Modeling the Ensemble-Based CPS Development Environment with the MPM4CPS Ontologies

Short Term Scientific Missions of the MPM4CPS COST Action

Detailed Report

Introduction

This document is an extension to the main report related to the short term scientific mission accomplished at the Telecom ParisTech engineering school in Paris in response to the call from Working Group 1.

Work Plan

The task that has been addressed is the modeling of the existing CPS Development Environments. Our team is developing a runtime environment which is known as jDEECo. Not only does it include a number of integrations with tools and transformations, but also it introduces a set of new concepts that targets CPS and at the same time improves the traditional software development approaches. That makes this environment a good candidate for the previously mentioned task. The task requires proposing a use case and modeling it using the existing ontologies. The process works as a validator for the MPM4CPS and catalog ontologies and helps in finding conflicts or missing concepts there. In the following is a list that describes my vision of the working plan during the suggested short term scientific mission:

1- Define a use case for Ensemble-Based CPS.
2- Model the use case using the existing concepts in the ontologies and express the integrations/transformations in the use case.
    a. Add any missing individuals to the defined classes in the catalog ontology.
    b. Check the conflicts and the shortcomings in the recent proposed concepts if there is any.
    c. Propose a solution for the encountered issues such as proposing a different classification.
    d. Use / define object properties.
    e. Use / define model operations.
3- Check the coverage of these ontologies over CPS development processes (i.e. starting from requirements to the complete implementation), and mention the missing points discovered during the system developing steps from both ontologies.
4- Prepare a questionnaire that sum up the resulted ontologies and their issues to get a feedback from all WG1 participants during the next meeting in Malaga.

The target is to have a validated example that we can go through during the next meeting. Following this, the future work will be to define a template of the CPS development environment use cases taking into account the changes proposed during the STSM.

Results

Use Case for Ensemble-Based CPS

Emergent systems are systems in which components form together in groups to achieve common goals. The composition of components is dynamic and depends on the context. Therefore, multiple models of the system are required to capture different views of the system, and find a suitable process that consists of a sequence of model operations which ensure consistency between system views. We present here an example of environment for developing such systems with models that is called Ensemble-Based Cyber-Physical Systems (EBCPS). EBCPS targets distributed dynamic and self-adaptive systems. The system consists of fully autonomic components with implicit communication.

EBCPS consists of many parts that cover system development process from the requirements elicitation phase to the system production phases. The parts in EBCPS are: requirements, design, runtime, self-adaptation, and simulation. Furthermore, a number of operations between these parts are available, and they are divided to two sets: transformation or integration operations.

Approach

The use case is implemented using the Protégé tool with the OWL ontology language. We depended on the WG1 defined ontologies which are the Core, Catalog, MPM and MPM4CPS ontologies as depicted in the following figure. These existing ontologies provided us with multiple classes, properties and individuals to characterize and model EBCPS. These ontologies are divided into two basic categories: one that holds classes and the other that holds individuals of these classes. Our example falls in the Mega-model category which is used to capture the development environment.

