Proceedings of the 16th Conference on Computer Science and Intelligence Systems pp. 485–488 DOI: 10.15439/2021F98 ISSN 2300-5963 ACSIS, Vol. 25

Modeling and Verification using Different Notations for CPSs: The One-Water-Tank Case Study

Sergey Staroletov
Polzunov Altai State Technical University /
Institute of Automation and Electrometry, Russia
Email: serg_soft@mail.ru

Horst Schulte, Thomas Baar
HTW Berlin, School of Engineering I,
Berlin, Germany
Email: {schulte, baar}@htw-berlin.de

Ivan Konyukhov, Nikolay Shilov
Innopolis University, Innopolis,
Republic of Tatarstan, Russia
Email: i.konyukhov@innopolis.ru, shiloviis@mail.ru

Andrei Rozov, Tatiana Liakh, Vladimir Zyubin
Institute of Automation and Electrometry / NSU,
Novosibirsk, Russia
Email: {rozov, liakh, zyubin}@iae.nsk.su

***Abstract*—The choice of an adequate notation and subsequent system formalization are the crucial points for the design of cyber-physical systems (CPSs). Here, an appropriate notation allows an explicit specification of the deterministic system behavior for specified initial states and inputs. We base our study on an industrial example (water tank) that comprises nominal as well as safety-critical states, and focus on the notation's support to validate/verify crucial safety properties. Several industrial notations (e.g. Matlab/Simulink©) to design and simulate such a hybrid system have been tried based on our physical model. In addition, we remodel our example using the well-founded mathematical formalism of hybrid automata. It enables us to formally express and verify important safety properties using the theorem prover KeYmaera.**

I. Introduction

The increasing complexity and use of cyber-physical systems (embedded in particular) in our lives requires a reassessment of the system design principles and tools. The most challenging designs are safety-critical systems, such as transportation systems (e.g., airplanes, cars, and trains), infrastructure systems (power grid, water management, traffic control), and medical equipment. Here, the correct behavior under different environmental conditions must be ensured. The system must be fault-tolerant, i.e. in the case of faults, it must be automatically responsive to prevent worse [1].

In practice, different mathematical and formal models are used for system design and analysis, validation and verification. In the modeling process, a distinction should be made between two kinds of models [2]. The models of the first type are used for simulation and should be able to represent the characteristic behavior of systems in the physical world; the value of a model of this type lies in how well its properties match those of the real world, but we should emphasize that the model fidelity is never perfect.

Models of the second type serve as blueprints or specifications for the design of real-world (control) systems; system design and implementation follow the specification of a model; thus, the value of a model of the second type lies in how well and easy a real system can be constructed to implement the behavior specified in the model.

Due to the complexity and heterogeneity, mathematical models of real-world control systems should be described by different mixed representations. Hybrid systems are a particular class of mixed models that focus on the combination of discrete and continuous subsystems. For example, controllers for local operating regions can be represented mathematically by continuous-time systems, where the switching mechanism between different control regions are mostly represented as discrete event systems. The whole behavior is described by a hybrid model including event-based, discrete state changes, and continuous property changes over time.

Because of space limitations, a survey of a vast educational and periodic literature on simulation approach to design, modeling and validating real-world cyber-physical systems is out of the scope of this short conference paper (interested reader can follow our survey [3]). An overview on the formalism and notation of hybrid systems is available in [4] and [5]. Topics related to hybrid systems model-checking-oriented specification and verification are addressed in many papers, e.g. [6], [7], and [8]. An introduction to proof-oriented logical analysis and verification of hybrid systems is presented in the monographs [9], [10].

There are also some publications related to model checking and verification of particular hybrid systems. For example, paper [11] discusses hybrid system modeling and control for large-scale power systems; an analysis of embedded control software in safety-critical systems like autonomous vehicles in urban environments is presented in [12].

The rest of the paper is organized as follows. The one-tank system as our benchmark is informally introduced in Section II; in addition, this section presents a mathematical analysis of the system. In Section III, we use prevailing industrial techniques to ensure model properties, including safety-critical behavior. An alternative approach is given in

This work was supported by the German Science Foundation (DFG-Deutsche Forschungsgemeinschaft), funding program for international research cooperation, grant no. BA 6655/1-1, SCHU3174/3-1.

