Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds.

DISPOSITIONS AND CAUSAL LAWS AS THE ONTOLOGICAL FOUNDATION OF TRANSITION RULES IN SIMULATION MODELS

Giancarlo Guizzardi

Federal University of Espírito Santo (UFES)
Computer Science Department
Av. Fernando Ferrari
s/n29060-970 Vitória, Espírito Santo, BRAZIL

Gerd Wagner

Brandenburg University of Technology
Institute of Informatics
P. O. Box 101344
03013 Cottbus, GERMANY

ABSTRACT

Discrete event simulation models define a state transition system, using some form of transition rules, or transition functions. In this paper we propose an ontological account of transition rules based on dispositions and causal laws. This account extends our previous presentation of DESO, a foundational ontology of objects and events for discrete event simulation modeling. We also show how the ontological concepts of dispositions and causal laws provide a semantics for the concept of transition rules in conceptual simulation modeling languages corresponding to transition functions in simulation languages.

1 INTRODUCTION

Any *discrete event simulation* (DES) formalism has one or more language elements allowing to specify, at least implicitly, the *transition functions*, or *transition rules*, that define the dynamics of the transition system specified by a simulation model by describing the dispositions of objects triggered by events. Unfortunately, this is often obscured by the standard definitions of DES, as we will discuss in the next section.

This paper is based on our previous research on ontological foundations of simulation modeling, reported in (Guizzardi and Wagner 2010, Guizzardi and Wagner 2011a, Guizzardi and Wagner 2011b, Guizzardi and Wagner 2012).

1.1 Standard Definitions of DES

We briefly discuss some of the standard definitions of DES that are repeatedly presented in simulation textbooks and tutorials. It is common to call the different computational paradigms, on which simulation languages and systems are based, "worldviews".

According to Pegden (2010), a *simulation modeling worldview* provides "a framework for defining a system in sufficient detail that it can be executed to simulate the behavior of the system". It "must precisely define the dynamic state transitions that occur over time". Pegden continues saying that "Over the 50 year history of simulation there has been three distinct world views in use: event, process, and objects":

Event worldview The system is viewed as a series of instantaneous events that change the state of the system over time. The modeler defines the events in the system and models the state changes that take place when those events occur. All DES systems implement their internal logic using this basic modeling approach, regardless of the worldview that they present to the user.

Process worldview The system is described as a process flow where a set of passive entities flow through the system and are subject to a series of process steps (e.g. seize, delay, release) that model the state changes that take place in the system.

Object worldview The system is modeled by describing the objects that make up the system. The system behavior emerges from the interaction of these objects.

The process worldview is still widely used in practice and many simulation systems based on it have incorporated some form of support for objects and *object-oriented* programming. According to Pegden (2010), agent-based modeling is typically implemented with the object world view. So, today's DES landscape is largely based on the process and object worldviews, and the fundamental concept of events is hardly mentioned anywhere.

On our ontological analysis, all three worldviews, and especially the two latter ones, which dominate today's simulation landscape, lack important conceptual elements. The event worldview does not support modeling objects with their properties. The process worldview does neither support modeling events nor objects. And the object worldview, while it supports modeling objects, it does not support modeling events. None of the three worldviews does support modeling the *dispositional* properties of objects with a full-fledged explicit concept of *transition rules*.

1.2 Transition Rule Concepts in DES Formalisms

We briefly discuss the concept of *transition rules* for the classical DES formalism of Petri Nets (PNs) and the Atomic DEVS formalism of Zeigler (1976), and for the agent-based DES formalisms of Brahms (Sierhuis 2001) and AORSL.

In a Petri Net, a transition rule is implicitly given by a 'transition' together with its input and output arcs and their multiplicity.

Atomic DEVS allows specifying transition rules as a combination of three functions: *external transition functions*, which define how input events change the states of the system, *internal transition functions*, which define how states of the system change internally based on a time advance function, and *output functions*, which defines how a state of the system generates an output event.

In Brahms (Sierhuis 2001), there are two forms of transition rules: *workframes* and *thoughtframes*. Also in AORSL there are two forms of transition rules: *environment rules* describing the causal laws of an inanimate environment, and *reaction rules* describing the behaviour of agents.

1.3 Events

In (Guizzardi and Wagner 2005; Guizzardi et al. 2008), we have presented early versions of our ontology of events called *UFO-B*. While the concept of an event is typically limited to instantaneous events in the area of DES, the general concept of an event, as discussed in philosophy and in other fields of computer science, includes composite events and events with non-zero duration. We believe that DES research would benefit from adopting the more general concept of events as proposed by ontological approaches.

