Presentation TaxonomyOfDEVS2011

Moon Ho Hwang
Dassault Systemes America Corp.

Taxonomy of DEVS Subclasses for Standardization

Moon Ho Hwang

ACIMS
University of Arizona
Tucson, AZ 48326, USA

April 6, 2011

Background

- Schedule-Preserving DEVS (SPDEVS) is a subclass of Finite& Deterministic DEVS (FDDEVS). [6][5]
- FDDEVS is a subclass of Alur's Timed Automaton (TA) [4].
- Some papers have attempted to convert DEVS into TA for verification [2], [3], [1]
- Q1. Is this conversion DEVS into TA always possible?
- Q2. What are subclasses, super-classes or equivalent classes?
- Q3. Which classes are sub, super, or equivalent classes of DEVS?
- By answering these questions, this paper enables us to standardize DEVS classes.

Contents I

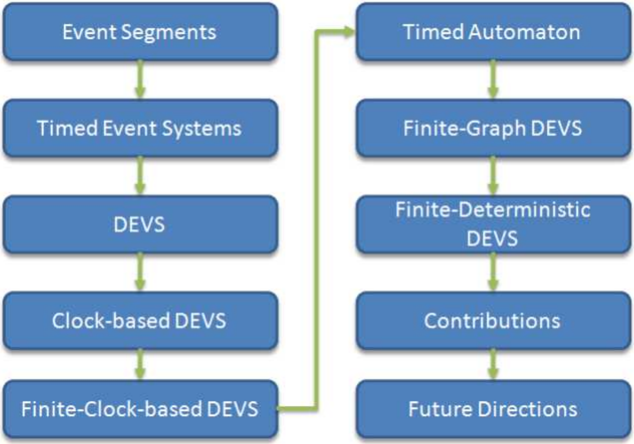


Figure 1: Presentation Organization

1.1 Example of Toaster Trajectories

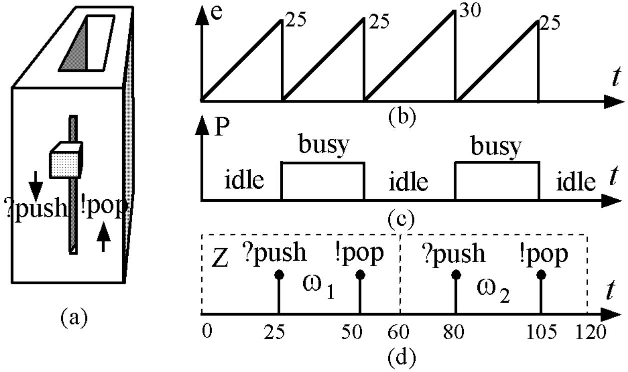


Figure 2: Trajectories of a Toaster (a) A Toaster, (b) Piecewise Linear Trajectory, (c) Piecewise Constant Trajectory, (d) Event Trajectory

1.2 Event Segments

- A *timed event*: (z, t) of $z \in Z$, $t \in \mathbb{T}$.
- The *null event segment*: $\epsilon_{[t_l, t_u]}$ where $\epsilon \notin Z$ and $[t_l, t_u] \subseteq \mathbb{T}$.
- An *unit event segment* is either a timed event or a null event segment.
- A *multi-event segment* $(z_1, t_1)(z_2, t_2) \dots (z_n, t_n)$ over Z and $[t_l, t_u] \subseteq \mathbb{T}$ is concatenations of unit event segments $\epsilon_{[t_l, t_1]}, (z_1, t_1), \epsilon_{[t_1, t_2]}, (z_2, t_2), \dots, (z_n, t_n)$ and $\epsilon_{[t_n, t_u]}$ where $t_l \leq t_1 \leq t_2 \dots \leq t_{n-1} \leq t_n \leq t_u$.
- Example: $\omega_{[0,120]}$=(?push,25)(!pop,50)(?push,80)(!pop,105).

