sure there are many types of knowledge and even there is a lot of confusion in the use of some words: information, data, knowledge, etc.
To address these issues the architecture presented in Fig.1 has been developed. This illustrates the main parts towards an AS, and how these parts are related and communicate with each other. The three main elements are:

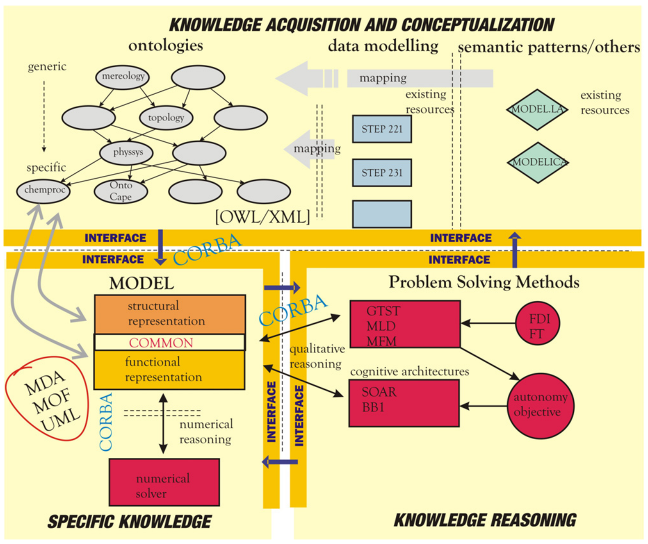

Fig 1. Architecture for the development of autonomous systems.

Knowledge acquisition (Information integration). This part establishes a methodology to integrate existing information resources of process systems. This integration will be based mainly on ISO data modelling standards (as 10303 [1] and 15926 [2]), although some existing ontologies are also considered, as OntoCAPE [3].

Independent model development (model-ontology-driven engineering). Using the information gathered in the process ontology an independent and reusable model has to be developed. This model will be expressed in sysML [4] (a UML extension). The transitions from data modeling/ontology to sysML and from sysML to the ontology are implemented. Transformation from an independent model to a specific model through existing tools is also addressed under this topic. Approaches similar to the Model Driven Architecture (MDA[5]) are used. The models developed have to be heterogeneous, i.e., have to be able to integrate different views of the system in the same model (qualitative view -functional model- and quantitative view -physical model).

Knowledge reasoning. The models developed have to be able to be used by heterogeneous applications such as simulation, hazard analysis, fault diagnosis, control reconfiguration, etc. Different techniques (besides numerical simulation) are applied to the model, to “reason” about the model, techniques such as: problem solving methods, cognitive architectures, …

Communication between the different parts and different (federated or agent) models is achieved by the use of middleware like CORBA [6].

Section 2 introduces the approach taken to make use of the functional and conceptual ontologies in order to exploit the advantages of both. Section 3 illustrates the proposed architecture with several examples and, finally, section 4 draws some conclusions of the presented work.

2. Merging functional and conceptual models

The idea behind the architecture is to look for a common nexus between both ontologies so a transition can be made from one to the other in a continuous way. This common space is composed of very primitive (basic) elements that can be expressed by both ontologies.

2.1. Functional models

Functional modeling is a technique that explains a system decomposing it through its functions. These are hierarchical methods to represent the system knowledge. The functional ontology used is generated based on existing functional modeling methodologies like the Goal Tree Success Tree described by Modarres in [7] or the Multilevel Flow Modeling described by Lind in [8] .

2.2. Conceptual models

Conceptual models are based on the structure of the model. These are the classic object oriented models to describe the behavior of a system. The conceptual ontology used is based on the mentioned COGENTS IST european project OntoCAPE, and on the data models developed under the ISO standards 10303/15926 (the use of standard based ontologies is the best way to ensure future reusability and compatibility between the developed models).

2.3. Architecture

Fig.2 shows the architecture to merge both mentioned ontologies. With a model based on this architecture (both ontologies) questions as: What do you have? (concept ontology) What do you know to do? Can you do this? (functional ontology) and even more important what happens if...? can be answered by the model.

The topology of the system is included in the boxes representing the structural part, each box has the name of the element that is connected to.

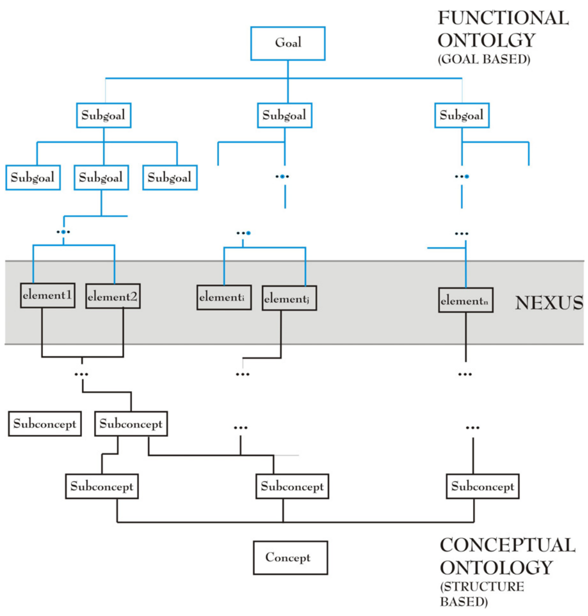

Fig 2. architecture for functional and conceptual models.

