A Probabilistic BPMN Normal Form to Model and Advise Human Activities

Hector G. Ceballos[1], Victor Flores-Solorio[1], and Juan Pablo Garcia[2]

[1] Tecnologico de Monterrey, Campus Monterrey, Mexico
[2] Universidad Autonoma de Baja California, Mexico
ceballos@itesm.mx, vmfsolorio@gmail.com, pablo.garcia@uabc.edu.mx

Abstract. Agent-based technologies, originally proposed with the aim of assisting human activities, have been recently adopted in industry for automating business processes. Business Process Model and Notation (BPMN) is a standard notation for modeling business processes, that provides a rich graphical representation that can be used for common understanding of processes but also for automation purposes. We propose a normal form of Business Process Diagrams based on Activity Theory that can be transformed to a Causal Bayesian Network, which in turn can be used to model the behavior of activity participants and assess human decision through user agents. We illustrate our approach on an Elderly health care scenario obtained from an actual contextual study.

Keywords: BPMN, Agent-Based Systems Engineering, Bayesian Networks, Activity Theory.

1 Introduction

BPMN is a standard notation for modeling business processes that provides a rich graphical representation that can be used for common understanding of processes [13]. Furthermore, BPMN has been used for process automation with support of agent technologies [10].

BPMN uses gateways for representing decisions, which are usually labeled with textual descriptions indicating the criterion followed. These decisions are based on information that is available at the moment of decision making and may refer to information of the process in course or to historical information (*data-based decisions*).

But when the BPMN workflow describes a human activity in terms of user tasks this decision criterion might be unknown or inaccessible to the modeler, e.g. the buying decision of a customer. For dealing with the uncertainty introduced by human intervention, approaches like [6] have proposed annotating edges with the probability of each alternative. Nevertheless, this approach does not permit to determine if the cause of such variability comes from some part of the process under the control of some participant, i.e. capture causal relationships between

non-consecutive nodes. And despite BPMN has been recently used for agent-based software engineering, decision making under uncertainty has not been addressed in current approaches [3, 7, 12, 10].

For these reasons, we propose a normal form of BPMN Process Diagrams for modeling human activities suitable for generating a probabilistic representation of activity's dynamic suitable for discovering causal relationships. Possible scenarios specified in the BPMN workflow can be used for predicting the behavior of human participants based on observable events. The BPMN normal form is inspired by Activity Theory [4], providing goal-oriented BPMN Process Diagrams capable of representing collective human activities.

This paper is organized as follows. In section 2, we present other applications of BPMN for agent-based software engineering and introduce probabilistic formalisms traditionally used for agent decision making. In section 3 we discuss the pertinence of using BPMN for modeling human activities and propose a BPMN normal form suitable for its transformation to a Bayesian Network. We provide an automatic transformation procedure that produces a probabilistic representation of activity's dynamics that can be used for agent decision making based on previous activity developments. In section 4, we present other probabilistic approaches to BPMN and compare our selection of BPMN elements with other agent engineering approaches. Finally, in section 5, we present our conclusions and future work.

2 Background

We revise current applications of BPMN for agent-based system engineering, and review probabilistic graphic models used for decision making.

2.1 Business Process Diagrams for Agent Engineering

Business Process Model and Notation (BPMN) is a standard notation used by organizations for understanding internal business procedures in a graphical notation. Due to its expressivity and its growing adoption by industry, it has been also used as a tool for modeling MultiAgent Systems [3, 7, 12, 10].

Endert et al [3] proposed a mapping of Business Process Diagram (BPD) elements to agent concepts. In particular they considered a BPMN fragment constituted by: event nodes (start, intermediate and end), activity nodes, subprocess nodes, split and merge gateways (XOR, OR, AND), and pools. They map each pool to an agent and the process itself constitutes a plan; properties of start (and end) events constitute inputs (respectively outputs) for the plan. Independent subprocesses are mapped to goals and embedded subprocesses are mapped to plans. Activity nodes are represented by plan operations, whereas control flows are mapped to sequences, if-else blocks and loops. Data flow, i.e. arguments passed to messages and operations, is captured in node attributes and it is used for modeling agent beliefs.

Hinge and colleagues developed a tool for annotating BPMN in order to provide a semantic description of events and actions [7]. Actions are described by their direct effects: the observable conditions that hold immediately after action execution. These annotations are used for calculating the current development of a process, this is, determining which events and actions have occurred by observing the accumulation of effects on a knowledge base. The knowledge base is considered non-monotonic as long as this approach counts with a procedure for detecting the removal of facts.

Muehlen and Indulska evaluated the combination of modeling languages for business processes and business rules [12]. Their overlap analysis look for the minimization of redundancy on constructors and the maximization of modeling expressivity. Modeling constructors were grouped in four categories: sort of things, states, events and systems. They conclude that the highest representation power is given by the combination of BPMN for representing the business process and SWRL [8] for representing business rules. Nevertheless, their analysis reveals that this combination, despite it is the most complete, lacks of a representation of states.

Jander and colleagues proposed Goal-oriented Process Modeling Notation (GPMN), a language for developing goal-oriented workflows [9]. The process is initially modeled by decomposing a main goal into subgoals, and then each subgoal is linked to a BPMN diagram that represents the plan to achieve that goal. This graphical language includes *activation plans* which decide subgoal parallelization or serialization, replicating the functionality of gateways in the goal hierarchy tree. A goal can be connected to multiple plans, enabling means-end reasoning.

Finally, Kuster and colleagues provide a full methodology for process oriented agent engineering that complements BPMN process diagrams with: declaration of data types (ontology engineering), a model for agent organization and distribution, low-level algorithms for activity nodes (service engineering), and use cases diagrams that link roles and process diagrams [10]. This framework implements the mapping of BPMN to agents described in [3] for agent engineering.