1.3 Universal Timed Language

Definition 1 (Universal Timed Language)

The *universal timed language* over an event set Z and a time interval $[t_l, t_u] \subseteq \mathbb{T}$, is denoted by $\Omega_{Z,[t_l,t_u]}$, and is defined as the set of all possible event segments. Formally,

$$\Omega_{Z,[t_l,t_u]} = \{(z,t)^* : z \in Z \cup \{\epsilon\}, t \in [t_l, t_u]\}$$

where $(z,t)^*$ denotes a none or multiple concatenations of null or timed events.

- Note that if L is a language over Z and $[t_l, t_u]$, then $L \subseteq \Omega_{Z,[t_l,t_u]}$.

2.1 Timed Event Systems

Definition 2 (TES)

$$G = (Z, Q, q_0, Q_A, \Delta)$$

- Z is the set of events;
- Q is the set of states;
- $q_0 \in Q$ is the initial state variable;
- $Q_A \subseteq Q$ is the set of accept states;
- $\Delta : Q \times \Omega_{Z,\mathbb{T}} \to Q$ is the state trajectory function that defines how a state q changes to another q' along with an event segment $\omega \in \Omega_{Z,\mathbb{T}}$. □

If ω is concatenation of two event segments, i.e. $\omega = \omega_1\omega_2$, then $\Delta(q,\omega) = \Delta(\Delta(q,\omega_1),\omega_2)$. In general if ω is concatenation of n-event segments, i.e. $\omega = \omega_1\omega_2\ldots\omega_n$, where $n > 1$ then

$$\Delta(q,\omega) = \Delta(\ldots\Delta(\Delta(q,\omega_1),\omega_2)\ldots),\omega_n) \qquad (1)$$

2.2 Determinism and Nondeterminism of Timed Event Systems

Example 1 (Deterministic and Nondeterministic Functions)

For example, assume that A and B are real numbers, then $f(a) = a + 5$ is deterministic. Given two sets A ={coin, dice} and B={head, tail, 1,2,3,4,5,6}, if the function f indicates outcomes of tossing a coin or a dice, f is non-deterministic. If $r \in \{head, tail\}$ represents the outcome of tossing coin, r is a nondeterministic (or random) variable.

Definition 3 (Deterministic and Non-Deterministic TESs)

A TES $G = (Z, Q, q_0, Q_A, \Delta)$ is deterministic if (1) q_0 is a constant variable, and (2) Δ is deterministic. Otherwise, G is non-deterministic. □

2.3 $L(G)$: Behaviors of a TES G

Definition 4 (Non-infinite length language)

If $0 \leq t < \infty$, *t-length observation language of G*, $L(G, t)$, is

$$L(G, t) = \{\omega \in \Omega_{Z,[0,t]} : \exists \text{ the case } : \ \Delta(q_0, \omega) \in Q_A\}. \tag{2}$$

Definition 5 (Infinite length language)

The *infinite length observation language of G*, $L(G, \infty)$ is

$$L(G, \infty) = \{\omega \in \lim_{t \to \infty} \Omega_{Z,[0,t]} : \exists \text{ the case s.t. } \mathit{inf}(\Delta(q_0, \omega)) \subseteq Q_A\}. \tag{3}$$

where $\mathit{inf}(\Delta(q_0, \omega)) \subseteq Q$ denotes the states where ω visits infinitely many times or stays infinitely long.

We would use just $L(G)$ instead of $L(G, t)$ or $L(G, \infty)$ if t is not important.

2.4 $E(A)$: Expressiveness of a formalism A

Given a formalism A that is a subclass of TES, it's expressiveness is denoted by $E(A)$.

Definition 6 (Expressiveness Inclusion)

Suppose that A and B are two TES classes.

- $E(A) \subseteq E(B)$, if for a given instance a of A, $\exists$ an instance b of B: $L(a) = L(b)$.
- $E(A) \subset E(B)$, if $E(A) \subseteq E(B)$ but for a given instance b of B, $\nexists$ an instance a of A: $L(a) = L(b)$.
- $E(A) = E(B)$, if $E(A) \subseteq E(B)$ and $E(B) \subseteq E(A)$.

