Annals of University of Craiova, Math. Comp. Sci. Ser.
Volume 36(2), 2009, Pages 69–78
ISSN: 1223-6934

Simulation Modelling for Businesses using BPMN and AORS

Oana Nicolae, Mircea Diaconescu and Gerd Wagner

Abstract. Our work is situated in the area of enriching Business Process Simulation (i.e. BPS) techniques with graphical modelling capabilities. We focus on the open-source standard BPMN as a graphical notation for the open-source Entity-Relationship (ER) /Agent-Object-Relationship (AOR) business process simulation framework. The goal is to envision an extension of the core BPMN for capturing essential simulation concepts such that the obtained BPMN extension can be used as a modelling language for simulations. To clarify this, the applicability of BPMN for modelling agent-based simulations has to be investigated. We use as a show case the example of a DriveThruRestaurant business management workflow and we present two methodologies of constructing the ER/AOR Simulation(s) of the business process i.e. a basic Discrete-Event-Simulation (i.e. DES) model where we abstract away the agents together with their acts of communication and, the agent-based DES implementation enhanced with activities and beliefs. We show that BPMN language still needs adjustments in order to be capable to model both approaches.

2000 Mathematics Subject Classification. Primary 68T42; Secondary 68T30.
Key words and phrases. BPMN, ER/AOR Simulation, Activities, Beliefs, BPEL, SOA, Business Process Modelling and Simulation.

1. Introduction and Motivation

The actual literature reveals two main research directions concerning the *Business Process Modelling and Simulation* (BPMS) topic: graphical modelling of business processes and simulation modelling (i.e. business process simulation). Our research aims at combining these two branches and obtaining a visual representation of business processes that can be further simulated under different circumstances with the purpose of better understanding the processes behaviour, reducing the costs of their implementation, while at the same time improving the quality of the delivered system.

Business modelling is often associated in the literature with Service-Oriented-Architecture (SOA) approach that supposes the construction of distributed business applications around the web-services technology. The SOA approach to business processes modelling supposes business diagrams that envision web-services collaborations (i.e. orchestrations or choreographies). The resulting business process model represents a workflow of activities that invoke appropriate web-services and supports an automated translation to a technical programming language. In this context, the creation of a business model means programming a process and the intended-final goal is the process enactment. BPEL[1], a block-structured programming language with an XML syntax, is often used as an automated and executable mechanism for orchestrating different web-services. An extension of BPEL language i.e. BPEL4CHOR was also envisioned for web-service choreography enactment. The literature reveals

[1]BPEL 2.0 - http://docs.oasis-open.org/wsbpel/2.0/wsbpel-specification-draft.html

a lot of academic work when trying to map a business process graphical documentation (graph-oriented) into BPEL/BPEL4CHOR languages with the purpose of gaining executable business processes orchestrations and choreographies. The translation procedure ([6], [9], [8]) is not always straight-forward, as graphs can be used to express more complex flow and loop patterns, which cannot be directly translated into block structures. Discussed in academia, the subject also passed in the industry area and proprietary frameworks such as ARIS[2] and Intalio[3] claim to provide automated mappings (BPMN2BPEL) for structured and semi-structured patterns[4].

While business process analysts are used to think in terms of business entities such as:

- each pool has associated an actor, a performer that owns or is responsible for the execution of the pool workflow (role-orientation)
- the actors conform to a delegation model (i.e. each actor can delegate its tasks to other actors - the pool is subdivided into multiple lanes)
- actors make use of some resources that can be or can be not available at a moment in time: the menu board service, the pickup window service or the pizza order.
- the actors have particular behaviours (i.e. the internal pool workflow that comprises events, activities and gateways for controlling the events and activities flow)
- the actors have particular goals they wish to achieve (e.g. a driver intends to buy a pizza from a DriveThruRestaurant). In order to achieve its goals, the actor must take some decisions: (e.g the driver arrives at the restaurant and if there are not so many cars waiting in line to be served, he decides to buy, other way he will leave).

a mapping to BPEL language will loose all this structured information for the sake of business process execution.

An agent-based simulation framework keeping all these information and assuring enactment proves to be more suitable to envision the process model of a particular business process diagram. This approach offers additional capabilities such as: the possibility to simulate the behaviour of the system at a given moment in time under some precise circumstances, resources management and continuous improvement of the business process. It also introduces the distinction between active entities (agents) and passive entities (objects), that does not exist in a SOA approach.

