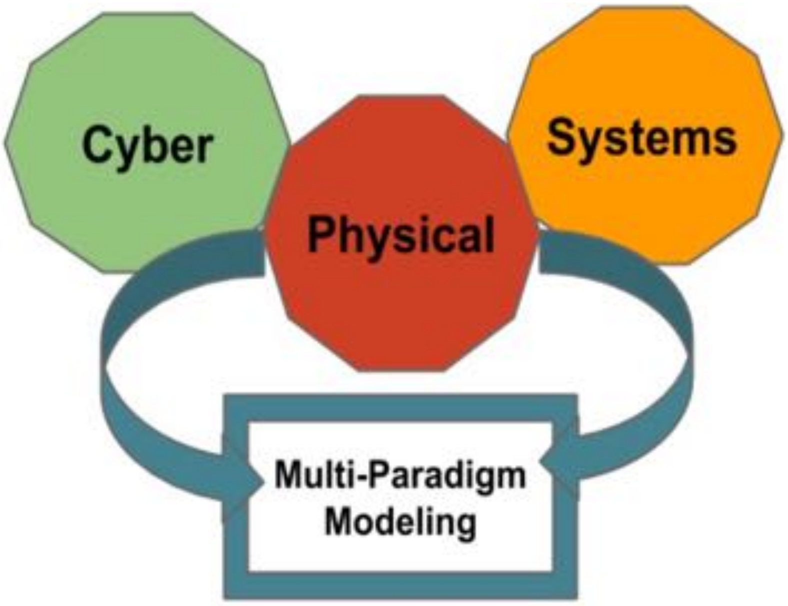
S



ICT COST Action IC1404

Struture and Approach for Developing the MPM4CPS Ontology (Draft)

Dominique Blouin, TODO.

Deliverable: D1.0

Core Team
University of Antwerp
New University of Lisbon
University of Malaga
Hasso Plattner Inst., Potsdam
University of Twente
fortiss GmbH, München
University of Lisbon

Document Info

Deliverable	D1.0
Dissemination	Restricted
Status	Draft
Doc's Lead Partner	HPI
Date	September 22, 2016
Version	0.1
Pages	8

Ver	Date	Author	Description
0.1	15-8-2016	Dominique Blouin	Initial version

Contents

1 Executive Summary

This document defines the approach for producing the deliverables of WG 1 and WG 2 of the MPM4CPS EUropean COST action. Such approach is centered on an ontology for Multi-Paradigm Modeling for Cyber-Physical Systems supporting the development of a catalogue of languages and tools for CPS (deliverable D1.1 of WG 1), a framework to relate /combine languages and tools (deliverabe D1.2 of WG 1), as well as a state-of-the art on MPM tools and techniques used in different disciplines for CPS development (deliverable TODO of WG 2). In addition, the choices made for the overall structure of this ontology are also presented and motivated.

2 Objectives and Deliverables

2.1 Working Group 1

As stated in the memorandum of the MPM4CPS COST action [MPM4CPS,], the objective of WG 1 consists of developing MPM foundations for CPS. In order to achieve this objective, WG 1 shall first characterize / categorize existing modelling languages used in the different disciplines of CPS development starting from typical industrial CPS development scenarios. From there, common formalisms and ontologies used in CPS shall be identified and a state-of-the-art report on the current formalisms used on CPS development shall be produced. Named D1.1, this deliverable D1.1 will consists of a structured catalogue of tools and modelling languages and a glossary of terms to be used throughout CPS development.

From there, WG 1 shall develop a framework to relate / combine (unify) modeling languages and techniques. The developed framework will be applied to combine MPM control, hybrid systems while dealing with the heterogeneity of CPS. This framework delivered as a report shall constitute deliverable D1.2.

2.2 Working Group 2

One of the deliverables of WG 2 has some overlap with D1.1 of WG 1 as it consists of providing a state-of-the art on MPM tools and techniques used in different disciplines for CPS development. Due to this similarity, it has been decided that WG 1 and WG 2 will work together to produce their deliverables from a common ontology.

3 Approach for Producing the Deliverables

The production of the deliverables for WG 1 and WG 2 involve activities of two different natures. Establishing a catalogue of languages and tools for CPS by looking at the various artefacts used during CPS development (languages, formalisms, tools, processes, etc.) involves describing and *explaining* the existing reality of current state-of-the-art CPS development. On the contrary, developing a framework to relate languages such as required by D1.2 aims at *constructing* a solution to the problem of coordinating the numerous models and tools used during CPS development.