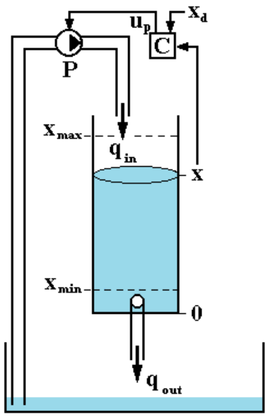

Fig. 1. The water tank system

Section IV with the formalization of the benchmark using the notation of hybrid automaton, which allows the rigorous logical verification of safety properties using the proof assistant KeYmaera. Finally, Section V concludes the paper.

II. Process Control Benchmark: Water tank

The water tank system (Fig. 1) can be viewed as a prototype (a core in some sense) of many industrial process control applications, e.g. in chemical plants or oil and gas systems. The typical control problem is to track the tank level by an input flow q_{in} under various disturbances. Moreover, the water tank process with one, two or three tanks is often used as a benchmark for fault diagnosis and isolation as well as fault-tolerant control [13].

The system consists of the water tank with cross-sectional area A_T and height H, orifice with cross-sectional area A_{out}, level sensor, pumping unit P, controller C and water basin. In this setup, the pump provides in-feed of the water q_{in} to the tank, and the outflow of the tank is denoted as q_{out}.

The following conditions with regard to the system are used to describe the level of the water x in the tank:

- The level of the water is measured by the sensor.
- The controller is able to force on the pumping unit by changing the voltage u_p applied to the input terminals of the pump.
- In nominal conditions, the controller allows to keep desired level of the water x_d in a predicted range $[x_{min}, x_{max}]$.
- The controller handles the situation of low level protection (when $x < x_{min} + \Delta_{IN}$) as well as the situation of the high level protection (when $x > x_{max} - \Delta_{IN}$).
- The level protection states are entered well before water the level x gets too close to x_{min}, x_{max}.

A. Model-based Hybrid Controller Design

The dynamic equation for the water level is derived as follows. The rate change of the water level in a time is given by

$$\dot{x}(t) = \frac{1}{A_T}(q_{in}(t) - q_{out}(t)), \tag{1}$$

Symbol	Description	Value	Unit
x	water level of the tank	$\in [0, H]$	m
x_d	desired water tank level (set point)	$\in [x_{\min}, x_{\max}]$	m
u_p	plant input / controller output	$\in [u_{p,\min}, u_{p,\max}]$	V
q_{in}	incoming flow rate	$\in [0, q_{\max}]$	m^3/s
A_T	cross-section of the tank	$7.9 \cdot 10^{-3}$	m^2
A_{out}	cross-section of the output orifices	$2.9106 \cdot 10^{-5}$	m^2
K_{pum}	pump coefficient	$8.374 \cdot 10^{-6}$	$m^3/(Vs)$
$x_{\min}$	lowest permitted water level	0.1	m
$x_{\max}$	highest permitted water level	0.5	m
H	tank height	0.65	m
$q_{\max}$	maximum flow rate of pump P	$100 \cdot 10^{-6}$	m^3/s
α_{out}	flow coefficient of output orifices	0.7	–
g	Earth gravity	9.81	m/s^2

TABLE I
Variables and parameters of the water tank system

where x, A_T, q_{in}, q_{out} are the water level, cross-sectional area of the tank, inflow rate, outflow rate, respectively. Next, note that the inflow rate to the tank is given by

$$q_{in}(t) = K_{pum} u_p(t), \tag{2}$$

where K_{pum} is the pump coefficient and $u_p(t)$ is the voltage applied to the pump. In addition, using the Torricelli's law for a flow through a small orifice, the outflow rate of the water from the tank is given by

$$q_{out}(t) = \alpha_{out} A_{out} \sqrt{2gx(t)}, \tag{3}$$

where g is the gravitational acceleration, A_{out} denotes the cross-sectional area of the orifice and α_{out} is the flow coefficient of the orifice. Using the (1–3), we obtain the dynamic equation for the water level in the tank as

$$\dot{x}(t) = \frac{1}{A_T}(-\alpha_{out} A_{out} \sqrt{2gx(t)} + K_{pum} u_p(t)) \tag{4}$$

or in simple notation,

$$\dot{x}(t) = -\gamma \sqrt{x(t)} + \beta u_p(t) , \tag{5}$$

where

$$\gamma := \frac{\alpha_{out} A_{out}}{A_T} \sqrt{2g} \quad , \quad \beta := \frac{K_{pum}}{A_T} \quad . \tag{6}$$

All variables and values of actual parameters are recorded in Table I.

