Elsevier Editorial System(tm) for Information Systems
Manuscript Draft

Manuscript Number: IS-D-14-436

Title: An Ontological Process Modeling Framework for Stochastic Systems

Article Type: Regular Article

Keywords: Process Modeling, Stochastic Systems, Ontology, Reliability, Failure

Corresponding Author: Dr. Atiq Waliullah Siddiqui, PhD

Corresponding Author's Institution: University of Dammam

First Author: Atiq Waliullah Siddiqui, PhD

Order of Authors: Atiq Waliullah Siddiqui, PhD

Abstract: Several process modeling techniques exist that are mostly without any firm theoretical foundation. This results in a lack of model validation in terms of model consistency, feasibility and compliance with its objectives. Furthermore, most of these techniques are essentially deterministic in nature and not applicable to widely present stochastic systems such as that related to computer, business and industrial applications. In this paper, we propose an ontology-based stochastic process modeling framework that further provides a specialization to failure and reliability issues. The framework is notation independent, and is primarily rooted in Bunge's ontology and two of its related process modeling frameworks i.e. Bunge-Wand-Weber (BWW) and Generic-Process-Models (GPM). To support the modeling of stochastic elements, several existing BWW/GPM definitions are revised or elaborated, while new definitions are proposed. The well-established Theory of Reliability constructs are also mapped to facilitate the modeling of failure prone systems. An example is also analyzed for illustrative proposes.

*Abstract

An Ontological Process Modeling Framework for Stochastic Systems

ABSTRACT

Several process modeling techniques exist that are mostly without any firm theoretical foundation. This results in a lack of model validation in terms of model consistency, feasibility and compliance with its objectives. Furthermore, most of these techniques are essentially deterministic in nature and not applicable to widely present stochastic systems such as that related to computer, business and industrial applications. In this paper, we propose an ontology-based stochastic process modeling framework that further provides a specialization to failure and reliability issues. The framework is notation independent, and is primarily rooted in Bunge's ontology and two of its related process modeling frameworks i.e. Bunge-Wand-Weber (BWW) and Generic-Process-Models (GPM). To support the modeling of stochastic elements, several existing BWW/GPM definitions are revised or elaborated, while new definitions are proposed. The well-established Theory of Reliability constructs are also mapped to facilitate the modeling of failure prone systems. An example is also analyzed for illustrative proposes.

Keywords: Process Modeling, Stochastic Systems, Ontology, Reliability, Failure

*Cover letter

30 November 2014

Profs. Dennis Shasha and Gottfried Vossen, Editor-in-Chiefs
Information Systems

Dear Professors,

Please find our manuscript titled "*An Ontological Process Modeling Framework for Stochastic Systems*", which we would like to submit for publication in Information Systems. The work presents an ontology-based stochastic process modeling framework with a specialization to failure and reliability issues. As a step towards automating the design and analysis tasks, we also show complete mapping of the proposed framework constructs to the Theory of Reliability constructs. An example is also analyzed for illustrative purposes.

We really appreciate if you kindly give it a due consideration and look forward to hearing from your office soon.

Sincerely,

Atiq Siddiqui, PhD
Assistant Professor,
College of Business Administration,
University of Dammam, KSA.
Email: awsiddiqui@ud.edu.sa
Tel.: +966.13.333.2100

An Ontological Process Modeling Framework for Stochastic Systems

Highlights

- A notation-independent stochastic process modeling framework is proposed
- The framework is mainly based on Bunge's ontology
- The framework provides specialization related to failure and reliability issues
- The framework constructs are mapped to the Theory of Reliability constructs

An Ontological Process Modeling Framework for Stochastic Systems

Manuscript for Information Systems

Atiq W. Siddiqui[1]

College of Business Administration, University of Dammam,

Dammam, KSA

[1]**Email**: awsiddiqui@ud.edu.sa | **Phone**: +966.13.333.2100 | **Mail**: PO Box 1982, Dammam 31451, KSA

An Ontological Process Modeling Framework for Stochastic Systems

ABSTRACT

Several process modeling techniques exist that are mostly without any firm theoretical foundation. This results in a lack of model validation in terms of model consistency, feasibility and compliance with its objectives. Furthermore, most of these techniques are essentially deterministic in nature and not applicable to widely present stochastic systems such as that related to computer, business and industrial applications. In this paper, we propose an ontology-based stochastic process modeling framework that further provides a specialization to failure and reliability issues. The framework is notation independent, and is primarily rooted in Bunge's ontology and two of its related process modeling frameworks i.e. Bunge-Wand-Weber (BWW) and Generic-Process-Models (GPM). To support the modeling of stochastic elements, several existing BWW/GPM definitions are revised or elaborated, while new definitions are proposed. The well-established Theory of Reliability constructs are also mapped to facilitate the modeling of failure prone systems. An example is also analyzed for illustrative proposes.

