ARTICLE IN PRESS

Mechatronics xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

Simulation of cyber physical systems behaviour using timed plant models[☆]

Nuno Canadas[a], José Machado[a,*], Filomena Soares[b], Carlos Barros[a], Leonilde Varela[c]

[a] *MEtRICs Research Centre, Mechanical Engineering Department, University Minho, Guimarães, 4800-058, Portugal,*
[b] *Algoritmi Research Centre, Industrial Electronics Department, University of Minho, Guimarães, 4800-058, Portugal*
[c] *Algoritmi Research Centre, Production and Systems Depart, University of Minho, Guimarães, 4800-058, Portugal*

ARTICLE INFO

Keywords:
Cyber physical systems design
Plant modelling
Controller modelling
Simulation

ABSTRACT

When developing a reliable controller for cyber-physical systems, one of the main issues is to guarantee that all behaviour properties of such systems will be accomplished. For this purpose, it is very important to find and use formalisms and tools in order to model the controller and the respective plant. Moreover, it is very important to consider the same formalism for modelling both. The accurate model of a plant is usually difficult to obtain. To solve this problem, several techniques for modelling the plant have been developed in the recent past years. However, some of them have lacks that do not allow a fast, reliable and flexible way for plant modelling. In this paper a systematic approach for modelling cyber physical systems using timed automata is presented. The proposed approach allows defining a systematic way for creating global models for both the plant and the controller. These models, may be used to simulate and validate the cyber physical systems. For this purpose software tools, as UPPAAL, and other simulation configurations, as Model-in-the-loop and Hardware-in-the-loop, can be used. In order to present and explain the proposed methodology, a virtual platform and a physical workbench have been developed, from a transport objects station example, where Model-in-the-loop and Hardware-in-the-loop configurations are used in a subsequent manner in the context of the presented system modelling methodology.

1. Introduction

Cyber-physical systems (CPS) are heterogeneous, hybrid, distributed, real-time systems. Successful development of CPSs is a challenging problem pushing the state of the art. One important technique for verification and validation is simulation, which requires the construction of a useful model. In general, this is a two-dimensional model. In one aspect, the CPS spans a variety of disciplines, including software engineering and control systems engineering. These different engineering paradigms have their own, very different, modelling techniques. So, a useful CPS model must necessarily be composed by other models of system software, system architecture, underlying computation platforms, and physical processes. The feedback loop between physical processes and computations will require modelling sensors, actuators, physical dynamics, computation, software scheduling, and networks with contention and communication delays. Producing an integrated model for those systems is a challenge to the state of the art [1].

Nowadays the conjunction of embedded computing and network technology enables to conceive such systems and some works have been proposed for modelling, simulation, testing and control of those systems [2].

Model-based designs can enhance the development of these systems by allowing alternative designs to be evaluated before building (expensive) physical prototypes. The behaviour of a CPS can be simulated and tested by coupling the controller model with the plant model in a co-simulation approach. A CPS model must be composed by models of the system software, system architecture, core computation platforms, and physical plants in order to obtain a model closer to the real system. The modelling of the closed loop system requires modelling sensors, actuators, physical dynamics, software scheduling, and networks with contention and communication delays. Producing an integrated model is a challenge [3]. It is then necessary to model the controller behaviour, the plant behaviour and the interaction between both models, following the schematic representation presented in Fig. 1 [4].

Due to different time scales - corresponding to controller behaviour and plant behaviour - during the real evolution of the system in real-world, it is necessary to use some variables for coordination of evolution of models, during simulation tasks, in order to be obtained more accurate simulation results.

[☆] This paper was recommended for publication by Associate Editor Prof. T.H. Lee.
[*] Corresponding author.
E-mail address: jmachado@dem.uminho.pt (J. Machado).

http://dx.doi.org/10.1016/j.mechatronics.2017.10.009
Received 31 December 2016; Received in revised form 7 October 2017; Accepted 21 October 2017

Please cite this article as: Canadas, N., Mechatronics (2017), http://dx.doi.org/10.1016/j.mechatronics.2017.10.009

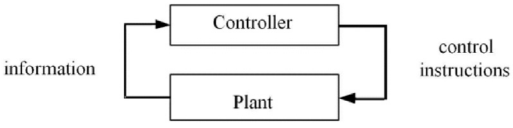

Fig. 1. Schematic representation of system interaction.

CPS spans a variety of disciplines, including software engineering and control systems engineering. These different engineering paradigms have their own, very different, modelling techniques.

To address the complexity of CPS, compositional semantics becomes central. It must address the following issues: models' consistency; realistic models of system architecture (including connectivity delays and timing properties); physical separation (including components with differing time granularity, faulty communication, information distribution, and consensus), [5,6]. This work is focused on the plant modelling and its influence on the system's behaviour in simulation.