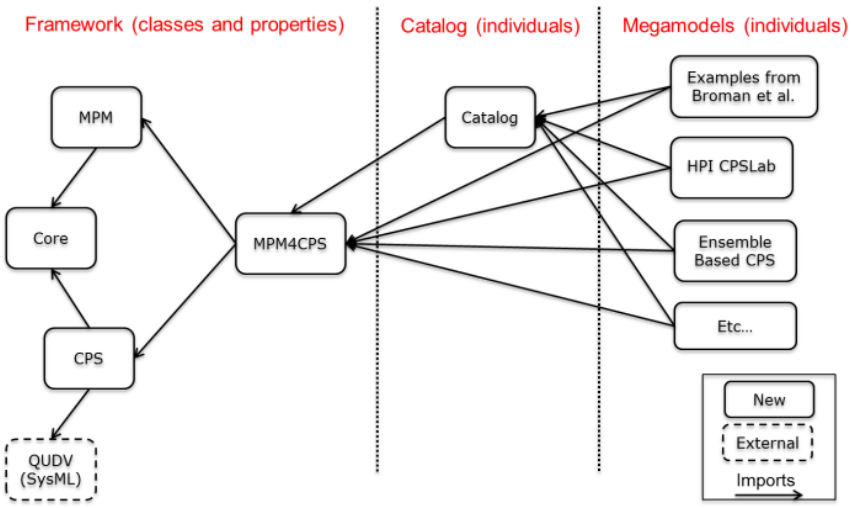

Figure 1 Overview of the structure of the MPM4CPS ontology

Modeling of the use case

To describe the structure of EBCPS, we are using the ontologies provided by mpm4cps. We use the Core, MPM, and Catalog ontologies to represent the models, model operations and tools used in EBCPS. Naturally, this requires starting by defining mega-model and mega-model fragments for EBCPS.

Protégé provides the users with a visual representation for the existing classes, individuals, and properties.

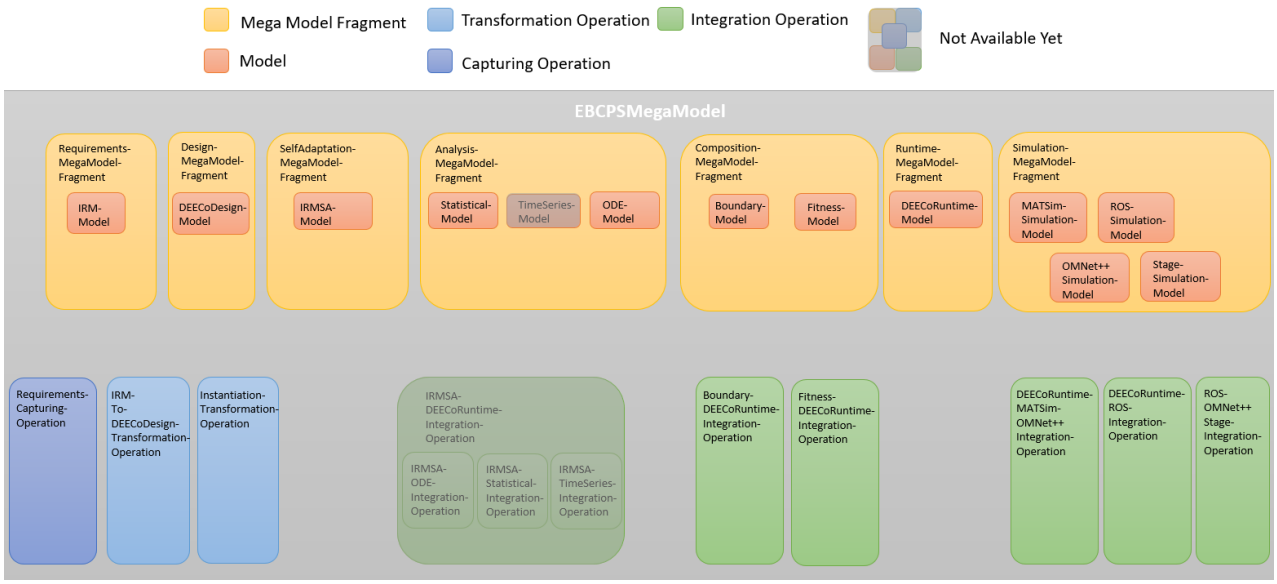

Figure 2 EBCPS Mega-Model, which includes mega-model fragments, models and model operations

For instance, in [Figure 3], we can see the previous diagram with all the included concepts and relations between them.