Keywords: Process Modeling, Stochastic Systems, Ontology, Reliability, Failure

1. Introduction

System and process modeling plays a key role in the analysis, design, development, and management of complex systems [1]. These systems may be related to computer, business and industrial applications. Given such a wide scope, numerous modeling techniques exist that capture a system's structure and its related processes such as activity sequencing, resource allocation and communications [2], which are addressed from different perspectives [3]. Examples of some commonly used modeling formalisms are Petri-nets and stochastic Petri-nets [4, 5], which are used to model distributed systems; Business Process Modeling Notation (BPMN), which aims to capture business procedures [6]; Integrated Definition (IDEF), which is a family of modeling languages that integrates process modeling and its related data structures [3]; Business Process Execution Language (BPEL), which is mainly used to model internet processes [7]; and Systems Dynamics, which models systems with feedback [8]. Unified Modeling Language (UML) is another example [9], which was initially designed to facilitate information systems design and analysis; however, it made its way to the general process modeling arena due to its flexibility. It is important to note that this is not an exhaustive list and many more such formalisms exist. Covering all of these is out of the scope of this paper; however, we refer to the following reviews that provide an extensive analysis of the broader process modeling literature. Anaya et. al., [10] provided an overview of the Unified Enterprise Modelling Language (UEML), which is a framework that integrates various IS and enterprise models; Recker et. al., [11], Aldin and de Cesare [12] and Oca et. al., [13] examined various aspects of business process techniques; and more recently García-Borgoñon et. al., [14] analyzed software process modeling languages.

Most of these cited and referred formalisms have an underpinning in practice as opposed to having a firm theoretical foundation. This usually results in a lack of model verification in terms of its completeness, consistency, feasibility and goal compliance in relation to the actual system or process [2, 15]. It is easy to see that such issues, due to the consequent inconsistencies, result in a lack of utilization of these models in the intended jobs stated earlier. There are also frameworks that have theoretical bases; however, most of these are limited in scope. For example, BPEL and Petri-nets have precise construct sets that allow formal analysis and verification, but these are applicable only to the modeling of existing systems and activity sequencing respectively. Accordingly, studies have also been conducted that examine modeling formalisms in terms of their ontological completeness, clarity and redundancy

[16-18]. For instance, Wand and Weber [19] assessed the structure of the Information Systems Analysis and Design (ISAD) methodologies (mainly Data Flow and Entity Relationship Diagrams) on these measures as based on Bunge's Ontology [20]. They showed a lack of expressiveness in these techniques due to incompleteness, construct overload and redundancy issues. Similarly, Parsons and Wand [21], also relying on Bunge's Ontology [20], evaluated Object Oriented modeling and suggested using representation-based as compared to implementation-based foundations. Likewise, Dussart, Aubert and Patry [22] evaluated the dynamic aspects of UML in the context of organizational flow; and, Green and Rosemann [18] evaluated the ARIS framework to determine the ability of the language to provide a good representation of the perception of an analyst.

It is important to note that there are also attempts at devising fully theory driven and notation independent modeling approaches, which is especially seen in the information systems and computer science domains. In this direction, Wand and Weber [16] presented a Bunge's ontology [20, 23] based information systems modeling framework (Bunge-Wand-Weber or BWW) by providing a set of precise modeling constructs. Similarly, Soffer and Wand [2, 15], extended BWW for goal driven processes (i.e. Generic-Process-Models or GPM) and added constructs for hard and soft constraints as well as process goals. Despite an elaborate and precise nature, both have a limited scope as these are intended for purely deterministic systems and processes only. This is a major limitation as many real world systems have stochastic behaviors such as random responses and failures. Examples of these can be found in business processes, computer and information systems, supply chains and computer networks etc. that contain process variability and failure prone elements. To overcome this limitation, we propose a generalization to BWW/GPM, further referred to as ***SSPM*** (or ***Stochastic System and Process Modeling Framework***), which aims to capture not only the stochastic nature of systems but also provides specialization to their failure and reliability issues. Consequently, we show a complete mapping of SSPM on the Theory of Reliability (TR) constructs [24]—perhaps the most important theory related to failure prone systems. The implication is that SSPM models can be employed directly and automatedly in the reliability analysis of such systems.

The rest of the paper is organized as follows: In section 2, we present the proposed SSPM modeling framework. In section 3, we present complete mapping of SSPM on TR constructs, which is followed by conclusions in sections 4.

2. The Modeling Framework

In this section, we present the SSPM modeling framework, where we first explain the basic constructs in section 2.1, followed by system, process and their related components in sections 2.2 and 2.3 respectively; failure and reliability constructs are proposed in section 2.4. We also comment on SSPM process model validity in section 2.5.

