DEV&DESS based Verification Tool for Cyber-Physical System

Jin Myoung Kim, Hae Young Lee, In Geol Chun, Won Tae Kim and Seung Min Park
Embedded SW Division, Electronic and Telecommunication Research Institutes, South Koreak
{jm.kim, haelee, igchun, wtkim, minpark}@etri.re.kr

Tae Ho Cho
Information and Communication Engineering Department, Sungkyunkwan University, South Korea
taecho@ece.skku.ac.kr

Keywords: Modeling and simulation, DEV&DESS, hybrid modeling, CPS, Verification

Abstract

Cyber-Physical Systems (CPS) is a highly complex system integrating physical components having sensors/actuators with computing components for the control. As the complexity of the system, the system should be designed using model-based approach and it should be verified before an implementation. In this paper, we exploit a simulation methodology for verifying the designed system by model-based approach. Simulation is an ideal way to predict behavior in complex systems that we cannot otherwise test. We describe CPS features and its hybrid modeling language that is based on the DEV&DESS formalism. Also modeled systems are simulated via hybrid simulation environment. In experimental result, we show the implementation of the simulation environment.

1. INTRODUCTION

Due to the increasing software complexity in modern cyber-physical systems (CPS), there is a high likelihood for latent defects in the software. CPS is a highly complex system integrating physical components having sensors/actuators with computing components for the control. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa [1].

CPS should have the essential features of reliability and autonomy. So, the applications of CPS include high confidence medical devices and system, traffic controls, avionics, defense systems and so on. For example, in an unmanned aerial vehicle (UAV) with active wings, an embedded controller monitors the airflow over the wing surface and modulates it through electromechanical actuators to ensure laminar flow such that the vehicle is capable of extreme maneuvers [2].

One particular use of modeling and simulation (M&S) is in the development of embedded systems [3]. The construction of embedded system models and their analysis through simulation reduces both development cost and risks, while enhancing system capability and improving the quality of the final products [4].

In this paper, we apply M&S methodology to the implementation of high reliable systems, and develop a verification tools. Since CPS is integrations of computation with physical processes, we define a hybrid modeling language [5] that is based on DEV&DESS formalism [6]. DEV&DESS is a combination of DEVS (Discrete Event Specification) and DESS (Differential Equation Specified System) [6]. We show the structure of the hierarchical classes and implement the proposed method using C++ language.

The rest of the paper is organized as follows: Section 2 briefly explains about the related works. Section 3 demonstrates a CPS modeling and simulation. Section 4 shows an implementation of proposed method. Finally, conclusion and future work in section 5.

2. COMBINATION OF DEVS AND DESS

DEV&DESS formalism comes into being by a combination of DEVS and DESS formalism. Figure 1 show the combination. This figure illustrates the modeling concept, which has both DEVS and DESS elements working together.

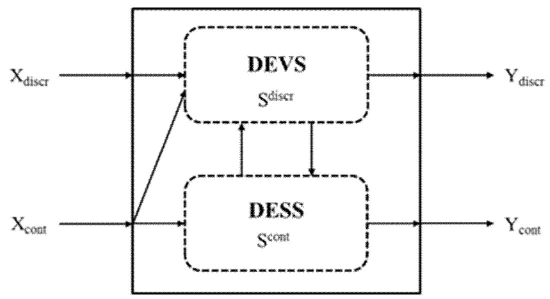

Figure 1. Combination of DEVS and DESS models

In the figure, X_{discr} and Y_{discr} are a set of discrete input and output, respectively. X_{cont} and Y_{cont} are a set of continuous input and output, respectively. DEV&DESS formalism is defined by the following equation:

$$DEV\ \&\ DESS = \left\langle \begin{matrix} X^{discr}, X^{cont}, Y^{discr}, Y^{cont}, \\ S^{discr}, S^{cont}, \delta_{ext}, C_{\text{int}}, \delta_{\text{int}}, \\ \lambda^{discr}, f, \lambda^{cont} \end{matrix} \right\rangle \quad (1)$$

where:

X^{discr}, X^{cont} are sets of discrete event inputs and outputs

Y^{discr}, Y^{cont} are sets of continuous event inputs and outputs

S^{discr}, S^{cont} are sets of discrete and continuous states

δ_{ext} is the external transition function

δ_{int} is the internal transition function

C_{int} is the event detection condition function

$\lambda^{discr}, \lambda^{cont}$ are the discrete event and continuous output function

f is the derivative function

A state event is considered to be the occurrence of a change in the value of the event condition predicate from false to true [7].

