Working Simulations with a Foundational Ontology

Robert Porzel and **Tobias Warden**[1]

Abstract. Simulation is a commonly used method, for example in economics, engineering or computer science. In many cases simulations become difficult to design and control, especially when the simulated actors need to possess, acquire or process knowledge in some form. Today there exist simulation systems that are based on the multi-agent paradigm and use formal ontologies for representing the world, internal states of an agent or domain knowledge. However, such systems frequently feature individual non-congruent models that are not aligned with state-of-the-art foundational ontology frameworks. In this work we outline a foundational simulation ontology (FSO) that models simulations themselves so that not only the domain-specific models can be aligned to a standardized upper level, but also scenario design for ontology-based simulations can be performed by applying state-of-the-art ontology engineering principles. Our showcase scenario is that of autonomous logistic processes, where we apply our model for the domain of transport logistics.

1 Introduction

Simulation constitutes a well-established method for the examination of specific properties of the individual approaches that are being simulated. Early examples of this method can be found in architectural engineering *simulations*, where small scale models were constructed to test the static properties of the real entity to be constructed. Today, simulations do not require *in vivo* physical models, but can be set up and examined *in silico*. Numerous techniques have been employed in these software-based simulations. In this field of study some well-established paradigms have emerged, such as equation-based approaches or ones using PetriNets [4]. More recently, also simulations that are based on the multi-agent paradigm have been examined and applied [30].

Some examples for such multiagent-based simulations are found for system analysis and evaluation in a variety of domains ranging from models of bee recruitment [22] to simulating complex business processes, such as for supply networks [3]. Generally speaking, the agent-based simulation paradigm lends itself well to the simulation of complex systems on the micro-level where individual decision makers are modeled explicitly as autonomous agents embedded in dynamic environments. More specifically, and in contrast to alternatives such as equation-based modeling, the agent-centered approach to modeling facilitates the design of complex technical systems [30] due to:

- the opportunity for task decomposition,
- a natural mapping from real-world actors or entities to agents,
- a focus on modeling of individual behavior,
- the availability of multiagent system development frameworks.

[1] University of Bremen, Germany, email: {porzel,warden}@informatik.uni-bremen.de

Still, the concrete decision to embrace multiagent-based simulation for evaluation purposes when starting from a blank slate, is often perceived as a mixed blessing as the effort required to design particular multiagent-based simulations, especially when increasing the number of involved agents and/or with higher environmental complexity, often exceeds that of comparable simulation approaches, such as PetriNets or queuing networks [22]. Off-the-shelf agent frameworks are typically not designed to consider simulation-specific issues, such as synchronization, for which solutions exist in standard simulation approaches [4]. In addition, the design of the simulation environment itself in which the multiagent system can be placed, requires significant development resources. Thus, there is still an engineering challenge for multiagent-based modeling and simulation to be addressed [34]. More recently, efforts have been undertaken to employ formal and explicit knowledge models, i.e. formal ontologies, for describing the simulation world model [15]. Despite the inferential benefits gained from employing ontologically modeled knowledge for multiagent-based simulation, we have identified several remaining issues that need to be addressed for creating portable and (re-)usable knowledge models simulation systems:

- there still remains a substantial hurdle for the engineer to design and configure a simulation,
- currently employed domain models provide little extensibility and do not scale well to other application domains and simulation systems,
- the need for contextual reification is not addressed so that agents only poorly cope with dynamic changes in the environment that need to be reflected in revisions of their individual conceptualizations of the (simulated) world.

In this work, we firstly present the state of the art in ontology-based multiagent simulation systems and show how the aforementioned challenges for creating scalable simulations[2] in complex domains can be approached while - at the same time - lowering the hurdle for the non-ontologist to design the simulations accordingly. To that end we propose a foundational simulation ontology which models simulations *per se* and thereby provides a set of ontological patterns for *plugging in* the domain-specific ground knowledge of what is being simulated. On that basis the proposed framework additionally allows for a context-dependent reification of *ground* domain entities by means of including an additional set of descriptive patterns.

[2] Please note that *scalable* as it is employed here does not refer to scaling the number of agents engaged in a simulation, but rather means going from comparatively simple simulations to more complex ones.