2.1. Basic Constructs

As in BWW/GPM, Bunge's ontology [20, 23] provides the ontological basis for SSPM. It defines the most primitive concept in the world as a *Thing*. Things exist independently and possess *properties* by which they are identified. That is, considering $x \in Z$ to be a substantial individual and $p(x) \in P$ its unarized properties (Z and P being sets of substantial individuals and the totality of properties), a thing X is defined as $X =_{df} \langle x, p(x) \rangle$ where $X \in \Theta$ (Θ being the totality of things) [20]. Things combine to form *Composite Things*. Whether simple or composite, all things possess a *state* at any fixed time instance *t*, which is defined by its current set of *attributes* (i.e. perceived properties). All possible states of X (represented as $s(X) \in S(X)$), reached deterministically or probabilistically, form its complete state space $S(X)$. For this state space, $PS =_{def} \left\langle 2^{S(X)}, \Pr \right\rangle$ represents a *general probabilistic structure iff* $S(\mathrm{X})$ is denumerable and $\Pr : 2^{S(\mathrm{X})} \to [0,1]$ (the probability measure defined for all states) satisfying $\Pr(S(\mathrm{X})) = 1$ [20]. The deterministic/probabilistic state space discussed above can be modeled by a general functional schema $X_m =_{def} \left\langle M, \tilde{F} \right\rangle$, where $M_x = [s(X)\ PS]$ (the input vector) and $\tilde{F} = \langle F_1, ..., F_n \rangle : M \to V_1 \otimes ... \otimes V_n$ is the *total probabilistic state function* of X – the Cartesian product of full range of all of the deterministic/ probabilistic state variables $F_i, 1 \le i \le n$. Here each F_i represents a property of X (an attribute) i.e. $x_i = F_i(M),\ 1 \le i \le n$. Note that this functional schema does not need to be inclusive in terms all properties of X, and may contain only a subset of these properties as required in a model. These state functions are governed by *state-law*s $l(X) \in L(X)$, which put restrictions on the possible values of the components of $\tilde{F}$ while establishing lawful/unlawful relationships amongst component of state function. That is, $l(X) : V_1 \otimes ... \otimes V_n \to \{\text{lawful, unlawful}\};$ the consequent *lawful state Space* is $S_L(X) = \{\langle x_1, ..., x_n \rangle \in V_1 \otimes ... \otimes V_n \to \left| \tilde{F} \right.$ satisfying every $l(X) \in L(X)\}$ [16]. Moving from one state to the other is

called a state *change*; which is a result of an *event* that is defined as an ordered pair of states $(s,s') \in S_L(X)$ [16]. An *event*, triggering a *change*, should be *lawful* as well. Thus, $G_L(X,t)$ denote the set of *transformations* from the lawful state space $S_L(X)$ into itself at a time instance *t*, which are given as lawful in the system i.e. $G_L(X,t) \subseteq S_L(X) \otimes S_L(X)$ and *t* > 0 [16]. In case of probabilistic change, the *Stochastic Lawful Transformation* can thus be defined as $G_L(X,t) \subseteq S_L(X) \otimes S_L(X)\ t \in M$ satisfying $\left\langle 2^{G_L(X,t)}, \Pr \right\rangle$ or $\Pr : 2^{S_L(X) \otimes S_L(X)} \to [0,1]$ and $\Pr[G_L(X,t)] = 1$; resulting from *lawful event* $(s,s') \in S_L(X)$ where $s' = g(s,t)$ and $g \in G_L(X,t)$.

2.2. System and its Structure:

Using the above basic constructs, we now present additional concepts needed to precisely define a *stochastic system* as well as its *structure*. However, we first refer to the Bunge's definition of a *system*, which in general, is a collection of interacting *things* that *acts* on or is *acted upon* by other things [20]. A system can thus be seen as an *assembly of things* which is accompanied by a loss or gain of *basic* or *emergent* system properties.

To define a *stochastic system* in this context, we first consider the notions of *History of X* in BWW [16], which is defined as a set of ordered pairs $h(X) = \{\langle t, \tilde{F}(t) \rangle\}$, at any time instant $t > 0$ for $X_m =_{df} \langle M, \tilde{F} \rangle$; and, *X acts* deterministically/stochastically on *Y* (Denoted by $X \rhd Y$) *iff* $h(X/Y) \neq h(X)$; consequently; a deterministic/stochastic coupling between *X* and *Y* exists *iff* $B(X,Y) = (X \rhd Y) \vee (Y \rhd X)$. A *Stochastic System* can thus be defined as:

- **Definition 1:** Let *C* be a set of things (with deterministic/probabilistic state space), then with $B_c = \{(X,Y) \mid X,Y \in C \wedge B(X,Y)\}$, $\rho(C,B_c)$ be a graph with *C* as vertices or nodes and B_c (deterministic/stochastic couplings) edges, then $\sigma = \rho(C,B_c)$ is a *General Stochastic System iff* it is a connected graph.