Since BPMN[5] is the result of the standardisation effort sustained by OMG[6] we expect it to be part of the business market landscape in the foreseen future. A short introduction to BPMN terminology and concepts can be found in [3]. BPMN is an evolving language and we wait the standardisation of its 2.0 version at the end of 2009. Its business process diagrams are intuitive and do not require specific knowledge about BPMN graphic symbols semantics to be understood. Additional specifications (usually expressed as graphical symbols attributes) come as a second language layer and contribute to the language expressiveness. For these reasons we decide to use BPMN as a visual language for modelling business processes.

Agent-based simulation environments often use goal-oriented, declarative process models. Agents may have assigned goals and the details of how to accomplish these

[2]ARIS - http://www.aris.de/
[3]Intalio BPM - http://www.intalio.com/products/bpm/
[4]Workflow Patterns - http://www.workflowpatterns.com/
[5]BPMN 1.2 - http://www.omg.org/spec/BPMN/1.2/
[6]OMG - htp://www.omg.org/

goals are not given, but depend on the agent perceptions during the process. Therefore, the ordering of tasks is not represented. Scheduling of tasks is done dynamically during process enactment, and is influenced by the triggered events and the process interactions.

A graphic representation of these kind of business processes is difficult and tends to become complex, since it would correspond to a visualisation of several possible scenarios. Our aim is to analyse how BPMN is dealing with representing such scenarios where the business process is not anymore static, but involves dynamic aspects. We will also investigate how a role-oriented language such as BPMN can represent a declarative and goal-oriented simulation language. The importance of a good business process model is well known: the more time it is spent for designing, the less time it is needed for coding and testing.

The implementation solution of the simulation show cases is obtained by using the open-source ER/AOR Simulation framework, which is an ontologically well-founded Discrete-Event-Simulation framework with a high-level rule-based simulation language and an abstract simulator architecture and execution model (see *AORS - http://www.AOR-Simulation.org*). The Simulation framework proposed in [7] supports both basic DES without agents (ER) and complex agent-based simulations with (possibly distorted) perceptions and (possibly false) beliefs (AOR). The simulation scenario is expressed with the help of the XML-based ER/AOR Simulation Language (ER/AORSL). The scenario is then translated to executable code, compiled to Java byte code and finally executed.

The rest of the paper is organised as follows: Section 2 describes the ER/AOR Simulation Language, with particular references at business process simulation domain. Basic concepts such as: Discrete-Event-Simulation (DES) and its different approaches will be discussed. Section 3 introduces the DriveThruRestaurant show-case and provides insights into ER/AOR (with beliefs and activities) implementation. BPMN standard will serve to graphically design the envisioned business processes, and will also cause the discussions presented in Section 4 about its capabilities to represent ER/AOR business process behaviour. We draw the conclusions and we specify the gained objectives in Section 5.

2. ER/AOR Framework for Simulation Modelling

Before introducing our ER/AOR framework for agent-based DES simulation we would like to briefly describe the simulation modelling topic, its characteristics and particular concepts and terminology used in the literature.

2.1. Simulation Modelling or DES Business Process Simulation. The simulation capability allows to experiment with a model in order to understand its behaviour under different scenarios. Simulation techniques were used for many years in business process re-engineering and logistics. Simulation involves two steps: construction of business processes models, and performing experiments on these models: repeatedly executing the models a period of time, with the purpose of obtaining data for statistical analysis. There are many types of simulation: static and dynamic, with dynamic simulations modelled continuous and discrete. We are interested in dynamic and discrete models, as they are the most appropriate to model the behaviour of an organisation. In discrete event simulations time is moved in discrete steps from one event to another. Briefly, the simulated system comprises entities (active resources) that have attributes and may consume work, in form of people or machines, termed as passive resources. The passive resources may be available for a specified time and the active resources (the entities) may wait in a queue if the passive resource is not available.

An agent-based DES framework, in order to perform simulation modelling needs to minimum comprise the followings: entities with attributes, events and resources.