We use this expressiveness inclusion when showing
$E(TA) \subset E(DEVS)$, and
$E(FDEVS) \subset E(FGDEVS) \subseteq E(FCDEVS)$.

2.5 Hierarchy of Formalisms

The hierarchy of difference formalism can be defined based on their expressiveness.

Definition 7 (Subclass, Equivalent class, and Superclass)

Suppose that A and B are two TES classes. Then

- A is called a *subclass* of B and B is called a *superclass* of A if $E(A) \subset E(B)$.
- A is called a *subclass or equivalent class* of B and B is called a *superclass or equivalent class* of A if $E(A) \subseteq E(B)$.
- A and B are called the *equivalent classes* if $E(A) = E(B)$. $\square$

2.6 Homomorphic Timed Event Systems

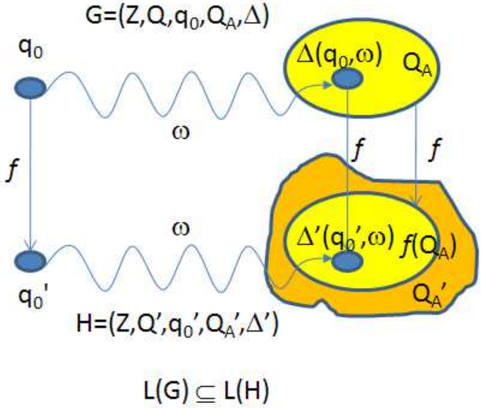


Figure 3: H is called a homomorphic system of G if such a mapping f exists. If H is a homomorphic system of G, $L(G) \subseteq L(H)$. We use this property when showing E(DEVS)=E(CDEVS).

3.1 Discrete Event System Specification(DEVS)

Definition 8 (DEVS)

$$M = (X, Y, S, s_0, ta, \delta_{ext}, \delta_{int}, \lambda)$$

- X and Y are the *set of input events* and the *set of output events*, respectively;
- S is the *set of states*; $s_0 \in S$ is the *initial state variable*;
- $ta : S \to \mathbb{T}^\infty$ is the *time advance function*;
- $\delta_{ext} : Q \times X \to S$ is the *external transition function* where $Q = \{(s, e) \in Q, e \in (\mathbb{T} \cap [0, ta(s)])\}$ is the *set of total states*, and e is the *piecewise linear elapsed time* since last event;
- $\delta_{int} : S \to S$ is the *internal transition function*;
- $\lambda : S \to Y^\phi$ is the *output function* where $Y^\phi = Y \cup \{\phi\}$ and $\phi \notin Y$ is a *silent event* or an *unobservable event*. □

3.2 Behaviors of the DEVS class

Let $M = (X, Y, S, s_0, ta, \delta_{ext}, \delta_{int}, \lambda)$ be a DEVS model. Then the behavior of M is explained by a TES $G(M) = (Z, Q, q_0, Q_A, \Delta)$ where the event set $Z = X \cup Y^{\phi}$; The state set $Q = Q_A \cup Q_{\bar{A}}$ where $Q_A = M.Q$ and $Q_{\bar{A}} = \{\bar{s} \notin S\}$ is called the non-accept state in which $\bar{s}$ is piecewise constant.
The initial state variable $q_0 = (s_0, 0) \in Q_A$.
The state trajectory function $\Delta : Q \times \Omega_{Z,\mathbb{T}} \rightarrow Q$ is defined for a total state $q = (s, e) \in Q$ at time $t \in \mathbb{T}$ and an event segment $\omega \in \Omega_{Z,[t,t+dt]}$, $dt \in \mathbb{T}$ as follows.
For a null event segment, i.e. $\omega = \epsilon_{[t,t+dt]}$,

$$\Delta(q, \omega) = q \oplus dt = \begin{cases} (s \oplus dt, e + dt) & \text{if } q \in Q_A \\ \bar{s} & \text{otherwise} \end{cases} \tag{4}$$

which is a *timed passage*.