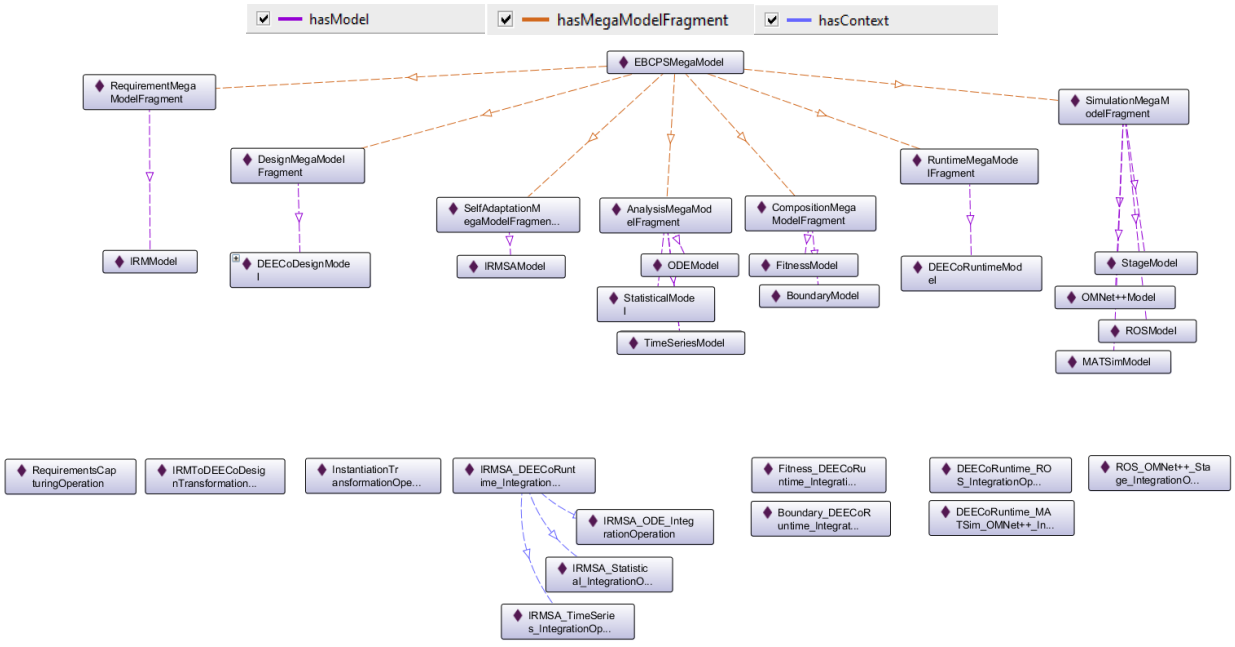

Figure 3 The OntoGraf view of the ontology

MegaModel

A MegaModel provides an overview of the many mega-model fragments and models for the system. For our use case EBCPS mega-model is instantiated to contain the EBCPS megamodel fragments, models and model operations.

MegaModelFragments

A MegaModelFragment is a part of mega-model that can be reused across different development organizations or projects, but that cannot exist by its own. For our use case the mega-model fragments are: RequirementsMegaModelFragment, DesignMegaModelFragment, SelfAdaptationMegaModel-Fragment, AnalysisMegaModelFragment, CompositionMegaModelFragment, RuntimeMegaModel-Fragment, and SimulationMegaModelFragment.

Models

A model contained in a megamodel is a representation of the actual real model of the system or a megamodel fragment. The purpose of such representation is to support model management. For our use case the models are: RequirementsModel, DEECoDesignModel, IRMSAModel, StatisticalModel, ODEModel, TimeSeriesModel, FitnessModel, BoundaryModel, DEECoRuntimeModel, MATSimModel, OMNet++Model, StageModel, and ROSModel.

Model Operations

A model operation represents all operations applied on models and orchestrated between them. The possible operations are: Integration, and transformation. A transformation operation can be also be a *capturing* operation, which is transforming information from engineer's mind or environment to a model.