2 RELATED WORK

As explained in (Silver et al. 2009), the DeMO project aims at establishing an ontology for discrete event simulation with the main concern to support *ontology-driven simulation*. The starting point for the DeMO methodology is the conceptual model of a system obtained as the first step in the process of making a simulation model. This model has to be provided in the form of an *OWL ontology*, i.e. a set of logical statements expressed in the *Web Ontology Language OWL* recommended by the W3C. It is then mapped to an instantiation of the DeMO ontology, which is, in turn, mapped to an executable simulation model. Thus, the DeMO ontology constitutes a high-level simulation language. The conceptual framework proposed by DeMO is, however, not based on a foundational ontology.

In (Benjamin, Patki, and Mayer 2006), it is recommended to use domain ontologies in the simulation modeling process for making simulation models unambiguous and consistent. It is argued that in distributed simulation, ontologies may play the role of a vendor/platform-independent modeling language that facilitates the translation of models into the different simulation platforms involved in a distributed simulation.

In (Livet et al. 2010), it is proposed to use ontologies (in the sense of conceptual domain models) for making the scientists' conceptual models more coherent with the simulation program code. This amounts to making an explicit conceptual model (using UML and/or OWL) before starting to code a simulation. However, although the paper refers to philosophical work on ontologies, foundational ontologies are not considered.

So, while there have been several proposals about how to use (OWL) ontologies in simulation engineering, we were not able to find any work on the ontological foundations of simulation (modeling) languages.

3 BACKGROUND: THE UNIFIED FOUNDATIONAL ONTOLOGY

Like other foundational ontologies, such as DOLCE (Masolo et al 2003) and GFO (Herre 2010), UFO makes a distinction between enduring and perduring entities, henceforth called *endurants* and *events*. Classically, this distinction can be understood in terms of their behavior w.r.t. time. Endurants are said to be wholly present whenever they are present, i.e., they *are in time*, in the sense that if we say that in circumstance c_1 an endurant *e* has a property P_1 and in circumstance c_2 the property P_2 (possibly incompatible with P_1), it is the very same endurant *e* that we refer to in each of these situations. Examples of endurants are a house, a person, the moon, a hole, an amount of sand. For instance, we can say that a person John weights 80kg at c_1 but 68kg at c_2. Nonetheless, we are in these two cases referring to the same entity.

Events, which are also called *perdurants* or *happenings*, are entities composed of temporal parts, they *happen in time* in the sense that they extend in time accumulating temporal parts. Examples of events are a conversation, a football game, a symphony execution, a birthday party, the Second World War, or a particular business process. Whenever an event is present, it is not the case that all its temporal parts are present.

UFO consists of three main layers: UFO-A is concerned with the ontology of *objects* (more precisely, of *endurants*), UFO-B is concerned with the ontology of *events*, and UFO-C is concerned with the *intentional* and *social* ontology of *agents*.

3.1 UFO-A: An Ontology of Objects

Among the categories of endurants, UFO makes a distinction between *substance individuals*, such as *objects*, and *trope individuals* (see Figure 1).

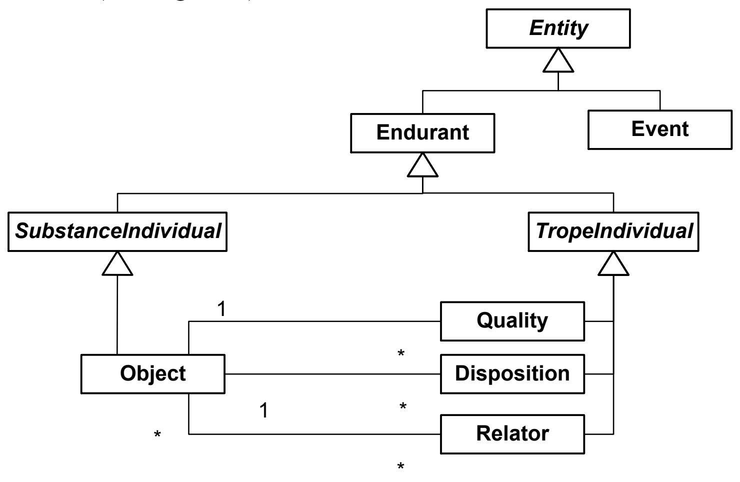

Figure 1: Fragment of the Unified Foundational Ontology (UFO).

Substance individuals are existentially independent entities. Examples include ordinary material objects such as a person, a dog, a house, a hammer, a car, Alan Turing and The Rolling Stones.

Trope individuals (also simply called *tropes*) are endurants, which are existentially dependent on other endurants (their bearers). The particular kind of existential dependency relationship between tropes and their bearers is called *inherence*. For instance, the eye color of Kate can only exist if Kate exists and cannot depend on anyone but Kate. Trope individuals include *qualities* such as the color of an eye, *dispositions* such as the fragility of a glass, and *relators* such as a marriage.

For being able to understand how dispositions are the foundation of causal laws, we need to understand how they relate to events. We, therefore, next turn to the topic of events.

