Ontological Issues in Conceptual Simulation Modeling

Gerd Wagner

Brandenburg University of Technology, Konrad-Wachsmann-Allee 5, 03046 Cottbus, Germany

Abstract
We present the approach of Object Event Modeling and Simulation as a foundation of Discrete Event Simulation and identify a number of open issues concerning the ontological grounding of several basic concepts of it, including discrete and continuous processes, activities, causal regularities and dynamical systems.

Keywords
Object Event Modeling, Object Event Modeling Simulation, ontological grounding

1. Introduction

Simulation models are to represent and imitate *dynamical systems*. They may take many forms and there are many languages and formalisms for making them. Based on the distinction between *discrete* and *continuous* dynamical systems, there is a distinction between discrete and continuous simulation models. While continuous simulation models are expressed with the help of object types and differential equations that capture the rate of change of certain object attributes considered as the relevant state variables, discrete simulation models are expressed, explicitly or implicitly, with the help of object types and the types of events that are responsible for the state changes of objects.

Discrete simulation models are also called *Discrete Event Simulation (DES)* models. Remarkably, the leading DES tools, AnyLogic, FlexSim and Simio, do not share a common conceptual/theoretical foundation, rather, each of them is based on its own concepts, terminology and diagram languages.

The *Object Event Modeling and Simulation* paradigm proposed in [1,2] is an attempt to provide a common conceptual/theoretical foundation for DES.

2. Object Event Modeling for Discrete Event Simulation

While research in Business Process Modeling has focused on events and processes, neglecting objects, research in Conceptual Modeling for Information Systems (IS) engineering has focused on objects, neglecting events. *Object Event Modeling (OEM)* reconciles both perspectives, giving equal weight to objects and events, which are the two most important ontological categories, called *endurants* (or *continuants*) and *perdurants* (or *occurrents*) in philosophy. The fundamental dichotomy between objects and events is reflected in two corresponding parts of the *Unified Foundational Ontology (UFO)*: UFO-A and UFO-B [3].

An *Object Event (OE) model* is a triple ⟨*OT, ET, R*⟩, consisting of a set of object types *OT*, a set of event types *ET* and a set of event rules *R*. While object types and event types can be defined in an information model, such as an *OE Class Model*, event rules can be defined in a discrete process model, such as an *Object Event Graph* or an *Activity Network* [2].

The formal semantics of an OE model ⟨*OT, ET, R*⟩ is defined in [4] in the form of an *Abstract State Machine*, which is a special kind of state transition system, whose state structure is defined by the object and event types of *OT* and *ET*, and whose transition functions are provided by the event rules of *R*. Since an OE model is executable, it is a special type of a *Discrete Event Simulation (DES)* model.

2.1. Information Modeling with Object Event Class Models

An OE class model is a UML class model with the class stereotypes «object type» and «event type» (and possibly also with «activity type»), defining the types of objects and events of a problem domain with their attributes, associations, operations and constraints. An example of an OE class model is shown in Figure 1.

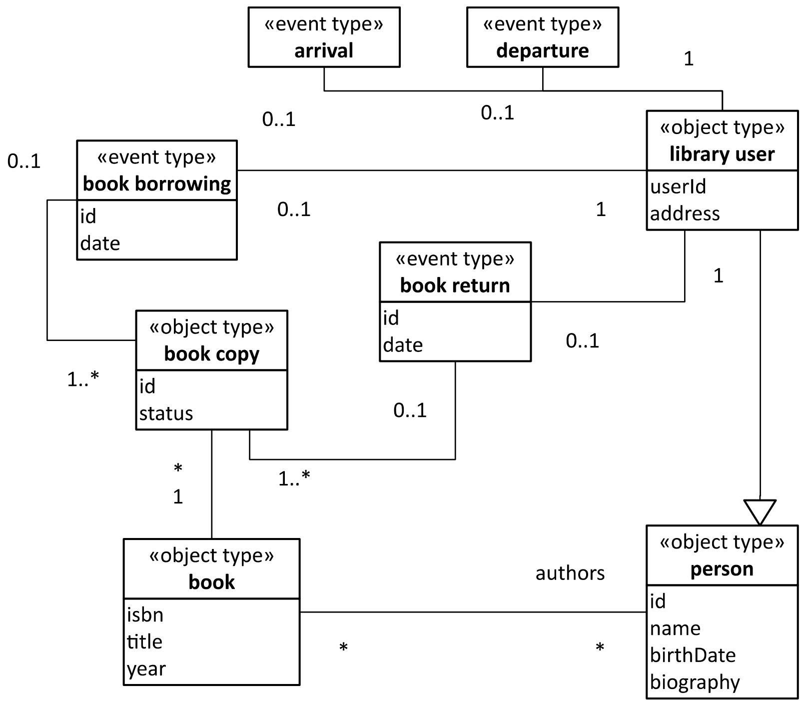

Figure 1: A conceptual information model in the form of an OE class model defining 4 object types and 4 event types.