Both of these activities will be performed with the help of models, but making use of two different modeling approaches or modes. This leads to the introduction of the explanatory and constructive modeling modes in the next section.

3.1 Explanatory and Constructive Modeling

Models used for CPS developmen and for systems engineering in general are of very diverse nature. It is common practice to distinguish between descriptive and prescriptive kinds of models [Rothenberg, 1990] but the same author also showed that such classification is not suficient and models are often further categorized in many different ways. More recently, explanatory and constructuve modeling were introduced [Atkinson et al., 2011] giving these terms deeper implications compared to the descriptive and prescriptive categorization as they refer to the supporting technology. Accordingly, explanatory modeling is better supported by *ontologies*, which typically assume *structural typing*. In such approach, the membership of an instance may only depend on whether or not it has some required properties. Therefore, it is typical to *infer* a classification from the properties of instances or individuals in ontology development.

On the other hand, constructive modeling predominantly uses *nominal typing*, meaning explicit typing relationships between instances and their types are used. The typing relationship is therefore defined by construction during instantiation. This is the case for most models used for software engineering, which are predominantly used for planning as blueprints for solutions for building a product. However, an important exception to this is models used for requirements elicitation and system analysis. Such models do not target solutions but instead aim at describing and understanding a problem domain.

These two different types of modeling modes are typically performed with different modeling technologies. Explanatory modeling is typically performed with ontologies, which alow defining sets of concepts describing domain knowledge. Classes can be specified in different ways by rich, precise logical definitions allowing knowledge discovery through the use of reasoners. For example, based on a set of instances, reasoners can be invoked to help improve the classification by discovering new classes or reclassifying instances more appropriately.

Such features are not provided by metamodeling languages such as UML class diagrams and therefore, ontologies seem to be the best available approach for performing the explanatory tasks on MPM4CPS such as creating a catalogue of languages and tools for CPS from existing artefacts (individuals / instances).

On the contrary, a framework to relate languages such as that of D1.2 is of a constructive nature, since it aims at constructing a solution to the problem of coordinating the numerous involved models and therefore, standard metamodeling techinques will be more appropriate for this task.

Therefore, the approach we shall use for producing the deliverables of WG 1 and partially WG 2 shall make use of both explanatory and constructuve types of modeling modes. Such approach is presented in the next section.

3.2 Approaches per Deliverable

3.2.1 WG 1: Deliverable 1.1

In order to derive the catalogue of languages and tools for CPS (D1.1), we will first establish an ontology serving as base classification. This ontology will define a first set of named classes for artifacts used in CPS development with MPM such as languages, formalisms, tools, etc. Then, a set of existing MPM for CPS development environments and processes will be tentatively classified using this first draft ontology, which will be iteratively refined following the discoveries made from the classified case studies using reasoning capabilities provided by the ontology approach.

From this refined ontology, a model transformation will be developed to automatically produce a first draft version of deliverable D1.1, which consists of a structured catalogue of tools and modelling languages for CPS development including a glossary of terms. This transformation will take care of preserving any manual editions of the document.

3.2.2 WG 2: State-of-the-art Deliverable

The state-of-the art on MPM tools and techniques used in different disciplines for CPS development to be delivered by WG 2 will also be produced from the ontology similar to D1.2 of WG 1.

TODO: anything to add there?

3.2.3 WG 1: Deliverable 1.2

The developed ontology will also support the development of a comprehensive GMM infrastructure including a framework to combine and relate languages as required for D1.2. The cental artifact of such infrastructure will be a megamodeling language acting as a registry for MDE resources such as formalisms, modeling languages, model relations between these languages such as model operations including transformations / synchronization.

Such megamodeling language shall be automatically derived from the MPM4CPS ontology, following a process as presented in [Walter et al., 2014] (TODO maybe more refs there), where ontologies are used to reason during DSLs development. The formal semantics of ontologies together with reasoning services allow for addressing constraint definition, progressive evaluation of the DLSs, suggestions of suitable domain concepts to be used and debugging of the DSL. However, we will not use the proposed combined ontology and metamodeling technical space proposed by [Walter et al., 2014]. As explained in the next section which motivates the choice of our technical spaces, stable and well proven technologies are required, which is not the case of this new approach. However, it should be possible to benefit from its analysis capabilities while maintaining the two separate ontology and metamodeling models consistent.

