An Ontology for Control Engineering

Francisco Rodríguez, Isaías García, Carmen Benavides, Héctor Aláiz, Javier Alfonso, Ángel Alonso

Dept. of Electrical and Systems Engineering, University of León, León, Spain, {diefrs@unileon.es, isaias.garcia@unileon.es, carmen.benavides@unileon.es, hmoreton@unileon.es, javi@unileon.es, dieaaa@unileon.es}

Abstract

Computers have always been a valuable tool in engineering domains. Within Control Engineering, it has been used as a tool for the modelling, simulation, analysis and design phases of a system's life cycle using its ability to perform fast numerical calculations. But today Control Engineering needs are demanding a shift in software applications toward highest levels of abstraction in the representation of the information. Numerical calculus is no longer enough but better human-to-machine and machine-to-machine interfaces are demanded and also interoperation among different applications is a must. Knowledge Engineering techniques and tools can be used to build such demanded software. This paper deals with the construction of an ontology for the Control Engineering field, in particular for the modeling of a design task. Based on this ontology an application is built that is able to give explanations to the user about the design rationale of the process. The particularities of this engineering field from the stand point of creating a conceptual model describing it will be presented as well as some examples of conceptualization issues found so far. A brief outline of the application's structure and functionality will be also presented. This application is intended to be used as an educational tool for Control Engineering students.

Keywords: Control education, Control engineering, Engineering ontologies, Knowledge engineering, Knowledge representation.

Resumen

Los ordenadores han sido una herramienta muy valiosa en los dominios relacionados con la ingeniería. Dentro de la Ingeniería de Control, siempre han sido usados como herramientas para las fases de modelado, simulación, análisis y diseño del ciclo de vida de un sistema haciendo uso de sus habilidades de realizar cálculos numéricos de forma muy rápida. Pero hoy las necesidades de la Ingeniería de Control están demandando un cambio en las

aplicaciones software hacia mayores niveles de abstracción en la representación de la información. El cálculo numérico ya no es suficiente sino que se necesitan mejores interfaces hombre-máquina y máquina-máquina y también es una obligación mejorar la interacción entre diferentes aplicaciones. Las técnicas y herramientas de la Ingeniería del Conocimiento pueden ser utilizadas para construir este tipo de software demandado. Este artículo trata de la construcción de una ontología para el campo de la Ingeniería de Control, en particular para el modelado de una tarea de diseño. En base a esta ontología se construye una aplicación capaz de ofrecer explicaciones al usuario acerca de los fundamentos de las decisiones de diseño tomadas durante ese proceso. Se presentarán las particularidades de este campo de la ingeniería desde el punto de vista de la creación de un modelo conceptual del mismo así como algunos ejemplos de aspectos relacionados con la conceptualización encontrados hasta el momento. También se presentará una breve descripción de la estructura de la aplicación y su funcionalidad. Se pretende que esta aplicación sea utilizada como una herramienta de apoyo a la educación para los estudiantes de Ingeniería de Control.

Palabras clave: Educación en Control, Ingeniería de Control, Ingeniería del Conocimiento, Ontologías para ingeniería, Representación del conocimiento.

1 Introduction

Computers have always been a valuable tool in engineering domains. Within Control Engineering[1], it has been used as a tool for the modelling, simulation, analysis and design phases of a system's life cycle into the so-called CACE software (Computer Aided Control Engineering) or also CACSD (Computer-Aided Control Systems Design). These kinds of applications exploited the computer's ability to perform fast numerical calculations. Simulation languages have led the evolution of CACSD software from the first applications, based on unstructured languages, to the modern object oriented ones. But today Control Engineering is demanding a shift in software applications toward highest levels of abstraction in the representation of the information. Numerical calculus is no longer enough but better human-to-machine and machine-to-machine interfaces are demanded and also interoperation among different applications is a must. Knowledge Engineering techniques and tools can be used to build such demanded software.

This paper deals with the construction of an ontology for the Control Engineering field, in particular for the modelling of a design task. Based on this ontology, an application is built that is able to give explanations to the user about the design rationale process. The particularities of this engineering field from the stand point of creating a conceptual model describing it will be presented as well as some examples of conceptualization issues found. A brief outline of the application's structure and functionality will be also presented. This application is intended to be used as an educational tool for Control Engineering students.

The paper is organized as follows: Section 2 presents Knowledge Engineering and ontologies as the structures where a conceptual model of the knowledge of a domain can be efficiently stored. This approach is compared to other modelling techniques coming from the Software Engineering fields. Section 3 presents the field of study where the conceptualization described

[1] Control Engineering deals with the creation of configuration and devices that attached to a system allows it keeping working properly despite the eventual external disturbances that may occur.

in this paper has been carried out: Control Engineering. In section 4 an outline of the ontology created is given and section 5 depicts the application built for using this ontology. Finally, section 6 presents some conclusions and future work.