2 State of the Art

As stated above, simulations are an appropriate and widely used means to evaluate the performance and adaptation abilities of systems in dynamic environments. For many domains, such as logistics or communication networks, discrete event simulation [4] has been the predominant simulation technique. However, in the case of more complex models, sequential simulation will exhibit poor runtime performance. For this, parallel distributed simulation systems have been developed [7] that enable the integration of simulation hardware as well as different simulation systems. Therefore, a so-called *High Level Architecture* [20] has been proposed defining services and interfaces to couple multiple simulation systems with potentially different simulation purposes. An alternative approach to distributed simulation of complex systems is constituted by multiagent-based simulation (MABS). In this case, the simulation model comprises multiple autonomous interacting software agents [37] that may run concurrently and distributed over multiple computers. The fundamental motivation behind multiagent-based simulations is to model agents as a direct and natural mapping of the simulated real-world entities acting in the simulation. This is particularly useful if the simulated world consists of multiple technical systems, organizations, and/or humans. Consequently, a major application domain has been social simulation [5, 30]. Beyond simulation distribution, the model decomposition to agents in multiagent-based simulation allows for extension, integration, and substitution of agents. Researchers or other system users that agree on an environment and interaction model for the domain of interest can add their own agents in a common scenario. The respective agents could be evaluated in separate simulation experiments or even concurrently in a competitive manner. The latter case, for instance, is applied in the RoboCup soccer simulation league [25].

Still, the research challenge of how best to design a simulation and create the corresponding agents' world models remains an open issue. For approaching these questions, we concentrate in particular on situation-aware agents which extend and substantiate the classical rational agent definition. For this category of agents, their respective belief about the world is no longer taken for granted. It is rather actively controlled by information acquisition as an additional meta-level reasoning process [14, 23]. Modeled human and artificial *decision makers* in our interpretation include both human controllers in the simulated processes as well as inanimate entities participating agentively in the simulation.

With regard to the representation of ground domain knowledge in the field of logistics, domain-specific ontologies have been compiled which formalize and explicate the domain at hand, e.g., transport networks, transport and production logistics, or physical goods in the logistics domain [34]. Despite these forerunners in building explicit formal models for simulating the properties of autonomous agents, there are several leaps remaining to be made in order to pass the aforementioned hurdles regarding the simulation design processes, the scalability and re-usability of the ensuing systems and the agent's respective ontological flexibility in terms of possessing the descriptive powers to construe their world dynamically and independently.

In the following, we will discuss how the research challenges, listed in Section 1, have been approached in the knowledge management sub-project of the Collaborative Research Center 637 on autonomous logistic processes and their limitations[3]. In this research effort, rational agents and multi-agent systems have been identified as a vantage point for conceptual modeling and simulation of cooperating logistic decision makers, with a particular focus on situation-aware agents which extend and substantiate the classical rational agent definition [36]. As mentioned above, for this category of agents, their respective belief about the world is no longer taken for granted.

Next, we will, therefore, discuss how a firm ontological grounding can be established that satisfies the requirements of simulating autonomous logistic processes with multi-agent systems.

3 Adding Foundational Knowledge

In case an ontology is to be used solely for capturing and representing knowledge for a specific resource within a given community and if the intended meaning of the terms used within the respective community is generally understood and agreed upon in advance by all its members, then little is to be gained by the employment of a foundational ontology, such as DOLCE or SUMO [24, 27]. If, however, an ontology is to be extended or re-used in different settings or even ported to new domains and applications, then a foundational ontology becomes indispensable, as previous attempts on building scalable knowledge models have shown [12, 11, 28]. While these matters have been discussed before and are by now widely accepted in the ontology engineering community a new level of complexity is added for the case of simulating logistic processes with autonomously acting and learning agents as we will discuss in Section 3.2 below.

Generally speaking, a foundational ontology constitutes an axiomatic theory about the high-level domain-independent categories in the real world [17], such as object, attribute, event, spatial and temporal connections and the like. The purpose of a foundational ontology is to act as a modeling basis for building individual domain ontologies, e.g., an ontology of logistic processes and objects as in our showcase application. Equally important is that foundational ontologies provide ontology design patterns that prescribe best practice modeling, avoid ontological idiosyncrasies and save a substantial amount of modeling effort [9]. In short, the benefits of using a foundational framework are that:

- foundational ontologies provide a reference point for comparing different possible ontological approaches, and are useful for analyzing, harmonizing, and integrating existing domain ontologies and specific metadata standards, such as electronic product catalogs of existing logistic IT systems;
- foundational ontologies also provide a starting point for the design of new domain ontologies - rather than having to begin from scratch a foundational framework provides a predefined set of ontological entities that can be extended for the specific domain ontologies;
- ideally, the foundational ontology also provides applicable ontology design patterns for handling re-occurring modeling needs, such as the location of an entity in space and time.