The derived megamodeling language shall require additional efforts so that it can be properly operationalized to relate languages and tools. For instance, existing approaches for model transformation / synchronization to ensure consistency is preserved may be integrated. TODO what else could we mention?

4 MPM4CPS Ontology

4.1 Selected Language and Tool

The development of the proposed ontology for MPM4CPS requires a solid ontology language and tool to make sure the classification can be developed and maintained correctly, taking into account that it will be developed collaboratively. Furthermre there is a need for reusing existing ontologies due to the vast coverage of domains for CPSs. Indeed, it is desirable that existing well defined ontologies can be reused. Such is the case for instance for measurement units, which is a domain strongly needed for CPS due to their interaction with the physical world. Such ontology initialy developed for the SysML language as the QUDV annex actually already exists expressed in the OWL ontology language and is of great interest for our work.

The Ontology Web Language (OWL) [OWL,] is currently one of the most used ontlogy language as of today. It is implemented with the Protégé tool developed at Stanford University [Protege,]. It was originally developed to hepl biologists define classifications for thir domain. Therefore, it has been extensively tested for the complex and large ontologies of that domain and exhibits good scalability and usability and should be sufficient for our relatively smaller MPM4CPS ontology. A detailed tutorial is provided to ease learning the language an tool [Protege-Tutorial,].

Another interesting ontology language and tool is Fluent [Fluent,]. It is a comprehensive tool for editing and manipulating ontologies using *Controlled Natural Language*. Such language closer to natural language may be a more suitable interface for human users than OWL-based editors such as Protege, especially for users prefering textual to graphical notations. Furthermore, the tool supports users by prohibits them from entering any sentence that is grammatically or morphologically incorrect and provides code completion to ease constructing sentences. Fluent is free of use for academics. However, it seems to be only available for the Windows platforms. In any case interoperability with Protege is provided by an additional plugin and users preferring a textual notation may want to use this second tool instead of Protege.

4.2 Overal Structure

The MPM4CPS ontology will be divided into two main parts respectively for the MPM and CPS domains. The objective of such architecture is to make these two ontologies independent and easily reusable. These ontologies will be unified through a topmost MPM4CPS ontology that will be linking them.

Each ontology wil be further separated into three layers as illustrated in figure 4.1. The core framework layer will include core generic classes for the main concepts for CPS development. The second layer will include subclasses for specializations of the core generic classes. This layer shall be the source of the catalogues of the deliverables. The glossary of terms shall be mainly derived from these two layers. Finally, a third layer will include the specific individuals of the existing CPS development environments and processes serving to iteratively refine the two other layers.

As a first source of information for the main concepts used in MPM4CPS, we will use the work of Broman et al. [Broman et al., 2012]. Such paper provides an initial coarse grained characterization of viewpoints, formalisms, languages and tools for CPS, providing initial definitions for important concepts that however may need to be refined throughout this work.

TODO: add any other references that could also be relevant?

