A Formal Framework for Stochastic DEVS Modeling and Simulation

Rodrigo Castro, Ernesto Kofman
Universidad Nacional de Rosario
rodrigocastro@ieee.org; kofman@fceia.unr.edu.ar

Gabriel Wainer
Carleton University
gwainer@sce.carleton.ca

Keywords: Discrete event simulation, DEVS, Stochastic systems.

Abstract

We introduce an extension of the classic Discrete Event System Specification (DEVS) formalism that includes stochastic features. Based on the use of Probability Spaces, the STochastic DEVS specification (STDEVS) provides a formal framework for modeling and simulation of general non deterministic discrete event systems. The main theoretical properties of STDEVS are shown. We illustrate its use in a stochastic-oriented simulation example with the main purpose of performance analysis in computer systems and data networks.

1. INTRODUCTION

The DEVS formalism was developed by Bernard Zeigler in the mid–seventies [15, 16]. Being a general system theoretic based formalism, DEVS can represent all the systems whose input/output behavior can be described by sequences of events. Thus, discrete event systems modeled by Finite State Automatas, Petri Nets, Grafcets, Statecharts, etc., can be also represented by DEVS models [17]. Moreover, discrete time systems can be also represented by DEVS [16]. The generality of DEVS converted it into a widely used language to describe and to simulate most classes of discrete systems. Moreover, numerical integration methods that approximate continuous systems (differential equations) by DEVS models have been developped [4] and several aplications and extensions of the DEVS formalism for modeling and simulation of continuous and hybrid systems have been proposed [6, 13]. Consequently, many DEVS–based modeling and simulation software tools have been developed in recent years [18, 14, 5, 11].

Nevertheless, a drawback of DEVS is that it is only formally defined for deterministic systems which limits the stochastic treatment of the systems under study. Although the relationship between DEVS and stochastic systems was studied in some early works [1, 12], and an extension for stochastic DEVS with finite states was already proposed [8], there is not a general theory nor a formal theoretic support for modeling general stochastic DEVS models. Stochastic models play a fundamental role in discrete event system theory. In fact, any system involving uncertainties, unpredictable human actions or system failures requires a non–deterministic treatment; and computer systems and data networks match all these properties. Examples of stochastic discrete event formalisms are Markov Chains, Queuing Networks [3] and Stochastic Petri Nets [2]. These tools permit analyizing and simulating stochastic models in several applications.

The first attempt to define a general DEVS–based formalism for stochastic systems was reported by two of the authors in [10]. In that work, a formalism called STDEVS that made use of probability spaces was proposed, and it was shown that the classic DEVS formalism is a particular case of STDEVS. A weakness of the original definition of STDEVS was that the different transitions did not define *independent* probability spaces as they shared their sigma–algebra. Thus, that definition of STDEVS could not capture the behavior of classic DEVS models equipped with random generators at the transition functions, which is the usual –but informal– practical way to incorporate stochastic behavior in DEVS. Also, in the aforementioned work, the crucial property of closure under coupling was conjectured but not proven (this property allows the usage of hierarchical model coupling).

In this new work, we continue with the preliminary work of [10] redefining the first idea of STDEVS proposed there in order to solve the mentioned difficulties. Using a different probability space for each transition, we prove that classic DEVS models that use random functions define STDEVS equivalent models (a corollary of this proof is that DEVS is a particular case of STDEVS). Also, we show that the property of closure under coupling holds in STDEVS. This property, combined with the previous one, ensures the correctness of hierarchically coupling classic DEVS and STDEVS models in an arbitrary way. In other words, in this paper we develop a complete theory of general stochastic DEVS.

The work is organized as follows. After recalling the principles of DEVS and Probability Spaces, Section 2 redefines the STDEVS formalism. Then, Section 3 shows that any DEVS model where the transition functions depend on random variables defines an equivalent STDEVS model. This property permits modeling STDEVS models without making use of probability space theory and also provides a formal framework for conventional DEVS simulation tools that make use of pseudo random sequence generators. Section 4 shows that STDEVS is closed under coupling, and, finally, Section 5 illustrates the use of the new formalism with a simulation example.

1.1. DEVS Formalism

A DEVS model [16] processes an input event trajectory and –according to that trajectory and to its own initial conditions– provokes an output event trajectory. Formally, a DEVS *atomic* model is defined by the following structure:

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

where

- X is the set of input event values, i.e., the set of all the values that an input event can take;
- Y is the set of output event values;
- S is the set of state values;
- δ_{int}, δ_{ext}, λ and ta are functions which define the system dynamics.

Each possible state s ($s \in S$) has an associated *time advance* calculated by the *time advance function* $ta(s)$ ($ta(s) : S \rightarrow \Re_0^+$). The *time advance* is a nonnegative real number saying how long the system remains in a given state in absence of input events.

Thus, if the state adopts the value s_1 at time t_1, after $ta(s_1)$ units of time (i.e., at time $ta(s_1)+t_1$) the system performs an *internal transition*, going to a new state s_2. The new state is calculated as $s_2 = \delta_{int}(s_1)$, where δ_{int} ($\delta_{int} : S \rightarrow S$) is called *internal transition function*.

When the state goes from s_1 to s_2 an output event is produced with value $y_1 = \lambda(s_1)$, where λ ($\lambda : S \rightarrow Y$) is called *output function*. Functions ta, δ_{int}, and λ define the autonomous behavior of a DEVS model.