That is, a general stochastic system σ may be depicted as a thing-on-node graph where the couplings are represented by edges of the graph. The full *structure* of the above stochastic system can elaborated through its *composition* and *environment*, where *composition* of a general stochastic system at time *t* is defined as $C'(\sigma,t) = \{x / x \in \sigma\}$ – the set of things belonging to σ, and *environment at time t* is

$E(\sigma,t) = \{x \,/\, x \notin C(\sigma,t) \wedge (\exists y)(y \in C(\sigma,t) \wedge B(x,y))\}$ – the set of things not belonging to σ but coupled to a thing in σ [16]**.**

Accordingly:

- **Definition 2:** *Structure of a general stochastic system* is a set of all couplings within a system and its environment i.e. $\Pi(\sigma,t) = \{R_i \in \bar{B}(\sigma,t) \cup \hat{B}(\sigma,t)\}$ where $\bar{B}(\sigma,t) = \{B(x,y) \big| x,y \in C'(\sigma,t)\}$ and $\hat{B}(\sigma,t) = \{B(x,y) \big| x \in C'(\sigma,t) \wedge y \in E(\sigma,t)\}$

To illustrate this structure of a *stochastic system*, we present an example of a supply chain segment doing processing of a product and delivering it to end customer(s) (Figure 1). This structure is shown as a connected graph, whereas system nodes (solid circles) represent things (e.g. transfer carts, processing units and inventory etc.) within the systems and the environmental nodes (dashed circles) things (i.e. suppliers and customers) belonging to the environment. Similarly the corresponding couplings are also shown (solid and dashed lines). This system allows raw material procurement from a supplier present in the system's environment, and then passes it through several processing and transfer stages to end customer who is also present in system's environment. Note that all these system elements have properties e.g. *Final Inventory* have: units-on-hand, location etc. as its properties, the current value of which defines its current state, while all possible values of it form its complete state space.

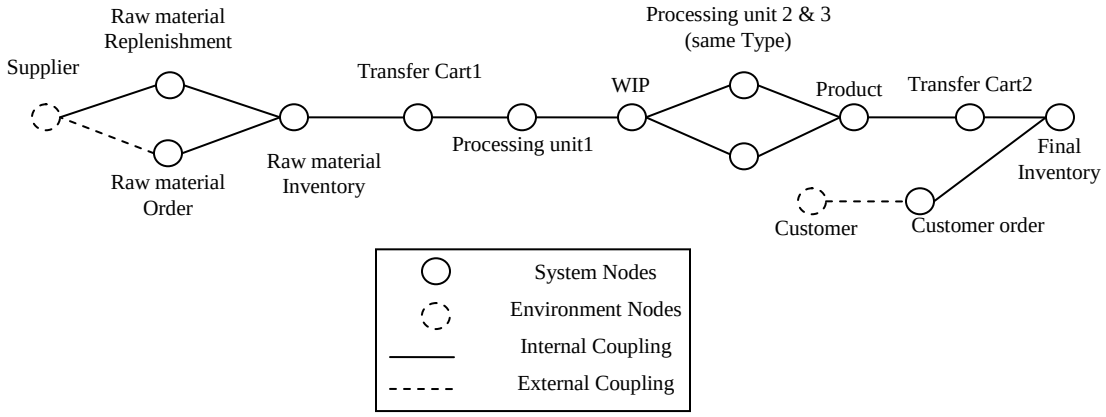

Figure1: A Product Processing System

Although definitions 1-2 are sufficient of describe a general stochastic system and its structure – its assembly of things through couplings within and in its related environment, it is still not sufficient to

express *system specific attributes* (i.e. perceived emergent properties) such as system operating/not operating, its failure etc. We thus present a *functional schema of a general stochastic system* σ as:

Definition 3: Let a general stochastic system be modeled by a functional schema as: $\sigma =_{def} \langle M'_\sigma, F'_\sigma \rangle$ where $M'_\sigma = \left[p(x \in C(\sigma))\ B_C\ \tau \right]$ and each component of function: $F'_\sigma = \langle F'_1, ..., F'_m \rangle : M'_\sigma \rightarrow V'_1 \otimes ... \otimes V'_m$, represent a property of σ at time $\tau \in M'_\sigma$, where $F'_j, 1 \leq j \leq m$ is the j^{th} deterministic/probabilistic state function (an attribute) of σ and F'_σ is the *total probabilistic state function* of σ. Also, $s'_L(\sigma) \in S'_L(\sigma)$ are the lawful states of system σ and $G'_L(\sigma, t) \subseteq S'_L(\sigma) \otimes S'_L(\sigma)$ the lawful system transformation.