In this work we will highlight the latter feature in our description concerning the reification of ground objects, where several design patterns can be applied. In this respect also our choice of employing DOLCE as foundational ontology was motivated by the need to enable agents to adapt their conceptualizations and create new ones *on the fly* because several dedicated DOLCE-modules provide a corresponding framework, as we will discuss in greater detail below. DOLCE belongs to the WonderWeb library of formal ontologies [24].[4] It is intended to act as a starting point for comparing and

[3] Web site: `www.sfb637.uni-bremen.de`

[4] More detailed comparisons of DOLCE to other foundational frameworks

elucidating the relationships with other ontologies of the library - so-called *modules*. DOLCE is based on the fundamental distinction between enduring entities (i.e., objects or substances) and perduring entities (i.e., events or processes). The central relation between *Endurants* and *Perdurants* is that of participation. For example an *Endurant*, such as a transportation vehicle, participates in a set of possible activities, such as driving or parking, which in their nature are *Perdurants*. DOLCE also introduces *Qualities* as another category that can be seen as the basic entities one can perceive or measure: shapes, colors, sizes, sounds, smells, as well as weights, lengths or electrical charges. Spatial locations constitute a special kind of physical quality and temporal qualities encode the spatio-temporal attributes of *Endurants* and *Perdurants*. Finally, *Abstracts* do not have spatial or temporal qualities and they are not qualities themselves. An important example for the domain of logistics are so-called *Regions* that are used to encode the measurement of qualities as conventionalized in some metric or conceptual space.

Furthermore, DOLCE features a rich reference axiomatization in modal logic (S5), which captures basic ontology design patterns, such as location in space and time, dependence or parthood. The ontology core is minimal in that it only includes the most general concepts and patterns. This, as pointed out above, makes it well-suited for modularization. In fact, there is a wealth of additional theories that can be included on demand, which we will take as our starting point for modeling simulations themselves as part of a foundational system of ontologies.

3.1 Separating the Real from the "Not So Real"

Before starting to model domain-specific entities and the corresponding logistic processes, we had to ask ourselves whether the existing ontological infrastructure provides suitable starting points for our modeling purposes. As a result of this, we have chosen to employ a set of ontological modules which have been designed along with the DOLCE framework, namely:

- the Ontology of Information Objects [16];
- the Ontology of Plans [10];
- the Core Software Ontology [29];
- the Descriptions and Situations module [13].

Before exemplifying the usage of these ontological modules, we provide an overview of this infrastructure and how our foundational simulation ontology is fitted into it in Figure 1.

As it is clear that a software agent that is simulating a logistic object - or any other agentively acting entity for that matter - is not the real thing itself, a proper ontological place for rational software agents needs to be established. Hereby the Core Software Ontology provides a fitting design pattern. Therein a differentiation is made between a *ComputationalActivity* and a *ComputationalObject*. While *ComputationalObjects* are actually modeled as physical objects, i.e. *Endurants*, in DOLCE - because they physically exist in the hardware and memory of a running computer system - a *ComputationalActivity* is a sub-class of *Perdurant* - because they refer to events or processes in which *ComputationalObjects* participate. It, therefore, becomes almost straightforward to model a software agent as an *AgentivePhysicalObject*. This is not because it might *control* a physical object, but because it itself is an agentive object constituted in the physical matter of a *ComputationalObject*. Consequently, a new

also reflect the individual ontological commitments that underly each modeling framework [28].

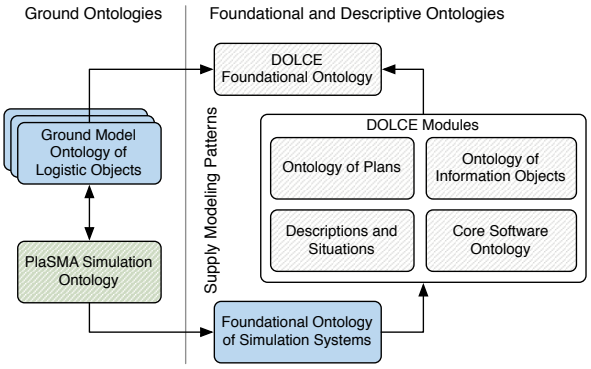

Figure 1. Framework for modeling multiagent-based simulation systems, grounded on DOLCE, using the showcase of simulating autnonomous logistic processes with the PlaSMA Simulation Framework [34].

class of *SimulatedAgentivePhysicalObjects* was added as a sub-class of *AgentivePhysicalObject*. This means that the entity simulating a logistic object is both a physical object and an agent itself and assumes the relation *internally-represents* which ranges over the class *Plan*. Thereby an explicit connection to the Ontology of Plans is established.