These approaches show how BPMN workflows can be used for designing agent specifications from the description of their interactions in a process. Nevertheless they assume that all the information needed by agent for making a decision is available, which in turn produces reactive agent specifications but is not sufficient for coping with uncertainty.

2.2 Decision Making based on Bayesian Networks

Bayesian Networks (BNs) have been used for quite a while for representing decision making under uncertainty and learning through observation/experience. BNs are suitable for identifying causal dependencies between random variables representing events and actions occurred at different time steps. Despite Markovian Decision Processes (MDPs) have gained popularity for their capacity for providing efficient probabilistic inference in long term processes, their represen-

tation lacks of memory, i.e. it only captures conditional dependencies between contiguous time steps.

Bayesian Networks. A *Bayesian Network* is a probabilistic graphical model that represents a set of events denoted by random variables, and their conditional dependencies via a directed acyclic graph (DAG), denoted:

$$M = \langle V, G_V, P(v_i|pa_i)\rangle$$

where V is a set of random variables, G_V is the graph consisting of variables in V and directed arcs between them, and $P(v_i|pa_i)$ is a conditional probabilistic distribution where the probability of v_i depends on the value of its parents (pa_i) in G_V.

A random variable V_i is a numerical description of the outcome of an experiment, and can be either discrete or continuous. The set of possible values a discrete random variable may hold, or *domain* $Dom(V_i) = \{v_{i1}, ..., v_{in}\}$, represents the possible outcomes of a yet-to-be-performed experiment, or the potential values of a quantity whose already-existing value is uncertain.

The *realization* of a random variable V_i to the value $v_{ij} \in Dom(V_i)$ is represented as $V_i = v_{ij}$, or v_i if the realization value is not relevant in a given context.

Random variables satisfy the probability theory requisite which dictates that in an experiment, a random variable can be realized to a single value of its domain. This means that all events represented by a random variable are disjoint.

G_V is an independence map (I-Map), i.e. a minimal graph where the presence of an arc from V_i to V_j indicates conditional dependence whereas its absence indicates conditional independence. An I-Map is called minimal because indirect dependencies are not included.

Bayesian Networks can be modeled from two distinct perspectives: evidential or causal. If arrows go from effects to causes the perspective is *evidential*, e.g. determining the disease based on the patient symptoms. A network is modeled from a *causal* perspective if arcs go from causes to effects. For instance, in *Dynamic Bayesian Networks* a different set of random variables represents the state of the system at time t, $t+1$, ... , $t+n$; arcs can go from a variable in t to another in $t+i$, but the opposite is not allowed.

Bayesian networks are used to find out updated knowledge of the state of a subset of variables $\bar{v}_1$ when another variables $\bar{v}_2$ (evidence) are observed, denoted $P(\bar{v}_1|\bar{v}_2)$. This process of computing the posterior distribution of variables given certain evidence is called *probabilistic inference*.

Influence Diagrams. An *Influence Diagram* is a generalization of a Bayesian Network devised for modeling and solving decision problems using probabilistic inference. Nodes may represent decisions (rectangles), uncertain conditions (ovals) or the utility obtained in a given scenario (diamond).

Arcs ending in decision nodes denote the information taken into account for making the decision. Arcs between uncertain nodes propagate uncertainty or information like in Bayesian Networks.

Decision nodes and their incoming arcs determine the *alternatives*. Uncertainty nodes and their incoming arcs model the *information*. Value nodes and their incoming arcs quantify the *preference* on the outcome. An alternative is chosen based on the maximum expected utility in the given scenario, calculated by the a posteriori probability of all nodes (including unknown values).

Causal Bayesian Networks. Judea Pearl introduced the notion of Causality on Bayesian Networks under the concept of *intervention*, where the value of a variable could be subject to alteration through a mechanism F or let its value being freely set [15]. Pearl and Robins proposed that nodes in a Bayesian network can be classified into purely observable variables, or *Covariates* (Z), and controllable variables (X) which are subject to intervention, denoted by $do(x_i)$ [16]. From this distinction they establish a graphical method for identifying the set of covariates ($W_k \subset Z$) that must be observed for determining the *causal effect* of a sequence of interventions $do(x_1), ..., do(x_k)$ on Y, i.e. $P(y|do(x_1), ..., do(x_k), w_k)$. This sequence of interventions constitutes a plan, which probability of success can be evaluated a priori and be revised once that the network is updated with information.

3 Probabilistic Decision Making on Business Process Diagrams

An Activity of Daily Living (ADL) modeled as a BPMN workflow is used for illustrating the proposed normal form. Then a procedure for transforming this workflow to a Bayesian Network is provided and some examples of probabilistic inference are given to validate the model.

The subset of graphical elements of the BPMN 2.0 specification [13] we use in our example and in our normal form is shown in Figure 1. BPMN Business Process Diagrams (BPDs) basically describe a process in terms of *events* and *actions* connected through *control flows* that indicate valid sequences in the process development. *Gateways* are special nodes connected through control flows that indicate whether the process develops in parallel (AND), alternatively (XOR) or optionally (OR). The beginning of the process is denoted by an *initial event* node and its conclusion by a set of *end event* nodes.

