An ETRI CPS Modeling Language for Hybrid System Simulation

Sanghyun Yoon

Konkuk University

TR-DS-2015-01

November 25, 2015

Konkuk University

Department of Computer Science and Engineering

Dependable Software Labortory

An ETRI CPS Modeling Language for Hybrid System Simulation

Sanghyun Yoon

Konkuk University

Abstract

This article describes ECML (ETRI CPS Modeling Language), a modeling langue for hybrid system simulation. ECML extends DEV&DESS (Discrete Event and Differential Equation System Specification), with conveniences in modeling and simulation. Modeling environment EcoPOD (ETRI CPS Open Developer) and visual simulator EcoSIM (ETRI CPS Simulator) support visual modeling and simulation. We introduce modeling and simulation environment of ECML, and also formal definition. The case study specifies a simple vehicle system with the formal definition.

1 Introduction

Hybrid system [1] is a dynamical system whose behavior is a combination of continuous and discrete dynamics. The discrete part usually models modes of system operations, while the continuous part does physical interactions with environments. Many approaches to modeling and analyzing hybrid system have been proposed. Timed automata [2], (linear) hybrid automata [3] [4], CHARON [5] and ECML are the examples of the modeling methods.

ECML (ETRI CPS Modeling Language) is an extension of the basic formalism DEV&DESS [6] with various conveniences in modeling and simulation [7] [8], recently proposed by ETRI (Electronics and Telecommunication Research Institute) in Korea. A modeling and simulation environment, EcoPOD (ETRI CPS Open Developer) and EcoSIM (ETRI CPS Simulator) supports ECML.

This paper introduces a formal definition for ECML and overviews modeling and simulation environment of ECML. The formal definition was proposed in [9] in Korean and various researches describes modeling and simulation environment. An ECML model is composed structural model and behavioral model. ECML model communicates with environment model. When conducting simulation, a specified ECML model is translated into *C++* programming codes, and environment model generates input scenario for ECML model and calculates outputs of ECML model. The translated model and scenario are executed and visualized.

This paper is organized as follows. Section 2 overviews related work about modeling and simulation environment of ECML. It also includes our previous work verifying ECML model with formal verification. Section 3 introduces ECML and formal definition of ECML. Section 4 shows an example with a *vehicle control system*. The system is specified with formal definition which introduced in Section 3. We conclude the paper in Section 5.

2 Related Work

An ECML is modeled in CPS (Cyber-Physical System) domain modeler, 'EcoPOD' [10] [11]. The model is loaded by hybrid simulation environment, 'EcoSIM.' EcoSIM translates the model into simulation program, and then executes and control the program based on scenario. The simulation result is visualized by 3D visualizer [12] as shown in <Figure 1>. Relation of model, simulation and physical system is described in [13].

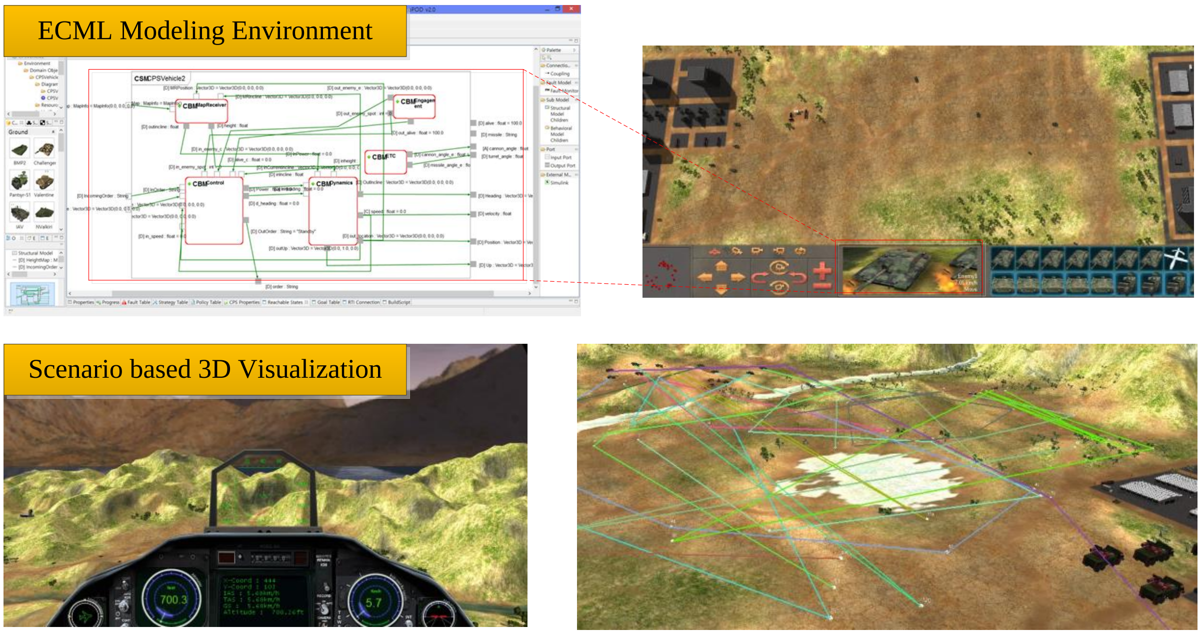
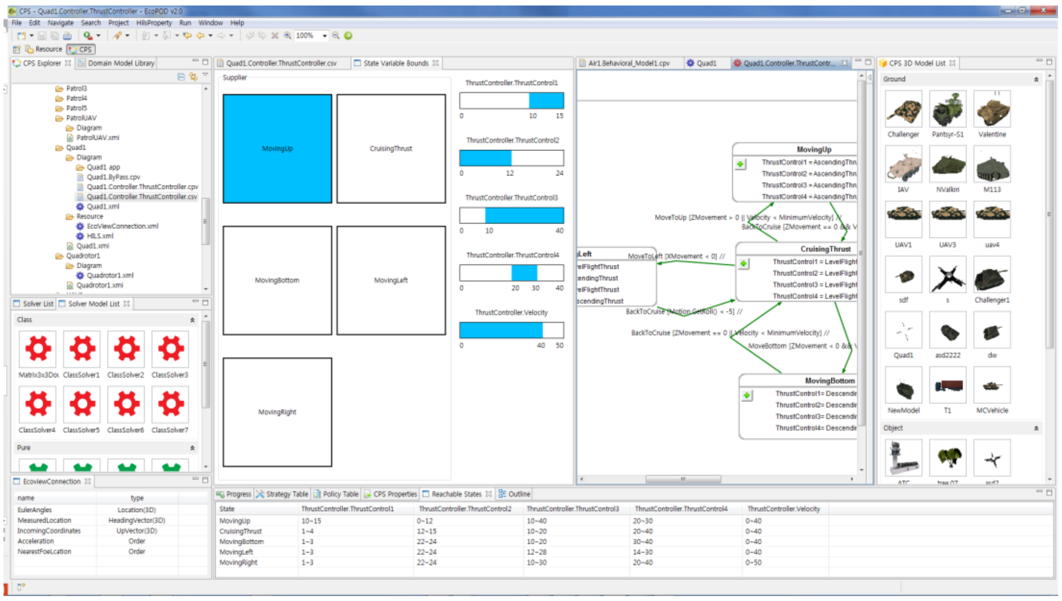