This example also constitutes a textbook showcase of the advantage of employing foundational systems and their modeling patterns: by *simply* finding the right place for the entity to be modeled, as shown above for the case of a simulation agent, one obtains a whole ensemble of relations and ranges that come from the corresponding super-classes that are part of the employed foundational module. These relations and ranges, in turn, actually serve the desired needs of the modeler. Of course, this is not a coincidence at all but a result of careful ontology engineering performed by the designers of the foundational framework. However, in many communities the employment of a foundational framework is still regarded as a cumbersome and often futile exercise, while - in our minds - rather the opposite is true.

Analogously, once a corresponding *SimulationObject* is inserted as a sub-class of *ComputationalObject*, we can express that:

- a *SimulationObject realizes* **some** *InformationObject*[5]
- a *SimulationObject participates-in* **some** *ComputationalActivity*

Also, by using the basic DOLCE design patterns to attribute spatial and other qualities to physical objects, we were able to model qualities - such as simulated traffic flow or simulated speed limits - of our simulated world by re-using the patterns established for real objects and their qualities.

For example, we also pointed to the additional challenge of allowing our software agents to learn in their environments and form new conceptualizations based on their respective body of experience in the simulated world, which will be described in the following section. Before that, however, we want to point at the emergent possibility to employ the Core Software Ontology even further, as the entire class hierarchy of the simulation software can now be described ontologically as well. This makes it feasible to interact - before and during a running simulation - not only with the simulation itself, but also with the underlying model. This, of course, presents additional challenges in the domain of human-computer interaction, but it puts the design of both the simulation and its software on an equal footing.

[5] Since *Software* is modeled as an *InformationObject* this solution expresses exacly what is needed for a foundational simulation model.

3.2 Reification of the Ground Domain

In recent work the notion of ontological design patterns has quickly become a central issue in ontology engineering research [8]. In its original form the first type of patterns, so called *logical patterns*, specifies ways of solving standard ontology modeling problems, such as how to model n-ary relations or how to employ subsumption macros [9]. The second type, so called *content patterns*, features applications of logical patterns, which means that instances of content patterns are composed of logical patterns and combinations thereof. Content patterns are concerned with specifying ways of representing everything that is not given by the logical vocabulary itself. For example, *is-a* relations come with the logical inventory, but *part-of* relations do not and are, consequently, part of the domain-specific content of the ontology. While the foundational layer provides the basic ontological distinctions, axiomatizations and design patterns for the development of further domain-independent and domain-specific layers of *ground* ontologies as well as additional layers of *descriptive* ontologies, which we will discuss below, it is important to note that this important distinction is primarily motivated as an ontological separation enabling an ontology engineer to express *reified* contexts [32] at the level of concepts or relations.

As a consequence of employing the Descriptions and Situations module [13], our simulation framework allows for the distinction between ground and descriptive models. Thus, we can provide a ground ontology of the logistic domain, featuring objects (such as trucks or containers), places (such as manufacturing and storage sites), as well as the transportation networks connecting them. However, these ground objects do not seek to model the roles that these entities can play - both in the real world or a simulated one.

For example, a truck can play the role of a means of transport. Nevertheless, it can also play the role of a freight object, if it is itself being transported. In an extreme case one can even consider that, for example, an entire storage facility is moved from one place to another and becomes logistic freight as a result and needs to be stored and transported itself.

The descriptive branch of the foundational ontology can then be applied for explicating such context-dependent reifications of real and simulated entities. For example, our proposed ontology framework consequently provides the logical and content patterns to express that a ground logistic class, such as a *Truck*, can be ontologically reified as a *MeansOfTransportation*, a *FreightObject*, or a *TrafficObstacle* depending on the context at hand.

