Robotics and Computer–Integrated Manufacturing 71 (2021) 102174

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcm

Transformation of semantic knowledge into simulation-based decision support

Wiking Jurasky [a,*], Patrick Moder [b], Michael Milde [c], Hans Ehm [b], Gunther Reinhart [c]

[a] TUM School of Management, Technical University of Munich, Arcisstraße 21, München, 80333, Germany
[b] Infineon Technologies AG, Supply Chain Innovation, Am Campeon 1-15, Neubiberg, 85579, Germany
[c] Institute for Machine Tools and Industrial Management, Technical University of Munich, Boltzmannstraße 15, Garching, 85748, Germany

ARTICLE INFO

Keywords:
Knowledge transformation
Decision support
Ontologies
Hybrid modeling
Pandemic simulation
Supply chain simulation

ABSTRACT

Simulation is capable to cope with the uncertain and dynamic nature of industrial value chains. However, in-depth system expertise is inevitable for mapping objects and constraints from the real world to a virtual model. This knowledge-intensity leads to long development times of respective projects, which contradicts the need for timely decision support. Since more and more companies use industrial knowledge graphs and ontologies to foster their knowledge management, this paper proposes a framework on how to efficiently derive a simulation model from such semantic knowledge bases. As part of the approach, a novel Simulation Ontology provides a standardized meta-model for hybrid simulations. Its instantiation enables the user to come up with a fully parameterized formal simulation model. Newly developed Mapping Rules facilitate this process by providing guidance on how to turn knowledge from existing ontologies, which describe the system to be simulated, into instances of the Simulation Ontology. The framework is completed by a parsing procedure for an automated transformation of this conceptual model into an executable one. This novel modeling approach makes model development more efficient by reducing its complexity. It is validated in a use case implementation from semiconductor manufacturing, where cross-domain knowledge was required in order to model and simulate the impacts of the COVID-19 pandemic on a global supply chain network.

1. Motivation

For global supply networks with multiple participants and interrelations, ever-changing market conditions lead to volatility, uncertainty, complexity and ambiguity [1]. In order to secure competitiveness, companies strive to achieve utmost flexibility and to adapt their supply chain efficiently in a timely manner. An emerging decision support strategy is based on semantic web technologies and incorporates ontologies that capture the knowledge required to make meaningful decisions. Ontologies in this sense serve two main purposes. First, they allow dynamic retrieval of yet implicit information. Second, ontologies act as *lingua franca* between domain experts due to their inherent nature of being broadly understandable and open to interpretation [2,3].

Despite the established usage of ontologies in complex, interdisciplinary or ambiguous industrial environments, this decision-support strategy is limited due to its merely static representation. Facing dynamic scenarios, separate technologies such as simulations are required for an effective analysis. Although simulations are widely used across manufacturing domains, related projects tend to lack general efficiency, especially when dealing with novel or domain-agnostic scenarios. Since a profound system understanding is required to develop simulation models, this process yet is very time-consuming and complex [4–6]. Usually ensured by knowledgeable experts, the model development is decelerated when external information is required to build up the model. This leads to long development times in simulation projects, which are contrary to the need for fast and agile decision making.

However, the spread of ontologies described above represents an opportunity to overcome the challenges in simulation projects and make their development more efficient. As pre-existing collections of knowledge, they can serve as a starting point for simulation development. Initial approaches from the literature already confirm that the use of ontologies has a beneficial effect on the development of simulation models [7,8]. Nevertheless, there is a lack of methods and tools in the industry on how to leverage knowledge from existing knowledge bases for more efficient simulation model development. With the help of such tools, companies can efficiently take advantage of simulation studies for their decision making, achieve better decisions, and ultimately secure competitive advantages.

* Corresponding author.
E-mail address: wiking.jurasky@mail.de (W. Jurasky).

https://doi.org/10.1016/j.rcim.2021.102174
Received 24 November 2020; Received in revised form 12 March 2021; Accepted 9 April 2021
Available online 21 April 2021

The paper on hand presents a framework for efficiently transferring knowledge from existing knowledge bases into executable simulation models. For this purpose, a general model for hybrid simulations in the form of an ontology is developed as part of the framework. Instantiated, this so-called Simulation Ontology represents a fully parameterized formal model within a simulation project. With the development of Mapping Rules as a further component of the framework, a structured procedure is created, with which knowledge from existing ontologies, the so-called Use Case Ontologies, can be efficiently transferred into instances of the Simulation Ontology. The transformation of the formal model into an executable simulation is fully automated by the third component of the framework, the Parser.

We evaluate the framework with a contemporary relevant use case scenario, investigating the impacts of the COVID-19 pandemic on a globally distributed semiconductor supply chain. The use case is chosen for two reasons: First, semiconductor manufacturing with its dynamic, complex and uncertain environment requires simulation models to provide decision support in disruptive situations [9]. Second, as such a disruptive situation is given in the course of the COVID-19 pandemic, it makes expert knowledge from different domains (epidemiology, supply chain management e.g.) necessary to be incorporated in the simulation model [10].

The remainder of this paper is structured as follows. Section 2 provides fundamentals on ontologies and simulation studies. Section 3 analyzes related work and shows respective limitations that the presented approach aims to address. Section 4 illustrates the developed framework and ontology together with mapping rules and parser details. Section 5 describes the use case application before discussing the framework based on the gained findings. Finally, Section 6 concludes the work and provides an outlook for future research.

2. Background

Ontologies are based on natural language design and serve as structured representation of knowledge in a specific domain [11]. This knowledge is represented by a set of uniquely defined terms (classes), which are related by certain attributes (relations). Attributes are further specified by restrictions, which represent the cardinality, i.e. the range of values an attribute can take [12]. Ontologies thus make it possible to formalize knowledge and information, so that it can be shared between people and machines and create a common understanding. In addition, they ensure easy extensibility as well as efficient reuse of knowledge. The instantiation of ontologies enables the separation of operational and domain-specific knowledge. An ontology with its class taxonomy, properties and relations represents a generalized model of the domain-related concepts, which integrates knowledge of the specific use case by instances. Through axiomatization, reasoning and query languages, one can obtain correlations and logical inferences in the domain. Ontologies thus represent a central tool for knowledge management and have been used increasingly in industry for the collection, structuring and storage of knowledge since the spread of the idea behind the Semantic Web [11–13].