Figure 1 ECML modeling and simulation environment

Including the ECML modeling and simulation environment, a dependable CPS software development environment 'EcoSuite' was proposed. EcoSuite [14] is includes EcoPOD, EcoSIM, EcoView, EcoHILS and EcoAce. EcoView visualizes simulation result, and EcoHILS [15] supports HILS (Hardware in the Loop Simulation) using hardware EcoAce. They are connected with real time control middleware EDDS [16].



EcoviewConnection

name	type
EulerAngles	Location(3D)
MeasuredLocation	HeadingVector(3D)
IncomingCoordinates	UpVector(3D)
Acceleration	Order
NearestFoeLcation	Order

Reachable States

State	ThrustController.ThrustControl1	ThrustController.ThrustControl2	ThrustController.ThrustControl3	ThrustController.ThrustControl4	ThrustController.Velocity
MovingUp	10~15	0~12	10~40	20~30	0~40
CruisingThrust	1~4	12~15	10~20	20~40	0~40
MovingBottom	1~3	22~24	10~20	30~40	0~40
MovingLeft	1~3	22~24	12~28	14~30	0~40
MovingRight	1~3	22~24	10~30	20~40	0~50

Figure 2 An example of ECML formal verification result (in EcoPOD)

Our previous researches applied formal verification to verify ECML model. [17] and [18] verified DEV&DESS model which is basic formalism of ECML with HyTech [19]. [20] translated ECML into linear hybrid automata, and [21] and [22] verified translated linear hybrid automata with HyTech. We also performed a case study with SpaceEx [23] to verify ECML model [24], and then ECML model to SpaceEx model translator and formal verification interface are imported in EcoPOD [25] as shown in <Figure 2>.

There are some approaches which use ECML in various ways. For example, [26] translated Simulink model into ECML, and an approach implementing simulation models for SAM fuzzy controllers without the use of external components is represented in [27].

3 ECML

3.1 Variables

ECML models three different rate types of variables: discrete event, discrete value and continuous value, specifically. The three typed variables have different behavior of value each other. When an event is occur, discrete event type variable is assigned a value and reset to zero. Discrete value type variable's value is held until next assignment is occurred. The value of continuous value type variable is changing continuously, and the rate defines rate of change of the value at a phase.

Variables in ECML also have *port types* and various *data types*. ECML has three *port types* such as *input* (X), *output* (Y) and *state* (S^C,S^D). An ECML BM gets data from connected model with *input variable*, changes internal states with *state variable*, and sends data with *output variable*. *Data types* determine the possible values for the type variable. *Integer*, *Double*, *Boolean* and *String* are the examples and ECML supports also *user-defined data type*. Modeling environment 'EcoPOD' and visual simulator 'EcoSIM' support visual modeling and simulation [28], as illustrated in <Figure 3>.