3. Examples

3.1. *Tank heated by a coil*

In this example a tank with two feeds is heated by steam through a coil. The functional ontology (Fig.3) shows some tasks to be performed on this system. For each task to be done all the paths to it must exist. For example, the tank can be filled as *change mass->mass balance ->material in/out and holdup->volume* are available for the tank. But the coil cannot be filled or emptied as the *holdup ->volume* relation does not exist for the coil. The architecture can be used the other way around, What happens if the input to the coil fails?, then the material and energy balances fail, so does the change energy function and finally the heat function (subgoal) cannot be achieved.

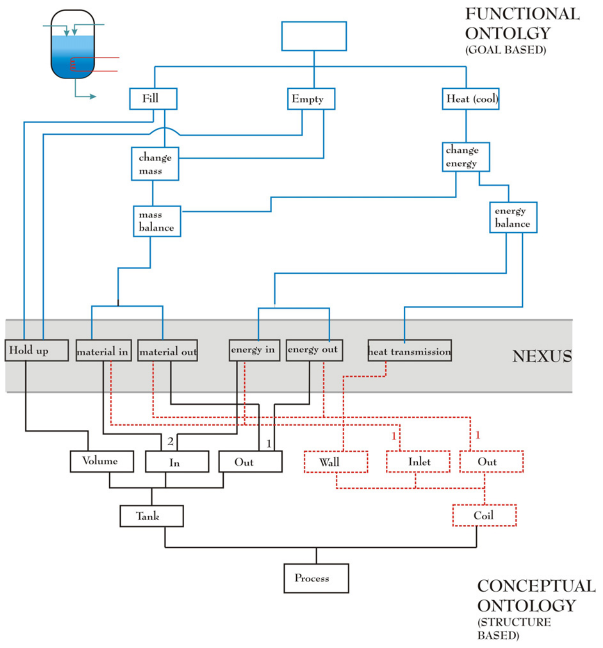

Fig 3. Model of a tank heated by a coil.

3.2. Mixing Process with level control

In this example a tank is fed with two streams, one of them is used to control the temperature. The tank has an output that is used to control the level (see Fig. 4). The same questions can be asked. What happens if the level starts to increase? Following the path *Volume-> Holdup* the control goal is reached, so this goal is affected and the possible causes are (going down from this goal) material in/out failure (which means inlet and outlet of the tank, and the valve) or signal transmission failure which means actuator/controller/sensor failure.

4. Conclusions and future work

This paper has presented an architecture to help in the development of autonomous systems. This architecture has three main parts: knowledge acquisition (using existing resources), model development (a whole integrating methodology that comprises the model of the system and the model of the software that implements it) and knowledge reasoning. Then, a new architecture to develop functional/structural models that can be used to answer questions from a conceptual as well as from a structural point of view is presented. This new architecture is connected with two different reasoning mechanisms, qualitative methods and numerical methods.

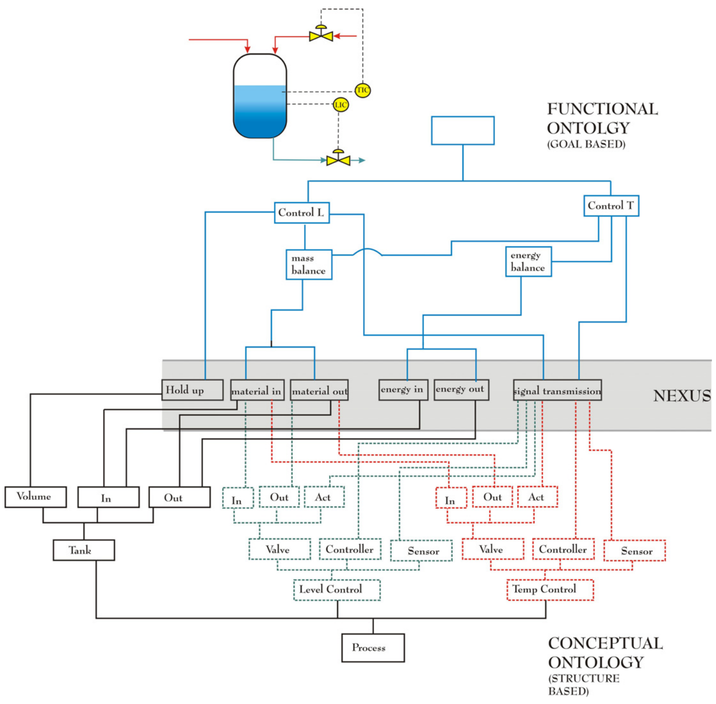

Fig. 4. Model of a tank with temperature and level control.

This allows to work in different abstraction levels depending on the application that is using the model. The proposed architecture is currently being implemented and the modeling methodology is being developed. Future steps will be to implement the whole process (from knowledge acquisition to the final application) in an agent based program.

References

1. M. Palmer (ed), ISO TC184/SC4/WG3 N745, Gaithersburg, USA, 1998
2. M.West, J. Sullivan and H. Teijgeler, ISO TC184/SC4/WG3 N1328, 2003
3. Braunschweig et al., COGENTS: Agent-Based Architecture For Numerical Simulation, Final Report, 2001
4. OMG, Systems Modeling Language Specification, 2006
5. OMG, MDA Guide Version v1.0.1, 2003
6. OMG, Common Object Request Broker Architecture (CORBA/IIOP) Specification version 3.0.3, 2006.
7. M. Modarres and S.W. Cheon, Function-centered modeling of engineering systems using the goal tree–success tree technique and functional primitives, Reliability Engineering and System Safety 64 (1999) 181–200
8. M.Lind, Modeling Goals and Functions of Complex Industrial Plant. Applied Artificial
9. Intelligence, Vol 8 No. 2 , April-June 1994.