For a timed input event, i.e. $\omega = (x, t)$ where $x \in X$

$$\Delta(q, \omega) = \begin{cases} (\delta_{ext}(s, e, x), 0) & \text{if } q \in Q_A, \\ \overline{s} & \text{otherwise.} \end{cases} \quad (5)$$

For a timed output or silent event, i.e. $\omega = (y, t)$ where $y \in Y^{\phi}$

$$\Delta(q, \omega) = \begin{cases} (\delta_{int}(s), 0) & \text{if } q \in Q_A, \underline{e = ta(s)}, \underline{y = \lambda(s)} \\ \overline{s} & \text{otherwise.} \end{cases} \quad (6)$$

If ω is a multi-event segment, we can apply Equation (1) using above three primitive cases described in Equations (4), (5), and (6).

3.3 Definition of Clock-based DEVS Structure

Definition 9 (CDEVS)

$$M_C = (X, Y, S, s_0, \delta_x, \delta_y)$$

- X and Y are the input and output events sets, respectively.
- $S = S_d \times \prod_{c \in C} (\mathbb{T}^\infty \times \mathbb{T})_c$ is the set of states that consists of two disjoint sets
  - S_d is the set of *piecewise constant states* which is called the set of *discrete* states.
  - C is the set of *clock names*. Each clock $c \in C$ has two clock variables
    - $\sigma_c \in \mathbb{T}^\infty$: the *schedule* of clock $c \in C$, which *is piecewise constant*.
    - $e_c \in \mathbb{T} \cap [0, \sigma_c]$: the *elapsed time* of clock $c \in C$, which *is piecewise linear*.

  Thus $s = (s_d, \ldots, \sigma_c, e_c, \ldots)$ denotes at phase $s_d \in S_d$, each clock c's schedule σ_c and the elapsed time e_c.

3.3 Definition of Clock-based DEVS Structure

Definition 10 (CDEVS (continued))

$$M_C = (X, Y, S, s_0, \delta_x, \delta_y)$$

- $s_0 = (s_{d0}, \ldots, \sigma_{0c}, 0, \ldots) \in S$ is the *initial state variable*
- $\delta_x : S \times X \rightarrow S$ is the *external transition function.*
- $\delta_y : S \rightarrow Y^\phi \times S$ is the *output and internal transition function*; □

Let *the remaining time function* $tr : S \rightarrow \mathbb{T}^\infty$ be

$$tr(s_d, \ldots, \sigma_c, e_c, \ldots) = \min_{c \in C} \{\sigma_c - e_c\} \tag{7}$$

for $(s_d, \ldots, \sigma_c, e_c, \ldots) \in S$.

3.4 A Example of CDEVS Toaster

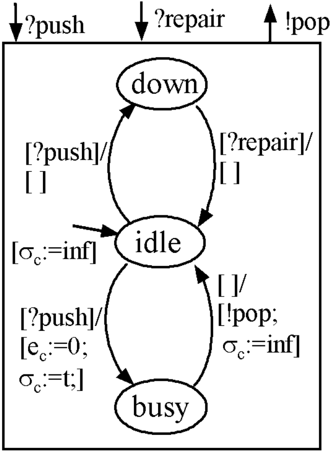


Figure 4: A Toaster CDEVS Model where $t \in [20, 30]$

3.5 Behaviors of the CDEVS class

Given a CDEVS $M_C = (X, Y, S, s_0, \delta_x, \delta_y)$, there exists a TES $G(M_C) = (Z, Q, Q_A, q_0, \Delta)$ defining the behavior of M_C as follows. The set of events is $Z = X \cup Y$. The set of states is $Q = Q_A \cup Q_{\bar{A}}$ where $Q_A = \{(s, t_s, t_e) : s \in S, t_s \in \mathbb{T}^{\infty}, t_e \in \mathbb{T} \cap [0, t_s]\}$ and $Q_{\bar{A}} = \{\bar{s} \notin S\}$ in which t_s and $\bar{s}$ is piecewise constant, and t_e is piecewise linear.
The initial state variable is given

$$q_0 = (s_0, t_{s0}, t_{e0}) = ((s_{d0}, \ldots, \sigma_{0c}, 0, \ldots), tr(s_0), 0).$$