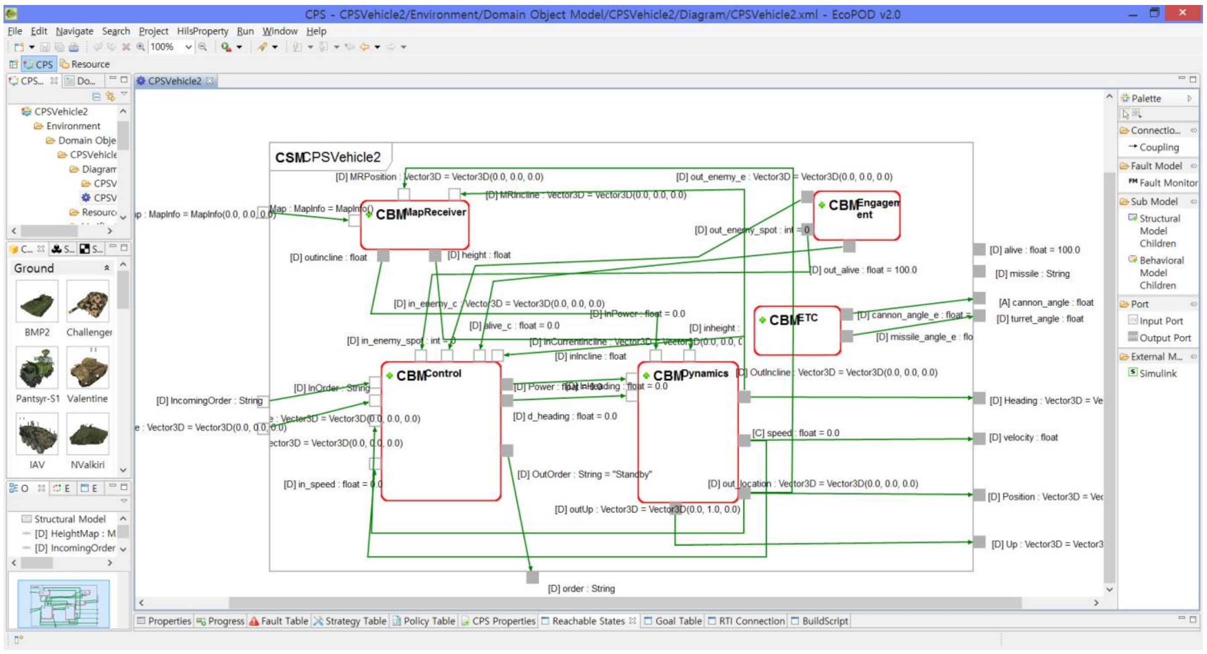


Figure 3 The modeling and simulation tool for ECML ('EcoPOD' and 'EcoSIM')

3.2 Structural Model & Behavioral Model

An ECML model is structured with a set of basic components, hierarchically. A basic component is modeled with BM (Behavioral Model), while their composition is modeled with SM (Structured Model). <Figure 3> shows an example model in 'EcoPOD.' CBM (CPS Behavioral Model) are BMs and CSM (CPS Structured Model) are SM. They communicate each other with connected input and output ports. The BM is defined as follows:

$$\text{BM} = \langle X, Y, S, Init, Cond^E, Trans^E, Trans^S, COnd^S, Rate, Out^C, Out^D, Out^E, Out^S \rangle$$

- $X = X^C \times X^D \times X^E$ is the set of inputs, where,
  - $X^C = \{(x_1^C, x_2^C, \ldots) | x_1^C \in X_1^C, x_2^C \in X_2^C, \ldots\}$ is the structured set of continuous value inputs with input variables x_i^C
  - $X^D = \{(x_1^D, x_2^D, \ldots) | x_1^D \in X_1^D, x_2^D \in X_2^D, \ldots\}$ is the structured set of discrete value inputs with input variables
  - X^E is the set of discrete event inputs
- $Y = Y^C \times Y^D \times Y^E$ is the set of outputs, where,
  - $Y^C = \{(y_1^C, y_2^C, \ldots) | y_1^C \in Y_1^C, y_2^C \in Y_2^C, \ldots\}$ is the structured set of continuous value outputs with output variables y_i^C
  - $Y^D = \{(y_1^D, y_2^D, \ldots) | y_1^D \in Y_1^D, y_2^D \in Y_2^D, \ldots\}$ is the structured set of discrete value outputs with output variables y_i^D
  - Y^E is the set of discrete event outputs
- $S = P \times S^D \times S^C$ is set of states; the Cartesian product of phases P, discrete states S^D and continuous states S^C
- $Init = S_0 \times X_0 \times Y_0$ is the initial condition set to define initial states and initial values of inputs and outputs
- $Cond^E : S \times X \rightarrow Bool$ is the external event transition condition function for conditioning the execution of the external events
- $Trans^E : S \times X \rightarrow S$ is the external event transition function
- $Out^E : S \times X \rightarrow Y$ is the output function for external event transitions
- $Cond^S : S \times X^C \times X^D \rightarrow Bool$ is the state transition condition function for conditioning the execution of the internal state events
- $Trans^S : S \times X^C \times X^D \rightarrow S$ is the internal state transition function
- $Out^S : S \times X^C \times X^D \rightarrow Y$ is the output function for internal state transitions
- $Rate : S \times X^c \times X^D \rightarrow S^C$ is the rate of change function
- $Out^C : S \times X^C \times X^D \rightarrow Y^C$ is the continuous value output function
- $Out^D : P \times S^D \times X^D \rightarrow Y^D$ is the discrete output function

A BM corresponds to an atomic DEV&DESS model. The semantics of an ECML BM are described as follows:

1. **Intervals $\langle t_1, t_2 \rangle$ with no events**: Only the continuous states S^C change. The continuous states at the end of the interval are computed form the state at the beginning plus the integral of the rate of change function $Rate(s(t), x^C(t), x^D(t))(\text{t} = \langle t_1, t_2 \rangle)$ along the interval. The continuous behavior of the model is specified by $Rate(s(t), x^C(t), x^D(t))$ and the continuous value output function $Out^C(s(t), x^C(t), x^D(t))$, while the discrete value output is generated by the discrete value output function $Out^D(p(t), s^D(t), x^D(t))$.
2. **An internal state event occurs first at time t in interval $\langle t_1, t_2 \rangle$**: The continuous states at the time of the transition are computed from the state at the beginning plus the integral of the rate of change function $Rate(s(t'), x^C(t'), x^D(t'))(\text{t}' = \langle t_1, t])$ along the interval until time t. Likewise, the hybrid output is generated until time t. At time t, the state transition condition function

$Cond^S(s(t), x^C(t), x^D(t))$ evaluates to true. That is, an internal state event occurs. Here, the internal state transition function $Trans^S(s(t), x^C(t), x^D(t))$ is executed to define a new state. The output function for internal state transitions $Out^S(s(t), x^C, x^D(t))$ is called to generated an output at time t.