In dynamic and complex systems, decision support systems solely based on the collection and evaluation of knowledge are not sufficient. The dynamic behavior of the system under different conditions over time is also to be studied in order to make meaningful decisions. Simulations, as model-based decision support systems, are suitable for this purpose and widely used in industrial practice [14,15]. A simulation is a virtual representation of a system, which can be used for experimental studies [14]. Simulations offer the possibility to investigate the system behavior in scenarios without interfering with the real system, and thus to make statements about its possible future development [15,16]. In the manufacturing and supply chain area, mainly three types of simulations are used: (1) Discrete-event Simulation (DES) for the analysis of processes, (2) Agent-based Modeling (ABM) for modeling the behavior and interaction of independent objects within a certain system, and (3) System Dynamics (SD) approaches for the continuous investigation of causal relationships between different parameters [17]. Often, given problems cannot be assigned to one particular type, so they are mapped in hybrid models that combine various submodels to address strategic, tactical as well as operational problem settings and dependencies at once [17]. Nevertheless, the general procedure behind all modeling approaches follows the same underlying logic.

According to Milde and Reinhart [6] and VDI [14], a simulation project can be divided into an introductory task definition followed by model development and generation, which is further composed of system analysis, creation of a conceptual model and its implementation into simulation software. In parallel, the process of data management identifies and prepares the required data. After a successful verification and validation, the model can be used for experimental simulation studies and thus to develop scenarios for decision support.

The burden of time-consumption in the current best practice of industrial simulations as well as the capabilities of semantic knowledge management regarding an efficient understanding and communication of complex systems raise the question of how to convert the knowledge within ontologies into simulation models in order to support fast and efficient decision making in complex and dynamic environments.

3. Related work

With regard to previous research, Fishwick and Miller [7] already emphasized in 2004 that the simulation domain should leverage the potentials of the Semantic Web. This section first presents further rather general related work. Subsequently, we follow McGinnis et al. [18] in distinguishing the research streams on ontologies for domain modeling and for simulation modeling. Whereas the former focuses on conceptual models for explicit application domains, the latter aims at a general knowledge representation for simulation studies.

Benjamin et al. [19] argue for a specific benefit of ontologies for every process step related to a simulation project. In particular, a harmonized terminology helps to establish its purpose and scope. The different levels of abstraction and the ease of analyzing an ontology can strongly support the identification of roles and relationships and thus the formulation of a conceptional model. Furthermore, the mining of data and interpreting text mitigates the challenges of data acquisition and analysis.

A theoretical concept for linking simulations to a semantic knowledge base is proposed by Rabe and Gocev [20]. A Web Ontology Language (OWL)-based reference ontology for manufacturing systems builds the backbone of a framework for the continuous enrichment of knowledge by the aids of human experience, an inference engine as well as simulation results.

The recent framework of Du et al. [21] addresses a prefabricated component supply chain of the Chinese construction industry. Local ontologies are used to store the specific information of each participant. Mapping to a global ontology of the entire supply chain enables information integration and knowledge sharing. To finally support decision making, the individual behavior of the agents is analyzed in a simulation model.

Research on ontologies for domain modeling particularly addresses the challenge of system understanding by provision of a general knowledge model. However, their interface to simulations is usually limited to ad-hoc transformations, which leads to an overall lack of generalizability. For instance, Fayez et al. [22] propose an ontology based on the Supply Chain Operations Reference (SCOR) model, from which specific simulation configurations can be derived. Being dynamic, large and complex with regard to space and time as well as based on various heterogeneous information technologies, the supply chain domain is stated to be particularly challenging for simulations.

A similar approach is presented by Cope [23]. The main contribution is a stand-alone tool for building simulation models from a SCOR-based ontology. In contrast to Fayez et al. [22], the ontology

of Cope [23] not only contains supply chain knowledge but also the knowledge required for building a related simulation model. In particular, classes for resources that perform a certain process and for the specification of the process duration are added.

Similar to the work of Cope [23], Chen and Chen [24] develop a knowledge model for supply chain simulation. This knowledge model also integrates supply chain knowledge and simulation know how in equal measure, focusing on modeling supply chain planning processes and the data and information exchanged between them. The knowledge model is not built as an ontology but as an object model and implemented using Extensible Markup Language (XML). By developing an interface, the XML model can be read into a simulation software and used for simulation studies.

Wagner et al. [25] provide an ontology-driven simulation framework for the specific application of automated material handling systems in semiconductor factories. The authors strive for fast decision support in this highly dynamic and complex environment by the use of simulation experiments. Nevertheless, their simulation interface only aims at direct experiments on specific KPIs in one particular problem setting and thus also lacks generalizability.

The counterpart to these domain-specific publications is represented by approaches for simulation modeling. However, since such simulation ontologies are in need of an appropriate interface to potential domain ontologies, further developments resulted in two major combined approaches [18]. Silver et al. [26] present the Discrete-event Modeling Ontology (DeMO), an OWL-based ontology derived from the mathematical foundations of a DES. DeMO is composed of four main parts, since the authors propose to distinguish models according to the general perspectives of activities, events, processes as well as states. A self-developed tool suite supports the user in mapping any domain ontology to DeMO. The resulting process instances are stored as a directed graph of activities. A transformation schema based on Java, the SPARQL Protocol and RDF Query Language (SPARQL) and XML is able to translate those graphs to actual simulation models. In the course of a case study from the biochemical domain, the general complexity of mapping properties from a domain ontology to corresponding classes of DeMO is mentioned. This can be reasoned by the general focus on the theoretical fundamentals of simulation modeling rather than on application-oriented concepts. Nevertheless, the works around DeMO provide valuable insights on how to bring a domain ontology, a simulation ontology as well as a transformation procedure into a conceptual framework.

The research of McGinnis et al. [18] can also be classified to the field of approaches that combine a simulation ontology with domain knowledge. But unlike OWL in [26], Systems Modeling Language (SysML) is used as a formal language for modeling discrete-event logistics systems. Their transformation procedure is based on the four-layer meta-object facility. The concept from software engineering implies the actual model being an instance of a user model and further has to be conform to a meta-model, i.e. a modeling language specification such as SysML. The definition of this meta-model is provided on the highest layer, the meta-meta-model. The authors present respective meta-models for both, SysML as well as the simulation applications Arena and AnyLogic and case-dependent transformation procedures to partially generate respective simulation models. Proofs-of-concept are provided for an electronic assembly system [27] as well as for general test problems from semiconductor manufacturing [28]. However, this approach also lacks practicability. Ehm et al. [29] argue, that trying to fit a company in a top-down meta-model architecture rather fits the world view of a computer scientist – mainly talking about objects and classes instead of machines, jobs or processes – than of particular domain experts or model developers.

