Grant Agreement: 644047

INtegrated TOol chain for model-based design of CPSs

Foundations Work Year 2 Overview

Deliverable Number: D2.2

Version: 0.3

Date: December 2016

Public Document

http://into-cps.au.dk

Contributors:

Ana Cavalcanti, UY
Simon Foster, UY

Editors:

Ana Cavalcanti, UY
Simon Foster, UY

Reviewers:

Consortium:

Aarhus University	AU	Newcastle University	UNEW
University of York	UY	Linköping University	LIU
Verified Systems International GmbH	VSI	Controllab Products	CLP
ClearSy	CLE	TWT GmbH	TWT
Agro Intelligence	AI	United Technologies	UTRC
Softeam	ST		

Document History

Ver	Date	Author	Description
0.1	09-12-2016	Simon Foster	Initial document version
0.2	11-12-2016	Ana Cavalcanti	Added content
0.3	12-12-2016	Simon Foster	Few small finalisations

Abstract

The work carried out by Work Package 2 in INTO-CPS underpins much of the technical work carried out in other packages, in particular, the integration of tools in Work Package 4. This document positions the work of the various tasks in the Work Package and briefly introduces the deliverables produced (D2.2a, D2.2b, D2.2c, and D2.2d).

The main goal of the research in Work Package 2 (WP2) is to provide theoretical foundations to justify the joint use of the notations and tools integrated in INTO-CPS with the technological support of the FMI standard. We face the challenge of unification of the various modelling paradigms involved, discrete, timed, continuous, and so on. We need to (1) understand the meaning of the composition of models written using disparate notations; (2) verify that the extra artefacts created for integration are sound; and (3) justify the analysis results that arise from the conjoined use of the composed models.

The basis of this work is Hoare and He's Unifying Theories of Programming (UTP), a relational semantic framework targetted especially at unification of diverse semantic models for a variety of programming paradigms. In the UTP, semantic models for each paradigm are developed independently and combined later via Galois connections. We use this approach in WP2 to define *CyPhyCircus*, a rich language characterised by a UTP model, including facilities to specify data models like in VDM, behavioural models like in CSP, time properties like in VDM-RT, and continuous models like in Modelica.

Model composition in INTO-CPS is formalised by translating all models and artefacts to *CyPhyCircus*. Figure 1 gives an overview of our approach. In INTO-CPS, a co-simulation is described at the architectural level using the INTO-SysML profile. For an INTO-SysML model, we define a CSP semantics; since CSP is a subset of *CyPhyCircus*, this is a semantics that captures abstract system-level properties of the co-simulation that can be verified using our framework. This CSP semantics is described in Deliverable D2.2a, where we also show that INTO-SysML can be used in a context wider than that originally predicted for INTO-CPS, involving other modelling languages.

Another formal characterisation of an INTO-SysML model, also presented in Deliverable D2.2a, captures the graph of dependencies between the ports of the models of a co-simulation. We use this information for two purposes. The first is to identify the possibility of an algebraic loop introduced by the model composition (rather than inside a particular model, which is problem already addressed by particular tools). This possibility is a warning of a possible problem to the designer of the co-simulation.

Second, the graph of dependencies is an important ingredient to support the automatic generation of a more concrete FMI-based model for the co-simulation. To define this concrete model, we need to combine *CyPhyCircus* models of the actual co-models and of the master algorithm chosen. In Deliverable D2.2d, we give a CSP (and, therefore, *CyPhyCircus*) model that gives a general characterisation of valid mater algorithms and a collection of examples of CSP models for some master algorithms. In the final year, we will give a CSP model of the INTO-CPS master algorithm and prove its validity.

We instantiate our approach by applying it to the engineering of mobile and autonomous robot applications in a collaboration with the UK EPSRC RoboCalc project.

CyPhyCircus models for VDM-RT and Modelica are the subject of Deliverables D2.2b and D2.2c. In this work, we provide a way to generate models automatically. They are useful to generate models for the FMUs of a given FMI co-simulation.

Any language with a *CyPhyCircus* semantics can be considered in our framework. In Figure 1, we name also RoboChart as an example. In Deliverable D2.2d, we give a CSP model that characterises a valid FMU according to the FMI standard. In the final year, we will define wrappers for our VDM-RT and Modelica semantics that define a valid FMU