3.1 An example of an ADL modeled in BPMN

We motivate the discussion using as example the medical consultation of an elder person, taken from an actual contextual study based on Activity Theory [5]. In this activity, the *subject* is an older adult who has a medical appointment (the *object*). The *objective* of the activity is having a medical appraisal and its

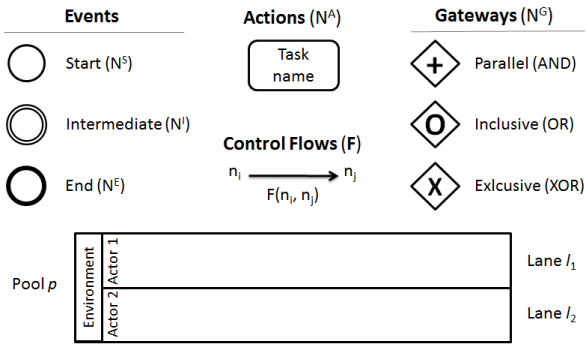


Fig. 1. BPMN graphical notation.

outcome includes getting a prescription, supply medicines and schedule a next appointment. The *community* involved in the activity includes a family member (optionally) and the doctor.

This diagram is used for compensating the lack of a formal representation of the activity's dynamic in Activity Theory (AT) [4]. At some extent, control flows and gateways formalize the set of rules specified in the AT specification.

Figure 2 shows the activity diagram modeled with BPMN. It illustrates two alternative ways the elder may choose for getting to the hospital: going by himself, or being carried out by a family member. It also shows five possible outcomes for the activity: 1) treatment finished, 2) taking new medication, 3) taking medication and follow up, 4) medication not available at the hospital's pharmacy, and 5) missing the appointment (failure outcome).

3.2 A Probabilistic BPMN normal form

The proposed normal form has the purpose of illustrating alternative sequences of actions performed by activity participants, mediated by intermediate events that the subject or other participants can observe. XOR gateways are used for representing disjoint alternatives. Activity's development has a triggering condition (initial event) and a set of successful or failure outcomes (end events). The resulting graph must be acyclic for facilitating its translation to a Bayesian Network through a graphical procedure. A BPMN BPD satisfies the probabilistic normal form if it observes the following constraints:

1. A Business Process Diagram $\mathbf{W}$ is represented by a set of pools ($\mathbf{P}$), lanes ($\mathbf{L}$), nodes ($\mathbf{N}$) and control flows ($\mathbf{F}$).

$$\mathbf{W} = \{\mathbf{P}, \mathbf{L}, \mathbf{N}, \mathbf{F}\}$$

2. Nodes ($\mathbf{N}$) allowed in the diagram are: start events (N^S), intermediate events (N^I), end events (N^E), atomic actions (N^A) and gateways (N^G).

$$\mathbf{N} = N^S \cup N^I \cup N^E \cup N^A \cup N^G$$

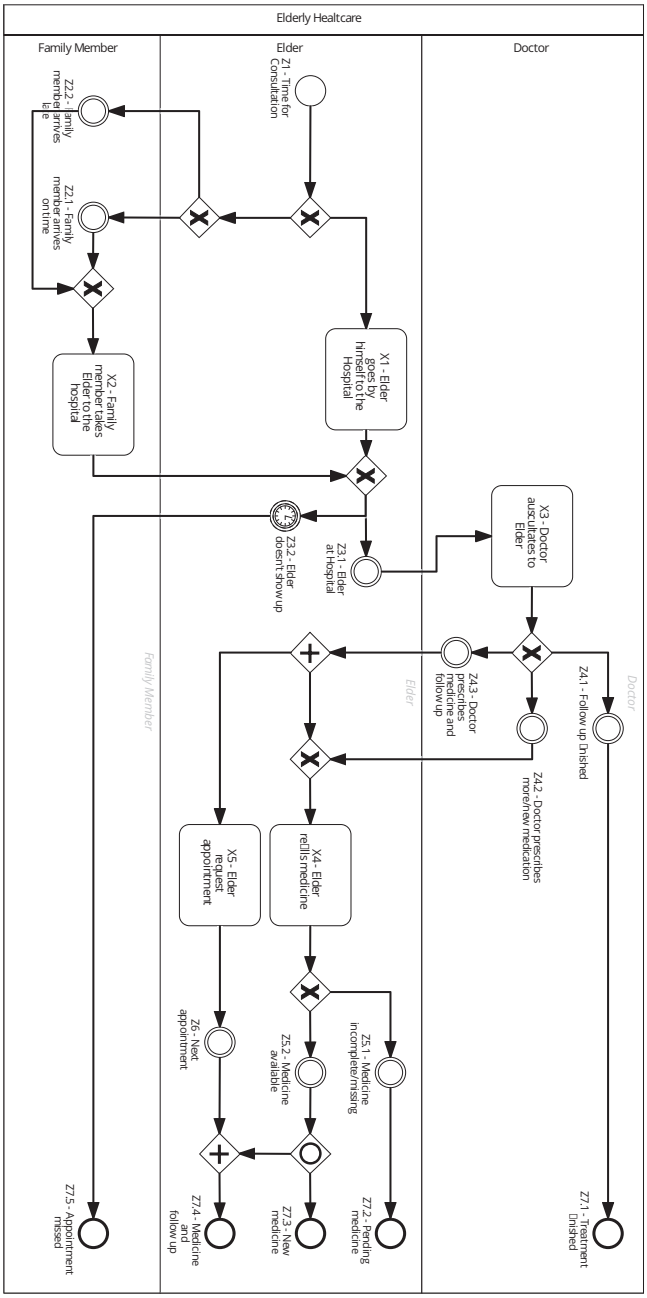


Fig. 2. Business Process Diagram of the Medical Consultation Activity.