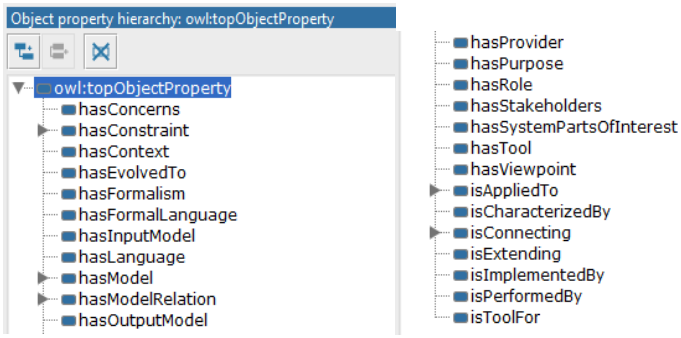

Figure 4 Properties defined in the MPM Ontology

Properties

Properties are used to relate an individual to another one. We will explain them and use them in the tutorial. The [Figure 5] illustrates most of the properties with their domain and ranges.

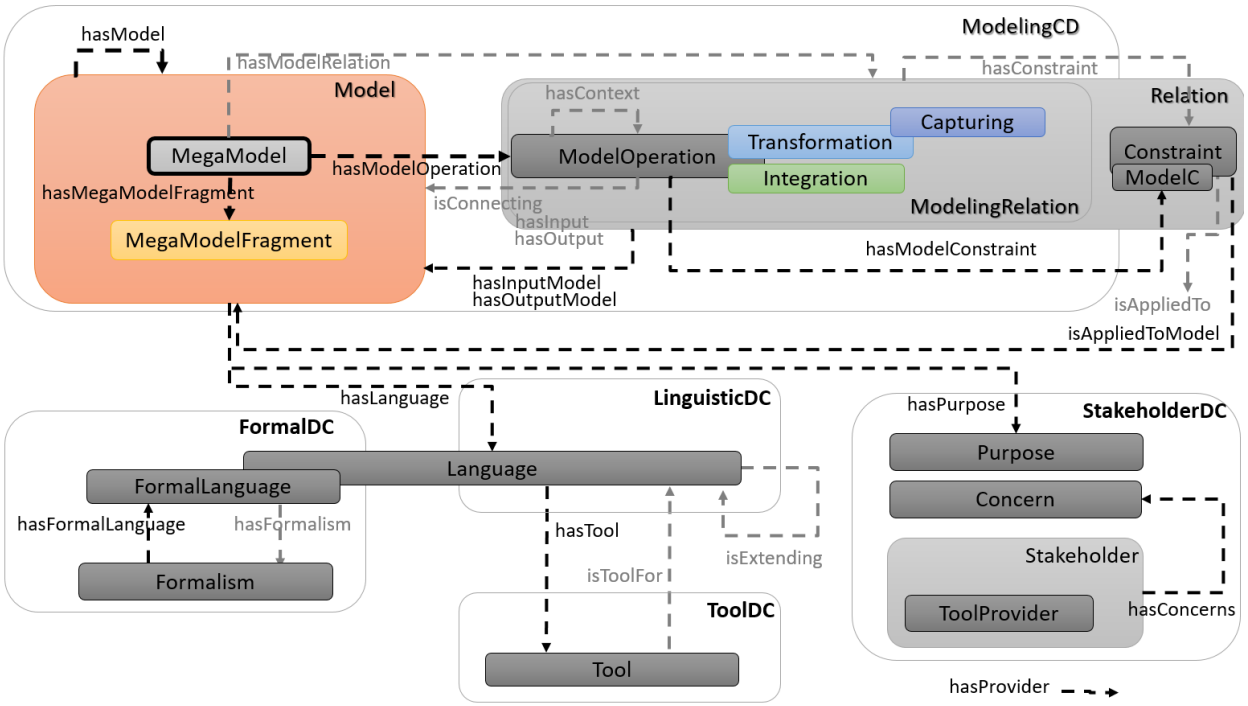

Figure 5 An overview of most of the properties which are defined in the MPM Ontology

hasMegaModelFragment

This property connects a mega-model with a mega-model fragment. In our use case the EBCPSMegaModel has all the previous named mega-model fragments (e.g. RequirementsMegaModelFragment).

hasModelOperation

This property connects a mega-model fragment to a contained model operation. In our use case the EBCPSMegaModel has many model operations (e.g. RequirementsCapturingOperation).

hasModel

Similarly, this property connects a megaModelFragment to a contained model.