The state trajectory function $\Delta : Q \times \Omega_{Z,\mathbb{T}} \rightarrow Q$ is given for $q \in Q$ and an unit segment ω as below.
For a null segment $\omega = \epsilon_{[t,t+dt]}$ and $t, dt \in \mathbb{T}$,

$$\Delta(q, \omega) = \begin{cases} ((s_d, \ldots, \sigma_c, e_c + dt, \ldots), t_s, t_e + dt) & \text{if } q \in Q_A \\ \bar{s} & \text{otherwise} \end{cases} \quad (8)$$

3.5 Behaviors of the CDEVS class

For a timed input event $\omega = (x, t)$, $x \in X$, and $t \in \mathbb{T}$,

$$\Delta(q, \omega) = \begin{cases} (\delta_x(s, x), tr(\delta_x(s, x)), 0) & \text{if } q \in Q_A \\ \bar{s} & \text{otherwise.} \end{cases} \tag{9}$$

For a timed output event $\omega = (y, t)$, $y \in Y^{\phi}$, and $t \in \mathbb{T}$,

$$\Delta(q, \omega) = \begin{cases} (s', tr(s'), 0) & \text{if } \underline{t_e = t_s}, \underline{\delta_y(s) = (y, s')} \\ \bar{s} & \text{otherwise.} \end{cases} \tag{10}$$

3.6 E(DEVS)=E(CDEVS)

Theorem 1 (E(DEVS)=E(CDEVS))

DEVS and CDEVS are equivalent classes to each other.

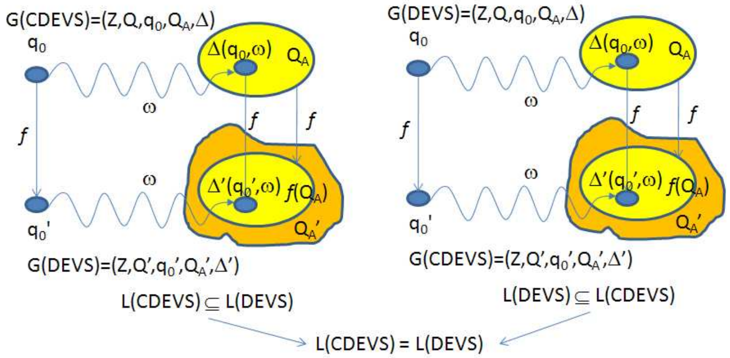


Figure 5: Proof of E(DEVS)=E(CDEVS) is available at https://sites.google.com/site/moonhohwang/publications

4.1 Finite Clock-based DEVS(FCDEVS)

Definition 11 (FCDEVS)

A Finite CDEVS (FCDEVS) is a subclass of CDEVS $M_{FC} = (X, Y, S, s_0, \delta_x, \delta_y)$ where the sets of X, Y, S_d, and C are finite. Note that $S = S_d \times \prod_{c \in C} (\mathbb{T}^{\infty} \times \mathbb{T})_c$ □

4.2 Timed Automaton(TA)

Definition 12 (Timed Automaton(TA))

$$TA = (Z, C, P, p_0, I, T)$$

- Z and C are the *finite sets of events* and the *finite set of clocks*, respectively.
- P and $p_0 \in P$ are the *finite set of phases* which are piecewise constant, and the *initial phase variable*, respectively.
- $I : P \rightarrow \Phi(C)$ is the *phase clock-constraint function* where $\Phi(C) = \{C \rightarrow \mathbb{I}_Q\}$ is the *set of partial clock constraints*.
- $T \subseteq P \times Z^{\phi} \times \Phi(C) \times \mathcal{P}(C) \times P$ is *a set of transitions*. A transition $(p, z, \varphi, C_R, p') \in T$ can be also inter-changeably represented by the notation $p \xrightarrow{z,\{(c,inv(c))\},C_R} p'$, requires the enabling condition of $I(p)$ and φ as a *precondition*, and the resetting clocks in C_R as a *postcondition*. □