3. The diagram must have a single pool ($p \in \mathbf{P}$) containing at least one lane ($l_i \in \mathbf{L}, i \geq 1$). Each lane represents a human participant in the activity, and nodes must be allocated in a single lane ($in(n, l_i), n \in \mathbf{N}$).
4. All sequence flows are unconditional, denoted as $F(n_i, n_j) \in \mathbf{F}$ where $n_i, n_j \in N$. Conditional or default control flows are not allowed; instead, intermediate event nodes are used for representing both data-based and event-based control flow.
5. A single start event $s \in N^S$ is defined ($|N^S| = 1$), given that the activity is modeled from the perspective of a single individual (the subject), and it must be labeled with the condition perceived by the subject that triggers activity's development.
6. Intermediate event nodes ($i \in N^I$) are labeled with a natural language description that corresponds to the condition (partial world state) that must hold for proceeding with the activity's course.
7. Similarly, atomic action nodes or tasks ($a \in N^A$) are labeled with a verb expressed in active voice that denotes the action performed by a participant, indicating other actors involved in collective actions, as well as required artifacts and locations.
8. Two consecutive action nodes must be mediated by at least one intermediate event node and as many gateways as needed, i.e. two action nodes are not connected directly through sequence flows. Observable intermediate events will permit monitoring the activity development and introducing agent assistance [2].

$$\forall a \in N^A, (F(n, a) \in \mathbf{F} \vee F(a, n) \in \mathbf{F}) \rightarrow n \notin N^A$$

9. Each split or merge of control flows must be mediated by a splitting gateway ($N^G_S \subseteq N^G$) or a merging gateway ($N^G_M \subset N^G$), respectively. Gateways can be of type Parallel-AND (A), Optional-OR (O), or Exclusive-XOR (X). Gateways with both multiple incoming and outgoing flows are not permitted.

$$\forall g \in N^G, type(g, t) \rightarrow t \in \{A, O, X\}$$

10. Splitting gateways XOR ($g \in N^G_S, type(g, X)$) must be followed by intermediate event nodes ($F(g, i) \in \mathbf{F}, i \in N^I$) or other XOR gateways, denoting alternative ways on which the activity can develop. Event node labels indicate the reason for selecting each alternative.
11. The diagram might have multiple end nodes, but two end nodes cannot represent the same outcome; their labels must reflect some difference. Control flows and gateways must be used for connecting all possible workflows ending in the same outcome.

$$\forall e_1, e_2 \in N^E \rightarrow Label(e_1) \not\equiv Label(e_2)$$

12. The graph G_N constituted by all $F(n_i, n_j) \in \mathbf{F}$ must not have any directed cycle or loop, i.e. it must be a Directed Acyclic Graph (DAG).
13. All other nodes, gateways and control flows are disallowed in the diagram. BPMN artifacts (associations, groups and text annotations) are ignored.

3.3 Translating BPDs to Bayesian Networks

Next we describe the rules and the procedure used for translating a BPD satisfying the previous normal form to a Bayesian Network. In short, events and actions are mapped to observable and controllable random variables, respectively, whereas control flows and gateways are used for building the conditional dependency graph and the probabilistic distribution of the model.

Events. Events represent partial world states in the activity context, hence their representation is associated to observable random variables (Z_i), whereas their occurrence is represented probabilistically by the realization of these variables ($Z_i = z_i$).

The *start event* is detected by the activity subject and it is represented by the boolean variable Z_S, which realization to $True$ holds on any process execution. Z_S has no parents and it is used for start process monitoring. In our example, the start event is the doctor's appointment time ($Z_S = Z_1$).

$$s \in N^S \rightarrow define(Z_S), Dom(Z_S) = \{True, False\}, map(s, Z_S = True) \quad (1)$$

The function $define(V_i)$ is used for declaring random variables, whereas the function $map(n, V_i = v_i)$, $n \in \mathbf{N}$, establishes the correspondence between elements of both representations.

A BPD might include multiple *end nodes* as shown in our example. Given that each end node corresponds to different outcomes of the activity, all of them are represented by a single random variable Z_E. Each outcome node e represents a possible realization of Z_E. In our example Z_7 represents Z_E, with $Dom(Z_7) = \{7.1, 7.2, 7.3, 7.4, 7.5\}$.

$$\forall e \in N^E \rightarrow e \in Dom(Z_E), map(e, Z_E = e) \quad (2)$$

Intermediate event nodes are used in the BPD for two reasons: 1) observing the evidence of actions performed by people in the real world (event-based control flow), and 2) controlling the workflow based on data produced during process execution (data-based control flow). Additionally to *generic intermediate event nodes* that can be expressed with expressions in First Order Logic, *timeout nodes* are introduced for representing temporal reasoning for process monitoring.

Intermediate event nodes are classified as subgoals or alternative events. *Subgoal events* are event nodes that must be performed in order to continue with process execution in a given workflow. A subgoal event is represented by a boolean random variable, where its realization to $True$ indicates that the condition/event was met and $False$ if it did not occurred during process execution. The node representing the scheduling of the *Next appointment* (Z_6) is an example of a *subgoal event*.

$$\forall i \in N^I, F(n, i) \in \mathbf{F}, (n \notin N^G \wedge (n \in N^G, \neg type(n, X)))) \rightarrow \quad (3a)$$

$$define(Z_i), Dom(Z_i) = \{True, False\}, map(i, Z_i = True) \quad (3b)$$

Alternative events are mutually exclusive world states denoted by intermediate event nodes preceded by a XOR gateway, and are represented by a single observable random variable. We define the set *Alt* for identifying these gateways in further steps of the transformation. For instance, the events *Follow up finished* ($Z_{4.1}$), *Doctor prescribes more/new medication* ($Z_{4.2}$), and *Doctor prescribes medicine and follow up* ($Z_{4.3}$), are represented by the random variable Z_4. Successor intermediate events mediated exclusively by XOR gateways are included in the set of disjoint events as well (see 4c–4e).