Previous works share two major deficits. None can be classified as a holistic approach, leveraging all potential benefits of ontology-based simulations, as argued by Benjamin et al. [19]. They either relied on ad-hoc transformations of a specific domain ontology or came up with a general model for simulations but lacking an appropriate mapping of domain knowledge. In order to leverage the capabilities of semantic knowledge for both the understanding of the underlying system and of the general concepts of simulations, guidelines on how to transfer objects and constraints from the real world to a digital replica are crucial. Furthermore, all previous works are limited to DES applications but the increasing complexity of modern manufacturing and supply chain systems is expected to raise problems that might be best addressed by an ABM, a SD or even a hybrid approach. Especially for the investigation of supply chain behavior, SD simulations are often used in literature (see [30,31]). Transferring these studies to actual supply chains in practice remains a challenge due to the lack of methods to support model building. Since simulation engineers usually have their personal expertise and experience in only one particular modeling technique, a standardized knowledge representation of all potential concepts is thus expected to facilitate the development of more profound simulation models.

4. Bridging ontologies and simulations

This section presents the framework proposed for the transformation of semantic domain knowledge to a hybrid simulation model. It exploits a given semantic knowledge repository of an industrial system and leads to a model executable in simulation software and thus capable for decision support. The knowledge-intensive tasks of system analysis and model conceptualization, i.e. the decisions of how to model the main objects and interrelationships, highly depend on the underlying problem setting and thus remain manual. The subsequent model generation is labor-intensive but with a conceptual model on hand rater straightforward and thus automatized. One can therefore classify the entire framework as semi-automatic.

4.1. Ontology-based simulation development framework

Fig. 1 depicts the general process of simulation projects [6] in the upper part and below how the respective tasks are addressed by our ontology-based approach. As outlined in Section 2, a simulation study can be split into an initial phase of orientation and project definition, followed by model building, running and result interpretation. The part in focus of our work, namely model development, is usually further accompanied by a verification and validation procedure as well as by an appropriate data management procedure. However, we assume the latter to be an integrated part in our approach, since ontologies and in particular knowledge graphs are sophisticated for managing complex input data.

The proposed framework aims at more efficient model development by supporting the respective tasks between the overall problem definition and the actual simulation analyses. It bases on a pre-existing industrial knowledge base, which is assumed to contain domain ontologies for the system to be analyzed. To better distinguish such domain ontologies from the later presented ontology for simulation, they are referred to as *Use Case Ontology* in the following. It may not contain knowledge explicitly related to simulations, but rather sufficient information and data of the system to understand its causal interrelationships. It is therefore expected to serve as central place for knowledge exchange between various stakeholders, such as simulation engineers.

In order to efficiently derive an executable simulation that is capable of providing insights into the dynamics and uncertainties, a transformation approach is proposed. Its core component is a novel *Simulation Ontology*, representing a formal meta-model for hybrid simulations. Aiming for an applicability to a broad range of potential problem settings, it brings all major elements a simulation model might be composed of in an appropriate class taxonomy and further adds semantics to improve their tangibility. One can formally define any simulation model by instantiating the respective classes and properties

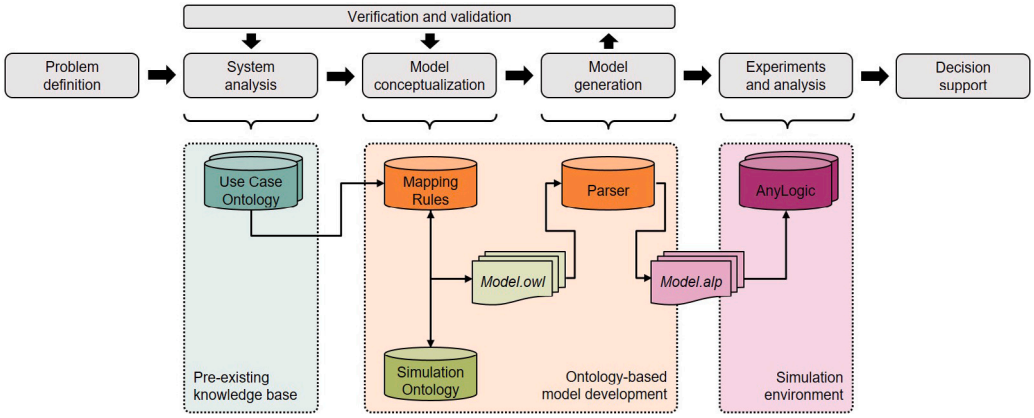

Fig. 1. Framework for ontology-based simulation development.

within the Simulation Ontology. This represents a novel approach for simulation modeling and in particular for model conceptualization. To support such a procedure, a set of general *Mapping Rules* provide guidance on which objects and constraints from the Use Case Ontology might be best addressed by which elements within this Simulation Ontology. Since one cannot directly conduct experiments on such a conceptual model, it still has to be transformed to source code in the model description language of an actual simulation application. This is addressed by the second step of knowledge transformation, namely a *Parser* for automated model generation.

From a technical perspective, OWL is recommendable for knowledge modeling within the Simulation Ontology and thus for model conceptualization. Being part of the backbone of the Semantic Web, it is proven to be powerful in practical applications. Despite a semantic triplestore and a structural class taxonomy, its additional constraints and characteristics enable a respective ontology to be reasoned. Furthermore, the related SPARQL protocol can query for any information or data and thus provides a flexible and efficient interface to other applications [8,32]. The latter also applies for AnyLogic Project (ALP), the model description language of AnyLogic. It brings the use of Java for simulation modeling in a text-based XML meta-structure. AnyLogic is further recommendable as simulation software due to the support of all techniques associated with hybrid modeling [33]. The entire framework can also be enabled for the use of other simulation software. For this purpose, the specific parts in the parser must be adapted to the corresponding model description language.