2.2. ER/AOR Simulation Language. An ER/AOR simulation model consists of: (1) an optional space model (needed for physical objects/agents); (2) a set of entity type definitions, including different categories of event, message, object and agent types; (3) each agent type may involve a set of agent rules representing its message interactions; (4) each agent may be responsible for the execution of a set of activities; (5) agents may have a mental structure in form of beliefs; and (6) a set of environment rules, which define causality laws governing the environmental state changes.

An entity type is defined by means of a set of properties and a set of functions. There are two kinds of properties: attributes and reference properties. Attributes are properties whose range is a data type; reference properties are properties whose range is another entity type.

The upper level ontological categories of ER/AOR Simulation are objects (including agents, physical objects and physical agents), messages and events. Notice that according to this upper-level ontology of ER/AOR Simulation, agents are special objects. For simplicity it is common, though, to say just object instead of using the unambiguous but clumsy term non-agentive object. Both the behaviour of the environment (i.e. its causality laws) and the behaviour of agents are modelled with the help of agent rules, which support high-level declarative behaviour modelling.

2.2.1. *ER Simulation.* Entity-Relationship (ER) modelling introduced in [1] involves entities, their classification by means of entity types, domain relationships between entities and their classification in relationship types. ER object-orientation capability allows domain independent relationships between entity types such as: generalisation and aggregation, and provide the entity types with the characteristic of having complex attributes.

Our DES-based ER Simulation deals only with two basic categories of entities: objects and events. A simulation model defines a number of object types and event types, each of them with one or more properties and zero or more functions (to be used for all kinds of computations such as for computing pseudo-random numbers following an empirical distribution). Among the event types, we distinguish those that define erogenous events (typically with some random periodicity) and those that define caused events that follow from the occurrence of other events.

The state of the environment (i.e. the system state) is given by the combination of the states of all objects. Environment rules define how the state of objects is changed by the occurrence of events. An environment rule is a 6-tuple (WHEN, FOR, DO, IF, THEN, ELSE) where: (1) WHEN denotes the type of event that triggers the rule; (2) FOR is a set of variable declarations, such that each variable is bound either to a specific object or to a set of objects; (3) DO represents the update of environment states that implicitly is performed; (4) IF is a logical formula allowing for variables and possible expressing a state condition; (5) THEN holds is the IF condition is evaluated to true and specifies an update of the environment state together with a list of resulting events to be created; and (6) ELSE holds is the IF condition is evaluated to false and also specifies an update of the environment state together with a list of resulting events to be created when the rule is fired.

In each simulation step, all those rules are fired whose triggering event types are matched by one of the current events and whose conditions hold. The firing of rules may lead to updates of the states of certain objects and it may create new future events to be added to the future events list. After this, the simulation time is incremented to the occurrence time of the next future event, and the evaluation and application of rules starts over.

2.2.2. *AOR Simulation.* In the form of agent-based DES, which we call Agent-Object-Relationship (AOR) simulation, we introduce a new category of entities: the agents, which can be physical or non-physical. When we introduce agents, we make further distinctions between different types of events, in particular, we need to consider perception events (or input messages that an agent receives) and actions events (or output messages that an actors sends) in order to account for the perception-action cycle defining the foundation of agent behaviour. ER modelling does not take into account the dynamic aspects of a process model such as: communication, interaction, activities, because it does not distinguish between active (the agents) and passive objects.

Optionally, when modelling business processes a new entity category extending the event type (the activities) is introduced (see [4], [5]), in order to represent the work to be done in a business process. Activities are complex events having a start event, an end event, an optional association with an actor (being an agent that plays the role of a special resource) and zero or more associations with other objects playing the role of further resources used by the activity.

An agent type is defined by means of: (1) a set of (objective) properties; (2) a set of (subjective) self-belief properties as well as (3) an optional set of (subjective) belief entity types; and (4) a set of agent rules, which define the agent reactive behaviour in response to perception events (and internal time events).