$$\forall g \in N_S^G, type(g, X), F(g, i) \in \mathbf{F}, i \in N^I \rightarrow \tag{4a}$$
$$define(Z_g), i \in Dom(Z_g), map(i, Z_g = i), g \in Alt \tag{4b}$$
$$\forall g \in N_S^G, type(g, X), F(g, g_1) \in \mathbf{F}, g_1 \in N^G, type(g_1, X), ..., \tag{4c}$$
$$F(g_{k-1}, g_k) \in \mathbf{F}, g_k \in N^G, type(g_k, X), F(g_k, i) \in \mathbf{F}, i \in N^I \rightarrow \tag{4d}$$
$$define(Z_g), i \in Dom(Z_g), map(i, Z_g = i), g \in Alt \tag{4e}$$

Observable random variables Z_i representing intermediate events are considered *mandatory* if Z_i is included in all paths connecting the start variable Z_S with the end variable Z_E. And it is considered *optional* if other alternative paths exist that connect Z_S and Z_E that do not pass through it. All optional random variables include the value $False$ in their domain for considering those cases where the process develops through an alternative path. This rule is applied after having obtained the conditional dependence graph G_V as explained next. Z_3 is an example of a mandatory variable with $Dom(Z_3) = \{3.1, 3.2\}$, whereas Z_4 is optional given the alternative path through $Z_{3.2}$, making $Dom(Z_4) = \{4.1, 4.2, 4.3, False\}$.

$$\exists p_i = path(Z_S, Z_E) \in G_V, Z_j \notin p_i \rightarrow False \in Dom(Z_j) \tag{5}$$

Actions. Action nodes in BPDs might represent atomic actions or subprocesses. In this analysis we only consider *atomic actions*, which correspond to the definition of action given by Leontiev [11], i.e. something that the person makes consciously to achieve a goal. This action might require the participation of other actors, like in the auscultation made by the doctor to the elder (X_3), or be performed individually, like when the elder going by himself to the hospital (X_1).

Similarly to subgoal events, *atomic actions* are represented by boolean random variables, denoted X_i, where the value $True$ denotes the execution of the action, and $False$ represents its omission. If the action is not performed, the value of the variable is set to $False$ at the end of activity's monitoring.

$$\forall a \in N^A \rightarrow define(X_a), Dom(X_a) = \{True, False\}, map(a, X_a = True) \tag{6}$$

Control Flows. Control flows encode necessary conditions for the development of a process, this is, the occurrence of previous events or actions enables event

observation or action execution. For instance, medical consultation (X_3) requires the patient being at the hospital ($Z_3 = 3.1$), and the next appointment (Z_6) requires that the elder had request it (X_5).

A control flow $V_i \rightarrow V_j$ indicates: 1) temporal precedence of the action/event V_i with respect to another action/event V_j, and 2) conditional dependence of V_j on V_i. For this reason, the equivalent representation of the BPD is a Bayesian Network modeled from a causal perspective.

In order to identify conditional dependencies between events and actions, we use control flows incoming and outgoing to their corresponding random variables. A copy of the DAG constructed with these control flows, denoted $G'_N : N \times N$, is modified according to rules (7a) – (7d) in Figure 3 for removing unnecessary gateways and unifying end nodes in a single one. In these rule we use graph operations such as adding/removing arcs and absorbing nodes. Absorbing n consists on adding control flows $F(n_i, n_j)$ for the cross product given by every pair $F(n_i, n) - F(n, n_j)$, and then removing the node n and those arcs connected to it.

$$\forall i \in N^I, F(g,i) \in G'_N, g \in Alt \rightarrow absorbe(i, G'_N) \quad (7a)$$

$$\forall g \in N^G_M \rightarrow absorbe(g, G'_N) \quad (7b)$$

$$\forall g \in N^G_S, g \notin Alt \rightarrow absorbe(g, G'_N) \quad (7c)$$

$$\forall e_i \in N^E, i > 1, F(n, e_i) \in G'_N \rightarrow remove(F(n, e_i), G'_N), add(F(n, e_1), G'_N) \quad (7d)$$

Fig. 3. Transformation of G_N to G'_n.

The resulting DAG G'_N and those mappings generated in the first stage of the process are used for defining the arcs that constitute the conditional dependence graph between random variables $G_V : V \times V$.

$$\forall F(n_i, n_j) \in G'_N, map(n_i, V_i = v_i), map(n_j, V_j = v_j) \rightarrow add(Arc(V_i, V_j), G_V) \quad (8)$$

At this point, the conditional dependence graph G_V of the medical consultation activity is shown in Figure 4. Random variables labeled Z_i represent observable variables, whereas X_i denote atomic actions. Note that alternative event nodes are grouped in random variables Z_2, Z_3, Z_4 and Z_5.

Gateways. Gateways, on the other hand, codify how likely is that two or more events/actions occur during process execution, which corresponds to the definition of the Conditional Probabilistic Distribution (CPD), i.e. $P(v_i|pa_i)$.