To show the application of this schema, using Figure 1, an example set of attributes for the system is presented in Table 1 i.e. the product processing system has attributes Cycle Time, Cycle Variance, Reliability, Operating etc.

System	Attributes	State Variable
Supplier's system	Cycle Time	CycleTime
	Cycle Variance	CycleVar
	Reliability	Relb
	Operating	Oper
	Condition	Cond

Table 1: Supplier's System with Attributes

2.3. Modeling the Process:

Systems perform processes; to express the process of a general stochastic system σ, we refer to Soffer and Wand [15] who defined a process over a *domain* as a sequence of unstable states leading to a stable state. They defined *domain* as a part of the world under consideration, which is to be modeled and is made up of things and their interactions defined by a set of state variables. By this definition we assume a system and a domain to be equivalent. Thus a process performed by system σ can formally be defined as:

- **Definition 4a:** A *process V* is a sequence of unstable states of system σ leading to a stable state [15]; whereas a system state $s'_L(\sigma) \in S'_L(\sigma)$ is defined as stable with respect to the set of system transformations $G'_L(\sigma, t)$ *iff* $(\forall g' \in G'_L(\sigma, t)),\ g'(s'_L(\sigma)) = s'_L(\sigma)$.

Soffer and Wand [15] also imposed a stability condition that dictates a domain to always attain certain stable states, formed as a set Γ called *Process Goals*, within a finite time. This condition is justifiable

based on the well-established notion of Finite-Time-Stability (FTS) [25], which deals with systems that operate in a fixed interval of time – for such systems reaching a stable state in any time greater than a given time bound is considered meaningless (in terms of stability). For systems where time bound is irrelevant, any sufficiently large time bound can be used. Thus, without a loss of generality, we apply this condition on our system σ i.e.

- **Definition 4b:** Under Soffer and Wand [15] stability condition**,** every execution of process *V* of system σ must reach Γ in a finite time bound.

To explain the above condition in the context of the example presented in Figure 1, a certain supply order not fulfilled in a given time limit is meaningless even if it can be fulfilled after that. To fulfill the criterion of reaching a stable state within a time bound, a state has to be defined that is achieved if a time bound is reached. For example an order is considered as declined or failed, in the supplier's system, if a given time bound is reached. Besides the stability constraint, other set of constraints can be imposed on a process [2], which are defined as:

- **Definition 4b:** *Criterion Functions* that are a predicate on the set of stable states - $Q: S \rightarrow$ '$True$', '$False$' , where *Q* is '*True*' for (and only for) states in the goal set.

2.4. Failure and Reliability

For the general stochastic system σ and its corresponding process *V* defined in sections 2.2-2.3, there are various scenarios that need specialized treatment. We have considered one such aspect where a system faces *random failures* and consequently *reliability* issues. For this process specialization, we first define *system failure* (or success) in terms of process goal set Γ defined above:

- **Definition 5a:** A *Failure* (*FG*) of process *V* of σ is defined as $FG \subseteq \Gamma$ i.e. a failure is reached when an execution of the process is terminated in *FG* with a probability of P_{FG}. While a *Success* (*SG*) is defined as $SG \subseteq \Gamma$ i.e. a success is reached when an execution of the process is terminated in *SG* with a probability of P_{SG}. An implicit condition on *FG* and *SG* in relation to Γ is $FG \cup SG = \Gamma$ and $FG \cap SG = \varnothing$.

FG represents a set of states of system attributes which are considered as failed states (vice versa for *SG*); for example, a failed order processing (or a successful order processing) in the example stated in figure 1.

- **Corollary 1:** A process always terminates in *FG*, except when a stable state in Γ is reached by the process *V* in a time less than or equal to the impose time bound.

As the notion of system of process failure is linked mainly to failure of a component of a system (i.e. a thing), we also define *failure of a thing* as:

- **Definition 6:** A thing *X* fails if $S(X) \in S_F(X)$ where $S_F(X)$ is the set of states of a thing that defines *a failure* of a thing – a state in which *X* does not act on any other thing or in other words decouples itself with other things i.e. $X \not\triangleright Y \mid S(X) \in S_F(X)$. Similarly an *intact thing* would be $X \triangleright Y \mid S(X) \notin S_F(X)$.
- **Corollary 2:** A process *may* fail when a thing, belonging to the system, fails.
- **Corollary 3:** A process *fails* when a thing, belonging to the system, fails and the system does not remain a connected graph (no complete path exists for a process to succeed).
- **Corollary 4:** A system fails when process *V* fails i.e. it does not remain a connected graph.

For example, in figure 1, if Processing unit 1 fails the system does not remain a connected graph. However, if one of Processing unit 2 or 3 fails the system still remain a connected graph and process *V* may still succeed.