Reasoner

The reasoner helps finding out all possible properties or classes for your individuals, which helps in finding some mistakes in the classification. After starting the reasoner, any highlighted information in yellow is inferred information. It needs to be checked if the inference is correct or not.

Previous Ontologies

During our modeling of EBCPS we encountered many conflicts and shortcomings from the previous ontologies. We have already made some modifications in the classifications and added more classes, individuals, and properties. The following sections will present the existing ontologies and their improvements. Worth mentioning that we used the unified definition of megamodel to capture the megamodel management in the ontologies. Found here: http://journal.ub.tu-berlin.de/eceasst/article/viewFile/704/713

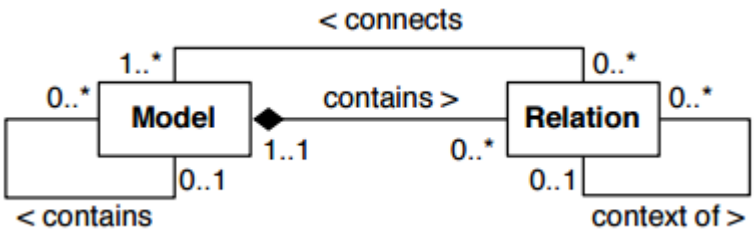

Figure 6 Unified metamodel for megamodels

Core Ontology

Classes

The class hierarchy of the core ontology was not changed much. The only change was the addition of the ToolProvider class.

Properties

We added there hasConstraint, hasEvolvedTo, hasProvider, and isAppliedTo.

Annotations

We added a set of annotations whose namespace starts with mpm4cps for storing additional information such as: author, reference, website, toDo, toBeMoved, toBeReviewed, toBeUpdated. This information will be used during the automated generation of the WG1 deliverables from the ontologies.