4.3 A Example of TA Toaster

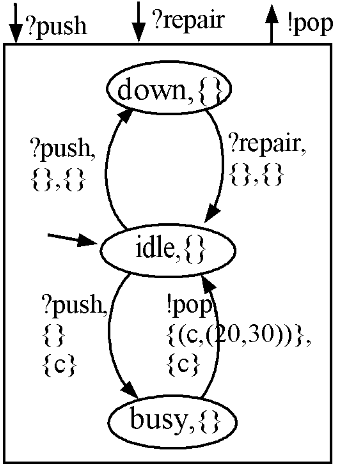


Figure 6: A Toaster TA Model

4.4 Behaviors of the TA class

Given a TA $A = (Z, C, P, p_0, I, T)$, there exists a corresponding FCDEVS $B = (X, Y, S, s_0, \delta)$ that defines the behavior of A. We consider all events in Z of A as output events of B so $X = \varnothing$ and $Y = Z$. The state set $S = P \times \prod_{c \in C} (\mathbb{T}^\infty \times \mathbb{T})_c$.

The initial state variable $s_0 = (p_0, \ldots, su(p_0, c), 0, \ldots)$ where $su : P \times C \times \mathbb{T} \to \mathbb{T}^\infty$ is called *the clock-schedule update function* that is given for a phase $p \in P$ and a clock $c \in C$

$$su(p, c) = \min\{t_S((M(I(p)) \cap M(\varphi))|_c \cap [e_c, \infty)) \\ : (p, z, \varphi, C_R, p') \in T\} \quad (11)$$

where M is defined in Equation (**??**) and $t_S : \mathcal{P}(\mathbb{T}^\infty) \to \mathbb{T}^\infty$ is the *sampling function* that is given for a set of time values $\mathbf{t} \subseteq \mathbb{T}^\infty$ which can be an time interval,

$$t_S(\mathbf{t}) = \begin{cases} \infty & \text{if } \mathbf{t} = \varnothing \\ t & \text{otherwise } t \in \mathbf{t}. \end{cases} \quad (12)$$

4.4 Behaviors of the TA class (continued)

The output and internal transition function $\delta_y : S \rightarrow Y^{\phi} \times S$ is given for $s = (p, \ldots, \sigma_c, e_c, \ldots)$, $y \in Y^{\phi}$: If $\exists(p, y, \varphi, C_R, p') \in T$, then

$$\delta_y(s) = (p', \ldots, \sigma'_c, e'_c, \ldots)$$

where $e'_c = t_R(c, C_R)$ where $t_R : C \times \mathcal{P}(C) \rightarrow \mathbb{T}$ is called the *resting function* that is defined for $c \in C$ and $C_R \subseteq C$,

$$t_R(c, C_R) = \begin{cases} 0 & \text{if } c \in C_R \\ e_c & \text{otherwise.} \end{cases} \tag{13}$$

and $\sigma'_c = su(p', c)$.
If $\nexists(p, y, \varphi, C_R, p') \in T$, then nothing changes because there is no such a transition from p, thus $\delta_y(s) = (p, \ldots, \sigma_c, e_c, \ldots)$. □

Proposition 1 ($E(TA) \subset E(FCDEVS)$)

$E(FCDEVS) \not\subseteq E(TA)$ because *TA* does not allow clock boundaries of real numbers which are allowed by FCDEVS.

4.5 Finite-Graph DEVS

Definition 13 (FGDEVS)

$$M_{FG} = (X, Y, S, s_0, \delta)$$

- X, Y and S are the same as those of CDEVS but they are finite sets; and $s_0 \in S$ is the initial state.
- $\delta \subseteq S_d \times Z^\phi \times \Psi(C) \times \mathcal{P}(C) \times S_d$ is the *finite set of transition relations* where $Z = X \cup Y^\phi$. A transition $(s_d, z, \psi, C_R, s_d')$ or its graphical notation $s \xrightarrow{z,\psi,C_R} s'$ denotes that the discrete state changes s_d to s_d' associated with an event z, together with *two post-conditions*: updating the schedule $\sigma_c = \psi(c)$ if $\psi(c)$ is defined for a clock $c \in C$, and resetting the elapsed time e_c of each clock $c \in C_R$. □