B. Hybrid Control for Low and High Level Protection

Due to the linear controller design for the nonlinear water tank system (that operates in a large operating range $x \in [x_{min}, x_{max}]$) and additional model uncertainties, the desired reference model with an overshoot free behavior is not exactly reached. Nevertheless, in order to be able to fulfill the requirements the system is extended by two control states. To distinguish them from the nominal states, these are denoted by *High Level Protection* (HP) and *Low Level Protection* (LP) state. During HP the controller output is set to $u_p = u_{p,min} = 0$. That means the pump is switched off as long as condition $x > x_{max} - \Delta_{OUT}$ holds. On the other hand, in the case of the LP state, the pump is set to the maximum flow rate q_{max}, that means (see (2)) the controller output is set to the constant value $u_p = u_{p,max}$. Note that the conditions from the nominal control state to the LP/HP state and from the LP/HP state to the nominal control state contain different delta values Δ_{IN} and Δ_{OUT}. With this, if $\Delta_{IN} < \Delta_{OUT}$ is fulfilled, it can be guaranteed that fast switching between states (scattering) is avoided.

III. Quality Assurance by Simulation

The so-called hardware (HIL) and software-in-the-loop (SIL) approaches are popular in industrial practice to ensure code quality. The difference between HIL and SIL is that the latter does not use target hardware to verify the code. In this paper, we use SIL-based verification, where the plant model and the controller code are simulated in the same environment.

For the verification by simulation with Matlab/Simulink©, the following controller settings for the nominal controller, the calculation rule of the controller coefficients and the switching conditions LP/HP are chosen as follows.

- **Nominal Controller Design**: We obtain the controller coefficients
$$k_p = 31.2634\,, \qquad k_I = 0.4016 \tag{7}$$
by using a desired reference dynamics with $\tau_{ref} = 30$, a chosen steady-state water level of $x^{ss} = 0.2\,\mathrm{m}$ and the given plant parameter of Table I.
- **Switching conditions**: The thresholds of the lowest and highest permitted water level are determined by the controller requirements. The values are listed in Table I. The relative thresholds $\Delta_{IN,OUT}$ are defined as
$$\Delta_{IN} = 0.01\,, \qquad \Delta_{OUT} = 0.05\,. \tag{8}$$

A selected simulation result for a given curve of reference values $x_d(t)$, $t = [0\,, 2000s]$ using the parameter setting (7), (8) is shown in Fig. 2. In the bottom diagram, the state values correspond to the implementation with the *high level protection* state denoted as STATE_HP = 1, the *nominal control* state as STATE_NC = 2, and the *low level protection* state as STATE_LP = 3. The simulation results clearly show (by visual inspection) that the controller meets the previously defined requirements for the given reference case.

To address new challenges of today's control software, some of the authors proposed a *process-oriented approach*, which has been implemented in a family of domain-specific programming languages such as Reflex, Industrial C and PoST. In this approach, control software is represented as a set of interacting processes, where processes are state machines enhanced with special operators that implement concurrent flow control and time-interval managing [14]. So, after modeling, these languages can be used for implementation of the system.

IV. Formal Verification with KeYmaera

In the previous sections, the water tank system has been formalized using different notations. Remark that all industrial notations discussed above use exclusively simulation as a technique to check, whether the system behaves as expected. Generally, there is a lack of tool support for proving system properties merely based on the static system description. Only

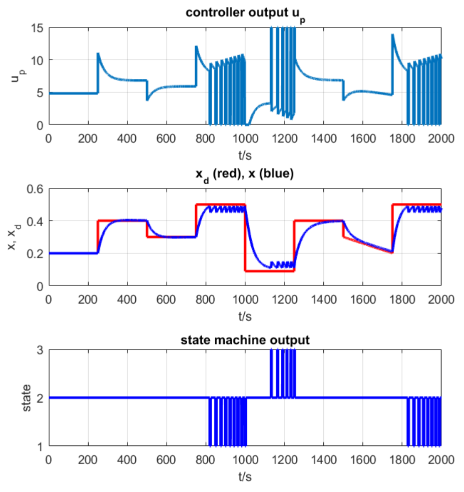

Fig. 2. Verification by simulation with Matlab/Simulink©

a rigorous analysis of the system can certify, that all expected safety properties actually hold under all circumstances.