Finally, for systems that fail randomly, *reliability*, which is the probability of not failing by time *t*, becomes a crucial systems attribute. In fact, it is usually a design parameter (set as a criterion) that may dictate the form or configuration of systems of our interest (e.g. computer, business or industrial systems). To define reliability we refer to a thing failing by time $\bar{t}$; for this, $f_X(\bar{t})$ is the probability density function and the life distribution is $\alpha(\bar{t}) - \Pr(X \in S_F)$ by time *t*. By definition, Reliability of a thing is thus defined as $R_x(t) = 1 - \alpha_x(t)$ [24].

2.5. Process Model Validity

Finally we comment on the *validity* of the process model i.e. model serving its intended purpose or not. The validation rules are mainly based on the approach presented in GPM by Soffer and Wand [2, 15],

whereas the deviations in our approach mainly arise from our consideration of stochastic nature of the system. Thus to validate model of process *V*, we make use of four concepts i.e. a *valid process path*, its *reachability*, a *successful process path* and a *failed process path*, which are defined as follows:

- **Definition 7(a):** A *valid process path* is a path $< s_1,...,s_n >$ such that $s_n \in \Gamma$ and has *finite steps* reached within a *finite time bound*.
- **Definition 7(b):** A goal set Γ is *reachable* if at least one valid path exit that terminates in Γ i.e., $s_n \in \Gamma$.
  - **Corollary 5:** A *Successful Process Path* is a valid process path $\langle s_1,...,s_n \rangle$ reached within the imposed time bound, such that $s_n \in SG$
  - **Corollary 6:** A *Failed Process Path* is a valid process path $< s_1,...,s_n >$ such that $s_n \in FG$
- **Definition 7(c):** A process model is termed *enactment valid* if when executed (an instability introduced due to an external event) always terminates in Γ.

Here, the *sequence of states* in a process depends upon the internal and external events occurrences and the transition laws. Soffer and Wand [2] defines a *good process design* to be the one for which an enactment will complete in the goal set Γ. If the contrary results, it is termed as *process design failure* under expected external events, and *process enactment failure* under *unexpected external events*. Definition 7(b) takes care of both these issue. Note that the addition of time bound condition ensures satisfying this definition in case a stable state is not reached within an imposed time bound. In terms of goal set (Γ) reachability i.e. to ensure at least one path of process *V* exits that leads to Γ definition 7(b) is set [2]. The corollary to both the definitions are that we have either successful of failed *process paths*. It is important to note that these definitions are with respect to *given* goal set Γ, however these does not validate the goal set itself, which must be validated.

3. Mapping the Theory of Reliability Constructs

Theory of reliability is perhaps the most important theory dealing with failure prone systems. This theory helps estimate the reliability of a system based on the configuration of various of its components [24]. To employ SSPM models in the reliability analysis, we need a mapping between the constructs of the theory of reliability (TR) and the constructs of SSPM. To do this mapping, we consider a TR

component equivalent to a SSPM *thing* as defined earlier in section 2.1. Similarly, the *connection* between any two TR components represents a SSPM *coupling* (section 2.1). With this mapping, the reliability of such systems (modeled as σ) can now be computed based upon its configuration using the TR. Examples of some basic (mapped to SSPM) system configurations are shown in figure 2 below:

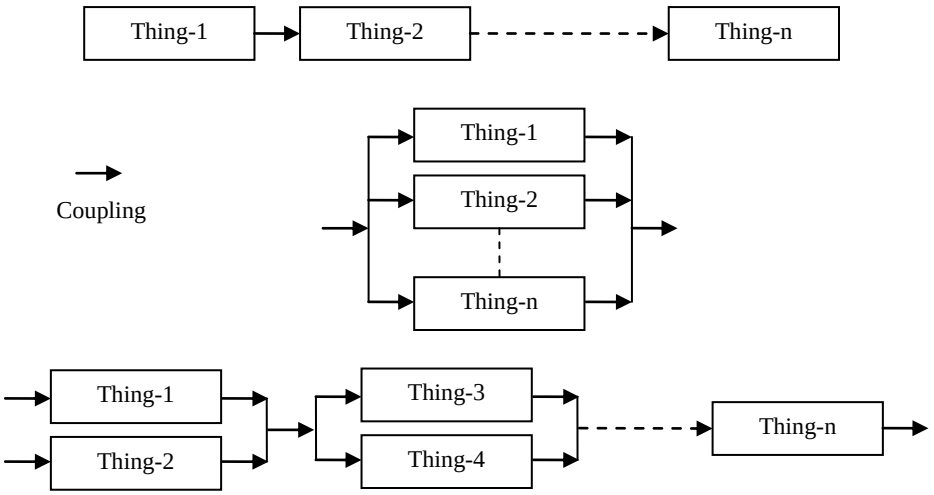

Figure 2: Common System configurations: (Top) Series, (Middle) Parallel & (Bottom) Hybrid