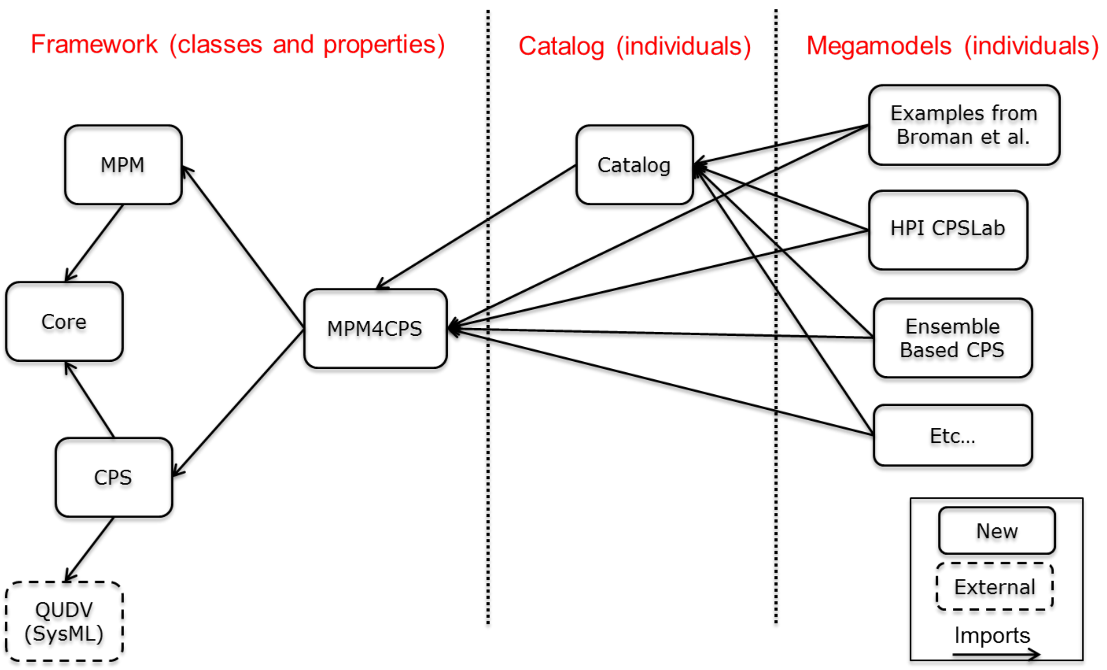


Figure 4.1: Overview of the structure of the MPM4CPS ontology

4.2.1 Core Ontlogy

4.2.2 MPM Ontlogy

The MPM ontology shall mainly focus on concepts for modeling in a multi paradigm approach; that is for every model related concept, forrmalism, languages, paradigms, and model combination / relation / operation etc.

4.2.3 CPS Ontology

The CPS ontology shall focus on concepts for describing CPSs such as systems, systems of systems, environment, networks, physics, mechanichs, electrics, etc TODO complete...

4.2.4 MPM4CPS Ontology

The MPM4CPS ontology is a unifying ontology including concepts of both previous ontologies and providing new concepts integrating them. Such concepts may typically include relationships between the actual system and its models, TODO complete...

4.3 Ontologies Development Protocol

The previously described ontologies will be developed collaboratively. In particular, the core and calalog ontologies will be used by many developers to capture existing CPS development environments and processes. This may lead to the identification of shortcomings in the core ontologies. We recommend that only a limited number of developers are granted the right to modify these core ontologies, since modifications may need to be propagated to existing other individuals and case studies. In such case, a web form (or Redmine issue tracking system?) with dedicated categories will be required for users demanding modifications to the core ontologies. Detailed justifcation for the need with a concrete example should be provided. From there, the core ontoloy management team will start discussing the required change, examine its potentian impacts and make a decision on its approval, upon which the changes will be made and notification send to update existing depenednt ontologies.

4.4 Multiple Files with Protege

Objects created under a class from an imported ontology are still stored under the actual ontology and not the imported ontology. One needs to select the imported ontology under the ontology combo to be abme to add to it.

Bibliography

[Atkinson et al., 2011] Atkinson, C., B. Kennel and B. Goß (2011), Supporting constructive and exploratory modes of modeling in multi-level ontologies, in *Procs. 7th Int. Workshop on Semantic Web Enabled Software Engineering, Bonn (October 24, 2011)*.

[Broman et al., 2012] Broman, D., E. A. Lee, S. Tripakis and M. Törngren (2012), Viewpoints, Formalisms, Languages, and Tools for Cyber-physical Systems, in *Proceedings of the 6th International Workshop on Multi-Paradigm Modeling*, ACM, MPM '12, pp. 49–54, ISBN 978-1-4503-1805-1, doi:10.1145/2508443.2508452. http://doi.acm.org/10.1145/2508443.2508452

[Fluent,] Fluent (2016), Fluent Homepage, http://www.cognitum.eu/semantics/FluentEditor/.

[MPM4CPS,] MPM4CPS (2016), MPM4CPS Homepage, http://mpm4cps.eu/.

[OWL,] OWL (2016), OWL Homepage, https://www.w3.org/OWL/.

[Protege,] Protege (2016), Protege Homepage, http://protege.stanford.edu/.

[Protege-Tutorial,] Protege-Tutorial (2016), Protege Tutorial, http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf.

[Rothenberg, 1990] Rothenberg, J. (1990), Prototyping as Modeling: What is Being Modeled?

[Walter et al., 2014] Walter, T., F. S. Parreiras and S. Staab (2014), An ontology-based framework for domain-specific modeling, *Software & Systems Modeling*, **13**, 1, pp. 83–108.