When an input event arrives, the state changes instantaneously. The new state value depends not only on the input event value but also on the previous state value and the elapsed time since the last transition. If the system goes to the state s_3 at time t_3 and then an input event arrives at time $t_3 + e$ with value x_1, the new state is calculated as $s_4 = \delta_{ext}(s_3, e, x_1)$ (note that $ta(s_3) > e$). In this case, we say that the system performs an *external transition*. Function δ_{ext} ($\delta_{ext} : S \times \Re_0^+ \times X \rightarrow S$) is called the *external transition function*. No output event is produced during an external transition.

DEVS models can be coupled in a modular way [16]. A DEVS coupled model N is defined by the structure:

$$N = (X_N, Y_N, D, \{M_d\}, \{I_d\}, \{Z_{i,d}\}, Select)$$

where:

- X_N and Y_N are the sets of input and output values of the coupled model.
- D is the set of component references, so that for each $d \in D$, M_d is a DEVS model.
- For each $d \in D \cup \{N\}$, $I_d \subset (D \cup \{N\}) - \{d\}$ is the set of influencer models on subsystem d.
- For each $i \in I_d$, $Z_{i,d}$ is the translation function, where

$$Z_{i,d} : \begin{cases} X_N \rightarrow X_d & \text{if } i = N \\ Y_i \rightarrow Y_N & \text{if } d = N \\ Y_i \rightarrow X_d & \text{otherwise} \end{cases}$$

- $Select : 2^D \rightarrow D$ is a tie–breaking function for simultaneous events that must verify $Select(E) \in E$, being $E \subset 2^D$ the set of components producing the simultaneity of events.

DEVS models are closed under coupling, i.e., the coupling of DEVS models defines an equivalent atomic DEVS model.

1.2. Probability Spaces

We recall here some concepts of probability spaces [7].

A sample space S of a random experiment is a set that includes all the possible outcomes of the experiment.

An event space (also referred as *sigma–field* or *sigma–algebra*) $\mathcal{F}$ of the sample space S is a nonempty collection made of subsets of S.

A sigma–field cannot be any arbitrary collection of subsets of S. A collection $\mathcal{F}$ must satisfy the following properties in order to constitute a sigma–field:

- if $F \in \mathcal{F}$ then $F^c \in \mathcal{F}$ (where F^c is the complement of F in S).
- if $F_i \in \mathcal{F}$ for $i = 1, \ldots, \infty$, then also $\bigcup_{i=1}^{\infty} F_i \in \mathcal{F}$

Notice that since $F^c \cup F = S$, the last two conditions imply that $S \in \mathcal{F}$ and also $\oslash \in \mathcal{F}$.

A particular sigma–field over S is the collection of all the subsets of S (2^S, called the power set of S).

Let $\mathcal{G}$ be a particular collection of subsets of S. The sigma–field generated by $\mathcal{G}$, denoted $\mathcal{M}(\mathcal{G})$, is the smallest sigma–field that contains all the elements of $\mathcal{G}$.

A pair $(S, \mathcal{F})$ consisting on a sample space S and a sigma field $\mathcal{F}$ of subsets of S is called a measurable space.

A probability measure P on a measurable space $(S, \mathcal{F})$ is an assignment of a real number $P(F)$ to every member F of the sigma-field, such that P obeys the following rules,

- Axiom 1. $P(F) \geq 0$ for all $F \in \mathcal{F}$.
- Axiom 2. $P(S) = 1$.
- Axiom 3. If $F_i \in \mathcal{F}$, $i = 1, \ldots, \infty$ are disjoint sets, then $P(\bigcup_{i=1}^{\infty} F_i) = \sum_{i=1}^{\infty} P(F_i)$

When $\mathcal{F} = \mathcal{M}(\mathcal{G})$ (the sigma field is generated from a collection $\mathcal{G}$), the knowledge of $P(G)$ with $G \in \mathcal{G}$ defines function P for every $F \in \mathcal{F}$.

Finally, a *probability space* is defined as a triple $(S, \mathcal{F}, P)$ consisting of a sample space S, a sigma–field $\mathcal{F}$ of subsets of S, and a probability measure P defined for all members of $\mathcal{F}$. Synthesizing, for every $F \in \mathcal{F}$, $P(F)$ expresses the probability that the experiment produces a sample $s \in F \subseteq S$.

2. STDEVS DEFINITION REVISITED

A STDEVS model has the structure:

$$M_{ST} = (X, Y, S, \mathcal{G}_{int}, \mathcal{G}_{ext}, P_{int}, P_{ext}, \lambda, ta)$$

where X, Y, S, λ, ta have the same definition as in DEVS.

$\mathcal{G}_{int} : S \to 2^S$ is a function that assigns a collection of sets $\mathcal{G}_{int}(s) \subseteq 2^S$ to every state s. Given a state s, the collection $\mathcal{G}_{int}(s)$ contains all the subsets of S that the next state might belong to with a known probability, determined by a function $P_{int} : S \times 2^S \to [0,1]$. When the system is in state s the probability that the internal transition carries it to a set $G \in \mathcal{G}(s)$ is calculated by $P_{int}(s, G)$.