3. CPS MODELING AND SIMULATION

3.1. Modeling

DEVS is a general formalism for modeling and simulation of any discrete systems using hierarchical composition of behavioral model and structural models. Models of CPS systems have a continuous and discrete factor. So, we exploit a combination of DEVS and DESS and define a modeling language [5]. Also, we implement a modeling tool based on the modeling language.

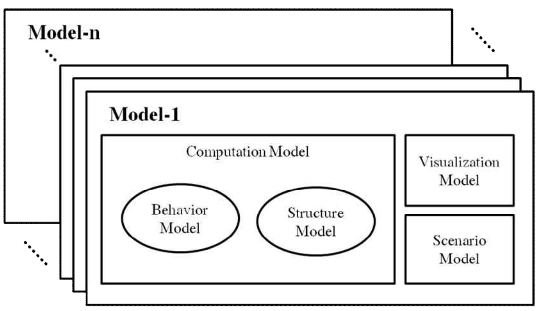

Figure 2. Model structure

Figure 2 shows a modeling structure. The model is basically composed of a visualization model, a scenario model and a computation model. The visualization model expresses objects, such as TANK and UAV, as 3D images. The scenario model includes the model's scenario in the in the simulation. The computation model consists of a structure model and a behavior model.

3.2. Simulations

Models that are defined by the modeling tool should be translated to model objects. Here, the model objects denotes translated models that are used to execute simulation as figure 3. The model objects are presented by C++ language. The model execution loads the translated models and executes these. Figure 3 shows a sequence of a model execution.

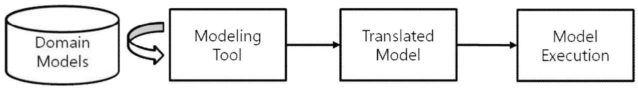

Figure 3. A sequence of model executions

Figure 4 is the class hierarchy of proposed hybrid simulation engine. Atomic class is behavioral model for DEVS and Digraph class is structural models. Hybrid class that inherits the atomic class defines DEV&DESS formalism. In hybrid modes, continuous parts cannot be simulated in their original form. So, the continuous parts should be replaced with approximations; with in fact, discrete event models amenable to computer simulation [8].

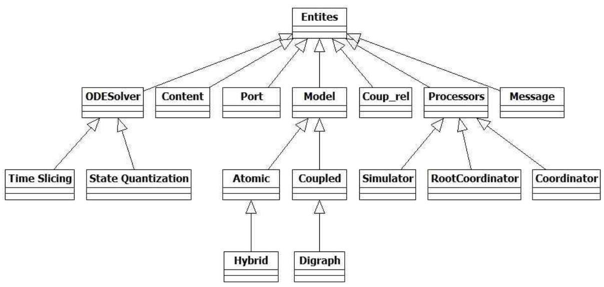

Figure 4. The class hierarchy of the hybrid simulation engine

Time Slicing class provides the method for numerical solution of an ordinary differential equation. It is the implementation of Euler's method. Since the time slicing method generates the event by elapsed time for continuous models, messages for simulation time synchronization bursts. It leads to the deterioration in simulation performance. State Quantization class provides a quantized state system [9].

4. IMPLEMENTED SOFTWARE

The simulation environment consists of the modeling tool, and the simulation engine. Eclipse based modeling tool is implemented by Java language. Figure 5 shows a computation model about an example UAV system.