3. **An external discrete event occurs first at time t in interval $\langle t_1, t_2 \rangle$:** The continuous states at the time of the transition are computed from the state at the beginning plus the integral of the rate of change function $Rate\big(s(t'), x^C(t'), x^D(t')\big)(t' = \langle t_1, t])$ along the interval until time t. Likewise, the continuous output is generated until time t. At time t, the external event transition condition function $Cond^E(s(t), x(t))$ evaluates to true. That is, the external event transition occurs. Here, the external event transition function $Trans^E(s(t), x(t))$ is executed to define a new state. The output function for external event transitions $Out^E(s(t), x(t))$ is called to generate an output at time t.

A SM of ECML corresponds to a coupled DEV&DESS model. A structured model contains basic models BMs, coupled each other with connecting ports describing flow of data typed of discrete, continuous and event. '*CSM CPSVehicle2*' in <Figure 3> is an example of ECML SM.

4 An Example: Vehicle Control System

This section models a '*vehicle control system*' with ECML. In actual vehicle control system, the vehicle moves along Euclidean shortest path in $\Re^2$ space. It receives current position, destination position, obstacle positon and the other information from environment. On the other hand, our case study approximates actual model, since the purpose is showing examples which are specified the introduced formal definition. There are no obstacles, and vehicle moves along east and north directions in case study model.

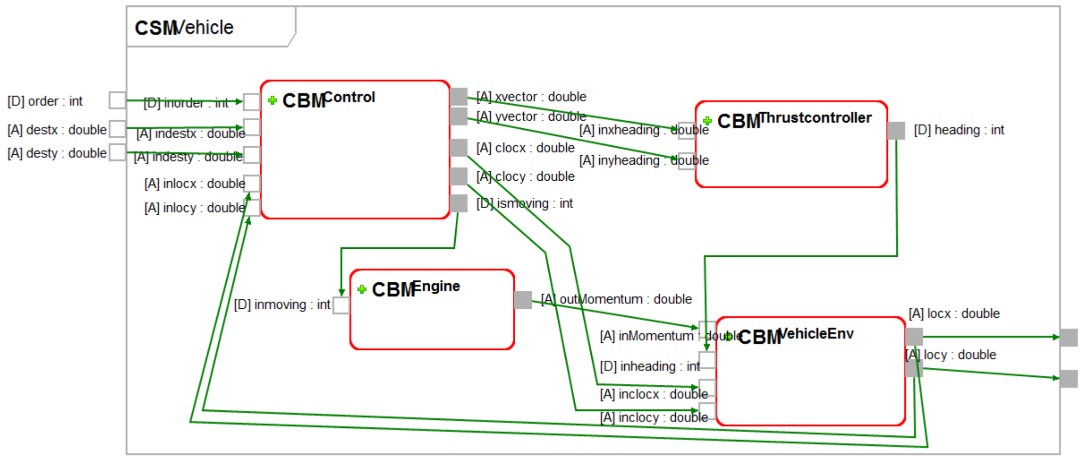


Figure 4 An ECML SM for simplified '*vehicle control system*'

<Figure 4> shows an SM *Vehicle* for 'vehicle control system.' Vehicle control system is structured with subsystems such as '*Control*,' '*Thruscontroller*,' '*Engine*' and '*VehicleEnv*.' It has three inputs and two outputs. *destx* and *desty* are inputs which set destination position of vehicle. When *order* is *true*, it changes current position of vehicle from initial position (0,0) to the destination position, and current position of vehicle is outputted through *outlocx* and *outlocy*.

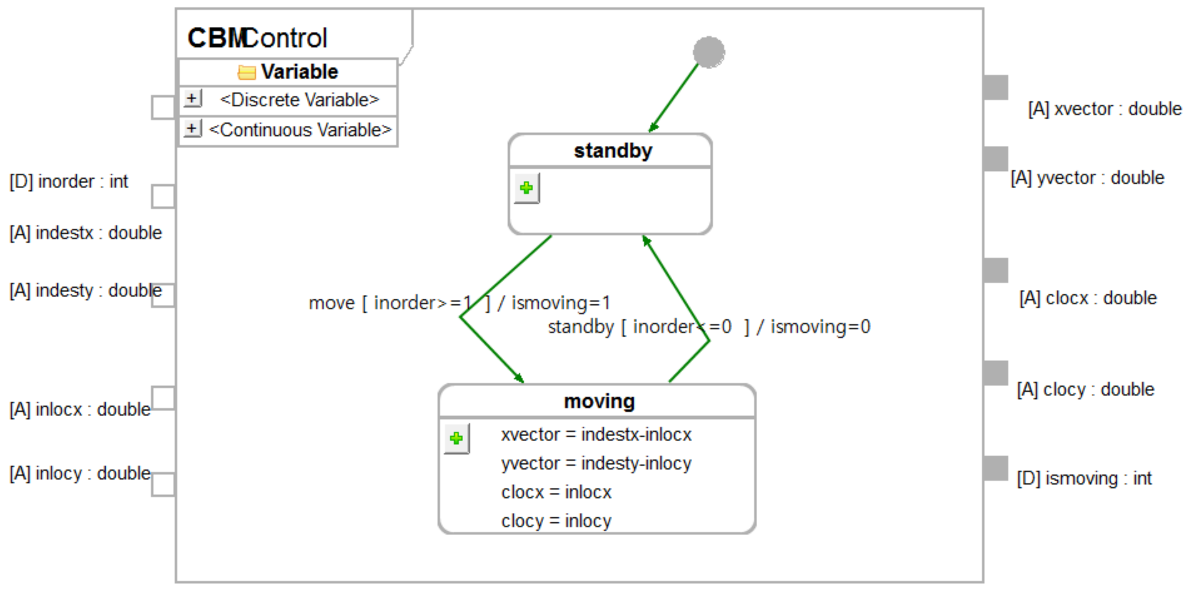


Figure 5 An ECML BM '*Control*'