Calling $\mathcal{F}_{int}(s) \triangleq \mathcal{M}(\mathcal{G}_{int}(s))$ to the minimum sigma-algebra generated by $\mathcal{G}_{int}(s)$, the triplet $(S, \mathcal{F}_{int}(s), P_{int}(s, \cdot))$ is a probability space for each state $s \in S$.

In a similar way, $\mathcal{G}_{ext} : S \times \Re_0^+ \times X \to 2^S$, is a function that assigns a collection of sets $\mathcal{G}_{ext}(s, e, x) \subseteq 2^S$ to each triplet (s, e, x). Given a state s and an elapsed time e, if an event with value x arrives, $\mathcal{G}_{ext}(s, e, x)$ contains all the subsets of S that the next state can belong to, with a known probability calculated by $P_{ext} : S \times \Re_0^+ \times X \times 2^S \to [0,1]$.

Calling $\mathcal{F}_{ext}(s, e, x) \triangleq \mathcal{M}(\mathcal{G}_{ext}(s, e, x))$ to the minimum sigma-algebra generated by $\mathcal{G}_{ext}(s, e, x)$, the triplet $(S, \mathcal{F}_{ext}(s, e, x), P_{ext}(s, e, x, \cdot))$ is a probability space for every triplet (s, e, x).

3. DEVS MODELS WITH FUNCTIONS RND

We will show that a DEVS model whose transition functions depend on random variables (typically generated using RND functions), always define a STDEVS model. Thus, in the first place it will be clear that STDEVS can represent any *practical* stochastic DEVS model defined by the usual method of using RND functions. In second place, this property allows us to define and simulate STDEVS models in a very simple and straight way, getting rid of the need for using probability spaces which add complexity to the model definition structure and terminology.

Theorem 1. *A DEVS model $M_D = (X, Y, S, \delta_{int}, \delta_{ext}, \lambda, ta)$ in which its state change functions δ_{int} and δ_{ext} depend dynamically on a random experiment through a random variable r (i.e., $\delta_{int} = \delta_{int}(s, r)$ and $\delta_{ext} = \delta_{ext}(s, e, x, r)$) with $r \in R \subseteq \Re^n$ characterized by a probability measure $P(r \in B \mid B \in \mathcal{B} \subseteq 2^R)$, defines an equivalent STDEVS model.* [1]

Proof: We shall obtain an STDEVS model $M_{ST} = (X, Y, S, \mathcal{G}_{int}, \mathcal{G}_{ext}, P_{int}, P_{ext}, \lambda, ta)$ equivalent to M_D, assuming that X, Y, S, λ, ta are identical for M_D and M_{ST}. Thus, we only need to find $\mathcal{G}_{int}, \mathcal{G}_{ext}, P_{int}$ and P_{ext}.

We start defining the collecting set $\mathcal{G}_{int}(s)$ in relation to the sigma-algebra $\mathcal{B}$ of the random experiment. For each set $B \in \mathcal{B}$ and for each state $s \in S$, we define the *image set* $G_{s,B} \subseteq S$ according to:

$$\hat{s} \in G_{s,B} \iff \exists r \in B / \delta_{int}(s, r) = \hat{s}$$

Then, we define $\mathcal{G}_{int}(s)$ as:

$$\mathcal{G}_{int}(s) \triangleq \{G_{s,B} | B \in \mathcal{B}\}$$

Therefore, for the system being in state s, the probability of transition to a new state belonging to $G_{s,B} \in \mathcal{G}_{int}(s)$ is:

$$P_{int}(s, G_{s,B}) = P(r \in B)$$

Then, for each state $s \in S$, the function $P_{int}(s, \cdot)$ is a probability measure in the measurable space $(S, \mathcal{F}_{int}(s))$, being $\mathcal{F}_{int}(s) = \sigma(\mathcal{G}(s))$ the minimum sigma-algebra generated by $\mathcal{G}_{int}(s)$. This is demonstrated by verification of the following axioms:

1. $P_{int}(s, G_{s,B}) \geq 0$ because $P_{int}(s, G_{s,B}) = P(r \in B) \geq 0$.
2. $P_{int}(s, S) = 1$, given $\delta_{int}(s, r) \in S, \forall s, r$.
3. Let $B_1, B_2 \in \mathcal{B}$. Then, if $G_{s,B_1} \cap G_{s,B_2} = \oslash \Rightarrow B_1 \cap B_2 = \oslash$. Therefore, the following holds true: $P_{int}(s, G_{s,B_1} \cup G_{s,B_2}) = P(r \in B_1 \cup B_2) = P(r \in B_1) + P(r \in B_2) = P_{int}(s, G_{s,B_1}) + P_{int}(s, G_{s,B_1})$

So far, we obtained $\mathcal{G}_{int}$ and P_{int} for the STDEVS model M_{ST} departing from the DEVS model M_D definition and the randomness condition incorporated in $\delta_{int}(s, r)$.

In the case of $\mathcal{G}_{ext}$ and P_{ext} we proceed analogously, this time replacing the state s by the triplet (s, e, x) for the analysis. This concludes the proof.

In the case that one (or both) of the transition functions is deterministic, it can still be defined as $\delta(\cdot, r)$, but in such a way that it results independent on r. Hence, the whole previous analysis remains valid. Following this reasoning, the theorem here presented is an alternative way for demonstrating that deterministic DEVS is a particular case of stochastic STDEVS, where randomness is removed from state transition dynamics.