As a result one can employ the same modeling instruments, including logical and content patterns, for descriptive entities as one employs for modeling ground entities. This, in turn, circumvents the need to resort to other logical instruments for describing these entities, such as to formulate so-called *theories* about the ground model, including theories about possible worlds or - in a weakened form - modal propositions about possible *situations* [2] that require universal algebra to express the semantics of the logical forms [35].

Therefore, the approach taken herein provides the possibility to employ the logical patterns, i.e. the specific logical vocabulary, of the given foundational ontology, for modeling descriptive entities in the same way as one does for the ground part. Again, this approach would be impossible without a foundational layer linking the ground and descriptive branches of the integrated ontology.

The DOLCE module *Descriptions and Situations* provides specific logical and content patterns for representing reified contexts and states of affairs [13]. In contrast to ground entities, such as physical objects or events, the extension of a descriptive ontology to include different conceptualizations that an autonomously learning logistic agent may derive of these entities poses a challenge to the ontology engineer. The reason for this circumstance is the fact that these conceptualizations are taken to assume meaning only in combination with some other entity. Accordingly, as discussed above, their logical representation is generally set at the level of theories or models and not at the level of concepts or relations.

In order to avoid potential terminological conflations and express it in natural language a descriptive statement is *about* something, e.g. it represents the *meaning* of something in a given context, while a ground statement is about the thing itself, e.g. for classifying instances within a given domain. In this modeling framework a situation is, consequently, clearly defined as a set of instances from the ground domain.

For example, a situation could be constituted by the instances of a specific object, e.g. *Truck:HB-IV-42*, a specific individual container and a specific road, e.g. the A27 in Germany, at a specific time, e.g. a certain day when the traffic flow was at a specific viscosity level. When seeking to describe this situation one would somehow like to express that the truck played the role of a means of transport, the container played the role of the freight object and the path was played by the specific road that the truck took under good conditions. In some other context however, e.g. that of an accident - one might seek to describe the truck as responsible for the accident and the capsized container as an obstacle.

In any case one would like to seek to refrain from simply multiplying the ground *isa* relations to express these states of affairs, as this is problematic both from a modeling as well as from a practical point of view. For avoiding such a taxonomic explosion, a dedicated descriptive pattern for a context-dependent reification of ground entities can be employed.

The employment of the foundational logical patterns described herein yields additional advantages. For example, the employment of such patterns - even when used with hindsight by means of *refactoring* - existing ontologies into pattern-based ones, has been shown to be beneficial for ontology quality when measured in terms of performance on a given task, e.g. ontology alignment [33]. Most importantly, however, a descriptive pattern for context-dependent reification, i.e. a coding of the functional meaning of some thing expressed in the terminology, introduced above, is to represent a pragmatically analyzed situation. It, therefore, enables the ontology engineer or an artificial agent to express that, using the example provided above, some ground entity is playing the *functional role* of a *MeansOfTransport*, which is transporting some other entity on a road, that plays the role of the path on a day where the actual traffic flow makes this possible.

The specified logical and content patterns of the *Descriptions and Situations* module feature three core descriptive entities, i.e. the classes *Courses of Events*, *Functional Roles* and *Parameters* [13]. These classes are linked by means of relations, which specify that:

- *Parameters* are *requisite-for* their *functional roles* and *Courses of Events*;
- *Functional Roles* are the *modality-targets* in the conceptualized *Courses of Events*.

Finally, the classes can be linked to the ground entities they describe, via the following relations:

- *Courses of Events* are *sequenced-by Perdurants*, i.e. processes within the ground ontology, such as *Locomotion*;

- *Functional Roles* are *played-by Endurants*, i.e. objects within the ground ontology, such as of type *Vehicle*;
- *Parameters* are *valued-by Regions*, i.e. phenomena that are sensed on scales, such as *TrafficFlow*.

Elaborating the example presented above in the domain of simulating logistic actors and processes, we can insert a class *LogisticActorRole* as a subclass of *AgentDrivenRole* that is already part of the Descriptions and Situations module. As individual subclasses of the class *LogisticActorRole* describe the functional roles of endurant entities, we proceeded by inserting specific logistic roles as further subclasses, such as *FreightActorRole*, *StorageActorRole* or *MeansOfTransportRole*. As defined by the Descriptions and Situations module all of these roles can only be played something that is an *Agent*, which is satisfied by our foundational simulation ontology, since *SimulatedAgentivePhysicalObjects* are *AgentivePhysicalObjects* which itself is a subclass of *Agent*. Additionally, these functional roles inherit further important relations, such as for the case of our class *LogisticActorRole (LAR)*:

- *LAR generically-dependent-on* **some** *PhysicalEndurant*;
- *LAR defined-by* **some** *Description*;
- *LAR participant-in* **some** *Perdurant*.