Because the simulation lacks soundness and comprehensiveness, we develop an additional system formalization using the hybrid automata approach [15], [4]. In this paper, we focus on the formal verification of the safety property that the current water level x never exceeds x_{max}, i.e. always $x < x_{max}$ holds.

As a verification tool we have chosen and used KeYmaera [16], [9]. We encoded a part of the problem in the special input code that is presented in Fig. 3. Actually, we have excerpted a part of the complete system description and focused on the state HP, which is most relevant for safety property $x < x_{max}$.

A hybrid automaton consists of states connected by transitions that encode the control flow. In contrast to classical automata, a state (e.g. HP) can be annotated with differential equations[1] (e.g. $x' = -\gamma\sqrt{x} + \beta u_p$, $u'_p = 0$) to encode, how the values of continuous variables (e.g. x, u_p) evolve while the system remains in the state. In addition, a state can be annotated with so-called domain constraints. A domain constraint is a condition to be true while the system is within the annotated state (e.g. $x >= x_{max} - \Delta_{OUT}$ for state HP); when some domain constraint of a state becomes invalid because of the change of some values of variables (e.g. value for x has fallen below $x_{max} - \Delta_{OUT}$), then the system is forced to leave the state via an outgoing transition (in case of HP, it can be left to state NC or to the final state). Note that a state can be left at any random time as long as there is an outgoing transition whose annotation condition allows its firing.

In order to check our safety property $x < x_{max}$, it is sufficient to show the property for the final state, since all states of the automaton (HP, NC, LP) are directly connected with the final state by an unconditioned transition that can fire any time and move the system to the final state. Thus, it is sufficient

[1]We use here x' instead of $\dot{x}$ to denote the derivative of x simply for consistency with the literature, e.g. [10].

```
\problem {
\[ R Xmin, X, Xmax, DeltaIn, DeltaOut,
 Up, Betta, Gamma, Aout, Alphaout, At, Kpum, g
\]
(0 < Xmin) & (Xmin < X) & (X < Xmax) &
 (DeltaIn = 0.01) & (DeltaOut = 0.05) &
 (Aout = 2.9106*10^(-5)) & (Alphaout = 0.7) &
 (At = 7.9*10^(-3)) & (Kpum = 8.374*10^(-6)) &
 (g = 9.81) &
 (Gamma = Alphaout * Aout * (2*g)^(1/2) / At) &
 (Betta = Kpum / At) &
 (Xmin = 0.1) & (Xmax = 0.5) & (X > Xmax - DeltaIn)
 ->
  (\[
     Up := 0;
     {
          X' = -Gamma * X ^ (1/2) + Betta * Up,
          Up' = 0,
           (X <= Xmax - DeltaOut)
     }
   \]
  (X < Xmax))
}
```

Fig. 3. HP in KeYmaera notation

to specify the safety property $x < x_{max}$ as part of the post-condition.

The only state that could violate safety property $x < x_{max}$ is *HP*, since for all other states this property is a weaker form of the state's domain constraint.

For state *HP*, the argumentation goes as follows:

- *Before entering HP, the value for voltage u_p is set to 0 and while the system stays in HP, this value remains 0 since $u'_p = 0$ holds in HP.*
- *Furthermore, upon entering HP, we know that $x < x_{max}$ as it is specified on all incoming transitions. Since we have $x' = -\gamma\sqrt{x} + \beta u_p$ together with $u_p = 0$, we know that x' is always negative in HP and x falls over time.*
- *Consequently, $x < x_{max}$ does hold when the process enters the state HP and for the whole period of staying in HP.*

The code for *HP* in KeYmaera notation is shown in Fig. 3.

The KeYmaera tool allows to transform the above informal mathematical argumentation into a formal proof and does check every proof step for correctness. At the end we get a formal verification that the desired safety property actually holds. Note that the verification process helps to make all assumptions explicit, e.g. it is very crucial to know that $0 < \Delta_{IN}$.

V. CONCLUSION

There are many different notations for the design, modeling, and analysis of control systems. They have proven to be useful in numerous industrial projects but also differ in terms of the used paradigms (e.g. object orientation, explicit state representation, etc.).

However, as our example shows, there is still a considerable gap in the usage of modeling concepts that still prevents an easy translation of, for example, Matlab/Simulink© models into input models for hybrid theorem provers like KeYmaera. It should also be noted that the KeYmaera tool cannot automatically find proofs for non-trivial properties and requires substantial user input (cmp. [17]).