BM *Control* is described in <Figure 5>. It receives order (*inorder*), destination position (*indestx*, *indesty*) and current position (*inlocx*, *inlocy*). Initial state of BM *Control* is *standby phase*. If '*inorder>=1*,' it transits to *moving phase*, and then it calculates distance from current position to destination position. The distance is outputted through *xvector* and *yvector* ports. Current position is passed to other subsystem through *clocx* and *clocy*, and order is done through *ismoving*. Formal definition of BM_{Con} for the subsystem *Control* is as follows:

$$BM_{Con} = \langle X, Y, S, Init, Trans^E, Cond^E, Out^E, Out^C \rangle$$

where,

- $X^D = \{inorder, indestx, indesty, inlocx, inlocy \mid inorder = order, indestx = destx, indesty = desty, inlocx = locx, inlocy = locy\}$
- $Y^C = \{xvector, yvector, clocx, clocy \mid \{xvector, yvector, clocx, cloccy\} \in \Re\}$
- $Y^D = \{\text{ismoving} \mid \text{ismoving} \in \{0,1\}\}$
- $P = \{standby, moving\}$
- $Init = \{P = \text{standby}\}$
- $Cond^E$:
  - $(P = standby, inorder \geq 1)$
  - $(P = moving, inorder \leq 0)$
- $Trans^E$:
  - $(P = standby, inorder \geq 1) \rightarrow (P = moving)$
  - $(P = moving, inorder \leq 0) \rightarrow (P = stanby)$
- Out^E :
  - $(P = standby, inorder \geq 1) \rightarrow (ismoving = 1)$
  - $(P = moving, inorder \leq 0) \rightarrow (ismoving = 0)$
- Out^C :
  - $(P = moving) \rightarrow$
    $(xvector = indestx - inlocx \wedge\ yvector = indesty - inlocy \wedge\ clocx = inlocx \wedge\ clocy = inlocy)$

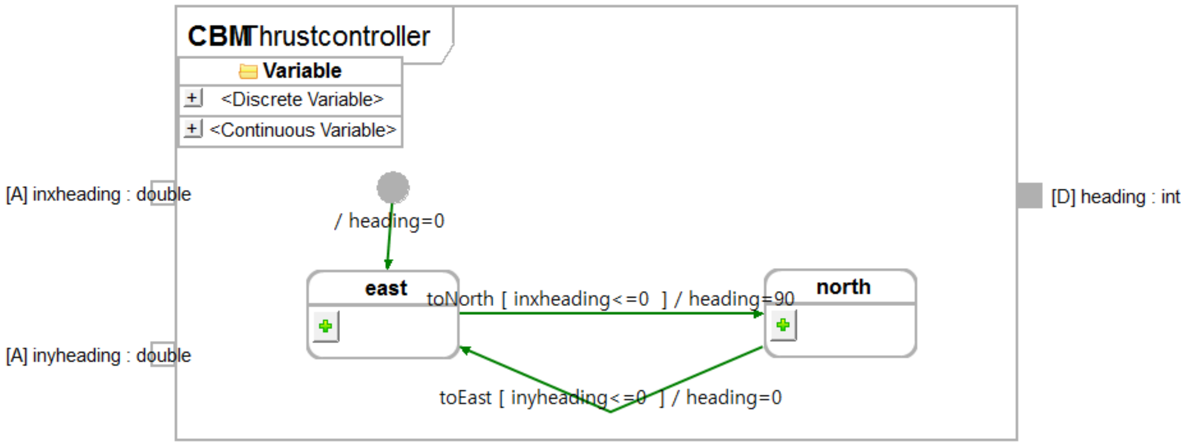


Figure 6 An ECML BM '*Thrustcontroller*'

Subsystem *Thrustcontroller* sets direction (*i.e.*, *heading*) of vehicle. <Figure 6> shows ECML BM_{TC} for a subsystem *Thrustcontroller*. *inxheading* and *inyheading* are variables which describe distance between current position and destination position. Vehicle of our case study has two heading, hence it has two *phases*. *Initial phase* is *east*, and it moves in the right direction (*i.e.*, *heading=0*). When the vehicle arrives to x-axis of destination (*i.e.*, *inxheading<=0*), it transit to *north* phase, and it moves in the up direction. Formal definition of the ECML BM_{TC} for the subsystem *Thrustcontroller* is as follows:

$$BM_{TC} = \langle X, Y, S, Init, Trans^E, Cond^E, Out^E \rangle$$

where,

- $X^C = \{inxheading, inyheading \mid inxheading = xvector, inyheading = yvector\}$
- $Y^C = \{heading \mid heading \in \{0, 90\}\}$
- $P = \{east, north\}$
- $Init = \{P = east,\ heading = 0\}$
- $Cond^E$:
  - $(P = east, \mathrm{inxheading} \leq 0)$
  - $(P = north, inyheading \leq 0)$
- $Trans^E$:
  - $(P = east, \mathrm{inxheading} \leq 0) \rightarrow (P = north)$
  - $(P = north, inyheading \leq 0) \rightarrow (P = east)$
- Out^E :
  - $(P = east, \mathrm{inxheading} \leq 0) \rightarrow (heading = 90)$
  - $(P = north, inyheading \leq 0) \rightarrow (heading = 0)$