[1] We call $\mathcal{B}$ to the sigma-algebra where function P is defined.

3.1. Particular Case: Random Variable r with Uniform Distribution

Consider now the particular case $r \in R = [0,1]^n \subset \Re^n$ with uniform distribution. We say that r is uniformly distributed when every component of r have uniform distribution over the interval $[0,1]$:

$$r_i \sim U(0,1),\ \ i = 1,2,\ldots,n$$

This is the typical case emulated by *pseudo-random sequence generators* used in most of the programming languages (we will call them *RND*). It is interesting to take a look separately for this particular case given STDEVS models will be usually simulated using *RND* functions.

The following is then, a corollary of Theorem 1, particularizing STDEVS model properties when using *RND* functions within the transition definitions.

Corolary 1. *A DEVS model in which $\delta_{int}(s,r)$ depends on n functions RND (i.e., $r \sim U(0,1)^n$) defines a STDEVS equivalent model.*

This corollary does not need a demonstration, given it is a particular case of Theorem 1, taking $R = [0,1]^n$. Anyway, we can make explicit reference of the components of the resulting STDEVS model.

Proceeding like the general case, for each *image set* $G_{s,B} \in \mathcal{G}(s)$, the probability of transitioning from state s to a new state belonging to the set $G_{s,B}$ will be:

$$P_{int}(s, G_{s,B}) = P(r \in B)$$

which turns out to be the Lebesgue Measure for the set B.

4. CLOSURE UNDER COUPLING IN STDEVS

We will show that a coupled DEVS model $N = \langle X_N, Y_N, D, \{M_d\}, \{I_d\}, \{Z_{i,d}\}, Select\rangle$ with $M_d \in \{M_d\}$ being STDEVS atomic models for all d, defines an equivalent atomic STDEVS model, thus verifying STDEVS closure under coupling.

To achieve this, we will find an atomic STDEVS model $M_{ST} = (X, Y, S_N, \mathcal{G}_{int_N}, \mathcal{G}_{ext_N}, P_{int_N}, P_{ext_N}, \lambda, ta)$ defined by the coupling expression N.

We begin defining the relationships that are shared with the classic proof for deterministic DEVS:

- $X = X_N, Y = Y_N$
- $S_N = \underset{d\in D}{\times} \{(s_d, e_d)\}$ with $s_d \in S_d, e_d \in \Re$. Each component of S_N has the form $s_N = (\ldots,(s_d,e_d),\ldots)$.
- $t_a(s_N) = min\{\sigma_d \mid d \in D\}$, with $\sigma_d = t_{a_d}(s_d) - e_d$.
- $d^* = Select(IMM(s_N))$
- $\lambda_{s_N} = \begin{cases} Z_{d^*,N}(\lambda_{d^*}(s_{d^*})) & \text{if } d^* \in I_N, \\ \oslash & \text{otherwise.} \end{cases}$

Then, we need to obtain the probability spaces that will represent the stochastic dynamics of the coupled model, as a result of the stochastic behavior of its atomic components.

First, for internal transitions, we define the set–collecting function:

$$\mathcal{G}_{int_N}(s_N) \triangleq \underset{d\in D}{\times} (\mathcal{G}_d \times \{\tilde{e}_d\})$$

where

$$\mathcal{G}_d = \begin{cases} \mathcal{G}_{int}(s_{d^*}) & \text{if } d = d^*, \\ \mathcal{G}_{ext}(s_d, \hat{e}_d, x_d) & \text{if } x_d \neq \oslash, \\ \{s_d\} & \text{otherwise.} \end{cases}$$

with

$$x_d = \begin{cases} Z_{d^*,d}(\lambda_{d^*}(s_{d^*})) & \text{if } d^* \in I_d, \\ \oslash & \text{otherwise.} \end{cases}$$

$$\tilde{e}_d = \begin{cases} 0 & \text{if } d = d^* \text{ or } x_d \neq \oslash, \\ \hat{e}_d & \text{otherwise.} \end{cases}$$

and

$$\hat{e}_d = e_d + ta_{d^*}(s_{d^*}) - e_{d^*}$$

The sets $G_N \in \mathcal{G}_{int_N}(s_N)$ will have the form $G_N = (\ldots(G_d, \{e_d\}),\ldots)$ and will verify $G_N \subseteq S_N$.

We also call $\mathcal{F}_{int_N}(s_N) \triangleq \mathcal{M}(\mathcal{G}_{int_N}(s_N))$ the minimum sigma-algebra generated by $\mathcal{G}_{int_N}(s_N)$. Then, the probability measure for the internal transition process in N, $P_{int_N} : S_N \times 2^{S_N} \to [0,1]$ is defined as:

$$P_{int_N}(s_N, G_N) \triangleq P_{int_{d^*}}(s_{d^*}, G_{d^*}) \prod_{d|x_d\neq\oslash} P_{ext_d}(s_d, \hat{e}_d, x_d, G_d)$$

and the triplet $(S_N, \mathcal{F}_{int_N}(s_N), P_{int_N}(s_N, \cdot))$ is a probability space. Similarly, for external transitions we define the set–collecting function:

$$\mathcal{G}_{ext_N}(s_N, e, x_N) \triangleq \underset{d\in D}{\times} (\mathcal{G}_d \times \{\tilde{e}_d\})$$

where

$$\mathcal{G}_d = \begin{cases} \mathcal{G}_{ext}(s_d, \hat{e}_d, x_d) & \text{if } x_d \neq \oslash, \\ \{s_d\} & \text{otherwise.} \end{cases}$$

$$\tilde{e}_d = \begin{cases} 0 & \text{if } x_d \neq \oslash, \\ \hat{e}_d & \text{otherwise.} \end{cases}$$

with

$$x_d = \begin{cases} Z_{N,d}(x_N) & \text{if } N \in I_d, \\ \oslash & \text{otherwise.} \end{cases}$$

and

$$\hat{e}_d = e_d + e$$

The sets $G_N \in \mathcal{G}_{int_N}(s_N)$ will also have the form $G_N = (\ldots(G_d,\{e_d\}),\ldots)$ and will verify $G_N \subseteq S_N$.

Again, we define $\mathcal{F}_{ext_N}(s_N,e,x_N) \triangleq \mathcal{M}(\mathcal{G}_{ext_N}(s_N,e,x_N))$ the minimum sigma-algebra generated by $\mathcal{G}_{ext_N}(s_N,e,x_N)$. Then, the probability measure for the external transition process in N, $P_{ext_N} : S_N \times \Re \times X \times 2^{S_N} \rightarrow [0,1]$ is defined as:

$$P_{ext_N}(s_N,e,x_N,G_N) = \prod_{d|x_d \neq \oslash} P_{ext_d}(s_d,\hat{e}_d,x_d,G_d)$$

and the triple $(S_N, \mathcal{F}_{ext_N}(s_N,e,x_N), P_{ext_N}(s_N,e,x_N,\cdot))$ is a probability space.

5. EXAMPLE MODEL

We will give a simple example for a system which dynamics fully depend on random experiments. Using the theory presented we will see that the practical DEVS representations of the random processes are consistent with their STDEVS specification in terms of probability spaces.

The example *Load Balancing Model* (LBM) is a simplification of a computing system that processes successive Tasks, consisting on the atomic models: *Load Generator* (LG), *Weighted Balancer* (WB) and two *Servers* (S1,S2) with no queuing policy (i.e., the Tasks arriving at a busy server are discarded). The set {WB,S1,S2} form the subsystem *Cluster* (CL), a coupled model.

As we did before, transition functions will be expressed in terms of $r \sim U(0,1)$, namely $\delta_{int}(\cdot) = \delta_{int}(s,r)$ and $\delta_{ext}(\cdot) = \delta_{ext}(s,e,x,r)$.

5.1. Load Generator

Consider a system that generates a number of Tasks in the unit time following a discrete Poisson random distribution being d_r the *mean expected departure rate*. It can be proven that the inter-departure time σ_k between tasks k and $k+1$ is exponentially distributed according $P(\sigma_k \leq t) = 1 - e^{-at}$ where $a = d_r$ and $1/a$ is the mean expected value. We will assume that LG generates only one type of task (Task: $task_1$) which goes out through the only output port (Port: out_1). LG does not have any inputs, thus only internal transitions are possible. The STDEVS definition for LG is:

$$M_{ST}^{LG} = (X,Y,S,\mathcal{G}_{int},\mathcal{G}_{ext},P_{int},P_{ext},\lambda,ta)$$

where the deterministic components are:

- $X = \oslash, Y = \{(task_1,out_1)\}$
- $S = \Re_0^+$
- $\lambda_s = \{(task_1,out_1)\}$
- $t_a(s) = s$

and the probabilistic-related elements are:

- $\mathcal{G}_{int} = \{A_t \mid t \geq 0\}$, $A_t = [0,t)$
- $P_{int}(s,G) = P_{int}(s,A_t) = 1 - e^{-at}$, $G \in \mathcal{G}_{int}$

As we can see the stochastic description for the inter departure time of tasks is mapped directly to the function P_{int} through the corresponding cumulative distribution function. Because only internal transitions are possible, we do not need to define $\mathcal{G}_{ext}, P_{ext}$.

Nevertheless, for implementing this STDEVS model M_{ST}^{LG} in a simulator, the probabilistic description must be translated into an algorithm to be evaluated into the internal transition code, representing the associated DEVS $\delta_{int}(\cdot)$ function. According our previous definitions we define:

$$\delta_{int}(s,r) = -(1/a)log(r)$$

where by means of the inverse transformation method we obtained an exponential distributed function making use of a uniform distributed variable $r \sim U(0,1)$ available as a RND() function in most languages.

Finally, the equivalent DEVS specification for LG will be:

$$M_D^{LG} = (X,Y,S,\delta_{int},\delta_{ext},\lambda,ta)$$

where:

$$\begin{cases} X = \oslash \\ Y = \{(task_1,out_1)\} \\ S = \Re_0^+ \\ \delta_{int}(s,r) = -(1/a)log(r) \\ \delta_{ext}(s,e,x,r) = s \\ \lambda(s) = \{(task_1,out_1)\} \\ t_a(s) = s \end{cases}$$

In this component, the next randomly calculated inter-departure time is stored in the real valued state s, which is then used by the time advance function $t_a(s) = s$ to "sleep" LG during the corresponding amount of time. Similar reasoning can be applied for the rest of the components, where the state values are used for storage purposes and models are specified in a shorter way.