In future, we plan to address these problems. We are developing an intermediate notation between industrial-strong modeling notations and hybrid automata to gain synergy. Our intermediate notation will semantically be strongly based on mathematically well-founded hybrid automata, but will also provide higher modeling concepts, that will facilitate the transition of industry models into our notation.

REFERENCES

[1] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*, 2nd ed. Springer-Verlag Berlin Heidelberg, 2006. ISBN 978-3-540-35653-0

[2] E. A. Lee, "Fundamental limits of cyber-physical systems modeling," *ACM Transactions on Cyber-Physical Systems*, vol. 1, no. 1, 2016. doi: 10.1145/2912149

[3] S. Staroletov, N. Shilov, I. Konyukhov, V. Zyubin, T. Liakh, A. Rozov, I. Shilov, T. Baar, and H. Schulte, "Model-driven methods to design of reliable multiagent cyber-physical systems," in *CEUR Workshop Proceedings*, vol. 2478, 2019, pp. 74–91.

[4] T. A. Henzinger, "The theory of hybrid automata," in *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996*, 1996. doi: 10.1109/LICS.1996.561342 pp. 278–292.

[5] R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, Eds., *Hybrid Systems*, ser. Lecture Notes in Computer Science, vol. 736. Springer, 1993. doi: 10.1007/3-540-57318-6

[6] R. Alur, T. A. Henzinger, and P. H. Ho, "Automatic symbolic verification of embedded systems," in *Proc. of the 14th Annual Real-time Systems Symp.*, 1993. doi: 10.1109/32.489079 pp. 2–11.

[7] B. I. Silva, O. Stursberg, B. H. Krogh, and S. Engell, "An assessment of the current status of algorithmic approaches to the verification of hybrid systems," in *Proc. of the 40th IEEE Conf. on Decision and Control*, 2001. doi: 10.1109/CDC.2001.980711 pp. 2867–2874.

[8] S. Mitsch and A. Platzer, "Modelplex: verified runtime validation of verified cyber-physical system models," *Formal Methods System Design*, vol. 49, no. 1,2, pp. 33–74, 2016. doi: 10.1007/s10703-016-0241-z

[9] A. Platzer, *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Heidelberg: Springer, 2010. ISBN 978-3-642-14508-7

[10] A. Platzer, *Logical Foundations of Cyber-Physical Systems*. Springer, 2018. ISBN 978-3-319-63587-3

[11] G. K. Fourlas, K. J. Kyriakopoulos, and C. D. Vournas, "Hybrid systems modeling for power systems," *Circuits and Systems Magazine, IEEE*, vol. 4, no. 3, pp. 16–23, 2004. doi: 10.1109/MCAS.2004.1337806

[12] K. G. Larsen, "Verification and performance analysis for embedded systems," in *Third IEEE International Symposium on Theoretical Aspects of Software Engineering (TASE)*. Tianjin, China: IEEE Computer Society, July 2009. doi: 10.1109/TASE.2009.66

[13] A. Kroll and H. Schulte, "Benchmark problems for nonlinear system identification and control using soft computing methods: Need and overview," *Applied Soft Computing*, vol. 25, no. 12, pp. 496–513, December 2014. doi: 10.1016/j.asoc.2014.08.034

[14] I. Anureev, N. Garanina, T. Liakh, A. Rozov, H. Schulte, and V. Zyubin, "Towards safe cyber-physical systems: the Reflex language and its transformational semantics," in *2019 International Siberian Conference on Control and Communications (SIBCON)*. IEEE, 2019. doi: 10.1109/SIBCON.2019.8729633 pp. 1–6.

[15] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems," in *Hybrid Systems*, 1992. doi: 10.1007/3-540-57318-6 pp. 209–229.

[16] J.-D. Quesel, S. Mitsch, S. Loos, N. Aréchiga, and A. Platzer, "How to model and prove hybrid systems with KeYmaera: A tutorial on safety," *STTT*, vol. 18, no. 1, pp. 67–91, 2016. doi: 10.1007/s10009-015-0367-0

[17] T. Baar and S. Staroletov, "A control flow graph based approach to make the verification of cyber-physical systems using KeYmaera easier," *Modeling and Analysis of Information Systems*, vol. 25, no. 5, pp. 465–480, 2018. doi: 10.18255/1818-1015-2018-5-465-480