3. Show Case: DriveThruRestaurant model

Show Case model description: As a car enters from the street, the driver, who we will call Fred,decides whether or not to get in line. If Fred decides to leave the restaurant, he leaves as a dissatisfied customer. One of the great things about a simulation is that it is easy to track these type of customers. In most real-world systems, it is usually difficult to track customers who leave dissatisfied. If Fred decides to get in line, then he waits until the menu board (with the speaker no normal human being can understand) is available. At that time, Fred gives the order to the order taker. After the order is taken, then two things occur simultaneously: (1) Fred moves forward if there is room. If there is no room, then he has to wait at the menu board until there is room to move forward. As soon as there is room, he moves so the next customer can order. (2) The order is sent electronically back to the kitchen where it is prepared as soon as the cook is available. As soon as Fred reaches the pickup window, then he pays and picks up his food, if it is ready. If the food is not ready, then Fred has to wait until his order is prepared. ([2])

3.1. ER Modelling and Implementation. In this section we show how to model and simulate our scenario described above purely with objects and events, without agents and activities. We use the BPMN-like graphical notation to visually describe it in the Figure 1. For the complete model, please check out *http://oxygen.informatik.tu-cottbus.de/aors/examples/Management/DriveThruRestaurant_withoutAgents AndActiv ities/scenario.xml*.

The information model for the envisioned system includes the object types entities: `Customer` object and `Order` object. We abstract away the representation of the DriveThruRestaurant in favour of the functionality that it offers: the menu board where the customer can order, the kitchen where the order is prepared and the pickup window where the customer receives its order. As maybe one may expect, the menu board, the kitchen and the pickup window are not modelled as objects either, but identified through some properties of the system, that describe their availability (e.g. `menuBoardBusy:boolean`) and the maximum size of the waiting line that customers may form (e.g. `maxMenuBoardQL:integer`). It is easy to see that the customer entity type represents some active resources that make use of the menu board, kitchen, pickup window passive resources. The last ones can be available or not, and generate the construction of a waiting line of customers willing to consume them. The simula-

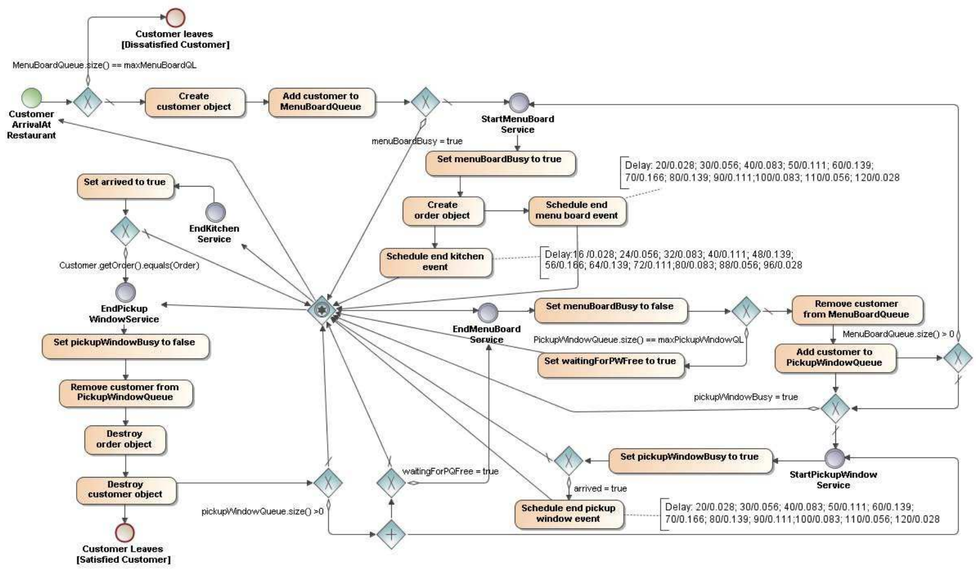

FIGURE 1. BPMN modelling of the ER DriveThruRestaurant model.

tion scenario begins with the arrival event of a customer at the restaurant. Based on some state information of the system, the customer decides if he leaves or get in line at the menu board to buy pizza. The customer can consume the menu board resource only when the resource is free, with other words, the menu board resource became free for a customer only when he became first in line in the menu board queue. The resource is busy with a customer, has a duration, for a predefined time period (we define it as a probability distribution defined for the service duration). We model this behaviour in the following way: after waiting in the queue, the customer finally arrives at the menu board and generates the start of the menu board service: the caused event `StartMenuBoardService`. The duration of the activity is modelled at an atomic action: a task (`Schedule end menu board event`) that schedules the end event of the menu board : `EndMenuBoardService`. In the same time, an `Order` object is created (the atomic task `Create order object`) and for simulation the fact that preparation of the order at the kitchen begins, we use schedule the `EndKitchenService` event.