This enabled both immediate relations to the corresponding physical objects as well as to the processes in which they participate. As a next logical step - for endowing the ground ontologies with a dynamic layer of agent-specific descriptions - these elementary patterns were employed to construct an underlying model of simulation patterns that will be exemplified in the following section.

3.3 Simulation Patterns

In our minds, the central determining factor in the construal of objects for their context-specific reification is constituted by their activities. In a sense one can regard the simulated world to afford the agents with an array of possible actions, which constrain and determine the roles which can be played by whom. Therefore, we take the concept of a *CourseOfEvents* as a starting point for our model of logistic tasks - whether they are simulated or real. Being a descriptive entity a *LogisticTask* is sequenced by perduring events, which means we need to specify the real world or simulated processes that can be construed, for example, as a *TransportationTask*.

Within this description, we can specify the kind of objects that can play the role of *MeansOfTransport* or *FreightObject*, which are taken from the ground domain ontology. Let us note, once more, that to say a truck can **play** the role of a *MeansOfTransport* in a given *TransportationTask* is quite different from saying that it **is** a *MeansOfTransport* in the ground ontology. Analogously, we can employ ground models of qualities and regions, e.g. for specifying traffic flow or speed limits, to supply the parameters that need to valued for making a segment of the street network suitable as a path in a *TransportationTask*.

The employment of such patterns for modeling simulations based on the multiagent paradigm becomes extremely significant in the light of allowing an individual agent to form novel conceptualizations autonomously. While the ontology engineer still has the chance to constrain the range of possible construals, e.g. what kind of ground objects could be construed to play obstacles, means of transportation or else, it is still up to the agent to base these conceptualizations upon individual *experiences*. Hereby, it should be noted that, our foundational simulation ontology does not constrain which learning approaches or heuristics are to be employed for the context-dependent reifications. Here one can employ a multitude of possible approaches ranging from the more traditional machine learning [26] to newer ones based on so-called *echo state* networks [21, 6] that feature a build-in short term memory. What the foundational simulation ontology does provide, however, is a target representation and modeling instrument to express the results of the learning processes in a formal and explicit manner, which automatically becomes part of the domain-specific simulation model.

4 Conclusion and Future Work

The work presented above assumes that simulations constitute a useful and important technique in many scientific branches, but that they become difficult to design and control, especially when the simulated actors need to handle non-trivial knowledge in some form or another. While the employment of ontology-based multiagent systems for those simulations where the simulated actors need to possess, acquire or process knowledge has been shown to be working well [34], we also found that an ontology that models simulations themselves is still missing. In our minds such a foundational simulation ontology offers potential advantages in several respects.

Design Phase: In the design phase of setting up a multiagent-based simulation the employment of a foundational simulation ontology makes it possible to exploit the ontological patterns provided by the foundational framework, which not only eases the task of scenario engineering, but also creates more congruent models across individual simulations and thereby facilitated their reusability.

Runtime Phase: During the runtime phase we anticipate the advantage that complex simulations can be rendered (more) accessible for human-computer interaction due to the more homogeneous explicit and formal representation of the involved agents and their tasks, as any presentation ultimately hinges on the underlying representation.

Analysis Phase: In the analysis phase easier and well-grounded analysis and evaluation of recorded simulation runs ought to be possible, as the logical instruments provided by the foundation simulation ontology come from the standard logical inventory of formal ontologies and can be employed for additional validation and inferential reasoning using standard reasoning engines [19, 18, 1].

Generally speaking, in this paper we have outlined that basic motivation and approach for employing foundational ontologies and a corresponding general simulation model together with some concrete examples and modeling choices. While some benefits of this approach can be shown qualitatively, e.g. the re-definition of simulation design as an ontology engineering task, others await a quantitative evaluation. This is especially true for evaluating how the corresponding augmentation of an agent's ontological knowledge influences their performance, for example, in the case of self-organized logistic processes.

Therefore, our future work will be on implementation and evaluation using task-based ontology evaluations methods [31]. In addition, we are currently exploring the question how the domain application in the domain of transport logistics can serve as a testbed for evaluation the interaction with complex simulations from a human-computer interaction perspective.