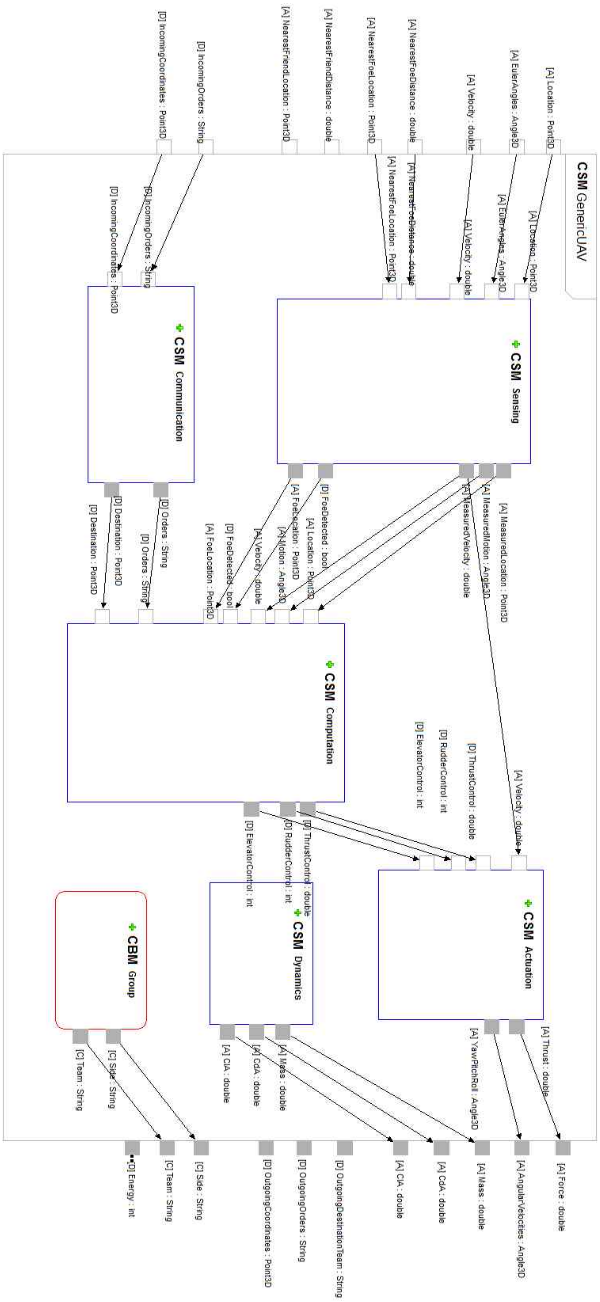

Figure 5. The computation model in the modeling tool

CSM (CPS structure model) and CBM (CPS behavior model) implies a structure model and a behavior model, respectively. As mentioned in section 3.2, to execute a simulation, the model should be translated the object model. Figure 6 is an example of translated model.

```
class Alpha1Computation : public digraph
{
protected:
        port portDestination;
        port portDesiredThrust;
        port portDesiredHeading;
        port portOrders;
        port portLocation;
        port portMotion;
        port portVelocity;
        port portDesiredAltitude;

public:
        // Constructor
        Alpha1Computation ()
        {
                // I/ O Port Naming
                portDestination.SetPortName("Destination");
                portDesiredThrust.SetPortName("DesiredThrust");
                portDesiredHeading.SetPortName("DesiredHeading");
                portOrders.SetPortName("Orders");
                portLocation.SetPortName("Location");
                portMotion.SetPortName("Motion");
                portVelocity.SetPortName("Velocity");
                portDesiredAltitude.SetPortName("DesiredAltitude");

                // Add port
                Add_inport (portDestination);
                Add_outport (portDesiredThrust);
                Add_outport (portDesiredHeading);
                Add_inport (portOrders);
                Add_inport (portLocation);
                Add_inport (portMotion);
                Add_inport (portVelocity);
                Add_outport (portDesiredAltitude);

                // Add models to global DB
                PutCPSModel ("Alpha1Computation", this);
        } ? end Alpha1Computation ? ;

        // Add sub model and coupling
        virtual void Couple ()
        {
                AddSubmodel ("Alpha1NavigationController");
                AddSubmodel ("Alpha1ThrustController");
```

Figure 6. Translated model that is based C++ language

5. CONCLUSIONS AND FUTURE WORKS

A simulation is an ideal way to predict behavior in complex systems. Through the simulation, risks and end cost can be reduced. CPS is integrations of computation and physical processes and requires a high reliability.