5.2. Weighted Balancer

The WB component delivers the incoming tasks arriving at input port (Port: inp_1) to the output ports out_1 and out_2 based on a balancing factor $b_f \in [0,1]$ that determines the weight relation between both ports. For $b_f = 0.5$ both outputs have the same weight and therefore the outgoing load will be balanced equiprobably. For $b_f > 0.5$ out_1 is privileged and for $b_f < 0.5$ out_2 is privileged, in a linear fashion. The tasks accepted belong to a set $T = \{task_1,...,task_m\}$ with m different possible tasks.

We will give the DEVS definition M_D^{WB} for WB:

$$M_D^{WB} = (X, Y, S, \delta_{int}, \delta_{ext}, \lambda, ta)$$

The corresponding equivalent STDEVS model M_{ST}^{WB} can be obtained following the same reasoning previously used for component LG. From now on, we will make use of Theorem 1 and will refer only to the DEVS form of components with some form of stochastic behavior, containing RND() functions in the algorithms that evaluate transitions.

Then we have:

- $X = T \times \{inp_1\}$, $Y = T \times \{out_1, out_2\}$
- $S = T \times \{out_1, out_2\} \times \Re_0^+$
- $\lambda(w, p, \sigma) = (w, p)$
- $t_a(w, p, \sigma) = \sigma$

The state is a triplet $s = (w, p, \sigma)$, where w represents the last task received, p is the port where that task is delivered and σ is the time advance. For our example $T = \{task_1\}$. After receiving an event (x_v, x_p) the new state must be evaluated by:

$$\delta_{ext}((w, p, \sigma), e, (x_v, x_p), r) = (x_v, \tilde{p}, 0)$$

with

$$\tilde{p} = \begin{cases} out_1 & \text{if } r < b_f, \\ out_2 & \text{otherwise.} \end{cases}$$

and $r \sim U(0,1)$. Finally, the internal transition will be:

$$\delta_{int}((w, p, \sigma), r) = (w, p, \infty)$$

in this case, independent of r.

5.3. Server1 and Server2

The servers S1 and S2 are components that receive the tasks delivered by the balancer WB. For each task received, a server processes it demanding a service time s_t and sends it out to a sink, where it is recognized as a processed task. The variable s_t is distributed exponentially with $P(s_t \leq t) = 1 - e^{-bt}$, and its mean expected value is $1/b$.

There is no queuing policy nor preemption defined for the servers. So, if a new task arrives to a server when it is busy processing a previous task, the arriving task is ignored.

We will give the DEVS definition $M_D^{S_n}$ with $n = 1, 2$ for S1 and S2 respectively:

$$M_D^{S_n} = (X, Y, S, \delta_{int}, \delta_{ext}, \lambda, ta)$$

where:

- $X = T \times \{inp_1\}$, $Y = T \times \{out_1\}$
- $S = T \times \{0, 1\} \times \Re_0^+$
- $\lambda(w, busy, \sigma) = (w)$
- $t_a(w, busy, \sigma) = \sigma$

The state is a triplet $s = (w, busy, \sigma)$, where w represents the last task received, *busy* represent the status of the server (if $busy = 1$ the server is processing a task and if $busy = 0$ the server is free) and σ is the time advance. For our example, we have $T = \{task_1\}$ and only one input port and one output port. After receiving an event (x_v, x_p) the new state will be evaluated according:

$$\delta_{ext}((w, busy, \sigma), e, (x_v, x_p), r) = (\tilde{w}, 1, \tilde{\sigma})$$

with

$$\begin{cases} \tilde{w} = x_v, \tilde{\sigma} = -(1/b)log(r) & \text{if } busy = 0, \\ \tilde{w} = w, \tilde{\sigma} = \sigma - e & \text{if } busy = 1. \end{cases}$$

and $r \sim U(0,1)$. The internal transition will be:

$$\delta_{int}((w, busy, \sigma), r) = (w, 0, \infty)$$

independent of r.

5.4. The Complete Model

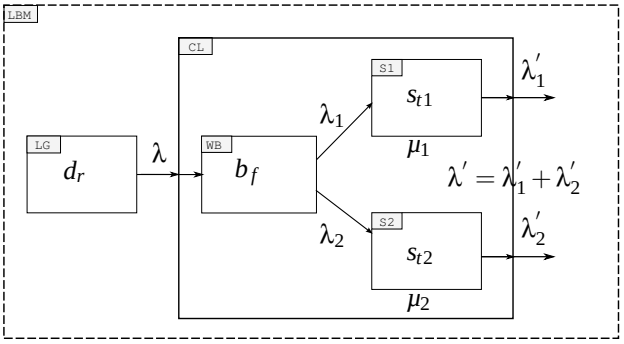

Figure 1. Topology of the Load Balancer Model (LBM) example.

The system is intended to show a scenario where random variables affect all of its building components. Here, we have a Poisson process dominating task generation, a Uniform process (with a latter deterministic bias) affecting the balancing between two servers and a Negative Exponential process representing task servicing times at servers. Nevertheless, the implementation always rely on the use of a uniform distributed variable $r \sim U(0,1)$.