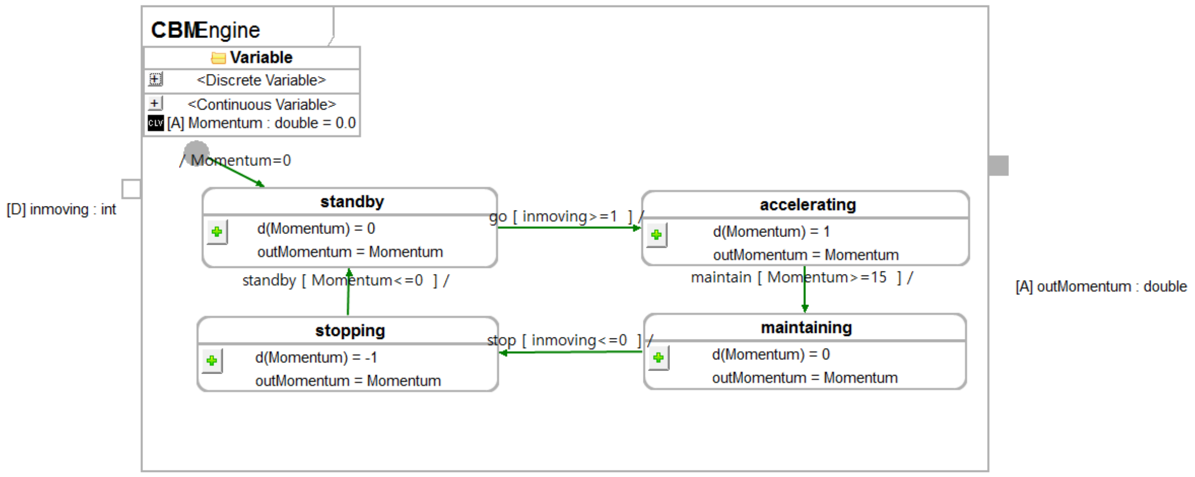


Figure 7 An ECML BM '*Engine*'

<Figure 7> shows ECML BM_{Eng} for a subsystem *Engine*. The model has an input and an output. *inmoving* is an order sent from BM_{Con}. If '$inmoving \geq 1$' in *phase standby*, it transits to *accelerating*. *Momentum* is a continuous state representing momentum of engine. In *accelerating phase*, the value of *Momentum* is increased by '*d(Momentum)=1*.' When *Momentum* reaches to own maximum value 15, it transits to *maintaining*, and maintains the *Momentum*. If '*inmoving<0*,' it transits to *stopping* and reduces *Momentum*. The value of *Momentum* is outputted through *outMomentum* port in every *phase* in BM_{Eng}. Formal definition of the ECML BM_{Eng} for the subsystem *Engine* is as follows:

$$BM_{Eng} = \langle X, Y, S, Init, Trans^E, Cond^E, Trans^S, Cond^S, Out^C, Rate \rangle$$

where,

- $X^D = \{inmoving \mid inmoving = ismoving\}$
- $Y^C = \{outMomentum \mid outMomentum \in \Re\}$
- $P = \{standby, accelerating, maintaining, stopping\}$
- $S^C = \{Momentum \in \Re\}$
- $Init = \{P = standby, Momentum = 0\}$
- $Cond^E$:
  - $(\mathrm{P = standby, inmoving} \geq 1)$
  - $(\mathrm{P = maintaining, inmoving} \leq 0)$
- $Trans^E$:
  - $(\mathrm{P = standby, inmoving} \geq 1) \rightarrow (P = accelerating)$
  - $(\mathrm{P = maintaining, inmoving} \leq 0) \rightarrow (P = stopping)$
- $Cond^S$:
  - $(P = accelerating, Momentum \geq 15)$
  - $(P = stopping, Momenmtum \leq 0)$
- $Trans^S$:
  - $(P = accelerating, Momentum \geq 15) \rightarrow (P = maintaining)$
  - $(P = stopping, Momenmtum \leq 0) \rightarrow (P = standby)$
- Out^C :
  - $(P = \{standby, accelerating, maintaining, stopping\}) \rightarrow (outMomentum = Momentum)$
- $Rate$:
  - $(P = \{standby, maintaining\}) \rightarrow (Mom\dot{e}ntum = 0)$
  - $(P = accelerating) \rightarrow (Mom\dot{e}ntum = 1)$
  - $(P = stopping) \rightarrow (Mom\dot{e}ntum = -1)$