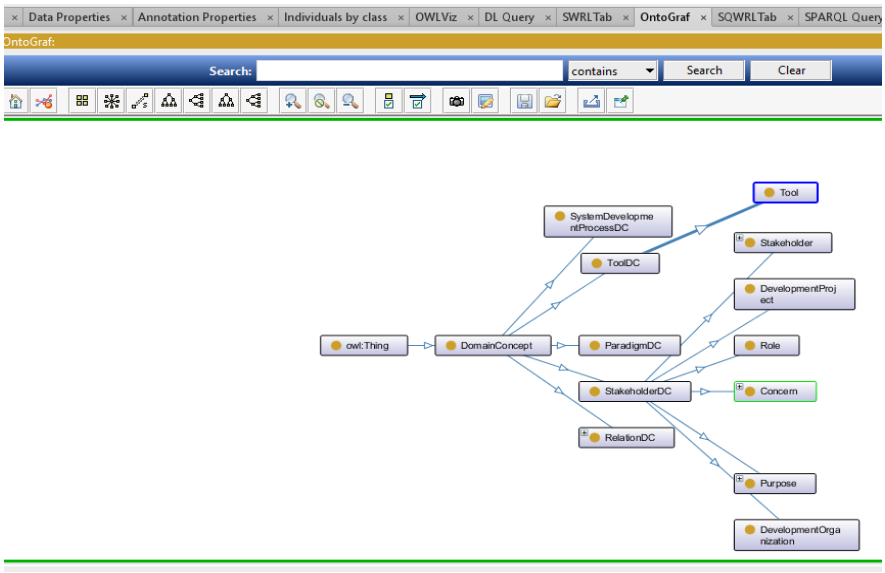

Figure 7 The Graph of Core Ontology

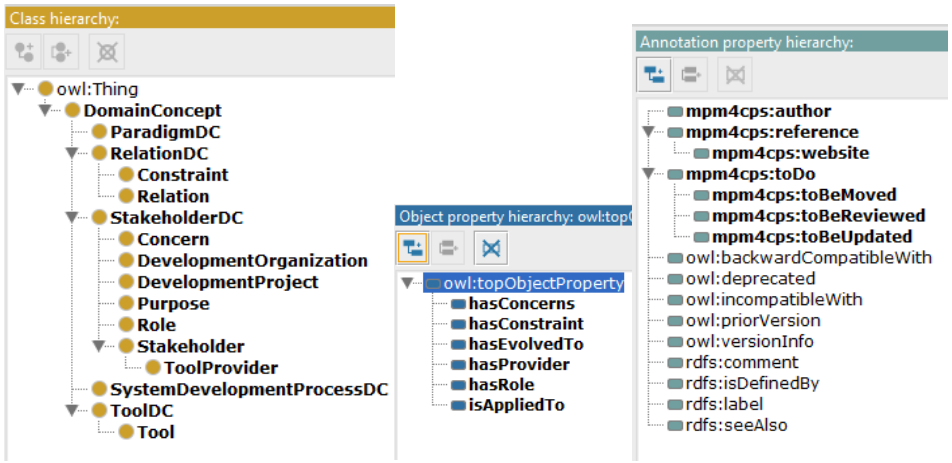

Figure 8 Core Ontology

MPM Ontology

The ontology is about multi-paradigm modeling and operations applied on models.

Classes

The added classes are ConstraintLanguage, all the subclasses of FormalSemantics, and all subclasses of Model, ModelRelation and Tool.

Properties

We described the properties of the MPM ontology [Figure 5] each property (relation) has a domain and a range. The defined properties in this ontology are in bold as it is shown in [Figure 9].

Catalog-tools-languages Ontology

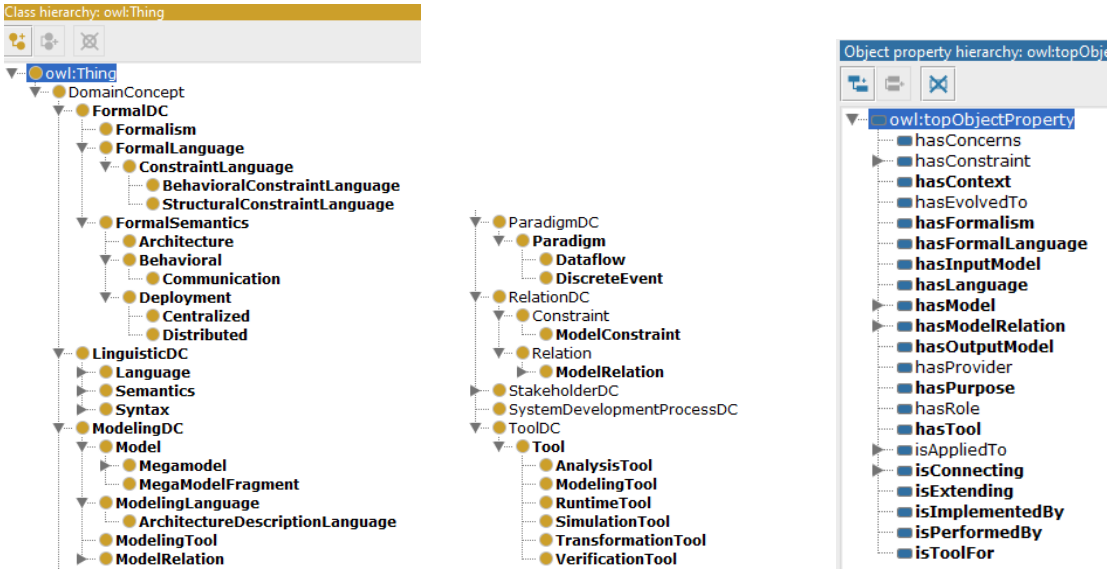

Figure 9 Classes and properties in MPM Ontology

I added some additional individuals in the catalog related to our EBCPS and other well-known languages and tools as well. The contribution in the catalog was in different areas such as tools. For instance, the tools that are related to our EBCPS were added which are Epsilon to model requirements, jDEECo as runtime tool, and simulation tools (i.e. MATSim, ROS, OMNet++, and Stage). Furthermore, more tools were added which were mentioned during the MPM4CPS meetings, schools and workshops (i.e. 20SIM, Fortiss, Motif ..), as well as a number of other tools (i.e. TTool, CHESS, ..).