3.2 UFO-B: An Ontology of Events

The second layer of UFO, called UFO-B, is concerned with the ontology of events, which consists of several theories about:

- the part-whole relationship between events;
- temporal relationships between events;
- the participation of objects in events;
- the relationship between events and situations;
- events as manifestations of the dispositions of objects.

3.2.1 The Extensional Mereology of Events

Events can be atomic or complex, see Figure 2. While atomic events have no proper parts, complex events are aggregations of at least two disjoint events.

Consider, for instance, the event e: the murder of Caesar. This event can be further decomposed into sub-events, namely: e1: the attack on Caesar, and e2: Caesar's death. Event e1 can, in turn, be decomposed into the events e11: Caesar's restraining by the conspirators, and e12: the stabbing of Caesar by Brutus.

In UFO-B, following (Simons 1997), the part-whole relationship between events is a partial order relation satisfying the axioms of *Extensional Mereology* where two events are the same if they are composed of the same parts.

3.2.2 Temporal Relationships between Events

The temporal properties of events (such as their start time, end time and duration) are verbalized by corresponding attribution statements using attributes that take their values from a temporal datatype based on a time model such as linear, branching, parallel or circular time. UFO does not make a commitment to a particular time model. It is only assumed that events have a start and an end time, and are strictly ordered by a *precedes* relation.

The set of temporal relations between two events corresponds to the well-known time interval relations *before*, *meet*, *overlap*, *starts*, *during*, *finishes* and *equal*, proposed by Allen (1983).

3.2.3 The Participation of Objects in Events

Events existentially depend on the objects that participate in them. Since an atomic event is a manifestation of a disposition of an object, we have that any atomic event depends on exactly one object, which is its unique participant. This dependency relationship is the perdurant counterpart of the inherence dependency relationship between tropes and their bearers.

A complex event is also an existentially dependent entity. Due to the extensionality principle of our event mereology, a complex event e' existentially depends on all its proper parts and, indirectly, on the participating objects these proper parts depend on.

The existential dependence of events on objects provides for an orthogonal way of partitioning complex events. Let us take as an example, the complex event e_{12}: *the stabbing of Caesar by Brutus*. This event can be decomposed into the events e_{Brutus}, e_{Caesar}, e_{dagger}, which depend (in the technical sense above) on Brutus, Caesar and the dagger, respectively. We call the portion of an event, which depends exclusively on a single object, a *participation event*. As an orthogonal way of partitioning events, participation events can be atomic or complex.

Figure 2 summarizes the results of this subsection and depicts these two aspects on which events can be analyzed, namely, as entities with certain mereological structures, and as ontologically dependent entities consisting of a number of participation events. As expressed in Figure 2, the relations of exclusively *depends on*, *participation of* and the notion of *participation* itself are all derived notions (derived from the relations of parthood and existential dependence). Nonetheless, making the notion of participation explicit is of great importance from an ontological as well as a conceptual point of view.

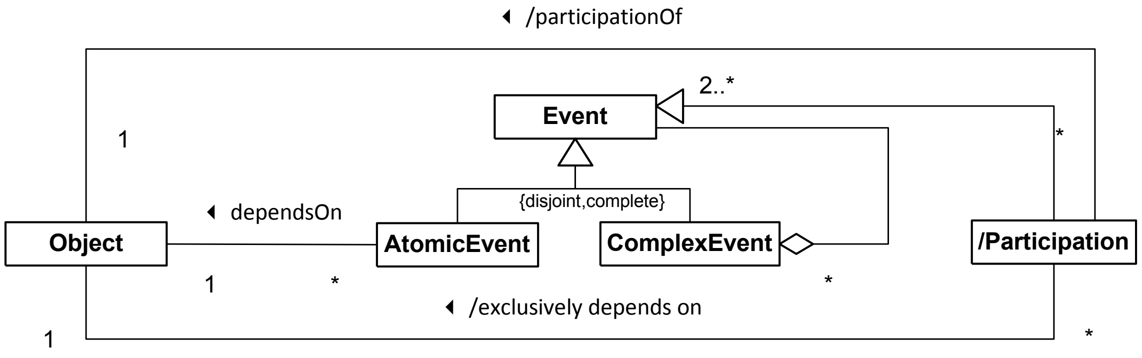

Figure 2: Complex and atomic events

While all temporal properties of objects are defined in terms of the events they participate in, all spatial properties of events are defined in terms of the spatial properties of their participants, as has been suggested in (Masolo et al. 2003).

3.2.4 Situations and Events

An event occurs in a certain situation at a certain point in time, and transforms it to another situation. Since events stand in such a close relationship to situations, we need to include situations as a basic concept in our ontological account of events.