The different process developments (scenarios) that can be generated according to gateway constraints provide the joint probabilistic distribution of the process. This distribution assumes that all scenarios are equally likely and it is used for learning the CPDs of random variables using the dependencies given

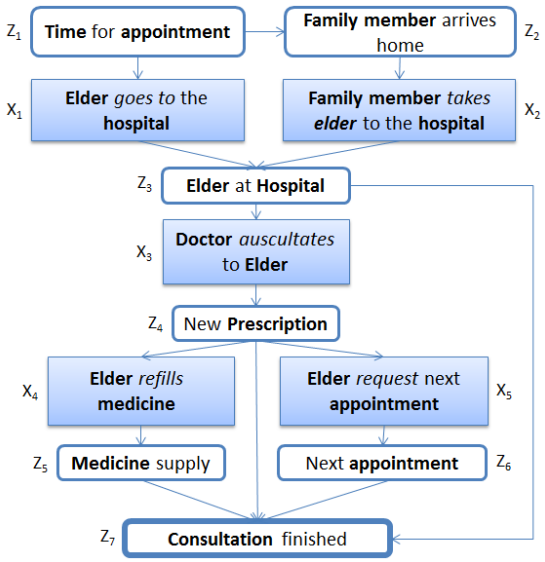


Fig. 4. The Medical Consultation Activity's conditional dependence graph

Z1	X1	Z2	X2	Z3	X3	Z4	X4	Z5	X5	Z6	Z7
T	F	2.1	T	3.1	T	4.1	F	F	F	F	7.1
T	F	2.1	T	3.1	T	4.2	T	5.1	F	F	7.2
T	F	2.1	T	3.1	T	4.2	T	5.2	F	F	7.3
T	F	2.1	T	3.1	T	4.3	T	5.2	T	T	7.4
T	F	2.1	T	3.2	F	F	F	F	F	F	7.5
T	F	2.2	T	3.1	T	4.1	F	F	F	F	7.1
T	F	2.2	T	3.1	T	4.2	T	5.1	F	F	7.2
T	F	2.2	T	3.1	T	4.2	T	5.2	F	F	7.3
T	F	2.2	T	3.1	T	4.3	T	5.2	T	T	7.4
T	F	2.2	T	3.2	F	F	F	F	F	F	7.5
T	T	F	F	3.1	T	4.1	F	F	F	F	7.1
T	T	F	F	3.1	T	4.2	T	5.1	F	F	7.2
T	T	F	F	3.1	T	4.2	T	5.2	F	F	7.3
T	T	F	F	3.1	T	4.3	T	5.2	T	T	7.4
T	T	F	F	3.2	F	F	F	F	F	F	7.5

Fig. 5. Valid process developments.

by the graph in Figure 4. Figure 5 shows the 15 scenarios that can be generated from the process in Figure 2, where columns indicate the realization of random variables in each scenario.

Table 1 shows the structures supported by our normal form, aligned with the corresponding transformation rules. The column Mappings shows the correspondence between BPD nodes and random variables, indicating the rule applied, and the last column indicates which nodes prevail in the reduced graph G'_N, indicating the rule that makes the reduction.

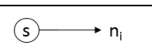
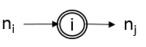
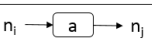
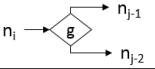
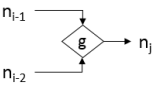
Structure			Constraint	Mappings	In G'_N
Trigger			$n_i \in \boldsymbol{N} \setminus N^S$	map(s, Z_s=True) (1)	s
Outcome			$n_i \in \boldsymbol{N} \setminus N^E$	map(e_i, Z_E=e_i) (2)	e_1 (7d)
Event			*i.* $n_i \notin N^G$ *ii.* $n_i \in N^G \wedge \neg type(n_i, X)$	map(i, Z_i=True) (3)	i
Actions			$n_i, n_j \in \boldsymbol{N} \setminus N^A$	map(a, X_a=True) (6)	a
Split gateways	Decision		$type(g, X), n_i \in \boldsymbol{N}$, *i.* $n_j \in N^I$ *ii.* $n_j \in N^G \wedge type(n_j, X)$	map(i, Z_g=i) (4)	g (7a)
	Alternative		$type(g, A) \vee type(g, O)$ $n_i, n_j \in \boldsymbol{N}$	n_i, n_j (see above)	n_i, n_j (7c)
Merge gateways			$n_i, n_j \in \boldsymbol{N}$	n_i, n_j (see above)	n_i, n_j (7b)

Table 1. Valid structures in the BPD normal form.

3.4 The Activity Causal Bayesian Network

The Bayesian Network produced by the transformation process described above is defined as follows.

Definition 1. *An Activity Causal Bayesian Network (ACBN) is represented by*

$$D = \langle G_V, X, Z, Z_S, Z_E, P(v_i|pa_i) \rangle$$

where G_V is a minimal DAG which arcs denote temporal precedence and conditional dependence between observable events (Z) and actions (X), $P(v_i|pa_i)$ encodes conditional probabilistic dependencies between random variables $V = Z \cup X$, and G_V has at least one directed path from the initial condition $Z_S \in Z$ to the outcome variable $Z_E \in (Z \setminus Z_S)$.

The Causal Bayesian Network of the activity modeled in Figure 2 has seven observable conditions or events (Z_1 – Z_7) and five human actions (X_1 – X_5). The initial condition is the appointment time (Z_1) and the outcome variable is Z_7. Its graph G_V is shown in Figure 4 and the corresponding $P(v_i|pa_i)$ is learned from the process instances shown in Figure 5.

Probabilistic inference. Figure 6 shows an example of probabilistic inference for a partially observed activity instance where the observed evidence (e) is: the family member arrived late to elder's house ($Z_2 = 2.2$), the elder arrived to the hospital on time ($Z_3 = 3.1$), and the doctor prescribed new medication only ($Z_4 = 4.2$). Posterior probabilities for the other variables are shown in Fig. 6.