In this paper, we exploit a simulation methodology for verifying the designed system by model-based approach. For modeling CPS, we define the hybrid modeling language that is based on DEV&DESS formalism. Developed modeling tool supports the hybrid modeling langue. Also, to simulate the CPS model, we implement simulation environment.

References

[1] Edward A. Lee. (2008). Cyber physical systems: Design challenges. ISORC, pp. 363-369.

[2] Gabor Karsai and Janos Sztipanovits. (2008). Model-Integrated Development of Cyber-Physical Systems. LNCS, 5287, pp. 46-54.

[3] Yu Y.H. and Wainer G. (2007). eCD++: An engine for executing DEVS models in embedded platforms. SCSC, pp. 323-330.

[4] Wainer G. and Glinsky E. (2004). Model-based development of embedded systems with RT-CD++. Proceeding of the WIP session, IEEE Real-Time and Embedded Technology and Applications Symposium.
[5] LEE H.Y. (2010). CPS Modeling Language Specification. Technical Repor, Embedded SW Division, Electronic Telecommunication Research Institutes.
[6] Zeigler B. P., Kim T. G. and Praehofer H. (2000). Theory of Modeling and Simulation. *Second Edition*, Academic Press, San Diego, California
[7] Pritsker, A.A.B. (1974). The GASP IV Simulation Language. John Wiley & Sons, New York, N.J
[8] James J. N. (2009). Building software for simulation. John Wiley & Sons, New York, N.J
[9] François Cellier and Ernesto Kofman. (2005). Continuous System Simulation. Springer, New York

Jin Myoung Kim is PhD candidate in Computer Engineering from Sungkyunkwan University. He was research engineer in Seocho R&D Campus, LG electronics. Currently He is a Research member of Engineering Staff, Embedded SW Research Department, ETRI, S. Korea. His research interests include cyber-physical system, modeling & simulation, artificial Intelligence, and wireless sensor Network.

Hae Young Lee received his Ph.D. in Computer Engineering and B.S. in Electrical and Computer Engineering from Sungkyunkwan University, Korea, in 2009 and 2003, respectively. He is currently a Senior Member of Engineering Staff, CPS Research Team, ETRI, Korea. His research interests include modeling & simulation, cyber-physical system, and wireless sensor network.

Ingeol Chun received his M.S. and B.S. degrees in Electrical and Computer Engineering from Sungkyunkwan University, Korea, in 1998 and 1996, respectively. He is currently a senior member of engineering staff in ETRI, Korea from 1999. His research interests are Cyber-Physical Systems, Autonomic Computing, Agent-oriented intelligence system, embedded systems and Software Engineering.

Wontae Kim received his B.E., M.E., and Ph.D. degrees in Electronic Engineering from Hanyang University, Seoul, Korea in 1994, 1996, and 2000, respectively. From Jan. 2001 to Feb. 2005, he was CTO of Rostic Technologies, a venture company which developed advanced mobile technologies. He joined ETRI, the major national research institute of Korea, in Mar. 2005. He is the team director of CPS (Cyber-Physical Systems) Research Team in Dept. of Embedded SW from Aug. 2010. His main research areas are CPS, RT Middleware, Autonomic Control, and High Confidential Computing.

Seungmin Park received his B.Sc. degree from Ulsan University, S. Korea, in 1981, M.Sc. degrees from Hongik University, S. Korea, in 1983; all major in electronics engineering. Since 1983, he is working at ETRI, S. Korea, as a principal member of technical staff. His research interests include computer communication, mobile computing, real-time OS, embedded software, sensor networking and Cyber-Physical Systems. He is in charge of embedded software research department of ETRI, since 2005.

Tae Ho Cho received the Ph.D. degree in Electrical and Computer Engineering from the University of Arizona, USA, in 1993, and the B.S. and M.S. degrees in Electrical Engineering from Sungkyunkwan University, Korea, and the University of Alabama, USA, respectively. He is currently a Professor in the School of Information and Communication Engineering, Sungkyunkwan University, Korea. His research interests are in the areas of wireless sensor network, intelligent system, modeling and simulation, enterprise resource planning.