The simulation goal is to collect statistic variables that gather information about: number of clients, number of lost clients, revenue, lost revenue, time-weighted average of resource utilisation for menu board, kitchen, pickup window resources.

Notice that BPMN, despite its proclaimed static representation of a business model, is able to represents the dynamic behaviour of our scenario that involves two queues of customers and events that are triggered in parallel for different `customer` instances.

For simplicity, we have discarded the code examples, but you can find them by check the above mentioned Web link.

3.2. AOR Modelling and Implementation. In this section we illustrate the same scenario envisioned by using the higher-level modelling constructs of agents with beliefs and activities. Like for higher-level programming constructs, they are not really needed, but they help a lot to make a model more well-structured and more readable (see Figure 2). The customers, the menu board, the kitchen and the pickup

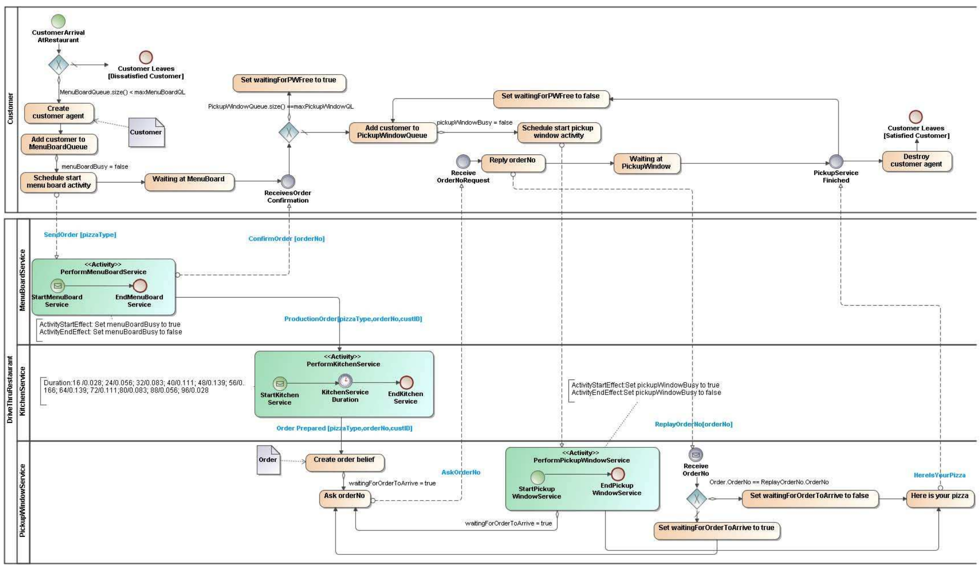

FIGURE 2. BPMN modelling of the AOR DriveThruRestaurant model.

window became agent types. They communicate by perceiving incoming messages and sending outgoing messages. The message tapes also have a complex structure that involves attributes and properties. The relation active resource entity - passive resource entity it is now much better represented, as we define activity types for each of the services that the DriveThruRestaurant offers. For example, the menu board agent is envisioned as performing an activity (`PerformMenuBoardService`) that encapsulates its entire workflow. Notice the conceptualisation manner: the menu board agent performs an activity that consume the menu board resource. In real life, the menu board agent is a person that takes your order, the corresponding activity is the work he/she performs and the used passive resource is the menu board desk that can be busy or not a duration of time, for a particular customer.

We will discuss in the following SubSection about the agents mentalistic structure that helps them to have a memory about events that happened in the past. Again, the entire model and code of the scenario can be found at the Web link: *http://oxygen.informatik.tu-cottbus.de/aors/examples/Management/DriveThruRestaurant_withAgentsAndActivities/scenario.xml*.

3.2.1. *AORS - Beliefs and their management.* AORS offers support for agent beliefs together with communication capabilities. Having such aptitudes, the agents are able to preserve information (possible false) about themselves, about other entities or about events that may occur. Moreover, they are also capable to exchange these information resulting in improvements of their knowledge. AORS offers support for two types of agent beliefs:

- self beliefs - are beliefs of an agent about itself. Such beliefs may refer to physical or non physical properties of the agent.
- belief entities - are beliefs of an agent about another agent, object or even events.