In philosophical logic and in linguistics, it has been noticed that natural language sentences should be evaluated with respect to (a model of) the situation to which they refer and not with respect to (a model of) a 'world', as in classical predicate logic. In the influential *situation semantics* proposed by Barwise and Perry (1981), a situation is identified with a *state of affairs* in the sense of a set of related facts providing an account of the state of one or more objects, including the relationships in which they stand to each other. Situation semantics has also been proposed as a theory of the information content of sentences.

Another important approach is based on the notion of *possible situations*, which are parts of possible worlds in the sense of possible worlds semantics as originally proposed by Kripke (1963). In this approach, the meaning of a sentence (called 'proposition') is given by a set of possible situations, which can also be considered to stand for a *situation type*.

In UFO-B, we postulate that for any event *e*,

1. there is exactly one situation *s* (called the *occurrence situation* of *e*), in which it occurs, at the start time of *e*. This situation is determined by the objects participating in *e*;
2. *e* brings about a situation *s'* (called the *resulting situation* of *e*), which obtains at the end time of *e*.

Since an event brings about a state change to its occurrence situation, one may be tempted to identify events with such state changes. But for allowing different types of events having the same type of state change (that is, for supporting an intensional concept of events), we prefer to distinguish events from the state changes they bring about. However, we may consider a state change brought about by an event as a special type of caused event.

In the light of this discussion, we can interpret a statement that a particular situation (such as a metallic object *o* being sufficiently close to a magnet *m*) *triggers* an event (such as *o* moving towards *m*), as an incomplete sentence form that leaves the event that has brought about the situation implicit (the event that is responsible for *o* getting close enough to *m*). Consequently, events, and not situations, trigger events.

3.2.5 Causation

In general, different accounts can be given for the causation of events. A caused event could be considered to be caused either by a preceding event, or by a state of affairs, or by a combination of both.

For keeping track of causation in the execution history of a system, it seems natural and sufficient, to consider just the resulting sequences of caused events, where each causation step consists of a causing, or *triggering*, event and a caused, or *resulting*, event (or set of events), and hence the causation relationship holds between events.

However, such an extensional causation trace does not provide a satisfactory account of causation. Rather, what we need is a concept of *causal laws*, which are responsible for particular causations of resulting events *e'* and resulting situations *s'* by triggering events *e* in occurrence situations *s*. We will discuss this issue in the following section.

4 DISPOSITIONS AND CAUSAL LAWS

As pointed out by Choi and Fara (2012), many terms have been used in philosophy to describe what is meant by dispositions: ‘power’, ‘ability’, ‘potency’, ‘capability’, ‘tendency’, ‘potentiality’ and ‘capacity’, among others. Following (Mumford 2003), we consider dispositions as properties that are only manifested in particular situations on the occurrence of certain triggering events, and that can also fail to be manifested. When manifested, they are manifested through the occurrence of resulting events and state changes.

Take for example the disposition of a magnet *m* to attract metallic objects. The magnet *m* has this disposition even if it is never manifested, for example, because *m* is never close to any magnetic object. Nonetheless, *m* can certainly be said to possess this intrinsic property, which it shares with other magnets. Now, consider a particular metallic object o that has the disposition of being attracted by magnets. Given a situation in which *o* is sufficiently close to *m* (on a surface with a sufficiently low friction, etc.), the disposition of these two objects can be manifested through the occurrence of a complex event, namely, the movement of *o* towards *m*.

As pointed out by Choi and Fara (2012), we can distinguish two ways of referring to dispositions:

1. by such simple predicates as ‘fragile’, ‘soluble’, ‘flammable’, and so on, which include no explicit reference to their stimulus conditions and manifestations;
2. by expressions that are explicit about stimulus conditions and manifestations, such as “the disposition to break in response to being struck” or “the disposition to cause death in response to being ingested”.

The two ways of expressing dispositional properties are related to one another. Water-solubility might be identified with the disposition to dissolve in response to being put into water, although this kind of identification is not readily available in many other cases.

In UFO-B, we postulate that

1. dispositions *inhere* in (and therefore existentially depend on) particular objects;
2. dispositions are causally explanatory of their manifestations
3. all atomic events are manifestations of dispositions triggered by an event (occurring in a preceding situation);
4. if a disposition *d* inheres in an object *o*, and an atomic event *e* is a manifestation of *d*, then *e* is an atomic participation event with *o* being its unique participant (implying that *e* existentially depends on *o*).