Further details on the three components proposed for ontology-based simulation model development are provided in the following. Insights into the structure and concepts of the novel Simulation Ontology are followed by the two-fold transformation procedure, namely the related Mapping Rules and the exemplary parsing procedure to AnyLogic.

4.2. Simulation ontology for hybrid modeling

The engineering process of the Simulation Ontology [34] followed the formal procedure of Noy and McGuinness [12]. Its overall scope is to represent a meta-model for hybrid simulations with a high degree of practicability. The latter is of particular importance since it enables users to directly define a formal simulation model by creating suitable instances of the required classes and properties. Previous works such as DeMO [26] merely addressed DES models. Furthermore, they lack wording and concepts of actual applications. The main approach to extend their general constructs is to use the set of all potential components or *building blocks* of a hybrid model as baseline for the class taxonomy. These elements are further distinguished as dependent on the model type and independent (i.e. *global*) in turn. Another important assumption is that each type implies a certain kind of *control logic* for the system interrelationships. For instance, a DES can be generalized as a flowchart, i.e. a diagram with nodes and directed edges to model the process. Table 1 shows, which general OWL concepts are used to model this novel Simulation Ontology.

Its core components, the various building blocks a simulation model might be composed of, are modeled in respective classes, which are further classified in a taxonomy that follows the structure of the different model types. The highest level therefore contains the `Model`, the corresponding `Controlchart` and `BuildingBlocks` as well as the type-independent `GlobalElements` and a general `ModelDescription`. Since each modeling method has a certain type of control chart, one distinguishes `DiscreteEvent`, `AgentBased`, `SystemDynamics` as well as `Flowchart`, `Statechart` and `StockFlowDiagram`, respectively. `Hybrid` models are then just a combination of submodels from different types. A flowchart is defined by instances of an `Activity`, with `Ports` linked via a `Connector`, as well as the processed `Entity` and some sort of `Resource`. Furthermore, a state chart is at least composed of an `EntryPoint`, `State`, `Transition` and `AgentType`, whereas a stock and flow diagram implies instances of `Stock`, `Flow` and `Link`. Subclasses of the type-independent and thus global model elements are among others a `Parameter` or a `Variable`.

The interrelationships between the classes are modeled by object properties. The actual triples are defined by respective domain and range restrictions. For instance, the `Entity` in a process-oriented model is linked to the different types of `Activity` building blocks, depending on whether they `generate`, `delay`, `queue`, `route` or `dispose` the entity. Despite such rather descriptive notions, there are also existential relationships in place — such as a `Connector` always has to be linked to `exactly1PortIn` and `exactly1PortOut` of an `Activity` block to model the particular step of the process flow. In general, the object properties provide semantics to support the user in understanding each potential component.

Data properties and respective domain restrictions define the parameters each building block might have. Despite rather general ones such as a `Name` or a `Location`, some solely have a single semantic domain. For instance, only the queue might have a `Capacity`. The amount of data properties modeled depends on the simulation application which the framework is applied to. Our implementation focuses on the ones required for modeling the in Section 5 presented use case in AnyLogic. However, we avoid highly software-specific elements for the sake of generalizability and rely on the ones that are most-likely transferable to other applications. Fig. 2 depicts the Simulation Ontology with this particular scope.

4.3. Mapping rules for guided model conceptualization

With a thorough understanding of the Use Case Ontology and thus the actual system, the simulation engineer can define a conceptual

Table 1
Ontology concepts within the Simulation Ontology.

OWL concept [8]	Modeling purpose	Example
Classes	Model elements (building blocks)	`Delay`
Class hierarchy	Classification of building blocks (taxonomy)	`SubclassOf( :Delay :EntityActivity)`
Class disjointness	Incompatibility relationships among building blocks	`DisjointClasses( :Delay :Queue)`
Object properties	Semantic relationships between building blocks	`delaysAgentType`
Data properties	Parameters of building blocks	`DelayTime`
Domain and range restrictions	Interconnection of building blocks by a certain object property or interconnection of a building block with a certain data property	`ObjectPropertyDomain( :delaysAgentType :Delay) ObjectPropertyRange( :delaysAgentType :Entity)`
Property restrictions (Complex classes)	Existential quantification relationships among building blocks	`EquivalentClasses( :EntityActivity ObjectSomeValuesFrom( :isPerformedOn :Entity))`
Property characteristics	Inverse, symmetric, functional or reflexive properties	`InverseObjectProperties( :hasIncomingRate :hasSinkStock)`
Individuals	Elements of an actual model	`InternalShipment1`