In Figure 1 the model topology is represented along with the main model parameters and derived traffic magnitudes that will be used in the next section for executing simulations. With the DEVS specification of these components and their defined interconnections, we built the same system in two different DEVS Simulation Tools (PowerDEVS [11] and CD++

[14]) parameterizing them with identical values, and run several simulations at different operating points for comparison and validation purposes.

5.5. Simulation Results

In order to validate results, we describe the given example model by means of basic queuing theory, derive the equations describing the system, and then compare simulation results against the expected theoretical values.

A single server with no queuing capacity can be described by a $M/M/m/m$ system with $m = 1$ [9]. This description assumes exponential inter-arrival times and exponential service times which match our case. For the i-*th* server we have the parameters λ_i (*arrival rate*) and μ_i *(service rate)*. The *traffic intensity* is defined

$$\rho_i = \lambda_i / \mu_i \tag{1}$$

Because of the limited buffering capacity (in our simplest case, only the servicing task can be "buffered") there is a probability of losing tasks, which will never be serviced. This probability is denoted P_{loss_i} (*probability of loss*) and is related with the traffic intensity by *Erlang´s loss formula* [9] in its simplest form for a single server:

$$P_{loss_i} = \rho_i / (1 + \rho_i) \tag{2}$$

The i-*th* server will see at its input port an *effective arrival rate*:

$$\lambda_i^{'} = \lambda_i (1 - P_{loss_i}) \tag{3}$$

which under stability conditions[2] is equal to the *server throughput* at its output port. In our LBM example, we have $i = 1,2$ for the two servers in the cluster (CL) sub-model. Clearly, the *total system throughput* $\lambda^{'}$ must be $\lambda^{'} = \lambda_1^{'} + \lambda_2^{'}$ hence being a function of the *total system arrival rate* λ and the traffic intensities ρ_1, ρ_2 at the servers.

These magnitudes are all calculated from model parameters set up for simulation: d_r (mean departure rate at LG, in *Tasks/second*), b_f (balancing factor at WB), s_{t1}, s_{t2} (mean service time at S1 and S2 respectively, in *seconds*) in the following way:

$$\begin{aligned} \lambda &= d_r \\ \mu_1 = 1/s_{t1} \qquad & \lambda_1 = b_f \lambda \\ \mu_2 = 1/s_{t2} \qquad & \lambda_2 = (1 - b_f)\lambda \end{aligned} \tag{4}$$

Now, with (1) and (4) in (2) we derive the internal loss probabilities:

$$P_{loss_1} = \frac{b_f d_r s_{t1}}{1 + b_f d_r s_{t1}}, P_{loss_2} = \frac{(1 - b_f) d_r s_{t2}}{1 + (1 - b_f) d_r s_{t2}} \tag{5}$$

[2] In lossy systems, the *effective traffic intensity* $\rho_i^{'} = \lambda_i^{'} / \mu_i$ is always $\rho_i^{'} < 1$ so the typical stability condition $\lambda_i / \mu_i < 1$ is not required. Finite buffer systems are always stable since arriving tasks are discarded when the number of tasks in the system exceeds system capacity.

Finally, we want to express the *total system throughput* in terms of a *total system loss probability* P_{loss} like we did for the individual servers. So with (3) and (5) we obtain:

$$\begin{aligned} P_{loss} &= b_f P_{loss_1} + (1 - b_f) P_{loss_2} \\ \lambda^{'} &= \lambda (1 - P_{loss}) \end{aligned} \tag{6}$$

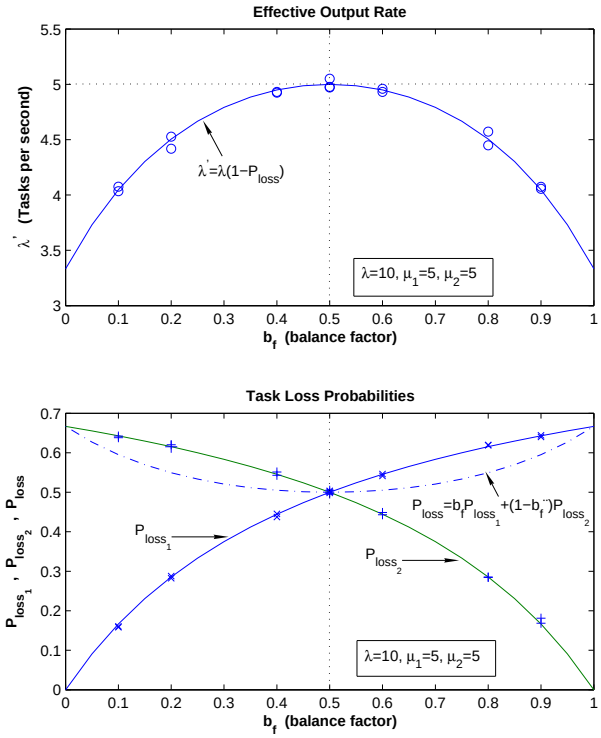

Figure 2. Simulation Results. Test Scenario 1 $d_r = 10, b_f = [0,1], s_{t1} = 0.2, s_{t2} = 0.2$