2 Overview of ontologies as conceptual structures

An ontology is a structure where a given conceptual domain is stored. It can be defined as a specification of a knowledge model that can be reused in different applications where the described knowledge applies (Gruber, 1993). Some authors stress the need for the ontology of having a community that commits to the representation it stores because ontologies are built to share knowledge across machines and humans and so some degree of consensus about the conceptualization should exist but, purely speaking, one can build a knowledge model without consensus, being a different issue its later degree of usability and success.

Ontologies were born in the Artificial Intelligence and Knowledge Engineering fields and are being adopted instead of the software models coming from the more traditional Software Engineering area because it offers some advantages over the usual modelling paradigms found there like for example the Unified Modelling Language (UML) tools and techniques. In particular, the model found in an ontology differs from the knowledge found in an UML model in the fact that the ontology model is built to be used in the application as a crucial part of the software while the models built with UML are used in design time for representing the problem and the software project itself. Further, an ontology is able of representing much richer knowledge structures than UML tools are. It must be stressed though that today Software Engineering and Knowledge Engineering are influencing to each other, sharing and adopting the other's concepts and somehow even merging ((W3C, 2006) and (Baclawski et. al., 2002) are some examples of this).

Ontologies are being used in several fields of application (see (Lancieri et. al., 2005)) or (DARPA, 2006) for a detailed list) and are more and more seen as a key data structure for the knowledge management in the organizations, just like databases are and have been during many years.

Building an ontology is a modelling process, a knowledge modelling process. The field is not so mature as to have well defined and standardized methodologies, though in (Gómez-Pérez et al, 2003) some approached are presented.

From the point of view of the knowledge lying inside, an ontology is built upon two main structures:

- One reflecting the static knowledge, comprising the concepts found in the domain and the relations among these concepts.
- Another one storing the dynamic knowledge, comprising the conceptualization of the tasks used in the domain to do problem solving.

With the static structure one would have a conceptual model that could be used to label documents, as is the case in the Semantic Web service paradigm. Those documents could then be efficiently stored, disseminated, shared or retrieved in a semantically enriched environment. Ontologies in this sense could be seen as an extended and flexible thesaurus.

The dynamic structure is the base for building applications that are able of reproducing the problem solving processes found in the domain. Here an application should be able to "interpret" this dynamic structure implementing into code the algorithms or reasoning processes needed to make the software evolve.

3 Building a domain ontology for Control Engineering

3.1 Characteristics of Control Engineering knowledge

In the engineering fields in general and in the Control Engineering field in particular the domain model is to be quite complex whenever one wants to reflect some amount of the complexity of the domain. This complexity arises from the characteristics of this knowledge area:

- Engineering is a human activity and so the problem solving knowledge involve human, many times heuristic, knowledge.
- Engineering deals with building physical systems, that is, the beginning and the result of an engineering process is always a real system. Thus, software in engineering will also deal with real artifacts in the end.
- The study of these physical systems is done by means of models –simplifications of the physical elements -, usually mathematical models that, in turn, use to be simplified generating different but related levels of representation and reasoning.
- The tools and techniques used for problem solving in the field are diverse: From mathematical reasoning and manipulations to graphical and spatial reasoning.
- The problem solving techniques involved in the analysis and design tasks in the field eventually result in an iterative process involving reengineering, that is, the repetition of the process in the light of the previous results.

The great majority of the engineering fields were in their beginnings merged and only the passing years allowed them to establish themselves as independent disciplines. In the case of Control Engineering, this maturity was produced in the post-World War Two years. The importance of language in Control Engineering may be noticed in the fact that this engineering branch is not based on the invention of new technological devices nor in significative changes in the physic theories about the world. It was instead a matter of giving new names and relating concepts that already existed, in such a way that it was able to explain and describe phenomena for which the traditional physical laws didn’t have enough expressive tools (Bisell, 1993). The process of Control Engineering to become independent involved the creation of new terms that denoted not only concepts but also properties and actions in such a way that not only names but also adjectives, verbs and adverbs are specific to this field. As in other engineering fields, it can be said that a "Control Engineering language" exists on its own.

Control Engineering is a horizontal discipline, that is, it appears applied to many different areas like mechanics, electrics, electronics, robotics, economy, social sciences, etc. The concepts lying in the core of the discipline are domain independent, that is, the control engineering models used to represent those domains allow talking about them without the need of knowing the kind of real system being treated. This fact shows how important and unique is the language of Control Engineering.