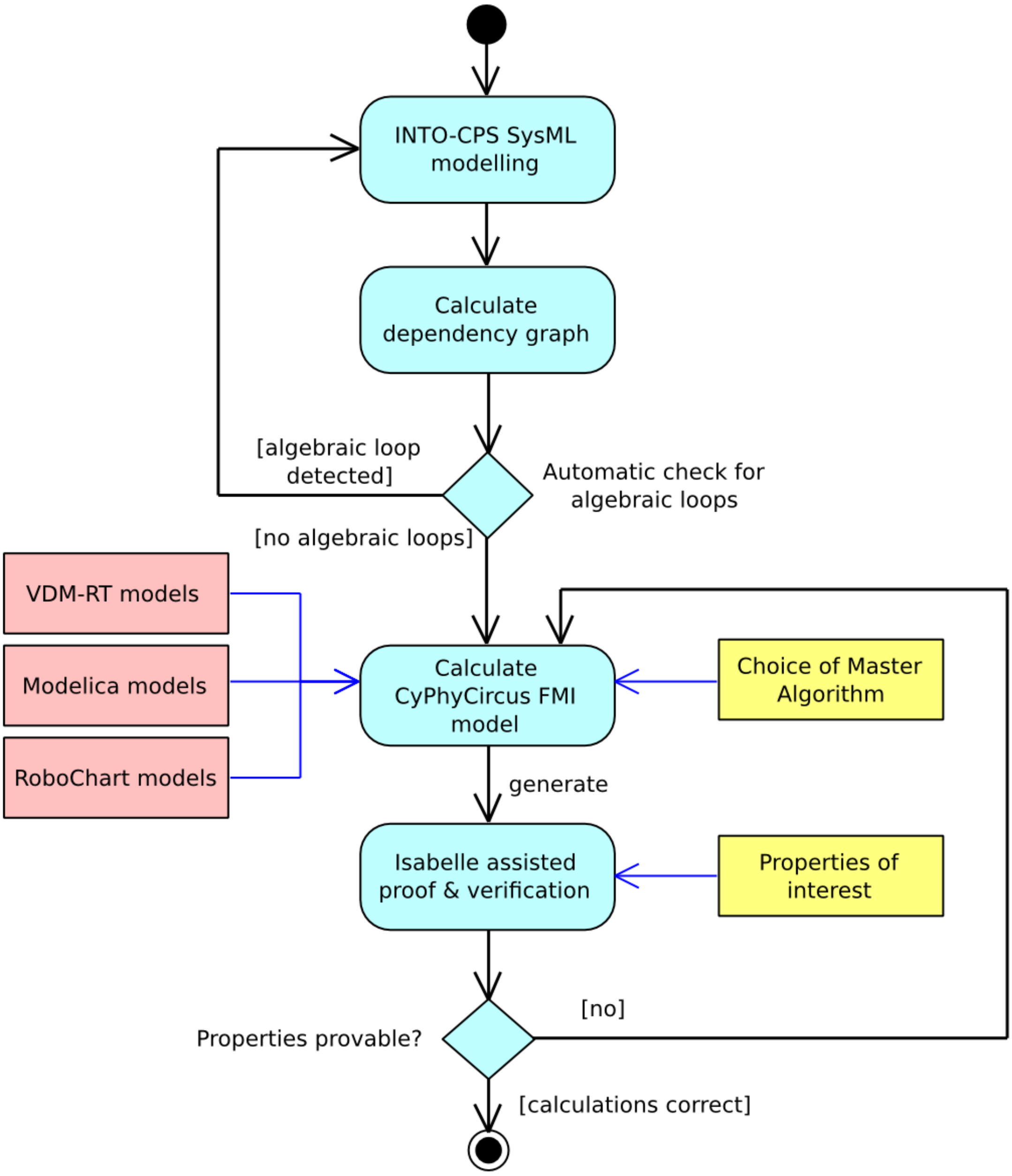
INTO-CPS SysML modelling
Calculate dependency graph
[algebraic loop detected]
Automatic check for algebraic loops
[no algebraic loops]
VDM-RT models
Modelica models
RoboChart models
Calculate CyPhyCircus FMI model
Choice of Master Algorithm
generate
Isabelle assisted proof & verification
Properties of interest
[no]
Properties provable?
[calculations correct]

Figure 1: Semantics and Verification infrastructure

model, given the *CyPhyCircus* semantics of a VDM-RT or Modelica model as defined in Deliverables D2.2b and D2.2c. The wrapper for VDM-RT is going to be based on the simulation view of this language taken in the INTO-CPS approach.

With a *CyPhyCircus* model that gives a comprehensive view of an FMI-based combination of co-models, we have an asset that supports reasoning beyond simulation. Our encoding of *CyPhyCircus* and its components in Isabelle provide practical support for theorem proving. For global system properties, model checking is unlikely to be feasible (although it has been proved useful for validation of the several model components, as described in Deliverables D2.2a, D2.2b, and D2.2d.) In the final year, a lot of effort will be devoted to mechanising the *CyPhyCircus* models of the co-simulation components in Isabelle, and using the produced infrastructure to verify INTO-CPS co-simulations.

Properties of interest to prove can come from the high-level INTO-SysML model and other sources. In the last year, we will consider proof of system properties of the railway case study. Proof can confirm that critical properties observed via co-simulation are universally true, and not just valid for the scenarios considered in simulation.