With equations (6) we completely characterize the system in terms of offered load, loss probabilities and effective throughput. Then, in Figure 5.5. we plot the theoretical curves for $P_{loss}, P_{loss_1}, P_{loss_2}$ and $\lambda^{'}$ as functions of b_f in a *test scenario 1* chosen as $TS_1 = \{d_r = 10, b_f = [0,1], s_{t1} = 0.2, s_{t2} = 0.2\}$. In the same figure we plotted simulation results for the STDEVS model LBM parameterized according the scenario TS_1 at a set of illustrative operational points sweeping b_f between 0 and 1. Simulation results matched closely the expected theoretical curves, for successive repetitions at each point. Simulation point values were derived from the output event log files produced by simulation runs, using calculated[3] *task rate* variables, thus obtaining $\lambda^{'sim}$ and

[3] A general λ_k^{sim} task rate at an arbitrary observation place k is: $\lambda_k^{sim} = NumberOfTasksLogged_k / TotalSimulationTime$.

$P_{loss_i}^{sim} = 1 - (\lambda_i^{'sim}/\lambda_i^{sim})$. The statistical properties of the random variables produced by the atomic models were verified to match with those expected: uniform distribution for b_f, discrete Poisson distribution for λ and exponential distribution for s_{t1} and s_{t2}. This also produced Poisson distributed series of values for all the observed task rates, as expected.

6. CONCLUSIONS

We presented a novel formalism for describing stochastic discrete event systems. Based on the system theoretical approach of DEVS and making use of Probability Spaces, STDEVS provides a formal framework for modeling and simulation of generalized non deterministic discrete event systems.

The development of STDEVS was motivated by a wider project aimed to provide a unified framework for analysis, design, modeling and simulation of automated control techniques targeting the performance optimization of computer systems and data networks; in interaction with continuous and hybrid systems. We have chosen STDEVS as the tool to provide that unified framework, exploiting the advantages of DEVS efficient approximation of continuous systems [4] (for classic control theory techniques representation) and DEVS high-performance execution features (for real-time model execution aims).

Thus, next steps will be oriented to develop STDEVS–based libraries in PowerDEVS and CD++ for modeling and simulation of general computer systems and data networks.

REFERENCES

[1] S. Aggarwal. *Ergodic Machines - Probabilistic and Approximate Homomorphic Simplifications*. PhD thesis, The University of Michigan, Ann Arbor, Michigan, 1975.

[2] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, 1995.

[3] Christos Cassandras. *Discrete Event Systems: Modeling and Performance Analysis*. Irwin and Aksen, Boston, Massachusetts, 1993.

[4] F.E. Cellier and E. Kofman. *Continuous System Simulation*. Springer, New York, 2006.

[5] J.B. Filippi, M. Delhom, and F. Bernardi. The JDEVS Environmental Modeling and Simulation Environment. In *Proceedings of IEMSS 2002*, volume 3, pages 283–288, 2002.

[6] Norbert Giambiasi, Bruno Escude, and Sumit Ghosh. GDEVS: A generalized Discrete Event specification for accurate modeling of dynamic systems. *Transactions of SCS*, 17(3):120–134, 2000.

[7] R. Gray and L. Davisson. *An Introduction to Statistical Signal Processing*. Cambridge University Press, Cambridge, UK, 2004.

[8] C. Joslyn. The Process Theoretical Approach to Qualitative DEVS. In *Proc. 7th Conf. on AI, Simulation, and Planning in High Autonomy Systems (AIS '96)*, pages 235–242, San Diego, California, 1996.

[9] Leonard Kleinrock. *Queuing Systems, Vol.1: Theory*. Wiley & Sons, New York, NY, USA, 1975.

[10] E. Kofman and R.D. Castro. STDEVS, A Novel Formalism for Modeling and Simulation of Stochastic Discrete Event Systems. In *Proceedings of AADECA 2006*, Buenos Aires, Argentina, 2006.

[11] E. Kofman, M. Lapadula, and E. Pagliero. PowerDEVS:A DEVS Based Environment for Hybrid System Modeling and Simulation. Technical Report LSD0306, LSD, UNR, 2003. Available at http://www.fceia.unr.edu.ar/∼kofman.

[12] B. Melamed. *Analysis and Simplifications of Discrete Event Systems and Jackson Queuing Networks.* PhD thesis, The University of Michigan,, Ann Arbor, Michigan, 1976.

[13] James Nutaro. *Parallel Discrete Event Simulation with Application to Continuous Systems*. PhD thesis, The University of Arizona, 2003.

[14] G. Wainer, G. Christen, and A. Dobniewski. Defining DEVS Models with the CD++ Toolkit. In *Proceedings of ESS2001*, pages 633–637, 2001.

[15] B. Zeigler. *Theory of Modeling and Simulation*. John Wiley & Sons, New York, 1976.

[16] B. Zeigler, T.G. Kim, and H. Praehofer. *Theory of Modeling and Simulation. Second edition*. Academic Press, New York, 2000.

[17] B. Zeigler and S. Vahie. Devs formalism and methodology: unity of conception/diversity of application. In *Proceedings of the 25th Winter Simulation Conference*, pages 573–579, Los Angeles, CA, 1993.

[18] Bernard Zeigler and Hessam Sarjoughian. *Introduction to DEVS Modeling and Simulation with JAVA: A Simplified Approach to HLA-Compliant Distributed Simulations*. Arizona Center for Integrative Modeling and Simulation, 2000.