Notice that a disposition d of an object implicitly defines two event types: one (TE_d) for the events triggering the disposition, and one (RE_d) for the resulting events being its manifestations. Consequently, a disposition d of an object o can be considered having the following form:

> *On the occurrence of an event e of type TE_d involving o in a situation satisfying certain standard state conditions, an event of type RE_d, involving o as a participant, and an accompanying state change, are triggered as a manifestation of d.*

A ***causal law*** is obtained by considering an object type *O* such that its instances o_i share a certain type of disposition *D*, which is instantiated by their particular dispositions d_i, and defining a triggering event type TE_D and a resulting event type RE_D:

> *On the occurrence of an event e of type TE_D involving an object o of type O in a situation satisfying certain standard state conditions, an event of type RE_D involving o as the unique participant, and an accompanying state change, are triggered as a manifestation of o's disposition of type D.*

An important question is how to define the *standard state conditions*, which correspond to a ceteris paribus clause. In philosophy, there is a debate if and how these conditions can be sufficiently defined, possibly in a probabilistic manner, such that the causal law is fully captured by the law statement. However, for our purpose of establishing an ontological account of the concept of transition rules in simulation languages, we can choose a form of rule that captures the underlying causal law only approximately, independently of this philosophical question.

In fact, we propose two possible approaches. In both approaches, the state condition of a transition rule is considered to be an incomplete approximation of the *standard state conditions* of the law to be captured by the rule. This is not deemed to be a problem since it is satisfactory for the simulation model expressed with the help of the transition rule to be an approximation only.

In the second approach, some form of probabilistic rules, where the resulting event(s) and state change(s) occur with some probability only, may be used for accounting for the approximate character of the rule's state condition.

5 ONTOLOGICAL FOUNDATIONS OF SIMULATION MODELING

In the DES literature, it is often stated that DES is based on the concept of "entities flowing through the system". However, the loose metaphor of a "flow" only applies to entities of certain types: events, messages, and material objects may, in some sense, flow, while many entities of other types, such as buildings or organizations, do not flow in any sense. Also, subsuming these three different kinds of flows under one common term "entity flow" obscures their semantic differences. It is therefore highly questionable to associate DES with a "flow of entities". Rather, one may say that DES is based on a *flow of events*.

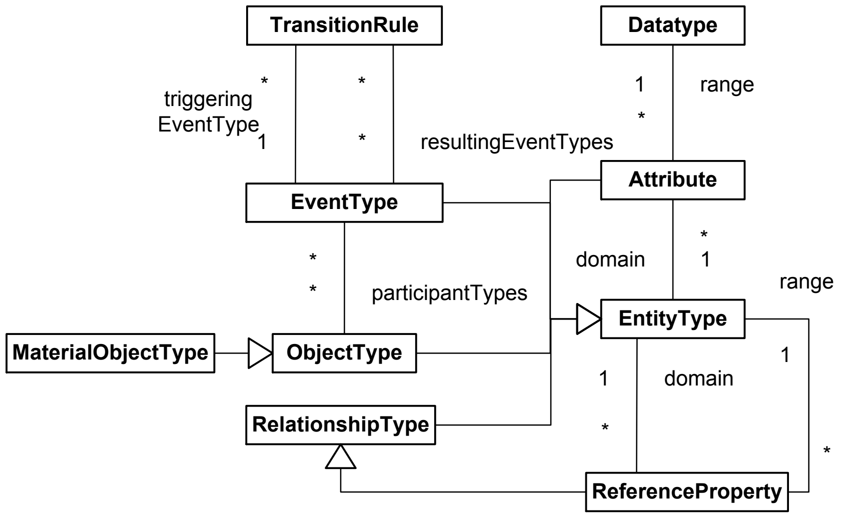

Figure 3: The basic type concepts of DESO.

Ontologically speaking, a discrete event system (or discrete dynamic system) consists of:

- ***objects*** of various types, whose *dispositions* may be triggered by
- ***events*** of various types occurring at times from a discrete set of time points;
- ***dispositions***, and ***causal laws***, determining the dispositional behavior of objects by relating disposition-triggering events and disposition-enabling situations with resulting state changes and resulting events.

For modeling a discrete event system, we have to do the following:

1. Define its ***object types*** $O_1,\ldots,O_n$ and ***event types*** $E_1,\ldots,E_m$.
2. Capture the *causal laws* of the system in the form of ***transition rules***, stating, for any disposition-triggering event type E_j, under which conditions the occurrence of an event e of that type involving one or more disposition-bearing objects $o_1,\ldots,o_k$ (each of some type O_i) leads to which ***resulting state changes*** of $o_1,\ldots,o_k$ and to which ***resulting events*** $e_1,\ldots,e_k$ (each of some type E_j) as manifestations of dispositions of $o_1,\ldots,o_k$.

This ontological account of discrete event simulation models improves the account given in (Guizzardi and Wagner 2010b, Guizzardi and Wagner 2011a) where we proposed the discrete event system ontology DESO and its agent-based extension ABDESO.