Acknowledgments

This research has been funded by the German Research Foundation (DFG) within the Collaborative Research Centre (CRC) 637 Autonomous Cooperating Logistic Processes A Paradigm Shift and its Limitations at the Universitat Bremen, Germany.

REFERENCES

[1] F. Baader, C. Lutz, and B. Suntisrivaraporn, 'Cel: A polynomial-time reasoner for life science ontologies', in *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR'06)*, eds., U. Furbach and N. Shankar, volume 4130 of *Lecture Notes in Artificial Intelligence*, pp. 287–291. Springer-Verlag, (2006).

[2] Jon Barwise and John R. Perry, *Situations and Attitudes*, MIT Press, Cambridge, Mass., 1983.

[3] M. Bloos, J. Schönberger, and H. Kopfer, 'Supporting cooperative demand fulfillment in supply networks using autonomous control and multi-agent-systems', in *INFORMATIK 2009. Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V. (GI)*, (2009).

[4] Carmen-Veronica Bobeanu, Eugene J. H. Kerckhoffs, and Hendrik Van Landeghem, 'Modeling of Discrete Event Systems: A Holistic and Incremental Approach using Petri Nets', *ACM Trans. Model. Comput. Simul.*, **14**(4), 389–423, (2004).

[5] P. Davidsson, J. Holmgren, H. Kyhlbaeck, D. Mengistu, and M. Persson, 'Applications of agent based simulation', in *Multi-Agent-Based Simulation VII, International Workshop (MABS 2006)*, pp. 15–27. Springer, LNAI, vol. 4442, (2006).

[6] Lutz Dickmann, Tobias Lensing, Robert Porzel, Christoph Lischka, and Rainer Malaka, 'Sketch-based interfaces: Exploiting spatio-temporal context for automatic stroke grouping', in *10th International Symposium on Smart Graphics*, (2010).

[7] R. Fujimoto, *Parallel and Distributed Simulation Systems*, Wiley & Sons, New York, NY, USA, 2000.

[8] A Gangemi and V. Presutti, *Handbook of Ontologies (2nd edition)*, chapter Ontology Design Patterns, 221–244, Springer, 2008.

[9] Aldo Gangemi, 'Ontology design patterns for semantic web content', in *M. Musen et al. (eds.): Proceedings of the Fourth International Semantic Web Conference*. Berlin, Springer, (2005).

[10] Aldo Gangemi, Stefano Borgo, Carola Catenacci, and Jos Lehmann. Task taxonomies for knowledge content. metokis deliverable d07, 2004.

[11] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Oltramari, 'Sweetening WordNet with DOLCE', *AI Magazine*, **24**(3), 13–22, (2003).

[12] Aldo Gangemi, Nicola Guarino, Claudio Masolo, Alessandro Oltramari, and Luc Schneider. Sweetening ontologies with dolce, 2002.

[13] Aldo Gangemi and Peter Mika, 'Understanding the Semantic Web through Descriptions and Situations', in *Proceedings of the ODBASE Conference*, pp. 689–706. Springer, (2003).

[14] Jan D. Gehrke, 'Evaluating Situation Awareness of Autonomous Systems', in *Performance Metrics for Intelligent Systems Workshop*, pp. 206–213. NIST, (2008).

[15] Jan D. Gehrke and C. Ober-Blöbaum, 'Multiagent-based Logistics Simulation with PlaSMA', in *Informatik 2007 - Informatik trifft Logistik, Band 1.*, Bremen, Germany, (2007). Springer.

[16] N. Guarino. Deliverable d2: Ontology of information objects, 2006.

[17] Nicola Guarino and Roberto Poli, 'Formal ontology in conceptual analysis and knowledge representation', *Special issue of the International Journal of Human and Computer Studies*, **43**, (1995).

[18] V. Haarslev and R. Möller, 'Racer: A core inference engine for the semantic web', in *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), located at the 2nd International Semantic Web Conference ISWC 2003, Sanibel Island, Florida, USA, October 20*, pp. 27–36, (2003).

[19] Ian Horrocks, 'The FaCT system', in *Proc. of the 2nd Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX'98)*, ed., Harrie de Swart, volume 1397 of *Lecture Notes in Artificial Intelligence*, pp. 307–312. Springer, (1998).

[20] *Standard for Modeling and Simulation High Level Architecture. IEEE Standard 1516-2000*, ed., IEEE, IEEE Computer Society, 2000.