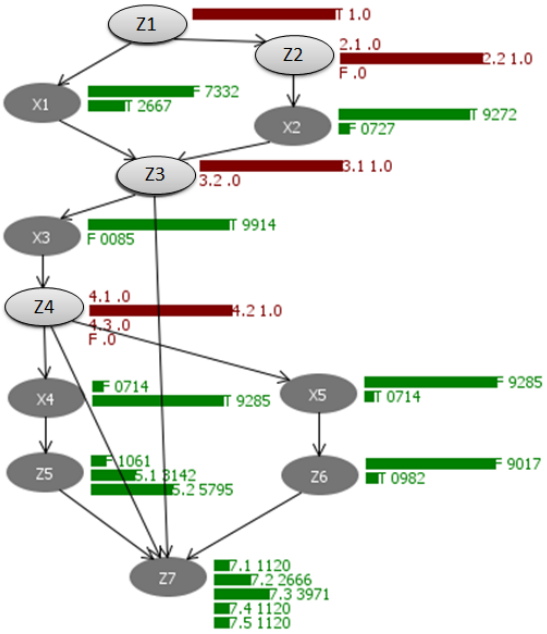


Fig. 6. Probabilistic inference on a valid scenario.

Posterior probabilities of human actions (X_i) indicate how likely is their execution despite the model is only feed with information of observable events. For instance, it predicts that the elder will refill medicine ($P(X_4 = True|e) = 0.9285$) but he will not request a new appointment ($P(X_5 = False|e) = 0.9285$), which is consistent with the BPMN workflow. On the other hand, the probability of the elder going to the hospital alone ($X_1 = True$) is slightly higher than he being carried out by his family member ($X_2 = True$), which can be explained

by the fact that the last arrived late to elder's home but the elder arrived on time at the hospital.

The probability of the elder getting all the medicine ($Z_7 = 7.3$) or part of it ($Z_7 = 7.2$) are slightly higher than the other outcomes. Both probabilities increase if more evidence is given (e.g. Z_5 and Z_6). Given that the Bayesian network was trained with valid process developments only, it predicts well the outcome on similar scenarios (see $P(z_7|e)$ in Fig. 6), but it assigns the same probability to all the five outcomes in invalid scenarios, which represents an uncertain outcome.

4 Discussion

First we analyze other probabilistic approaches to BPMN. Then we compare our selection of BPMN elements with other approaches that transform BPDs to agent-based system specifications. Finally we discuss the applications of probabilistic workflows as agent engineering tool.

4.1 Probabilistic approaches to BPMN

In 2008, Prandi and colleagues [17] proposed a formal semantics for BPMN based on the process calculus COWS. Each BPD node is considered as a COWS service and the translation describes the message flow between them. They provide a COWS formula for each node-centered structure supported by their normal form and produce a single composite formula that represents the flow of tokens across the BPD. BPDs are formalized as Continuous Time Markov Chains, a model used for automated verification of Web Service composition. Thanks to the implementation of COWS in the probabilistic model checker PRISM, the probability of observing certain event or condition at a time t can be estimated. Tasks, annotated with a duration range, occur at a different time in each alternative workflow produced by gateways present in the workflow; hence the probability of observing an event or action at a time t is expressed probabilistically.

Herbert and Sharp [6] proposed *stochastic BPMN workflows*, an extension of Core BPMN that includes: probabilistic flows (sequence flows with a given probability) and rewards associated to the execution of tasks. Using PRISM, authors transform BPMN workflows into Markovian Decision Processes (MDPs). A PRISM module is generated for each task based on a structure supported by their normal form; code templates codify transitions between states (represented by tasks), mediated by actions (represented by gateway conditions and task completion). PRISM is then used for generating all valid action sequences and calculating: 1) transitory and steady state probabilities of process conditions, 2) the probability of occurrence of an event (at a time t), 3) best and worst scenarios, and 4) the average time of process execution.

On the other hand, Bobek and colleagues [1] proposed a transformation of BPMN workflows to Bayesian Networks (BNs). The translation is straightforward, each node (action, event or gateway) is translated into a Boolean random

variable whereas control flows are used for constructing the conditional dependency graph. The Bayesian Network is trained with BPMN workflows obtained from a process library, producing CPTs that indicate how likely is to observe a node N_1 followed by another node N_2. The resulting BN is used for recommending missing nodes during a new process specification. This approach lacks of a mechanism for recognizing disjoint events and detecting equivalent events/tasks across different BPDs.

4.2 Translatable Fragments of BPMN workflows

The BPMN fragment of our approach differs from the one used in the translation of BPMN to BPEL [14] in two aspects. First, in [14] exist two types of end events, one for indicating that the participation of a component has finished, and another for indicating process termination. Given that we model the process from the perspective of the activity's subject, end events represent the different ways on which the process might terminate, successfully or on failure for the subject. Second, in our approach we don't consider data/event-based XOR gateways as long as an equivalent expressivity is provided by XOR gateways followed by intermediate events that might represent the event to observe for deciding which branch is followed during process execution.

Unlike the mapping of BPMN to agents proposed in [3], we only consider a single pool on which every lane represents a role. The use of multiple pools forces to specify illocutions between agents as part of the activity description, which produces a low-level specification which is not the purpose of our approach at this point. In contrast, BPEL, used for specifying systems based on Web Services, does not capture the attribution of agent capabilities (perceptions and actions) grouped around roles, which is evidenced on that it does not consider BPMN pools and lanes on its translation [14].

A limitation of our normal form is that we do not permit the representation of cycles in the BPMN workflow as long as it would produce non-acyclic graphs. This can be solved by replacing the feedback arc by a subprocess that replicates the cyclical section and it is called recursively until reaching the stop condition.

Another limitation is that the definition of random variables from intermediate events relies in a single fixed structured (XOR gateways); this mechanism can be generalized by calculating the different ways on which the graph can be traversed and determining which events never occur together, establishing a criterion for grouping proximate disjoint nodes.