One of the main characteristics of the concepts found in Control Engineering is that they appear in many places at different levels of abstraction, related to many different concepts, playing different roles and being used for different types of reasoning processes. It is also very common in Control Engineering to simplify the way in which concepts are cited. For example it is common to talk about "the poles of a given system" when, properly speaking, it should be said "The roots of the denominator of the (maybe simplified) mathematical transfer function model of the (maybe linearized) set of differential equations obtained from the lumped parameter model of a given physical system". We can see how the different levels of representation (that correspond to different models and thus to different simplifications) are bypassed and also how new terms (pole) refer to existing mathematical terms (root) but related to a given scenario (when roots belong to the denominator of an algebraic expression of a mathematical model of a system). This is done so to save time and complexity in the expressions and experimented engineers usually are aware of what they are doing, but in the case of students or engineers from other fields, these kind of simplifications and shortcuts are a source of misunderstanding and may lead to a bad assimilation or interpretation of the concepts.

This is the most challenging issue when representing control engineering concepts: To make the user aware of what concepts she is using, what simplifications have been done to use these concepts and so in which scenarios these concepts are valid or not. And further, relate these concepts with others at different levels of representation or abstraction. It is very common in Control Engineering the fact that the user loses the path from the numbers and calculations she performs to the concepts and physical meanings underlying these numbers.

4 Description of the ontology

First, a subdomain within Control Engineering has been chosen to be modeled. This subdomain is the lead-lag compensator design with techniques from the classical control theory (mostly with the root locus techniques). There are several reasons to choose this domain: First, this domain has huge amounts of semantics in its core (much more than is found in the so-called modern control theory, for example); second, the methods used to get a design involve several and different reasoning types; and third, it is a well-known field with lots of bibliographical data describing the processes. See for example (Nise, 2003).

4.1 The static knowledge structure of the ontology

The static structure of the ontology is comprised of the concepts and relations of the domain. It has been built by using bibliographic data and interviews with experts in the field. As an example, concepts used in this field may be *signal flux graph, point, polynomial, root, pole, percent overshoot*, etc.

All these concepts have been arranged into different hierarchies in order to show the relations that hold among them. Relations may vary from the well-known "is-a" inheritance relation to more complex ones like parthood, composition, connection, causality, etc. This way, several hierarchies of concepts could be obtained by just changing the focus on the relationship to be inspected. The concepts along with the relations among them form a complex networked structure of knowledge that build up the static structure of the ontology.

As was previously mentioned, all in Control Engineering comes from the use of a mathematical model of the systems involved in the design problem. Any concept in the

ontology is, in the end, related to a given model. Another interesting issue is the fact that any characteristic or any other object can be obtained from the manipulation of this simple mathematical model.

In figure 1 the concept PolynomialRoot appears in two different networks where the links mean different relations. In figure 1 (left) the traditional is-a relation is shown with the meaning "PolynomialRoot is a MathematicalConcept that is a Concept that (eventually) is a Thing" (Thing represents here the topmost concept in the is-a hierarchy). Figure 1 (right) also has the same PolynomialRoot concept but appearing in a different relation network that could have a kind of "parthood", "has associated" or "is composed of" meaning; this way, the branch could read "A TransferFunctionSystemModel has an associated PolynomialQuotient that has associated a (denominator)Polynomial that has associated a RootsAndGainDescription description that has associated a collection of Root that has associated a PolynomialRoot".