In the meta-model shown in Figure 3 above, we summarize the ontological type categories of DESO that form the foundation of conceptual simulation modeling languages. Entity types classify entities, which are said to be their instances. An entity type may be the domain of attributes and reference properties, which are also entity types since their instances, which are attributions and references, are entities (in fact, they are trope individuals). The range of an attribute is a datatype, which is an abstract thing (namely a structure consisting of a symbol set as the datatype's *lexical space*, an abstract set as its *value space* and a mapping from the lexical space to the value space). The range of a reference property is an entity type. Reference properties are (binary) relationship types.

Since objects may participate in events, an event type may have a number of object types as participant types. Transition rules capture the relevant causal laws defining the dynamics of the discrete event system. A transition rule has one type of triggering events and zero or more types of resulting events.

5.1 The Conceptual Process Modeling Language Onto-PMN

We briefly discuss how event types and causal laws can be expressed in conceptual models. Conceptual information models can be expressed in a variant of UML, which we call *Onto-UML*. Conceptual process models can be expressed in a variant of BPMN, which we call *Onto-PMN*, where a transition rule takes the form of a sub-process type (called *Sub-Process* in BPMN) that is part of a process type and has its own contextual scope. While a conceptual information model describes the informational aspects of objects and events, the associated conceptual process model describes the dynamic aspects, including causal laws and the succession of events. As opposed to Onto-UML, which has originally been proposed in (Guizzardi 2005) and has been used and validated in many modeling projects since then, Onto-PMN is still under development.

In the same way as the Onto-UML information modeling concepts do not depend on the language of UML Class Diagrams, the concepts of Onto-PMN do not depend on BPMN, which is used only due to its wide adoption in computer science and information systems, and its practical relevance.

Ontologically, BPMN Activities, including Tasks and Sub-Processes, are special event types. However, this subsumption of activities under events is not supported by the semantics of BPMN. It is one of the departures of Onto-PMN from standard BPMN.

5.2 Example: A Service Queue System

In the classical service queue system example, as implemented in the Simurena Library (Simurena 2012), customers arrive at random times at a service desk where they have to wait in a queue when the service desk is busy. Otherwise, when the service desk is not busy, they are immediately served by the service clerk. Whenever a service is completed, the next customer from the queue will be served, if there is any.

5.2.1 The Conceptual Information Model

Typically, in a simulation model we would make several simplifications and, for instance, abstract away from the object type ServiceClerk,. But in a conceptual system model, we include all entity types that are relevant for understanding the real-world system, independently of the simplifications we make in the solution design and implementation models.

After modeling all relevant object types in the first step, we model the relevant event types in a second step, as shown in Figure 4. The main type of association between events and objects is ***participation***. When adding event types to the object types in our conceptual information model, we therefore also model the participation types between them. For instance, in Figure 4, we have modeled that a customer arrival event has exactly one customer and one service desk as its participants.

From an analysis of the problem description, we may infer that there are six potentially relevant types of events in this system: customers arriving, customers getting in line, customers being invited by the clerk for being served, start of service, end of service and customers departing. Since the service queue system is an example of a business system, it is not surprising that all these events are *actions* (or *action events*) performed by human actors. In fact, we obtained this list of event types by asking ourselves: what are the relevant actions of the two actors of the system, *customers* and *service clerks*?

Notice that in order to complete the model, we would have to add the attributes needed for describing objects and events of these types.

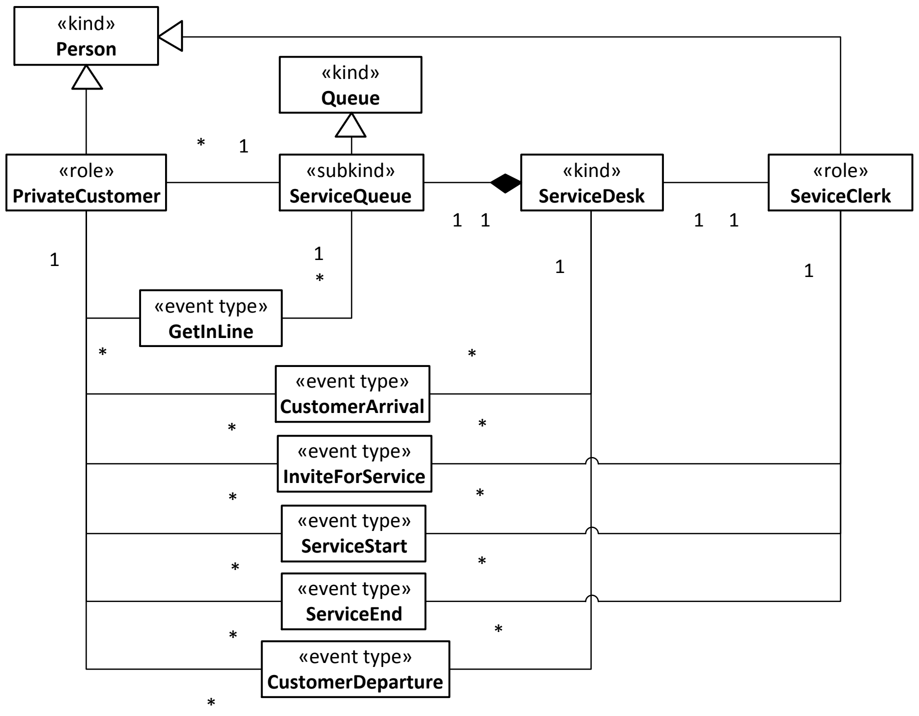

Figure 4: Adding event types.