Ontologies Coverage over CPS Levels

The ontology covered most of the parts related to system levels: Requirements, Design, analysis, and Runtime in the model. In the future work, we are planning to continue working on the ontology to include formalisms, languages, and tools.

Discussion

As part of the MPM4CPS COST activities, we provided a starting point for further collaborations within WG1. In order to achieve this, we prepared a tutorial to be presented to the group members and a list of issues to be discussed among members. The discussion aims for enriching the existing ontology and involving the group members further in the process of enhancing the ontologies.

Tutorial

We prepared a tutorial that tells the user step by step how to model a CPS Development Environment. The tutorial can be found in the Redmine ontology wiki.

Questionnaire

The performed work raised many questions. To resolve those questions, we requested feedback from WG1's group members during the meeting in Malaga and the rest of the issues will be documented in the WG1 ontology project server. The issues are the following:

no	Question	suggestion1	Answer
1	Definition of Paradigm and examples	model every part and aspect of a system, including development processes, explicitly, at the most appropriate level(s) of abstraction, using the most appropriate modelling formalism(s)	Formalisms + processes
2	Should we recognize between first and second class entities?	Define first class entities in CPS Ontology	
3	Where should we represent dynamic component composition representation?	Under Relation Class as a view to represent the relation between system components	
4	Where should we represent hierarchical component composition?	Under Relation Class as a view to represent the relation between system components	
5	Where should we represent UML/SysML?	Formal Language Class/ Modeling Language Class/ define semi-Formal Language Class	UML is a set of languages
6	Do we keep abstract and concrete syntax?	Yes/no	
7	MAPE-K loop, is it used in only at Software level?	Yes/no	
8	Do we distinguish between runtime and design time level and if yes under which class we would add it?	e.g. runtime verification / design time verification	
9	Do we add property class? Where?		
10	Do we add who performs the model operations?	we can consider it manual as a default (human)/automatic	
11	Do we add Discrete Events & Continues Time representation?	-Have discrete and continues components -have property for the tools as support for discrete event or/and continues time	
12	How to represent the human as input model to Model instance?	-add new property between stakeholder and model -add human as a model	
13	Do we put the property "isExtending" to be more general?	e.g. model extends model	hasModel property works like model is extending another model
14	Do we consider tractability as Modeling operation?	Yes/no It is not an operation. We can add it as a relation.	
15	Should we adding different relation for integration than input/output models… it should be symmetric relation	I am using isConnecting	

16	isReturingTo is not inverse of hasNext in general but it works like that in integration operation	Maybe we should define a property for integration only?	
17	Is it interesting to define topology as a class?		

Meeting results

The meeting included presenting the work done in the STSM, and we went through the tutorial and described the use case to the group members and the difficulties we faced during the modeling. We also mentioned the improvements of the existing ontologies.

Moreover, we discussed the open issues related to the mega-model fragments concept, a definition of the paradigm concept and the difference between formal, informal, and semi-formal languages.

Future work

In the light of the accomplished STSM work and the discussions at the meeting in Malaga, I highly recommend the following:

1- Encouraging members of WG1 to model their CPS development environments using the prepared tutorial and provide feedback about potential improvements of the modeling process and ontologies.
2- Continue developing the example ontology and add the used formalisms, languages, and tools.
3- Study what kind of information could be used from WG2 to enrich our ontologies.
4- Study what kind of information could be used from WG3 to enrich our ontologies.
5- Prepare a call for the modeling of other CPS development environment to continue the development of the MPM4CPS ontologies.

Following the results we accomplished during the STSM, WG1 is preparing for a journal paper for which I will be one of the main authors.