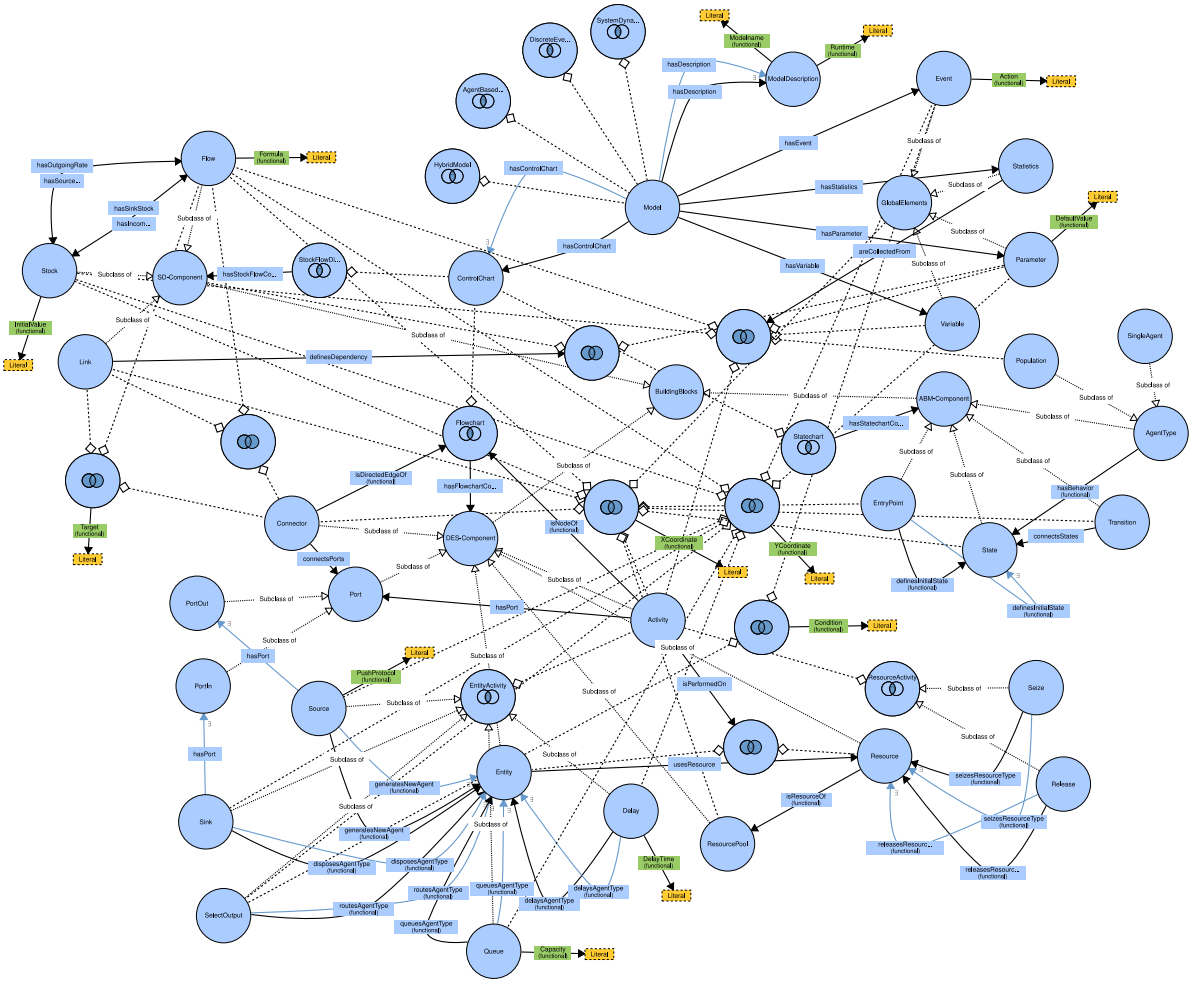

Fig. 2. Simulation Ontology for hybrid modeling [34].

model by instantiating the Simulation Ontology accordingly. Since this represents an entirely novel modeling procedure, a set of general Mapping Rules serves the need for appropriate guidance. Table 2 presents an overview of all modeling questions, inferred rules as well as the corresponding ontology classes.

It is recommendable to go through the rule set in a hierarchical manner since they closely follow the taxonomy of the Simulation Ontology. Hence, one preferably first defines the scope of the project and provides sufficient description. Subsequently, the level of abstraction or detail of the underlying system provides a key indicator for the best fitting modeling technique. For instance, a rather detailed operational process usually implies a DES approach, whereas a rather abstract compilation of general global relationships is better addressed by a SD model. In case the system cannot be clearly assigned to one of the three types, one might try to split it into subproblems and thus most likely come up with a hybrid model, i.e. a composition of different submodels. The third modeling question is independent of this decision, since it addresses the group of type-independent building blocks. In case of a

Table 2
Mapping rules for domain knowledge with Simulation Ontology.

What is the simulation project all about?		
1	Sufficient description of the project is to be provided.	`ModelDescription`
Which level of abstraction/detail is intended?		
2	The system to be analyzed implies a certain model type.	`Model`
2.1	The system can be represented as an operational process.	`DiscreteEvent`
2.2	The individual behavior of objects shall be observed.	`AgentBased`
2.3	The system continuously observes global relationships.	`SystemDynamics`
2.4	The problem infers an interaction of systems/individuals from different levels of abstraction.	`Hybrid`
Which global elements are referred to the model?		
3	Model components that are type-independent.	`GlobalElements`
3.1	An action triggered under a certain condition.	`Event`
3.2	A numeric factor with a fixed value, e.g. input data.	`Parameter`
3.3	A numeric factor with a varying value.	`Variable`
3.4	A plotted observation of a variable over time.	`Statistics`
Of which type-dependent building blocks is the model composed?		
4	Each model type implies a certain kind of control chart.	`BuildingBlocks`
4.1	Discrete-event models are controlled by a flowchart.	`Flowchart`
4.1.1	An object that flows through the process.	`Entity`
4.1.2	An object which is used by an entity within the process.	`Resource`
4.1.3	A group of a certain kind of resources.	`ResourcePool`
4.1.4	A location/node defining a discrete process step.	`Activity`
4.1.4.1	A process step where an entity ...	`EntityActivity`
4.1.4.1.1	... is generated, i.e. the initial process step.	`Source`
4.1.4.1.2	... is disposed, i.e. the final process step.	`Sink`
4.1.4.1.3	... is queued.	`Queue`
4.1.4.1.4	... is delayed.	`Delay`
4.1.4.1.5	... is routed.	`SelectOutput`
4.1.4.2	A process step where a resource ...	`ResourceActivity`
4.1.4.2.1	... is seized by an entity.	`Seize`
4.1.4.2.2	... is released from an entity.	`Release`
4.1.5	The transition/edge between two successive activities.	`Connector`
4.2	Agent-based models are controlled by a state chart.	`Statechart`
4.2.1	A certain kind of an individual or population.	`AgentType`
4.2.2	A condition an agent can be in at a specific point in time.	`State`
4.2.3	The condition for an agent to change its current state.	`Transition`
4.2.4	The marker that defines the initial state.	`EntryPoint`
4.3	System Dynamics models are stock and flow diagrams.	`StockFlowDiagram`
4.3.1	An element that retains a value at a discrete point in time.	`Stock`
4.3.2	The continuous change rate of a stock per time interval.	`Flow`
4.3.3	Implicator of a math. relationship between elements.	`Link`

hybrid one, the notion of *global* gets even more clear, since respective elements might be linked to several subsystems at once. This decision for a certain model type is crucial for the last and most detailed part of model conceptualization, namely the actual building blocks, since one can only choose from the subset corresponding to the implied type. In the fourth and last section of simulation modeling, the actual elements and thus the required type-dependent building blocks are to be identified. These nodes finally form a flowchart, a state chart or a stock and flow diagram by defining the required arcs, i.e. connectors, transitions or links, respectively.