Thus estimation of reliability of a system, using SSPM can be given as follows. For a system having *n* things (i.e. TR components), functioning of an individual thing is a random variable defined as:

$$w_i = \begin{cases} 1 & \text{for } X_i \text{ when } S(X_i) \notin S_F(X_i) \\ 0 & \text{for } X_i \text{ when } S(X_i) \in S_F(X_i) \end{cases}$$

Now for $\forall X \in C(\sigma)$, vector $W_\sigma = (w_1, ..., w_n)$ is a state vector indicating functioning of all things in the composition of a system. The functioning of the systems is thus represented by a structure function as:

$$\phi(W_\sigma) = \begin{cases} 1, & \text{if } \sigma \text{ is functioning given } W_\sigma \\ 0, & \text{if } \sigma \text{ is failed given } W_\sigma \end{cases}$$

For each system configuration $\phi(W_\sigma)$ can be determined, where the probability of a system not failing (or failing) will be defined as a life distribution; in other words, reliability as a function of $\phi(W_\sigma)$ and can be given as: $\alpha[\phi(W_\sigma) = 0]$ *or* $R\{\phi(W_\sigma) = 1] = 1 - \alpha[\phi(W_\sigma) = 0]$.

Analysis of Example in Figure 1: Now consider the example presented in Figure 1, we apply the above mapping to calculate the system reliability. Here all things are considered to have a reliability of 1 except the processing units and the transfer carts having reliabilities given as (Table 2):

Things	Reliability
Processing unit1	R_{PU1}
Processing unit2	R_{PU2}
Processing unit3	R_{PU3}
Transfer Cart1	R_{TC1}
Transfer Cart2	R_{TC2}

Table 2: Things with Respective Reliability Attributes

A simplified representation of the system (from figure 1) can be given as in Figure 3:

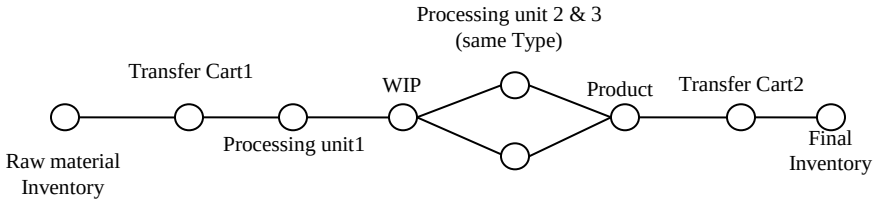

Figure 3: Simplified Version of Product Processing System Presented in Figure 1

Now in order to estimate the reliability of the whole system we have, $R[\phi(W_\sigma)=1]$ given $W_\sigma=(R_{PU1},R_{PU2},R_{PU3},R_{TC1},R_{TC3})$, which can be evaluated as follows:

1. For Processing units 2 & 3, if one of these keeps functioning the system functions. Therefore the probability of functioning of this sub-system will be the probability of at least one of the things functioning. Therefore the reliability will be:

$$R_{PU2,3}=1-\alpha[(X_7=0)\cap(X_8=0)]=1-\alpha(X_7=0)\alpha(X_8=0)$$
$$=1-(1-R_{PU2})(1-R_{PU3})$$

2. Considering $R_{PU2,3}$ (Reliability of Processing units 2 and 3 jointly as a single composite thing), all the things are now in series, which means that the system will fail even if one of the things in this series connection fails. Therefore the probability of the functioning of system will be the probability of all things functioning:

$$R[\phi(W_\sigma)=1] = \alpha[(X_6=1)\cap(X_{7,8}=1)\cap(X_9=1)\cap(X_{10}=1)]$$
$$= R_{PU1}R_{PU2,3}R_{TC1}R_{TC2}$$
Substituting $R_{PU2,3}$

We have: $R[\phi(W_\sigma)=1] = P_{SG} = R_{PU1}(1-(1-R_{PU2})(1-R_{PU3}))R_{TC1}R_{TC2}$

Now consider the case where there can be at most *n* parallel Processing units that can be installed instead of a configuration with Processing units 2 and 3 only, and the reliability of the rest of components is considered '1' i.e., $R_{PU1}=R_{TC1}=R_{TC2}=1$ a goal criterion can be:

Minimize: cost of installation

Subject to: reliability of system ≥ β.