5.2.2 The Conceptual Process Model

For making the conceptual process model, we start by identifying those types of events that account for the causation of state changes and follow-up events. They trigger a *causal law*. Any event type modeled in the information model could potentially trigger a causal law. So, we could model six causal laws, one for each type of event described in the information model. For simplicity, however, we consider only two laws: one for customer arrival events and one for customer departure events. We omit service start events, since they can be viewed to coincide either with customer arrival events, when the service queue is empty, or with customer departure events, when the queue is non-empty. In a similar way, service end events can be viewed to coincide with customer departure events.

Our conceptual process model for the service queue system thus consists of two causal laws, as shown in Figure 5. The CustomerArrival law is triggered by a customer arrival event and causes a corresponding customer departure event, which triggers the CustomerDeparture law that, in turn, causes another customer departure event (for the next customer), if the queue is non-empty.

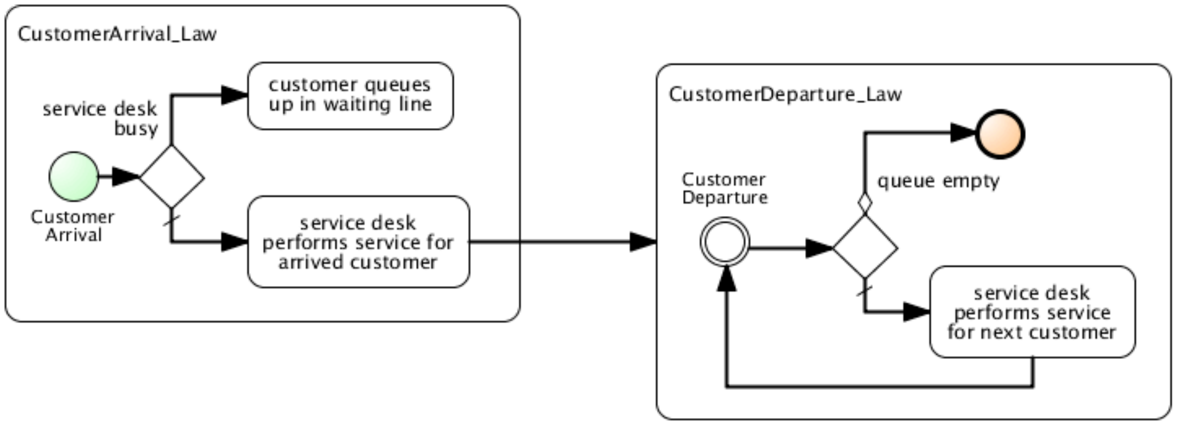
CustomerArrival_Law
service desk busy
customer queues up in waiting line
Customer Arrival
service desk performs service for arrived customer
CustomerDeparture_Law
Customer Departure
queue empty
service desk performs service for next customer

Figure 5: A conceptual process model of the service queue system describing two transition rules.

AKNOWLEDGEMENTS

This work is supported by FAPES (PRONEX grant #52272362).

REFERENCES

Allen, J.F. 1983. Maintaining Knowledge about Temporal Intervals, *Communications of the ACM*, Vol.26, no. 11.

Barwise, J., and J. Perry, 1981. Situations and Attitudes. *The Journal of Philosophy*, 78: 668–91.

Benjamin P., M. Patki, and R. Mayer. 2006. Using Ontologies for Simulation Modeling. In L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, (eds.), *Proceedings of the 2006 Winter Simulation Conference*, pp. 1151–1159.

Choi, S. and M. Fara. 2012. Dispositions. *The Stanford Encyclopedia of Philosophy*, Spring 2012 Edition, Edward N. Zalta (ed.), http://plato.stanford.edu/archives/spr2012/entries/dispositions/

Guizzardi, G., and G. Wagner. 2005. Some Applications of a Unified Foundational Ontology in Business Modeling, Business Analysis with Ontologies, IDG Group.

Guizzardi, G., Falbo, R.A., and R.S.S. Guizzardi. 2008. Grounding software domain ontologies in the Unified Foundational Ontology (UFO): The case of the ODE software process on-tology, 11th Iberoamerican Conference on Software Engineering (CIbSE 2008). pp. 244–251.