Once an instance of a class is defined to represent an object or constraint from the real system, it inherits all properties of its parenting class. The aforementioned object properties further guide the user in how the different elements might interrelate and thus support the conceptualization of the formal model. Its parametrization is enhanced by the inherited data properties, since they show the user which parameters a certain object might have and thus which data is to be gathered. For instance, the simulation engineer is provided with information on how many ports for connectors an activity object in a DES model has and further that all connectors are forced to have a single particular source and target.

4.4. Parser for automated model generation

The mapping procedure of the required system knowledge results in a fully-instantiated Simulation Ontology and thereby completes the task of model conceptualization. However, this formal model still needs to be transferred to appropriate simulation software in order to conduct the desired experiments. Since this implementation becomes highly labor-intensive with an increasing number of elements but does not directly add value, a parsing script for the automation of this task is proposed. In a nutshell, the Parser [34] is supposed to take the instances of the Simulation Ontology as input and translates them into the required model description language to receive an executable simulation model.

Since the SPARQL protocol is applicable to OWL-based ontologies, respective queries can be made on the Simulation Ontology in order to export the formal model. The first query retrieves a list of all building blocks by composing all defined class instances. The second one retrieves a list of the corresponding parameters and their respective values, i.e. the inherited and instantiated data properties per building block.

It shall be noted, that the argument of the `VALUES` statement in the first query can be extended to all classes of the Simulation Ontology. As presented here, only the elements required by the formal model of our later use case are considered for the sake of clarity and consistency. Similarly, the scope of the respective statement in the second query depends on the amount of implemented data properties and here only contains the ones necessary for the case study conducted in Section 5. In the example above, the user for instance retrieves information such as the number of intended delay building blocks and their respective delay time parameters as well as all connectors with their respective source and target.

```
SELECT ?type ?instance
WHERE {
  ?instance a owl:NamedIndividual.
  VALUES ?type {:ProjectDescription :Stock :Flow :Parameter :Link :Connector :Event :Source :SelectOutput :Delay :Queue
                :Sink}.
  ?instance ?property ?type.
}
ORDER BY ?type
```

```
SELECT ?subject ?property ?value
WHERE {
  ?instance a owl:NamedIndividual.
  VALUES ?property {:Action :Capacity :Condition :DefaultValue :DelayTime :Formula :InitialValue :Modelname :PushProtocol
                    :Runtime :Source :Target :XCoordinate :YCoordinate}.
  ?subject ?property ?value.
}
ORDER BY ?subject
```

The script used to finally transform this query output into an executable simulation model is supposed to run through the following general logic:

(I) A blank ALP file is read-in and the lists of elements and parameters are retrieved via SPARQL. The latter two are preferably stored in a file format for flexible and efficient data exchange, such as Comma-separated Values (CSV). The use of a dummy model is recommended over a direct encoding of the entire model description schema, since it enables the approach to be easily adjusted to potential future version of AnyLogic.

(II) Depending on the actual data formats, the input might have to be cleaned with regard to textual spelling for an easier processing. For instance, the ontology prefix or the CSV delimiters are redundant information for the simulation.

(III) An algorithm then runs through the list of building blocks and creates a respective element at the right position in the XML-tree. After a single element is inserted, it first checks the list of parameters to create a respective tag for each parameter related to the current element, before going to the next item and repeating the parameter insertion. However, the list of building blocks is not simply processed from top to bottom but rather according to the general sequence required by the ALP schema. In particular, all elements of the `Variable` class are followed by `Dependences`, `Connectors`, `StateChartElements`, `Events`, `AnalysisData`, `AgentLinks`, as well as `EmbeddedObjects` sequentially. Besides the actual elements, general data such as the name or the runtime of the model have to – if defined in the Simulation Ontology – replace their preexisting dummies.

(IV) The fully tagged XML-tree has to be written into a new ALP file in order to represent the source code of the desired simulation model, which is then executable in AnyLogic.

The resulting model generation workflow is summarized in Fig. 3. It further highlights which tasks remain manual, namely the SPARQL querying process, and what is automated by the Parser. Overall, the user only has to execute three codes in order to generate the simulation model from the instantiated Simulation Ontology.

So far, the script is able to build up any simulation model for AnyLogic. In case of an application to another simulation application, only step (III) of the script needs to be adapted. In particular, the structure of the XML-tree has to follow the general schema of the respective model description language. The underlying logic and the sequence however can remain unchanged. Besides, the simulation software needs to be able to execute all desired model types. For instance, many common tools such as Arena are only capable of DES models and would therefore limit the potentials of the entire framework.

5. Application and validation

To proof this novel approach for developing a decision-support model based on a semantic knowledge base, it is applied to a recent use case problem, namely the COVID-19 pandemic and its impact on the global supply chain of a semiconductor manufacturer. Based on the derived insights, the general validity of the approach and its practicability for industrial applications are discussed.

5.1. Use case: Semiconductor SCM in times of a pandemic

The unforeseeable emergence and spread of COVID-19 not only astonished governments and the general public but was and is still challenging industrial corporations with regard to best possible adoption on heavily disrupted business conditions. Due to the lack of prior knowledge and experience, the semiconductor manufacturer tried to gather as much data and information as possible in order to derive all potential implications. The resulting pandemic knowledge base was built by use of an ontology, since its semantic capabilities can cope with the broad range of affected domains and incorporate the ever-changing opinions and new findings. It further enabled the stakeholders to find a consensus on the core impact factors. However, the related uncertainty with the future development of the pandemic led to the additional and urgent need for profound scenarios in order to proactively define countermeasures. Since semiconductors typically have a cycle time of several months as well as and highly expensive production capacities, simulations are a proven tool for proactive decision support [9,25]. The main problem in the course of simulating COVID-19 is that simulation engineers might be familiar with the overall supply chain but lack knowledge on how to model a pandemic as well as its impacts — which argues for an ontology-based modeling approach.