The problem can therefore be defined as:

$$Let\ X_j = \begin{cases} 1 & \text{if a production unit } j \text{ is installed} \\ 0 & \text{otherwise} \end{cases} \qquad \text{where } j = 2,3,\ldots,n+1$$
and: C_j = Cost of installing production unit j

The goal criterion for the process model can thus be defined as:

$$\min \sum_j C_j X_j$$
Subject to:
$$1-(1-R_{PU2})(1-R_{PU3})\ldots(1-R_{PU(n+1)}) \geq \beta$$

4. Conclusions

In this paper, we suggested an ontology based systems and process modeling framework i.e. SSPM, which builds upon the Bunge-Wand-Weber [16] and the Generic-Process-Models [2, 15] frameworks. The intention is to provide a generic, notation independent and theory driven modeling method that can be employed to model processes that are stochastic and failure prone in nature. As the framework is generic in nature, it may be applied to the modeling of a wide variety of complex engineering systems as well as other business, computer and industrial systems and processes. Furthermore, as SSPM is a theory driven framework, other process modeling languages as well related theories can be mapped or compared to with this generic framework and their validity evaluated regardless of specific notations.

5. References

[1] B. Curtis, M.I. Kellner, J. Over, Process modeling, Commun. ACM, 35 (1992) 75-90.

[2] P. Soffer, Y. Wand, Goal-Driven Multi-Process Analysis, Journal of the Association for Information Systems, 8 (2007) 175-203.

[3] G.M. Giaglis, A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques, International Journal of Flexible Manufacturing Systems, 13 (2001) 209-228.

[4] T. Murata, Petri nets: Properties, Analysis and Applications, Proc IEEE, 77 (1989) 541-580.

[5] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, Modelling with Generalized Stochastic Petri Nets, 1998.

[6] M. Hammer, J. Champy, Reengineering the Corporation: A Manifesto for Business Revolution, Internet Delivered Business Book Summaries - summaries.com, (1994).

[7] W.M. van der Aalst, Business Process Execution Language, Encyclopedia of Database Systems, Springer2009, pp. 288-289.

[8] J. Sterman, J.D. Sterman, Business Dynamics: Systems Thinking and Modeling for a Complex World with CD-ROM, 1 ed., McGraw-Hill/Irwin2000.

[9] U.M.L. Omg, Unified Modeling Language (UML), Version 2.1.2, 2009.

[10] V. Anaya, G. Berio, M. Harzallah, P. Heymans, R. Matulevičius, A.L. Opdahl, H. Panetto, M.J. Verdecho, The unified enterprise modelling language—Overview and further work, Computers in Industry, 61 (2010) 99-111.

[11] J. Recker, M. Rosemann, M. Indulska, P. Green, Business process modeling-a comparative analysis, Journal of the Association for Information Systems, 10 (2009) 333-363.

[12] L. Aldin, S. de Cesare, A literature review on business process modelling: new frontiers of reusability, Enterprise Information Systems, 5 (2011) 359-383.

[13] I.M.-M.d. Oca, M. Snoeck, H.A. Reijers, A. Rodríguez-Morffi, A systematic literature review of studies on business process modeling quality, Information and Software Technology, (2014).

[14] L. García-Borgoñon, M. Barcelona, J. García-García, M. Alba, M.J. Escalona, Software process modeling languages: A systematic literature review, Information and Software Technology, 56 (2014) 103-116.

[15] P. Soffer, Y. Wand, Goal-Driven Analysis of Process Model Validity, Advanced Information Systems Engineering, (2004) 521-535.

[16] Y. Wand, R. Weber, An Ontological Model of an Information System, Software Engineering, IEEE Transactions on, 16 (1990) 1282-1292.
[17] Y. Wand, R. Weber, An Ontological Evaluation of Systems Analysis and Design, in: E.D. Falkenberg, P. Lindgreen (Eds.) Information System Concepts: An In-depth Analysis, North-Holland, Amsterdam, 1989, pp. 79-107.
[18] P. Green, M. Rosemann, Integrated Process Modeling: An Ontological Evaluation, Information Systems, 25 (2000) 73-87.
[19] Y. Wand, R. Weber, On the Ontological Expressiveness of Information Systems Analysis and Design Grammars, Information Systems Journal, 3 (1993) 217-237.
[20] M. Bunge, Treatise on Basic Philosophy: Ontology I, Ontology I: The Furniture of the World ed., Reidel, Boston, 1977.
[21] J. Parsons, Y. Wand, Using Objects for Systems Analysis, Commun ACM, 40 (1997) 104.
[22] A. Dussart, B. Aubert, M. Patry, An Evaluation of Inter-Organizational Workflow Modelling Formalisms, Journal of database management, 15 (2004) 74-104.
[23] M. Bunge, Treatise on Basic Philosophy: Ontology II, Ontology II: A world of Systems ed., Reidel, Boston, 1979.
[24] P. Tobas, D. Trindade, Applied reliability, 2nd ed., Van Nostrand Reinhold, New York, 1995.
[25] L. Menini, L. Zaccarian, C.T. Abdallah, An Overview of Finite-Time Stability, Current Trends in Nonlinear Systems and Control, Birkhauser Boston2006, pp. 185-195.