Typically, such systems are designed and analysed using a variety of modelling formalisms and tools where different models emphasise certain features and disregard others to make analysis tractable. Moreover, a particular formalism is used to represent either the cyber or the physical process, but not both. Whereas differential equations are used for modelling physical processes, frameworks such as Petri nets and automata are used to represent discrete behaviour and control flows. Although this approach to modelling and formalisms may suffice to support a component-based "divide and conquer" approach to CPS development, it poses a serious problem for verifying the overall correctness and safety of designs at the system level and component-to-component physical and behavioural interactions [7]. This is a gap that makes crucial to develop models for both (controller and plant) that can use the same formalism and the same software tools, using analysis techniques in order to improve the reliability of the developed system.

1.1. Plant modelling

A plant model is a (more or less) precise representation of the physical system dynamics. It can be used to simulate the behaviour of the plant at its operating point, answering questions about its behaviour in real operation, via the analysis through simulation or other related techniques. There may be multiple models for a single physical system, with different levels of fidelity depending on the phenomena of interest and what it is it intended for.

Models are used to simulate physical systems behaviour instead of doing experiments on the real plant. They allow a vast number of tests and simulations in a shorter time, without the practical problems associated to an experiment (plant's components wear, experience's high costs or high danger process).

Plant models can be developed considering a continuous-time (CT) approach, a discrete time (DT) approach or a discrete event (DE) approach.

For CT modelling of physical plant there are some languages and formalisms that can be used as, for instance, Modelica [8] or Simulink [9]. Modelica is a non-proprietary, object-oriented, equation-based language to conveniently model complex physical systems. A large number of Modelica model libraries are available for modelling. Simulink is the world's leading modelling and simulation tool for CT systems.

For DT and/or DE modelling of physical plants, some formalisms and tools can be used, such as Net Condition/Event systems or Timed Net Condition/Event systems [10], Finite-state machines [11], Petri nets [12], bond-graphs [13] and others.

For all those approaches, concerning modelling the plant, there is a huge limitation: it is not possible to model a realistic system, from industrial point of view. The size of the systems that can be modelled – and analysed by using analysis techniques (such as simulation and formal verification) - is always very small comparing with industrial needs.

In order to deal with realistic problems, the domain of DT or DE is more adequate and there are some works that consider simulation and formal verification of those systems [14]. On this context (DT and/or DE) there are two main approaches to build plant models: the monolithic approach and the modular approach.

The monolithic approach, as can be observed in [15–18] considers obtaining a plant model in a single block. This approach is not modular neither refined. This way, for obtaining plant models, it is limited to systems with small size. In fact, realistic (industrial) systems are not possible to be obtained due to the limitation of human capacity for modelling them. On the other hand, all states of those models are pertinent because they have been obtained by the correct reasoning of the designer.

The modular approach, for modelling physical plants, consists in "dividing" the plant in components or parts that compose them. Those components are named as "modules" and can represent different components of the physical system [19–22]. A module is a component of the physical system, as, for instance, a valve, a cylinder, a sensor,… Considering this approach, modelling tasks start with modelling each module and, then, a final monolithic model will be obtained by composition of all modular models of the system. Many times, the final monolithic model is obtained by the Cartesian composition of the modules' models [23]. In this case, the global obtained model has a higher size (number of states and transitions) than the number of states corresponding to the behaviour of the real plant because some of those states have no physical behaviour meaning: they exist because of the composition of the modules' models.

Taking into account the presented gaps, this paper presents a methodology for obtaining plant models that considers modular modelling; considers DT domain; considers the same formalism for final versions of the plant model and controller model; uses the same software for analysing the behaviour of the system (controller and plant); allows using simulation and formal verification techniques, with the same models and same software tool; allows considering systems with realistic (industrial) size; and, mainly, focuses on discrete event simulation. Moreover, it is presented a case study, for illustration of the methodology, when obtaining the plant models for simulation purposes. In this context, a virtual platform is created and Model-in-the-Loop (MiL) simulation is performed. Also, a physical workbench is developed and Hardware-in-the-Loop (HiL) simulation is performed taking into account the same developed plant models.

1.2. Controller modelling

The controller model is a representation of the system's controller (more or less detailed, depending on the simulation purpose and the features to be tested), used to simulate its behaviour while it is in operation, or to test specific modules. Controller and plant exchange electrical signals: orders (outputs) are sent from the controller to the plant and inputs (sensor's information, buttons and levers signals, etc.) are sent from the plant to the controller, which are modelled as continuous-time or discrete-time variables, in order to trigger specific actions from both blocks, as it is illustrated in Fig 1.

Using these variables, the interaction between the controller model and within the plant models may be established and used to observe through simulations and animations the behaviour of a system. This is very useful for developing dependable and robust controllers for automated systems because the system's reactions may be observed in different environments and adverse situations.

In this paper a methodology to build a Discrete Time controller model is presented.