An abstract version of the company's ontology for the impacts of COVID-19 on its supply chain is depicted in Fig. 4 and for this purpose referred to as Use Case Ontology. It includes the main strategic echelons along the internal supply chain, such as the factories or distribution centers, as well as the key drivers of related planning decisions. Since semiconductors tend to require a comparably high number of internal shipments along their operational journey from a silicon wafer up to a final product for the customer, their routing and thus the accumulation of all transit times in between the different supply chain echelons is crucial. Despite the knowledge area of supply chain management, the domain of COVID-19 and its affects with a certain population is incorporated. The ontology shows the semantic interrelationships between general notions such as the reproduction rate, the capacity of hospitals or governmental restrictions. The latter factor further represents the key connection to the supply chain knowledge area, since governmental reactions on the pandemic spread in form of legal restrictions occurred

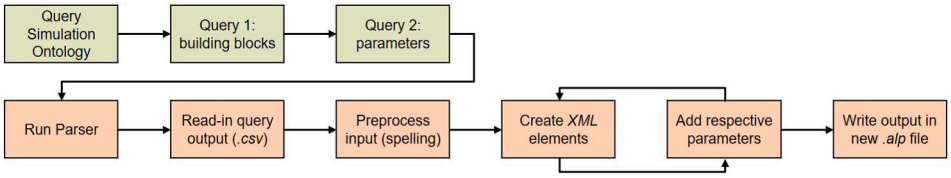

Fig. 3. Workflow of the parsing procedure.

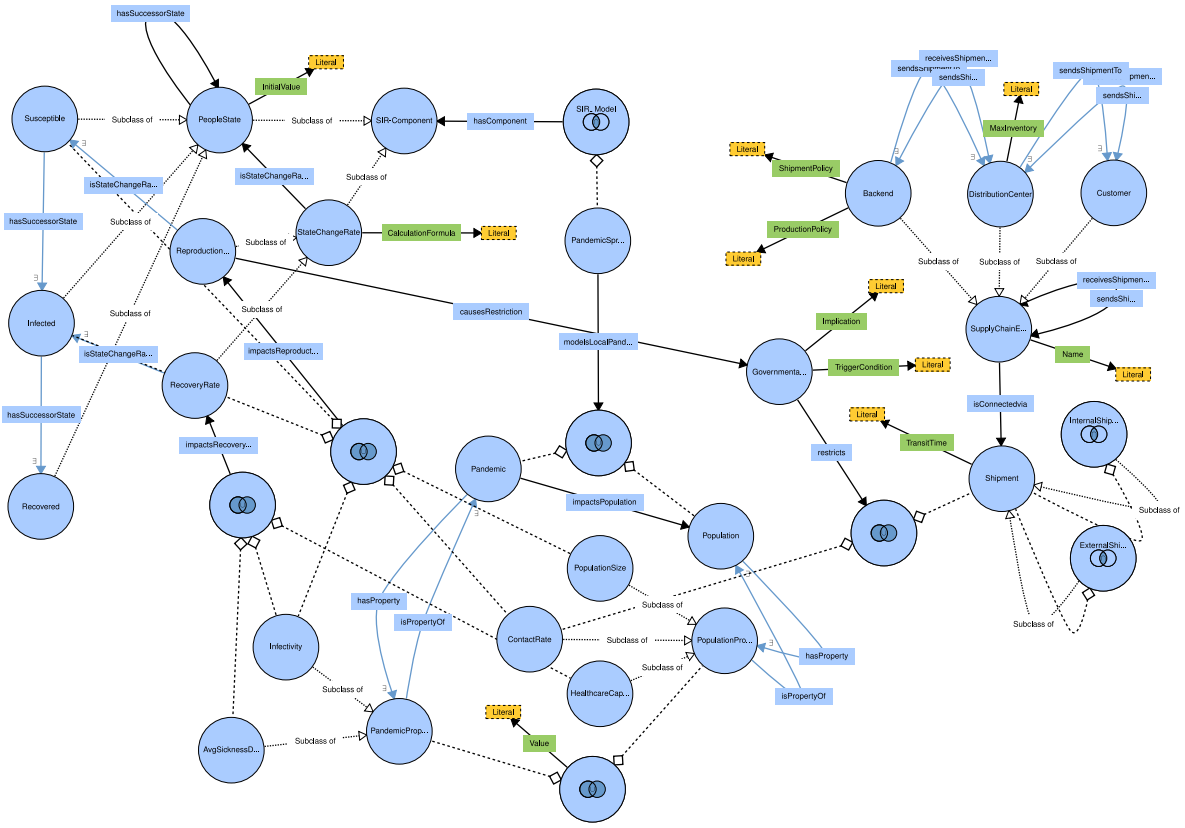

Fig. 4. Use Case Ontology for the impacts of COVID-19 on a semiconductor supply chain.

to be the main disrupting factor. In particular, it could be observed that regional lockdowns heavily delay the transit times of affected shipments. In order to predict these restrictions and thus be able to proactively define countermeasures for the supply chain, a third knowledge area on the dynamic development of the pandemic over time is incorporated. The so-called Susceptible–Infected–Recovered (SIR) model provides the mathematical baseline on how a virus is spread among a population and in particular how the number of infections develops. The related reproduction rate is seen as the key trigger for lockdowns and thus disruptions on the semiconductor supply chain.

Overall, the Use Case Ontology is an actual knowledge graph, i.e. an ontology instantiated with real data. It supports the user in understanding the causal relationships between the collected data, such as the master data of the actual supply chain echelons, the observed transit times of shipments and the numeric measures related to the COVID-19 pandemic. Based on these, a simulation study is supposed to come up with actual scenarios for the spread of the pandemic, the potentially resulting lockdowns and finally the best possible routing decisions along the global supply chain in order to proactively adjust manufacturing or inventory capacities and thereby increase the overall resilience.

The first step in applying the proposed framework on this Use Case Ontology is to transfer the contained knowledge to the simulation domain. By using the introduced Mapping Rules (cf. Table 2) in combination with the Simulation Ontology, a formal model for the pandemic spread and the supply chain can be derived. The modeling question of the most appropriate model type leads to a hybrid approach, since the underlying problem infers the rather global interrelationships associated with COVID-19 as well as the operational order-fulfillment process of a semiconductor manufacturer. According to rules 2.1–2.4, a combination of a continuous SD model with a discrete-event process flow is most suitable to address both levels of abstraction. Regarding the required building blocks, Table 3 provides an exemplary overview of the instantiated elements associated with the third modeling question, namely the type-independent ones. For instance, the governmental restrictions are best to be modeled by an `Event` called `Lockdown`. The Use Case Ontology reasons this by lockdowns being linked to a certain trigger condition, the pandemic reproduction rate, and particular implications, namely a decreased contact rate within the population and an increased transit time of shipments throughout the affected region. These transit times as well as all properties of the pandemic and the population are fixed input data with a certain value and thus imply respective instances of the `Parameter` class. The same procedure is applied for the set of DES building blocks as well as the set of SD building blocks in order to transform all further instances of the ontology into the required elements and corresponding parameters.