4.6 A Example of FGDEVS Toaster

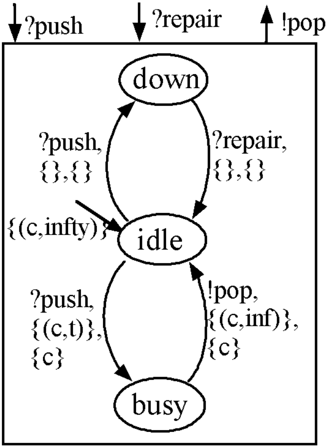


Figure 7: A Toaster FGDEVS Model where $t \in [20, 30]$

4.7 Behavior of FGDEVS

The behaviors of an FGDEVS $M_{FG} = (X, Y, S, s_0, \delta)$ model are given through an FCDEVS $M_{FC} = (X, Y, S, s_0, \delta_x, \delta_y)$ as follows.
The initial state variable is $s_0 = (s_{d0}, \ldots, \sigma_{0c}, 0, \ldots)$.
The external transition function $\delta_x : S \times X \rightarrow S$ is given for $s = (s_d, \ldots, \sigma_c, e_c, \ldots) \in S$ and $x \in X$, if $\exists s_d \xrightarrow{x,\psi,C_R} s_d{}' \in \delta$, then

$$\delta_x(s, x) = (s'_d, \ldots, \sigma'_c, e'_c, \ldots)$$

where

$$\sigma'_c = \begin{cases} t_S(\psi(c)) & \text{if } \psi(c) \text{ is defined} \\ \sigma_c & \text{otherwise,} \end{cases}$$

and t_S is the sampling function defined in Equation (12), and

$$e'_c = t_R(c, C_R)$$

where $t_R(c, C_R)$ is the resetting function defined Equation (13). If $\nexists s_d \xrightarrow{x,\psi,C_R} s_d{}' \in \delta$, nothing changes by x, thus $\delta_x(s, x) = (s_d, \ldots, \sigma_c, e_c, \ldots)$.

4.7 Behavior of FGDEVS (continued)

The output and internal transition function $\delta_y : S \rightarrow Y^\phi \times S$ is given for $s = (s_d, \ldots, \sigma_c, e_c, \ldots) \in S$ and $y \in Y^\phi$, if $\exists s_d \xrightarrow{y,\psi,C_R} s_d{}' \in \delta$, then

$$\delta_y(s) = (y, (s_d', \ldots, \sigma_c', e_c', \ldots))$$

where $\sigma_c' = \psi(c)$ if $\psi(c)$ is defined, otherwise,$\sigma_c' = \sigma_c$; and $e_c' = t_R(c, C_R)$. If $\nexists s_d \xrightarrow{y,\psi,C_R} s_d{}' \in \delta$, nothing changes by an internal transition from s so $\delta_y(s) = (\phi, (s_d, \ldots, \sigma_c, e_c, \ldots))$. □

Proposition 2 ($E(FGDEVS) \subseteq E(FCDEVS)$)

It is given by the definition. We still don't know if $E(FCDEVS) \subseteq E(FGDEVS)$ so $E(FGDEVS) = E(FCDEVS)$.

4.8 Finite & Deterministic DEVS (FDDEVS)

Definition 14 (FDDEVS)

$$M_{FD} = (X, Y, S, s_0, \tau, \delta_x, \delta_y)$$

- X and Y are the same as those of FCDEVS.
- S is the *finite discrete states* which are piecewise constant.
- $s_0 \in S$ is *the constant initial state*.
- $\tau : S \rightarrow \mathbb{Q}_{[0,\infty)}$ is *the time schedule function* where $\mathbb{Q}_{[0,\infty)}$ is the none negative rational numbers plus infinity.
- $\delta_x : S \times X \rightarrow S \times \{0, 1\}$ is *the external transition function*.
- $\delta_y : S \rightarrow Y^{\phi} \times S$ is *the output and internal transition function*.

As the name explains, τ, δ_x and δ_y of FDDEVS are deterministic.