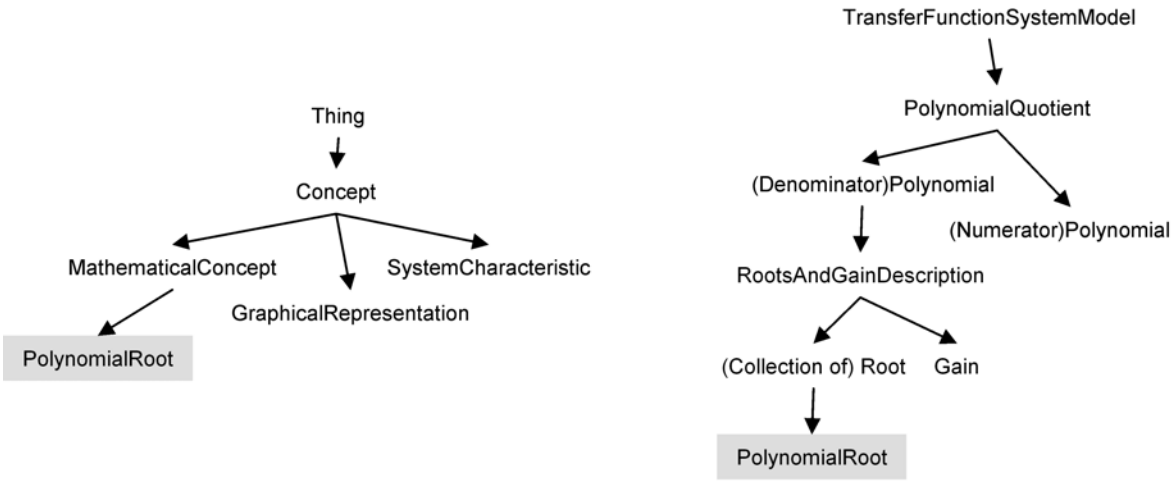

Figure 1. Two Networks of Relations

This static structure of the ontology is crucial to the system in order to being able of answering basic, definitional, questions of the type "what is this?"

As well as building this network of concepts and relations an ontology also contains the representation of rules by means of which new knowledge may be inferred from existing one. For example, the next rule states that when any pole of a system is located on the right half-plane the system is unstable:

```
If
A Pole isLocated on RighHalfPlane
Then
The System (representedBy the TransferFunctionSystemModel that has
associated the PolynomialQuotient that has associated the
(Denominator)Polynomial that has associated the
RootsAndGainDescription that has associated the collectionOfRoots
that has associated the PolynomialRoot that stands for the Pole) is
unstable
```

This rule has been written in almost natural language and in a simplified manner in order to show the basic structure and also how the network of concepts shown in figure 1(right) is used to relate the pole (the root of a polynomial) to the system that is represented with a denominator having this root.

This previous example also serves as a way of showing how some kind of simple explanation may be obtained. Say that some design decision has lead to the whole system to present instability and this fact has been notified by putting some red flag somewhere in the user's graphical interface. The user will see something like "the system is unstable with this design". Then she could ask "why?" and an explanation would be generated based on the structure of the rule (or set of rules if the reasoning used is more complex). For example, the first answer to the mentioned question could be simply "because there are poles of the system in the right-half plane", then the user could ask for further information, for example, the system could show which pole/s is/are in the right-half plane, but could even tell the user what a pole is or how the roots of a polynomial are calculated.

As well as the inheritance and parthood hierarchies, other relations have been built devoted to describe topology (for graphical reasoning purposes), mappings, causality, connection, etc. Also, as well as concepts, relations and rules, the static structure of the ontology also comprises constraints, expressed in a subset of first order predicate logic.

4.2 The dynamic knowledge structure of the ontology: Modeling the Design Task

The dynamic structure of the knowledge in the ontology represents the tasks used to solve the problems pertaining to that domain. In this case it is clear that the main task could be named as "design". So making the dynamic conceptual structure of the ontology is making the modeling of the design task in Control Engineering (actually a given type of design as was mentioned in section 3.3). In order to being able to give explanations the system must have a complete and explicit representation of this task and all the structure of subtasks that are involved on it.

To build the dynamic knowledge structure, the design task is divided into a set of subtasks. Those subtasks are arranged into a general workflow, allowing the specification of a control mechanism on their execution. A subtask may comprise different processes and comparison operations which, in turn, may be of procedural or heuristic nature. So, these processes may be carried out invoking a different reasoning facility that implements the kind of process involved.

The basic structure for controlling the execution of the dynamic structure is the rule. Rules have been proven as the most suitable way of representing expert knowledge and also allow building explanations from their structure.

The structure of the rule has been formalized in the ontology and thus it is easy to introduce new rules or to maintain the rule base because they are built by using near natural language constructs coming from the ontology's static knowledge.

A Control Engineering design task is almost always an iterative process that involves re-engineering. So, a different treatment has been carried out with this issue. Information about the result of any of the subtasks or processes earlier mentioned is incorporated to the system in such a way that this information guides the execution of the following ones.

5 Description of the application

A prototype has been built in order to test the ontology structure of concepts, relations and dynamical knowledge. The application may be thought of as a kind of simple Intelligent Tutoring System (because no model of the student is used nor assessment mechanism is performed). As the first aim is to know if the system has a correct knowledge representation, it was decided to build an application that automatically performed control design problems on its own, using the knowledge inside. To test the validity of the represented knowledge, explanation capabilities has been built that allows a user, once the design process ends, to ask about the reasons of any step of the design or about any concept that may have appeared. For this purpose, all the intermediate design steps are presented to the user through a graphical interface. Any concept or graphical element in the interface coming from the ontology may be clicked and a contextual menu appears where the user can choose the kind of questions she may have.