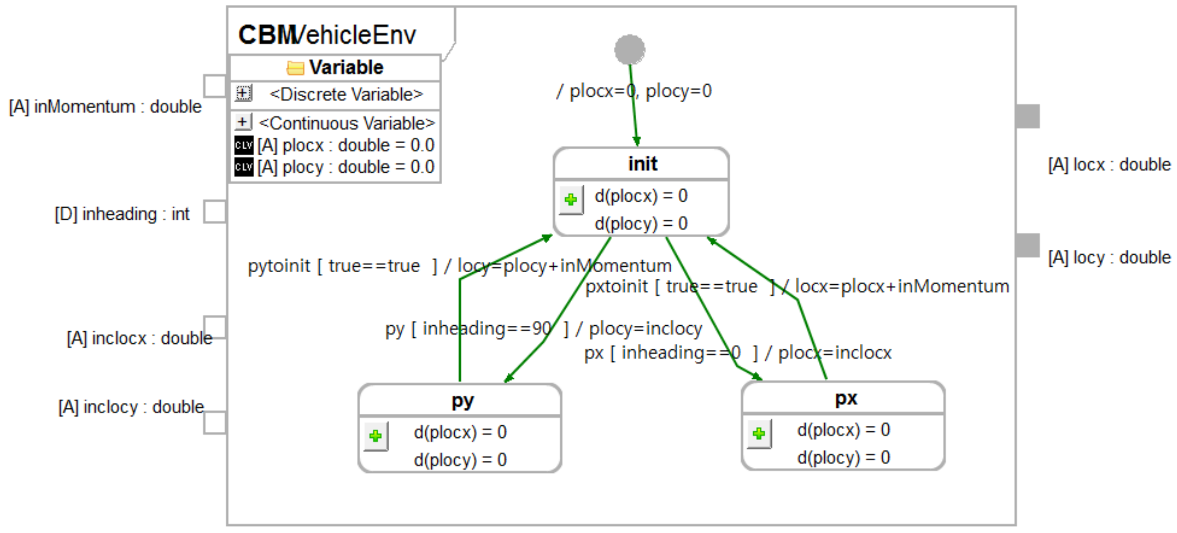


Figure 8 An ECML BM '*VehicleEnv*'

The subsystem *VehicleEnv* models change of vehicle's current position. <Figure 8> is ECML BM_{Env} for the subsystem. *Initial phase* is *init*, and it transits to *px* or *py phase* according to heading value which is sent from BM_{TC} (*i.e.,inheading*). '*inheading==90*' represents vehicle is moving in the right direction, so horizontal value of current position is increased by engine momentum. Otherwise, when '*inheading==0*,' vehicle is moving in up the direction, hence vertical value of current position is increased. Formal definition of the BM_{Env} for the subsystem *VehicleEnv* is as follows:

$$BM_{VE} = \langle X, Y, S, Init, Trans^E, Cond^E, Trans^S, Cond^S, Out^S, Rate \rangle$$

where,

- $X^C = \{inMomentum, inclocx, inclocy \mid inMomentum = outMomentum, inclocx = clocx, inclocy = clocy\}$
- $X^D = \{inheading \mid \text{inheading} = \text{heading}\}$
- $Y^C = \{locx, locy \mid \{locx, locy\} \in \Re\}$
- $P = \{init, px, py\}$
- $S^C = \{plocx, plocy \mid \{plocx, plocy\} \in \Re\}$
- $Init = \{P = init, plocx = 0, plocy = 0\}$
- $Cond^E$:
  - $(P = init, inheading = 0)$
  - $(P = init, inheading = 90)$
- $Trans^E$:
  - $(P = init, inheading = 0) \rightarrow (P = px, plocx = inclocx)$
  - $(P = init, inheading = 90) \rightarrow (P = py, plocy = inclocy)$
- $Cond^S$:
  - $(P = px)$
  - $(P = py)$
- $Trans^S$:
  - $(P = px) \rightarrow (P = init)$
  - $(P = py) \rightarrow (P = init)$
- Out^S :
  - $(P = px) \rightarrow (locx = plocx + inMomentum)$
  - $(P = py) \rightarrow (locy = plocy + inMomentum)$
- $Rate$:
  - $(P = \{init, px, py\}) \rightarrow (\dot{plocx} = 0 \wedge \dot{plocy} = 0)$

5 Conclusion & Future Work

Hybrid system is a dynamical system whose behavior is a combination of continuous and discrete dynamics. Many approaches to modeling and analyzing hybrid system have been proposed. ECML is an extension of basic formalism DEV&DESS, proposed by ETRI. This paper introduced ECML with formalism, modeling and simulation environment, EcoPOD and EcoSIM. The dependable CPS software development environment, EcoSuite supports HILS using ECML. The vehicle control system example shows that the introduced formalism can specify ECML model. Our previous work applied formal verification to verify ECML model. We will make a paper about formal verification with SpaceEx. The paper will describe translation rule from ECML model to SpaceEx model, supporting tool and case study.

References

[1] P. J. Antsaklis, J. A. Stiver and M. D. Lemmon, "Interface and controller design for hybrid control systems," *Hybrid Systems II, LNCS,* vol. 999, pp. 462-492, 1995.

[2] R. Alur and D. L. Dill, "A theory of timed autoata," *Theoretical computer science,* vol. 126, no. 2, pp. 183-235, 1994.

[3] R. Alur, C. Courcobetis, N. Halbwachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoretical Computer Science,* vol. 138, no. 1, pp. 3-34, 1995.

[4] R. Alur, T. A. Henzinger and P. H. Ho, "Automatic symbolic verificiation of embedded systems," *IEEE Transactions on Software Engineering,* vol. 22, no. 3, pp. 181-201, 1996.

[5] R. Alur, T. Dang, J. Esposito, Y. Hur, F. Ivančić, I. L. Vijay Kumar, P. Mishra, G. J. Pappas and O. Sokolsky, "Hierarchical modeling and analysis of embedded systems," *Proceedings of the IEEE,* vol. 91, no. 1, pp. 11-28, 2003.

[6] B. P. Zeigler, H. Praehofer and T. G. Kim, Theory of modeling and simulation, Academic Press, 2000.