Guizzardi, G. 2005. *Ontological foundations for structural conceptual models*. PhD thesis, University of Twente, Enschede, The Netherlands. CTIT Ph.D.-thesis series No. 05-74 ISBN 90-75176-81-3.

Guizzardi, G., H. Herre, and G. Wagner. 2003. On the General Ontological Foundations of Conceptual Modeling", 21st *International Conference on Conceptual Modeling (ER-2002)*. Springer-Verlag, Berlin, Lecture Notes in Computer Science 2503, 65-78.

Guizzardi, G., and G. Wagner. 2010. Towards an Ontological Foundation of Discrete Event Simulation. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E (Eds.), Proceedings of Winter Simulation Conference, Baltimore (MD), USA, 652−664. Available from: http://www.informs-sim.org/wsc10papers/059.pdf

Guizzardi, G., and G. Wagner. 2011a. Towards an Ontological Foundation of Agent-Based Simulation. In: S. Jain, R.R. Creasey J. Himmelspach, K. P. White, and M. Fu. Proceedings of Winter Simulation Conference. Phoenix, Arizona, USA, 284−295. Available from: http://www.informs-sim.org/wsc11papers/024.pdf

Guizzardi, G., and G. Wagner. 2011b. Can BPMN Be Used for Making Simulation Models? In: J. Barjis, T. Eldabi and A. Gupta (Eds.). *Enterprise and Organizational Modeling and Simulation*. Springer Lecture Notes in Business Information Processing, vol. 88.

Herre, H. 2010. General Formal Ontology (GFO): A Foundational Ontology for Conceptual Modelling, *Handbook of Theory and Application of Ontologies*, Springer-Verlag.

Kripke, S. 1963. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, 16: 83-94.

Livet, P., J.-P. Müller, D. Phan, and L. Sanders. 2010. Ontology, a Mediator for Agent-Based Modeling in Social Science. *Journal of Artificial Societies and Social Simulation* **13**:1, Available via <http://jasss.soc.surrey.ac.uk/13/1/3.html> [accessed April 9, 2010].

Masolo, C., S. Borgo, A. Gangemi, N. Guarino and A. Oltramari, 2003. 'Ontology Library', WonderWeb Deliverable D18.

Mumford, S. 2003. *Dispositions*. Oxford University Press.

Pegden, C.D. 2010. Advanced Tutorial: Overview Of Simulation World Views. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E (Eds.), Proceedings of Winter Simulation Conference, Baltimore (MD), USA, 643−651.

Sierhuis, M. 2001. "Modeling and Simulating Work Practice. BRAHMS: a multiagent modeling and simulation language for work system analysis and design." Ph.D. thesis, Social Science and Informatics (SWI), University of Amsterdam, SIKS Dissertation Series No. 2001-10, Amsterdam, The Netherlands, ISBN 90-6464-849-2.

Silver, G. A., K. R. Bellipady, J. A. Miller, W. S. York, and K. J. Kochut. 2009. Supporting Interoperability Using the Discrete-Event Modeling Ontology (DeMO). *Proceedings of the 2009 Winter Simulation Conference (WSC'09)*, Austin, Texas, December 2009, pp. 1399–1410.

Simons, P. 1997. Parts: *A Study in Ontology*, Oxford University Press.

Simurena Library. 2012. A public library of simulations and games for education and entertainment. Accessed May 23, 2012. http://portal.simulario.de/public/.

Turnitsa, C., J.J.. Padilla, and A. Tolk. 2010. Ontology for Modeling and Simulation. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E (Eds.), Proceedings of Winter Simulation Conference, Baltimore (MD), USA, 643−651. Available from: http://www.informs-sim.org/wsc10papers/058.pdf

UFO (Unified Foundational Ontology). 2012. http://www.ufo-ontology.info/. Accessed May 23, 2012.

Zeigler, B. 1976. *Theory of Modeling and Simulation*. Wiley Interscience, New York.

AUTHOR BIOGRAPHIES

GIANCARLO GUIZZARDI is Associate Professor at the Computer Science Department, Federal University of Espírito Santo (UFES), Brazil, and senior member of the Ontology and Conceptual Modeling Research Group (NEMO). His work is focused in the development of domain ontologies and foundational ontologies and their application in computer science and, primarily, in the area of conceptual modeling and organizational modeling. He has been involved in a number of industrial projects in domains such as off-shore software development, petroleum and gas, medical informatics, telecommunications and news information management. His email address is gguizzardi[at]inf.ufes.br.

GERD WAGNER is Professor of Internet Technology at the Department of Informatics, Brandenburg University of Technology, Germany. His research interests include agent-oriented modeling and agent-based simulation, foundational ontologies, (business) rule technologies and the Semantic Web. In recent years, he has been developing an agent-based discrete event simulation framework, called *ER/AOR Simulation* (see www.AOR-Simulation.org). His email address is G.Wagner[at]tu-cottbus.de.