The application is written in Java and the ontology has been built with the tool Protégé, using a "traditional" frame based representation (not OWL). The application has been built in Java, using the Protégé Java API (Application Program Interface) for accessing the ontology structure. The dynamic knowledge structure of the ontology is putted into work also by means of an ad-hoc Java parser, while a translation of these structures into jess rules is also being studied in order to use the built-in jess rule engine.

The application uses numerical calculus software to perform function calls for evaluating some numerical characteristics or obtaining the value of some parameters by simulation. The interface between the main application and the numerical calculus ones is also conceptualized in the ontology. In this sense, this project is also an experience of the integration of engineering software applications using the concepts in an ontology as the top level communication facility.

6 Conclusions and future work

This work has presented the use of an ontology in the domain of Control Engineering for building an application to help the user in a given design process carried out in the discipline. This project is one of the first attempts to apply Knowledge Engineering techniques in the field of Control software development. Based on the experience in the development of this application some general conclusions can be presented.

Today research efforts on ontologies and Knowledge Representation have mostly shifted to the Semantic Web application field. This has led the research towards the static knowledge structure of ontologies (though research on a Semantic Web Rule Language has recently appeared), assuming an “open world” character for the knowledge and stressing highly formal knowledge representation schemes and languages. Research in the dynamic knowledge structure of ontologies has decreased if compared with the late 1990’s where research on generic tasks and Problem Solving Method was an important issue.

Though reusing is one of the biggest declared benefits of ontologies, there are few engineering-related ontologies so far and much work is still to be done for studying and conceptualizing the kind of structures, tasks and reasoning processes found in these fields.

The interaction of Software Engineering and Knowledge Engineering today is very interesting and worth studying. The advent of the Internet and the information burden produced in the Web service gave Knowledge Engineering the opportunity of applying its mostly theoretical studies carried out since the late 1980's crisis of the Expert Systems. Domain models (represented in ontologies) then have showed to be a very convenient way of building software applications in complex domains and to improve the machine-to-human and machine-to-machine interactions, and not only for web environments. On the other hand, Software Engineering techniques used modelling so far for generating a description of a software project but not to use those models at run time. Today, this discipline is going beyond its traditional object orientation approach and building and developing the so called Model Driven Architectures (MDA), partly inspired on ontologies. It seems that both branches are trying to reach the same "software based on models" objective, driven by their most representing organizations, the OMG (Object Management Group) and the W3C (World Wide Web Consortium).

While the interaction mentioned in the previous paragraph is serving to speed up the process of maturing in software based on domain models, it is important in the application fields to find the modelling strategies and structures that best fit the knowledge in this field's domain. This is what has been done in the present paper, within the field of Control Engineering. Being able to adapt the last achievements in software development will help to cope with the challenges that Control software is facing today.

References

BACLAWSKI, K. [et al.]. *Extending* the unified modeling language for ontology development. *International Journal Software and Systems Modeling (SoSyM)*, 2002, vol. 2, n. 1, p. 142-156.

BISELL, C. C. A *New way of talking: aspects of the creation of the language of control engineering*. Technical report SAG/1993/RR31/CCB, Department of Telematics, Faculty of Technology , Open University, UK, 1993.

DARPA (Defense Advanced Research Projects Agency). *DAML ontology repository*, 2006 [electronic resource] http://www.daml.org/ontologies/. [Consulted: 30 oct. 2006]

GÓMEZ-PÉREZ, A. [et al.]. *Ontological Engineering*. Springer Verlag, 2003.

GRUBER, T. A translation approach to portable ontology specifications. *Knowledge Adquisition*, 1993, n. 5, p. 199-220.

LANCIERI, L. [et al.]. *Success Stories and Best Practices, Knowledgeweb* deliverable 1.4.2, 2006 [electronic resource] <http://www.knowledgeweb.semanticweb.org/semanticportal/home.jsp?_origin=%2Fhome.jsp&instance=D1.4.2%3A+Success+Stories+and+Best+Practices&ontology=Documentation+Ontology&content=instance.jsp&instance_set=kweb&var_name=instance.> [Consulted: 30 oct. 2006]

NISE, N. *Control Systems Engineering*. 4th ed. Willey, 2003.

TETLOW, *P.* [et al.] (eds.). *Ontology driven architectures and potential uses of the Semantic Web in systems and software engineering*, 2006 [electronic resource]. http://www.w3.org/2001/sw/BestPractices/SE/ODA/. [Consulted: 10 dec. 2006]