□

4.9 Behavior of FDDEVS

Given an FDDEVS model $M_{FD} = (X, Y, S, s_0, \tau, \delta_x, \delta_y)$, there is a corresponding FGDEVS $M_{FG} = (X, Y, S_G, s_{0G}, \delta)$ can describe the behavior of the original model M_{FD} as follows. The events sets of M_{FG} are the same those of M_{FD}. The state set of $S_G = \{(s, \sigma_c, e_c) : s \in S, c \in C\}$ where C={'c'}. The initial state $s_{0G} = (s_0, \tau(s_0), 0)$. The state transition relation δ of M_{FG} is defined corresponding to each state transition.

$$\delta_x(s, x) = (s', 0) \text{ implies } s \xrightarrow{x, \varnothing, \varnothing} s' \in \delta,$$

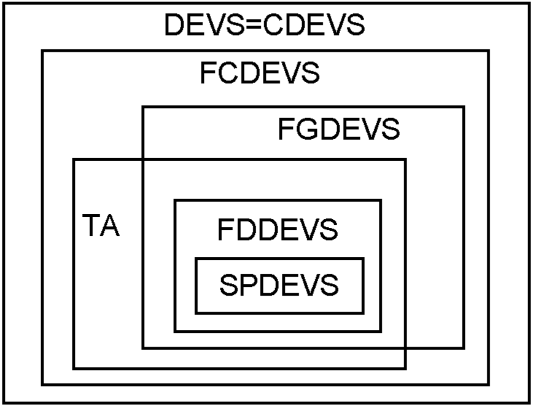
$$\delta_x(s, x) = (s', 1) \text{ implies } s \xrightarrow{x, \{(c, \tau(s'))\}, \{c\}} s' \in \delta,$$

$$\delta_y(s) = (y, s') \text{ implies } s \xrightarrow{y, \{(c, \tau(s'))\}, \{c\}} s' \in \delta.$$

□

5.1 Contributions

- Provided a formal framework that clarifies expressiveness of different formalisms.
- Expressive inclusion among DEVS equivalent and subclasses:



5.2 Future Directions

- The question whether $E(FCDEVS) \subseteq (FGDEVS)$ or not is still an open problem.
- In addition to TA, expressiveness comparison among other popular formalisms like Colored (timed) Petri-Nets, UML Start-Charts are possible in the same way of timed language approaches.
- **Similarity (or Distance) of Two models**: Given two DEVS instances M_1 and M_2, the distance of M_1 and M_2 can be done by their (1) event segments, or (2) states. Then we will have a metric space of discrete event systems using DEVS. That may be answer of simulation model validity for closeness or similarity of two given systems (one can be a target system, the other its simulation model).

References I

L. Nigro F. Pupo F.Cicirelli, A. Furfaro.
Temporal verification of RT-DEVS models with implementation aspects.
In *Proceeding SpringSim '10 Proceedings of the 2010 Spring Simulation Multiconference*, 2010.

A. Furfaro and L. Nigro.
Embedded Control Systems Design based on RT-DEVS and temporal analysis using UPPAA.
In *Proceedings of the International Multiconference on Computer Science and Information Technology*, pages 601–608, 2008.

H. Saadawi and G. Wainer.
Rational Time-Advance DEVS (RTA-DEVS).
In *Proceedings of 2010 Spring Simulation Multi-Conference: Proceedings of 2010 DEVS Symposium*, 2010.

References II

M. H. Hwang and B.P. Zeigler.
Expressiveness of Verifiable Hierarchical Clock Systems.
International Journal of General Systems, 37(4):391–413, 2008.

M.H. Hwang and B.P. Zeigler.
Reachability Graph of Finite & Deterministic DEVS Networks.
IEEE Trans. on Automation Science and Engineering, 6(3):454–476, 2009.

Moon H. Hwang and B. P. Zeigler.
A Reachable Graph of Finite and Deterministic DEVS Networks.
In *Proceedings of 2006 Spring Simulation Multi-Conference: Proceedings of 2006 DEVS Symposium*, pages 48–56, Huntsville, AL, Apr. 2-8 2006. SCS.