[7] J. M. Kim, H. Y. Lee and I. G. Chun, "Hybrid modeling and simulation for verifying cyber-physical systems," *Proceedings of 2011 International Symposium on Embedded Technology (ISET 2011),* 2011.

[8] H. Y. Lee, I. Chun and W. -T. Kim, "DVML: DEVS-based visual modeling language for hybrid systems," *In Proceedings of Control and Automation, and Energy System Engineering - International Conference,* 2011.

[9] S. Yoon., I. -G. Chun, W. -T. Kim, J. Jo and J. Yoo, "An etri cps modeling language for specifying hybrid systems," *Journal of KIISE (in Korean),* vol. 42, no. 7, pp. 823-833, 2014.

[10] J. -H. Jeon, I. -G. Chun, W. -T. Kim and S. -M. Park, "Design and method in modeling of cyber-phsical systems," *In Proceedings of JCICT & The first Yellow Sea International Conference on Ubiquitous Computing (YES-ICUC) 2011*, 2011.

[11] J. Jeon, I. Chun and W. Kim, "Metamodel-based CPS modeling tool," *Embedded and Multimedia Computing Technology and Service (LNCS),* no. 181, pp. 285-291, 2012.

[12] E. -I. Kim, M. -J. Park, I. -G. Chun, W. -T. Kim and S. -M. Park, "Reliability support framework for cyber physical systems," *Proceedings of 2011 International Symposium on Embedded Technology (ISET 2011),* pp. 1-5, 2011.

[13] J. -Y. Kim, D. -N. Choi, H. -J. Kim, J. -M. Kim and W. -T. Kim, "Abstracted CPS model: a model for interworking between physical system and simulator for CPS simulation," *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium (TMS/DEVS 2012),* 2012.

[14] W. -T. Kim, I. -G. Chun, S. Lee and J. Park, "A large-scale autonomous CPS software platform (in Korean)," *Communications of the Korean Institute of Information Scientists and Engineers,* vol. 12, no. 31, pp. 16-28, 2013.

[15] M. -J. Kim, S. Kang, I. -G. Chun and W. -T. Kim, "A research on effective cyber-physical systems tests using EcoHILS (in Korean)," *IEMEK Journal of Embedded Systems and Applications,* vol. 4, no. 9, pp. 211-217, 2014.

[16] S. Lee, J. Kim, W. T. Kim and J. Ryou, "Communication entities discovery in complex CPS system," *Control and Automation, and Energy System Engineering,* no. 256, pp. 213-219, 2011.

[17] H. Choi, S. Cha, J. Jo, J. Yoo, H. Y. Lee and W. -T. Kim, "Formal verification of DEV&DESS formalism using symbolic model checker HyTech," *Control and Automation, and Energy System Engineering,* pp. 112-121, 2011.

[18] H. Choi, S. Cha, J. Jo, J. Yoo, H. Y. Lee and W. -T. Kim, "Formal verification of basic DEV&DESS formalism using Hytech," *Information-An international interdisciplinary journal,* vol. 16, no. 1 B, pp. 821-826, 2013.

[19] T. A. Henzinger, P. -H. Ho and H. Wong-Toi, "HyTech: a model checker for hybrid systems," *Software Tools for Technology Transfer,* vol. 1, no. 1-2, pp. 110-122, 1997.

[20] J. Jo, J. Yoo, H. Choi, S. Cha, H. Y. Lee and W. -T. Kim, "Translation from ECML to linear hybrid automata," *Embedded and Multimedia Computing Technology and Service,* pp. 293-300, 2012.

[21] J. Jo, "A systematic verification of ECML model using HyTech," *Master's thesis, Department of Computer & Information Communication Engineering, Konkuk University,* 2013.

[22] S. Yoon, J. Jo, I. -G. Chun and J. Yoo, "Verification and analysis of ECML model using HyTech (in Korean)," *In Proceeding of Korea Computer Software Engineering 2014 (KCSE 2014),* pp. 2-10, 2014.

[23] G. Frehse, L. G. Colas, D. Alexandre, C. Scott, R. Rajarshi, L. Olivier, R. Rodolfo, G. Antoine, D. Thao and M. Oded, "SpaceEx: Scalable verification of hybrid systems," *Computer Aided Verification,* pp. 379-395, 2011.

[24] J. Jo, S. Yoon, J. Yoo, H. Lee and W. Kim, "Case Study: Verification of ECML model using SpaceEx," *Proceedings of Korea-Japan Joint Workshop on ICT,* 2012.

[25] J. Jo, I. Chun and W. Kim, "EcoVerifier: Tool support for formal verification of ECML using SpaceEx," *Proceedings of 2014 International Symposium on Embedded Technology (ISET 2014),* pp. 104-105, 2014.

[26] H. S. Son, W. Y. Kim, R. Y. Kim and H. -G. Min, "Metamodel design for model transformation from Simulink to ECML in cyber physical systems," *Computer Applications for Graphics, Grid Computing, and Industrial Environment,* no. 351, pp. 56-60, 2012.

[27] H. Y. Lee, S. -M. Park and T. H. Cho, "Simulation modeling of SAM fuzzy logic controllers," *IEICE Transactions on Information and Systems,* vol. 7, no. E93-D, pp. 1984-1986, 2010.

[28] H. Y. Lee, I. Chun and W. -T. Kim, "DEV&DESS-based cyber-physical systems modeling language with uncrtainty consideration," *Proceedings of the 2013 Spring Simulation Multiconference Poster Session,* 2013.