Notice that the possible values of the *status* attribute of the class *book copy* are AVAILABLE and LENDED (to be defined in an enumeration datatype in a design model).

For the participation associations between object types and event types, the model shows a *snapshot multiplicity* at the side of the event types. For instance, for the association between *library user* and *book borrowing*, the multiplicity 0..1 shown at the side of *book borrowing* implies that at any moment in time, a library user participates in at most one book borrowing event. By default, if it is not shown as a second multiplicity annotation of such an association end, the *history multiplicity* is 0..*, which means that over time, a library user may participate in zero or many book borrowing events.

By default, the event types in an OE class model are types of instantaneous events, which implicitly have an attribute *occurrence time*.

2.2. Process Modeling with Object Event Graphs

Object Event Graphs extend *Event Graphs* [5] by (1) allowing to attach for any type O of objects participating in events of type E a corresponding object rectangle O to the event circle E and (2) by allowing for an Object-Oriented state change language for expressing state change statements in such an object rectangle.

The names of event circles and object rectangles in an OE graph refer to the corresponding types defined in an underlying OE class model.

OE Graphs define visually for all relevant types of events *E*,

1. the (possibly conditional) *state changes* and
2. the (possibly conditional) *follow-up events*

caused (or triggered) by an event of such a type. Thus, an OE graph defines a set of ***event rules*** of the form

ON *E* STATE CHANGES … FOLLOW-UP EVENTS …

capturing ***causal regularities***.

An example of an OE graph that is based on the OE class model of Figure 1 is shown in Figure 2.

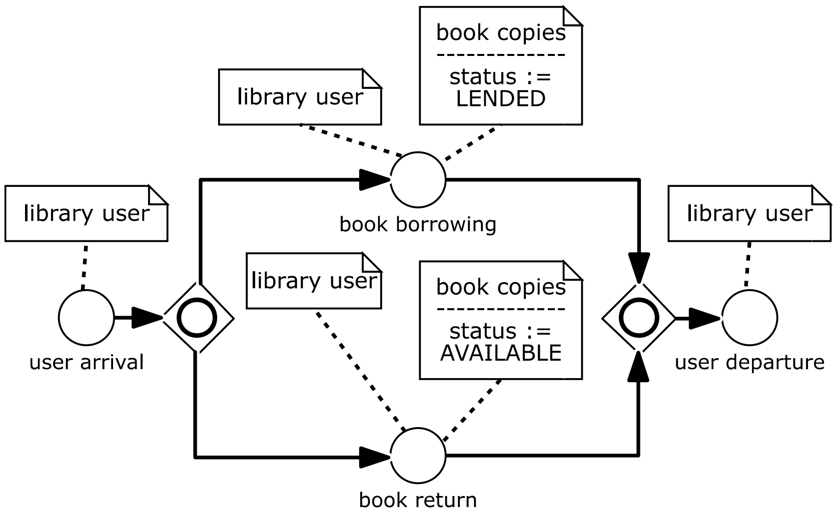

Figure 2: A conceptual process model in the form of an OE graph showing for each type of events, which state changes and follow-up events are caused by an event of that type.

OEM's notation for OE Graphs is based on BPMN, thus the gateway shape with the inner circle denotes non-exclusive conditional branching (and merging).

Notice that when a *book borrowing* event occurs, for all borrowed books, their *status* attribute is set to the value LENDED. Likewise, when a *book return* event occurs, for all returned books, their *status* attribute is set to the value AVAILABLE. Thus, the event rules for the two event types *book borrowing* and *book return* as described in the OE Graph of Figure 2 are as follows:

ON *book borrowing*
STATE CHANGES: *status* := LENDED for all borrowed book copies
FOLLOW-UP EVENTS: *user departure*

ON *book return*
STATE CHANGES: *status* := AVAILABLE for all returned book copies
FOLLOW-UP EVENTS: *user departure*

2.3. Process Modeling with Activity Networks

In OEM, an activity is a special kind of non-instantaneous event that is composed of an instantaneous activity start event followed by an instantaneous activity end event. OE class models allow to model activity types, in addition to object and event types, as shown in Figure 3, where the event types *book borrowing* and *book return* of the model of Figure 1 have been replaced by the corresponding activity types *book lending* and *book take back*.