[21] Herbert Jaeger, 'The echo state approach to analysing and training recurrent neural networks', Technical report, Technical Report 148, German National Research Institute for Computer Science, (2001).

[22] Franziska Klügl, Christoph Oechslein, Frank Puppe, and Anna Dornhaus, 'Multi-Agent Modelling in Comparison to Standard Modelling', in *Artificial Intelligence, Simulation and Planning in High Autonomy Systems (AIS 2002)*, pp. 105–110. SCS Publishing House, (2002).

[23] Hagen Langer, Jan D. Gehrke, Joachim Hammer, Martin Lorenz, Ingo J. Timm, and Otthein Herzog, 'A Framework for Distributed Knowledge Management in Autonomous Logistic Processes', *International Journal of Knowledge-Based and Intelligent Engineering Systems*, **10**(4), 277–290, (2006).

[24] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, and Alessandro Otramari, 'D18: Ontology Library (final)', Project deliverable, WonderWeb: Ontology Infrastructure fo the Semantic Web, (2003).

[25] N. Mayer, J. Boedecker, R. da Silva Guerra, O. Obst, and M. Asada, '3D2Real: Simulation league finals in real robots', in *RoboCup 2006: Robot Soccer World Cup X*, pp. 25–34. Springer, LNAI, vol. 4434, (2008).

[26] Tom M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[27] Ian Niles and Adam Pease, 'Towards a Standard Upper Ontology', in *Proceedings of the Int. Conference on Formal Ontology in Information Systems*, pp. 2–9, (2001).

[28] Daniel Oberle, Anupriya Ankolekar, Pascal Hitzler, Philipp Cimiano, Michael Sintek, Malte Kiesel, B. Mougouie, S. Vembu, S. Baumann, Massimo Romanelli, Paul Buitelaar, R. Engel, D. Sonntag, N. Reithinger, Berenike Loos, R. Porzel, H.-P. Zorn, V. Micelli, C Schmidt, Moritz Weiten, F. Burkhardt, and J. Zhou, 'Dolce ergo sumo: On foundational and domain models in swinto (smartweb integrated ontology))', *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, **5**(3), 156–174, (2007).

[29] Daniel Oberle, S. Lamparter, Andreas Eberhart, and Steffen Staab, 'Semantic Management of Web Services', in *Service-Oriented Computing - ICSOC 2005*, volume 3826 of *LNCS*, pp. 514–519. Springer, (12 2005).

[30] H. V. D. Parunak, R. Savit, and R. L. Riolo, 'Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide', in *Multi-Agent Systems and Agent-Based Simulation, First International Workshop*, volume 1534 of *LNCS*, pp. 10–25, Paris, France, (July 4–6 1998). Springer.

[31] Robert Porzel and Rainer Malaka, 'A task-based framework for ontology learning, population and evaluation', in *Ontology Learning from Text: Methods, Evaluation and Applications*, eds., Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini, volume 123 of *Frontiers in Artificial Intelligence*, 107–122, IOS Press, (JUL 2005).

[32] Erich Herrmann Rast, *Reference and Indexicality*, number 17 in Logische Philosophie, Logos Verlag, 2007.

[33] Ondrej Svb-Zamazal, Vojtech Svtek, Christian Meilicke, and Heiner Stuckenschmidt, 'Testing the impact of pattern-based ontology refactoring on ontology matching results.', in *CEUR Workshop*, eds., Pavel Shvaiko, Jrme Euzenat, Fausto Giunchiglia, and Heiner Stuckenschmidt, volume 431. CEUR-WS.org, (2008).

[34] Tobias Warden, Robert Porzel, Jan Gehrke, Otthein Herzog, Hagen Langer, and Rainer Malaka, 'Towards Ontology-Based Multiagent Simulations: The Plasma Approach', in *24th European Conference on Modelling and Simulation*, Kuala Lumpur, Malaysia, (2010).

[35] Alfred North Whitehead, *A Treatise on Universal Algebra with Applications*, Cambridge Universitity Press (Reprint 1960), Cambridge, UK, 1898.

[36] Michael Wooldridge, *Reasoning about Rational Agents*, The MIT Press, 2000.

[37] Michael Wooldridge and N. R. Jennings, 'Intelligent agents: Theory and practice', *Knowledge Engineering Review*, **10**(2), 115–152, (1995).