Since beliefs may be updated during the communication, there is possible that the received information be not up to date, false or even contradictory. The receiver is responsible for actions taken when an incoming message containing beliefs data is received. Mainly, these decisions are results of agent rules or communication rules.

AOR allows to express beliefs of an agent type as a structure that has a name (the belief entity type name) and a set of properties with predefined types. A belief entity name may be the same as the name of the believed object/agent type. Belief entities may or may not have the same properties as the believed entity. A belief entity is referenced via its ID. In the case of beliefs about real entities, the its ID must be the same as the ID of the referenced entity. During the simulation time, new beliefs may be created and existing beliefs may be destroyed and/or updated. Beliefs management is performed via agent rules that are responsible for the behaviour of agents of that type. Agent rules are mainly related to agent types, but they are also personalised to individual agents via their conditions and/or results.

A belief entity type of an agent type must be defined before it can be instantiated, updated or destroyed. A belief type is defined as a substructure of an agent type definition. For instance, the above presented scenario considers an agent type namely `PickupWindow` that have beliefs about the orders which have to be delivered to clients. Therefore, a belief type, `Order` may be defined and used for this purpose.

When beliefs about an entity are created or updated, values for the `id` and the belief entity type may be provided as static values or as expressions (e.g. Java expressions when use the AOR-JSim implementation). After a a belief is created, its ID cannot be changed or reallocated, but only if the belief it is destroyed. When update actions are performed, it is not necessarily to provide the belief entity type, but only the slots that need to be updated and their new values, along with the belief entity ID. The beliefs management may be performed via agent rules or communication rules. Rules allows to define belief entity variables by using their IDs (as static value or as expressions) along with belief entity types that result in a collection with the defined type.

4. BPMN modelling of agent-based DES scenarios

Although the overall conclusion is that BPMN is a graphical language that is capable to represent both basic DES and agent-based DES scenarios (in particular ER/AOR scenarios), there are issues that still need improvements and we would like to discuss some of them in the followings (see also the proposed eBPMN metamodel in Figure 3):

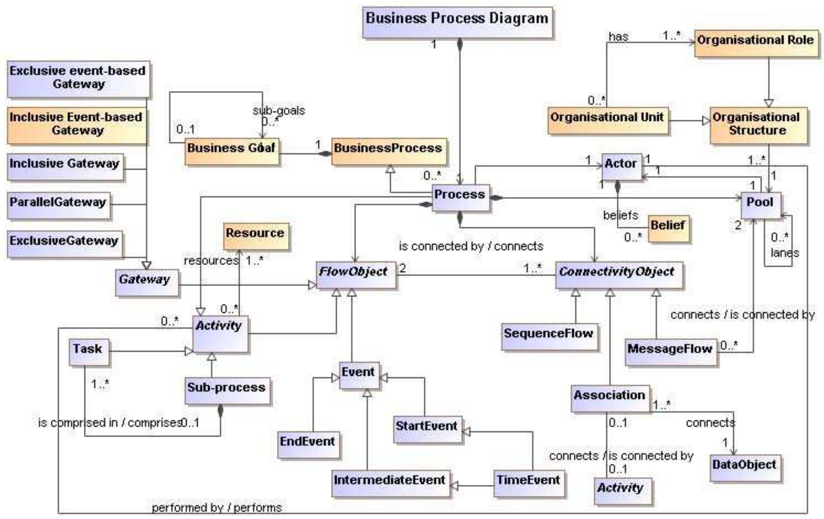

FIGURE 3. Draft eBPMN metamodel.

- BPMN does not have appropriate structures to represent complex data. For example, the creation of `customer` and `object` instances is represented using a data artifact connected to the atomic task using an association arrow. Two weak points: (1) the data artifact does not allow us to represent complex types that have attribute and properties; (2) the artifacts and the association arrows do not have an explicit semantics stipulated in the Specification.
- The accumulated semantics of the sequence flow graphic element: it can bind a task to another task, a task with an event, an event with a task, an event with a sub-process or with the boundary of the pool.
- The message flow elements fails when it should represent the message type it carries and also the internal structure of the message. We choose an ad-hoc manner to attach the parameters of the message in square brackets, just after the message name.
- the simulation constructs as: number of steps, statistics variables, resources do not find an equivalent in BPMN Specification.
- also the dynamic nature of the ER/AOR scenarios suppose the use of a customers queue. BPMN is not yet capable to represent this queue. We used a gateway whose semantics differs from the ones existent in the actual specification: the inclusive event-based gateway. It has an OR semantics, meaning that the gateway waits for many events to happen in parallel, at the same time, event that belong to different customer type instances.
- BPMN does not have an equivalent structure for representing agents beliefs.
- BPMN is a role-oriented language, and is not capable to represent plans and their associated goals, although the goal-orientation seems natural in the business process modelling and management area. Business goals should have a natural relation with BPMN activities, and the completion of the business goals should update the activity(ies) state. A plan can be envisioned as an activity or a set of