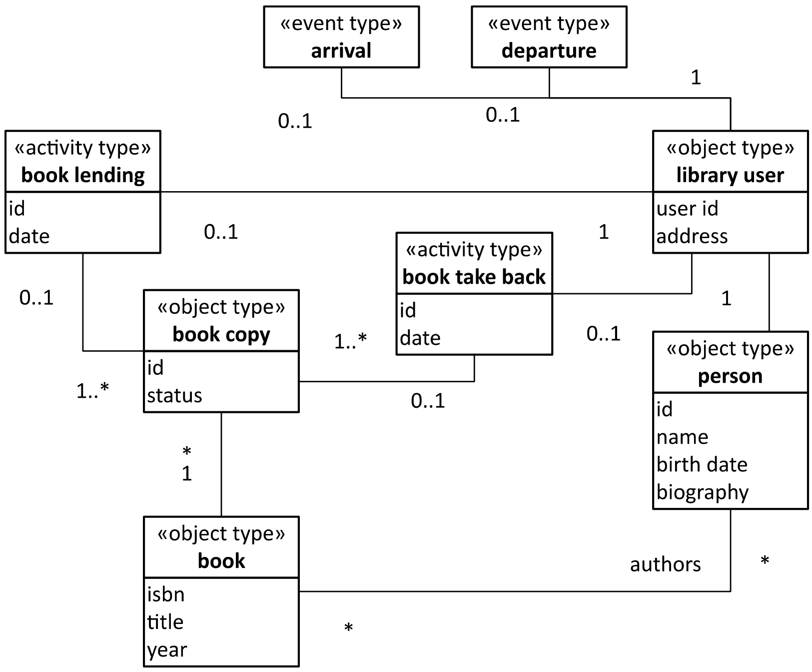

Figure 3: A conceptual information model in the form of an OE class model defining 4 object types, 2 event types and 2 activity types.

Modeling *book lending* and *book take back* as activity types implies that lending and taking back books takes some time and there are implicit activity start and end events. In OEM, activity types have the implicit attributes *start time*, *duration* and *occurrence time*. Activities, as events, occur when they end (or complete). Thus, it holds that

duration = occurrence time − start time.

Whenever the start events and the durations of activities of a certain type do not matter, an event type that represents the corresponding activity end events can replace the activity type. This is how we could derive the OE class model of Figure 1 from the one of Figure 3.

Conversely, whenever an event type in an OE class model, or an event circle in an OE graph, represents a type of activity end events, it can be replaced with the corresponding activity type or activity rectangle. This is how we obtain the activity network shown in Figure 4 from the OE graph of Figure 2.

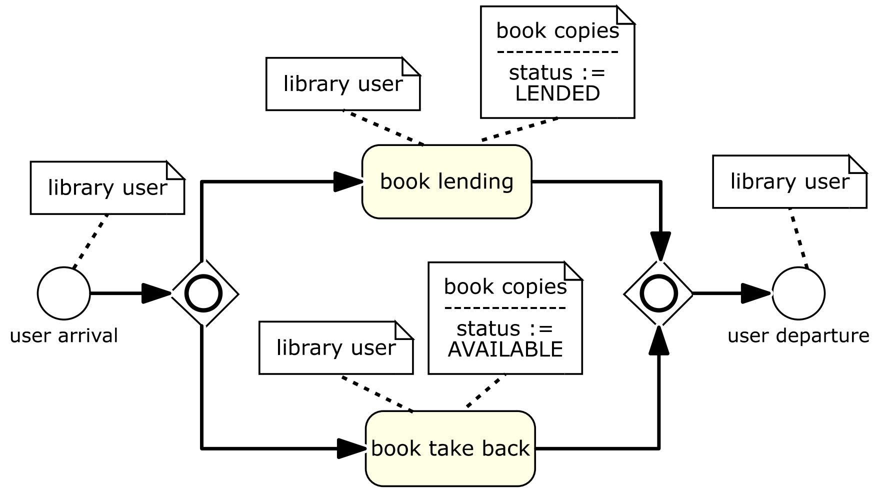

Figure 4: A conceptual process model in the form of an activity network showing for each type of events and activities, which state changes and follow-up events are caused by an event or activity of that type.

3. Ontological Issues of Object Event Models

Although the ontological grounding of OEM's concepts of objects (and object types) as well as events (and event types), and the new types of *participation* associations between object types and event types, are provided by UFO-A and UFO-B [3], there is still some mismatch between OEM and UFO-B/OntoUML, and there are several open issues concerning the OEM concepts of (discrete and continuous) processes, activities, causal regularities and dynamical systems.