4.3 Probabilistic workflows as agent engineering tool

Modeling human activities using BPMN from an Activity Theory perspective provides a goal-oriented plan representation for the User agent representing to the activity's subject in a MAS. The corresponding ACBN can be used for modeling other participants and providing recommendations to the user. Given that human actions are not directly observable, observable events between them can be used for estimating what happened or what will occur next.

As we show in [2], the causal network can be further used for introducing the participation of software agents and generating Prometheus scenarios. In this work we propose the use of BPMN as a user-friendly way of specifying the activity dynamics and its probabilistic distribution.

In this paper we illustrate how the BPD can be modeled from the perspective of a single actor (the subject) meanwhile it captures his interactions with other participants. Modeling a complex system where other actors should achieve their own goals requires capturing in a single BPD the perspective of other participants, or modeling their perspectives in separate BPDs and calculating their intersections. For instance, the participation of the Doctor is conditioned to his presence at the hospital previous to the appointment time; this is not represented in Figure 2, but such precondition should be available in the Doctor's consultation workflow.

5 Conclusions

We introduced a BPMN Business Process Diagram (BPD) normal form based on Activity Theory that can be used for representing the dynamics of a collective human activity from the perspective of a subject. We introduce a novel automatic procedure that transforms this workflow into a Causal Bayesian Network that can be used for modeling human behaviors and assessing human decisions.

The resulting Bayesian Network is not only consistent with the valid process developments encoded in the BPD, but it can be further complemented with causal dependencies discovered by algorithms like Pearl's Inferred Causation [15] from actual process developments in order to improve goal achievement's prediction.

Providing a semantic representation of event and action nodes will permit to overcome the limitations of other approaches for detecting equivalent nodes and will provide the platform for the composition of workflows, the generation of agent role descriptions and plans, and the implementation of a process monitoring procedure. Using these descriptions, the proposed transformation can be extended with a proper translation of loops and subprocesses, which in turn could be used for providing a work around for cycles.

Acknowledgments

This research was supported by Tecnologico de Monterrey through the "Intelligent Systems" research group, and by CONACyT through the grant CB-2011-01-167460.

References

[1] Bobek, S., Baran, M., Kluza, K., Nalepa, G.J.: Application of bayesian networks to recommendations in business process modeling. In: Proceedings of the Workshop AI Meets Business Processes 2013. vol. CEUR 1101, pp. 41–50 (2013)

[2] Ceballos, H.G., Garcia-Vazquez, J.P., Brena, R.: Using activity theory and causal diagrams for designing multiagent systems that assist human activities. In: Proceedings of the 12th Mexican International Conference on Artificial Intelligence, MICAI 2013. pp. 185–198. Mexico City (Nov 2013)
[3] Endert, H., Kuster, T., Hirsch, B., Albayrak, S.: Mapping BPMN to agents: An analysis. In: Agent, Web Services, and Ontologies Integrated Methodologies - International Workshop MALLOW-AWESOME 2007. pp. 43–58 (2007)
[4] Engeström, Y., Miettinen, R., Punamäki, R.: Perspectives on Activity Theory. Learning in Doing: Social, Cognitive and Computational Perspectives, Cambridge University Press (1999)
[5] Garcia-Vazquez, J.P., Rodriguez, M.D., Tentori, M.E., Saldana, D., Andrade, A.G., Espinoza, A.N.: An agent-based architecture for developing activity-aware systems for assisting elderly. Journal of Universal Computer Science 16(12), 1500–1520 (jun 2010)
[6] Herbert, L., Sharp, R.: Precise Quantitative Analysis of Probabilistic Business Process Model and Notation Workflows. Journal of Computing and Information Science in Engineering 13(1), 011007(1–9) (2013)
[7] Hinge, K., Ghosey, A., Koliadisz, G.: Process seer: A tool for semantic effect annotation of business process models. In: Proceedings of 13th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2009. pp. 54–63 (2009)
[8] Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A semantic web rule language combining OWL and RuleML. W3C Member Submission 21 May 2004
[9] Jander, K., Braubach, L., Pokhar, A., Lamersdorf, W., Wack, K.J.: Goal-oriented processes with GPMN. International Journal on Artificial Intelligence Tools (20), 1021–1041 (2011)
[10] Kuster, T., Lutzenberger, M., Hessler, A., Hirsh, B.: Integrating process modelling into multi-agent systems engineering. In: Multiagent and Grid Systems. pp. 105–124 (2012)
[11] Leont'ev, A.: Activity, Consciousness, and Personality. Prentice-Hall, Inc. (1978)
[12] zur Muehlen, M., Indulska, M.: Modeling languages for business processes and business rules: A representational analysis. Information Systems 35(4), 379–390 (2010)
[13] OMG: Business Process Model and Notation (BPMN), Version 2.0 (January 2011), http://www.omg.org/spec/BPMN/2.0
[14] Ouyang, C., van der Aalst, W., Dumas, M., Hofstede, A.: Translating BPMN to BPEL. BPM Center Report BPM-06-02, BPMcenter.org (2006)
[15] Pearl, J.: Causality. Models, Reasoning, and Inference. Cambridge University Press (2000)
[16] Pearl, J., Robins, J.: Probabilistic evaluation of sequential plans for causal models with hidden variables. In: Besnard, P., Hanks, S. (eds.) Uncertainty in Artificial Intelligence 11. pp. 444–453 (1995)
[17] Prandi, D., Quaglia, P., Zannone, N.: Formal analysis of bpmn via a translation into cows. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5052 LNCS. pp. 249–263 (2008)