activities. Performing the plan would mean to execute the steps from the plan i.e. interaction with the environment, with other actors, performing some kind of computation, sending and receiving messages from other actors in the process.

5. Conclusions and Future Works

Business processes represent important assets for any organisation and BPEL is already a standardised language for business process execution with great industry acceptance. Then why using Simulation? The simulation capability allows to experiment with a model in order to understand its behaviour under different scenarios. The enactment is not anymore the final-must for an organisation. What is desirable is a facile and intuitive graphical support for simulation framework. Standardised languages that assures interoperability and open-source tools are envisioned in the academia. We propose the use of a BPMN extension to serve as a graphical documentation for the ER/AOR simulation scenarios (Please check *http://oxygen/aors/exampl es/Management/* web link). We have showed that BPMN is capable to model simple DES and agent-based DES simulations that are suitable for capturing the entire conceptualisation procedure of a business process. Moreover, an organisation comprises many business processes and a target of our research is to obtain goal oriented business processes, that are performed by actors enhanced with beliefs. The business goals are gathered into business plans (the activities or set of activities) that consume resources.

References

[1] P.Chen, 1976,*The Entity-Relationship model - toward a unified view of data.*ACM Transactions on Database Systems, 1(1), pages 9-36.

[2] R.G. Ingalls, 2008. *Introduction to Simulation.* In Mason, S.J. Hill, R.R. Mnch, L. Rose, O. Jefferson, T. Fowler, J.W. (Eds.), Proceedings of the 2008 Winter Simulation Conference, 17-26.

[3] O. Nicolae, M. Cosulschi, A. Giurca and G. Wagner, 2008. In: Nicolae Constantinescu (Eds.)*Towards Standards for Modelling and Executing Collaborative Business Processes.* Proceedings of the 8th International Conference on Artificial Intelligence and Digital Communication (AIDC 2008). September, 13-14, 2008. pages. 137147. Craiova, Romania.

[4] O. Nicolae, G. Wagner, J. Werner, 2009. *Towards an Executable Semantics for Activities using Business Process Simulation* edBPM Workshop, BPM 2009 Conference, Ulm, Germany. (to appear - September 2009).

[5] O. Nicolae, G. Wagner, J. Werner, 2009. *Extending DES by adding an Activity concept for Business Process Modelling and Simulation* Winter Simulation Conference (WSC) 2009, Austin, Texas, USA. (to appear - December 2009).

[6] C. Ouyang, Dumas, A. H. M. ter Hofstede and W. M. P. van der Aalst. (2007) *Pattern-based translation of BPMN process models to BPEL web services*, International Journal of Web Services Research (JWSR).

[7] G.Wagner, 2004. *AOR Modeling and Simulation - Towards a General Architecture for Agent-Based Discrete Event Simulation.* In Agent-Oriented Information Systems, Lecture Notes in AI, volume 3030, pages 174-188. Springer-Verlag.

[8] P. Wohed, W.M.P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, and N. Russell. *Pattern-based Analysis of BPMN - An extensive evaluation of the Control-flow, the Data and the Resource Perspectives (revised version)*, BPM Center Report BPM-06-17, BPMcenter.org, 2006.

[9] P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. *Analysis of Web Services Composition Languages: The Case of BPEL4WS.* In ER 2003, pages 200-215.

(Oana Nicolae, Mircea Diaconescu and Gerd Wagner)
Department of Internet Technology, Institute of Informatics, Brandenburg Technical University at Cottbus, Germany
E-mail address: {Oana.Nicolae, M.Diaconescu, G.Wagner}@tu-cottbus.de