3.1. Mismatch between OEM and UFO-B resp. OntoUML

OEM has been developed independently of, and prior to, the extended version of OntoUML that covers event modeling [6]. There are the following mismatch items:

1. In OEM, creation and termination of objects have not (yet) been considered and a commitment to a “historical semantics” not only for event types, but also for object types, has been avoided. For practical modeling purposes it is preferable not to impose a “historical semantics” on all object types, which would require to keep terminated objects in the universe of discourse (and in the extension of corresponding database tables), but only impose such a semantics on event types (requiring to keep historical event records in the underlying database).
2. While OntoUML only considers history multiplicities at the event type association ends of participation associations, OEM allows both for snapshot multiplicities and for history multiplicities. For instance, in the OE class model of Figure 1, the participation association between the object type *book copy* and the event type *book return* has an explicit snapshot multiplicity of 0..1 and an implicit (default) history multiplicity of 0..* expressing that at any moment in time a book copy can participate in at most one book return event, but over time it can participate in many book return events.

3.2. Processes

UFO-B does not (yet) cover the ontological foundations of processes as a particular category of perdurants. Only recently, in [7], Guarino and Guizzardi have been proposing a new ontological theory of processes essentially stating that

1. For allowing the same process to be ongoing (while “accumulating events”) and later completed, processes should be considered to be “variable embodiments” in the sense of [8].
2. The completion of a process creates a corresponding (process-as-)event with the same start and end time.
3. There are a number of conceptual distinctions among basic process kinds: (1) structurally *homogeneous* processes, (2) *intentional* processes, and (3) *telic* processes. E.g., business processes are complex telic processes.

However, Guarino and Guizzardi do neither consider continuous processes, which are not “accumulating events”, nor the distinction between continuous and discrete processes.

3.3. Activities

Activities are a category of processes: they may be both ongoing and completed. While completed activities may be loosely identified with the corresponding (activity-as-)event, this does not apply to ongoing activities.

In [7], activities are defined as “processes whose manifestations are sequences of intentional acts of the same kind, and are described by verbal expressions such as walking, running, eating apples, etc.” However, this concept of activities as quasi-homogeneous processes deviates from the ordinary language use of the term, e.g., in the area of Business Process Management, where any subprocess performed by the same acto(s) can be considered to be an activity.

3.4. Causal Regularities

In UFO-B, causation has only be considered at the level of individuals (in the form of events causing other events), but not as a pattern, or *causal regularity*, at the level of types.

In OEM, *event rules* express causal regularities where events of a certain type cause state changes of affected objects and follow-up events.

3.5. Dynamical Systems

The concept of dynamical systems is widely used in mathematics and the natural sciences, but also in certain social sciences such as economics. In some works, the term "discrete dynamical system" is mistakenly defined as a dynamical system for which time is discrete.

The general concept rather refers to the nature of the state changes of a system. If a system has only continuous state changes, it is a continuous dynamical system, while if it has only discrete state changes, it is a discrete dynamical system.

UFO does not (yet) have a theory of dynamical systems.

4. Conclusions

We have presented a general approach to Discrete Event Simulation (DES) modeling, called Object Event Modeling, and have discussed the ontological grounding of its basic concepts. The goal of our discussion is to better understand the possibilities of an ontological grounding of DES modeling and the potential and shortcomings of UFO for this purpose.

References

[1] G. Wagner, Information and Process Modeling for Simulation – Part I: Objects and Events, *Journal of Simulation Engineering* 1 (2018) 1–25..

[2] G. Wagner, Object Event Modeling and Simulation, 2023. URL: https://dpmn.info/reading/OEMS.

[3] G. Guizzardi, A. Botti Benevides, C. M. Fonseca, D. Porello, J. P. A. Almeida, T. P. Sales, UFO: Unified Foundational Ontology, *Applied Ontology* 17 (2022) 1–44.

[4] G. Wagner, An Abstract State Machine Semantics for Discrete Event Simulation, in W. K. V. Chan et al (Eds.), Proceedings of the 2017 Winter Simulation Conference, IEEE, Piscataway, New Jersey, pp. 762–773.

[5] L.W. Schruben, Simulation Modeling with Event Graphs. *Communications of the ACM* 26 (1983) 957–963.

[6] J.P.A. Almeida, R. Falbo and G. Guizzardi, Events as Entities in Ontology-Driven Conceptual Modeling, Proceedings of 38th International Conference on Conceptual Modeling, , ER 2019, Salvador, Brazil, November 4–7, 2019.

[7] N. Guarino and G. Guizzardi, Processes as variable embodiments. Synthese 203, 104 (2024). https://doi.org/10.1007/s11229-024-04505-2

[8] K. Fine, Acts and embodiment, Metaphysics, 5.1 (2022) 14–28.