Table 3
Applied mapping rules for general elements.

Which global elements are referred to the model?		
3.1	An action triggered under a certain condition.	**Event**
	`GovernmentalRestriction has Implication & TriggerCondition; ReproductionRate causesRestriction GovernmentalRestriction; ContactRate & Shipment isImpactedByRestriction GovernmentalRestriction`	`Lockdown`
3.2	A numeric factor with a fixed value, e.g. input data.	**Parameter**
	`Shipment has TransitTime`	`TransitTime1, TransitTime2`
	`PandemicProperty has Value; AvgSicknessDuration & Infectivity isSubclassOf PandemicProperty`	`AvgSicknessDuration, Infectivity`
	`PopulationProperty has Value; ContactRate & HealthcareCapacity & PopulationSize isSubclassOf PopulationProperty`	`ContactRate, HealthcareCapacity, PopulationSize`

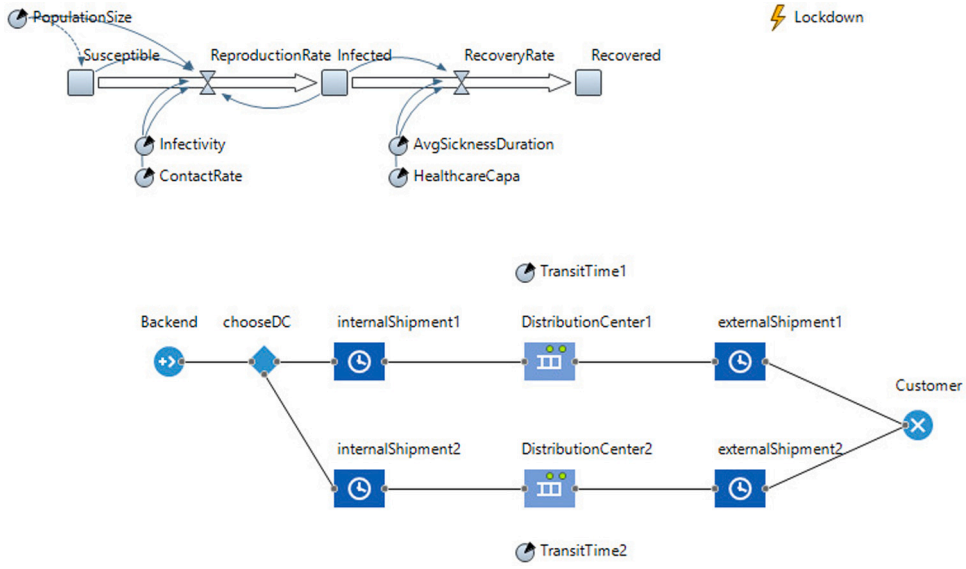

Fig. 5. Automatically generated simulation model in AnyLogic.

Protégé is used for mapping the knowledge between both ontologies. The free and open-source software application of Stanford University is chosen due its large community in academia and industrial corporations. The parsing workflow (cf. Fig. 3) starts with querying the instantiated Simulation Ontology on the *Apache Jena Fuseki* server by using the two SPARQL queries. Their output is stored in respective CSV files, which are then read in by the Parser - a script encoded in *Python*. It transforms the formal model into the XML-structure of an ALP file. The thereby resulting AnyLogic simulation model is depicted in Fig. 5.

Its dynamic behavior follows the causal relationships as defined in the Use Case Ontology: The upper submodel is related to the spread of the pandemic over time in an SIR-based approach. As soon as the reproduction rate between susceptible and infected people exceeds a certain threshold, a lockdown is imposed. This governmental restriction not only reduces the contact rate within the observed population but also increases the transit time of shipments throughout the respective country or region. The order-fulfillment process in the semiconductor industry is represented by the lower submodel. Its exact configuration highly depends on the nature of the specific company as well as the products it is specialized in. Since this application aims at a high degree of generalizability and not on a thorough company-specific case study, only two downstream distribution channels, i.e. ex-backend fabrication, are distinguished. Which actual supply chain a product takes to a certain customer depends on a complex decision logic. Nevertheless, the transit times of an internal shipment from the backend to a certain distribution center as well as the subsequent external shipment to the customer have a major impact on this routing decision and thus on the manufacturing, inventory and transport capacities required along the respective path.

The parameters associated with all depicted building blocks, such as the infectivity of SARS-CoV-2 or the actual transit times throughout the two supply chains, originate from the Use Case Ontology as well. They were linked to the data properties of the Simulation Ontology in the course of model conceptualization and further transferred to the model source code by the Parser. When executing the model, one can observe a change in routing as soon as a lockdown is caused that affects one of the potential supply chains. This enables the simulation engineer to conduct detailed experiments with different data sets for all regions of interest and thereby develop profound scenarios. Based on these, decision support can be provided on which governments are most likely to impose new lockdowns and which manufacturing and distribution sites of the particular company are affected by delayed shipments. This enables supply chain managers to proactively adjust the global capacities and thus increase the robustness of their entire network.

5.2. Discussion

The following discussion of the proposed framework is twofold. First, the approach itself is validated based on the findings from the

use case application. Second, additional considerations and enablers for industrial practice are identified.

The application scenario proofed that our framework is able to exploit the semantic capabilities of an ontology describing a certain system for the efficient development of a respective simulation model. The scope of the use case has been chosen to provide a clear end-to-end exemplification rather than a thorough company-specific case study. The setting enabled the simulation engineer to efficiently gain the required knowledge about the pandemic and its impacts on the supply chain without having to rely on tedious discussions with domain experts. Model conceptualization by using the novel Simulation Ontology appeared to be viable. However, the Simulation Ontology so far only incorporates the basic elements of DES, ABM and SD models and only a few basic parameters per element. The addition of further simulation-related concepts would require an extension of the class taxonomy and respective object and data properties as well as new mapping rules. Another critical aspect to consider is the representation of complex control and decision logics, as for instance often found in advanced manufacturing equipment. The current best practice in simulation modeling revealed, that such mechanisms are typically best to be addressed by direct program code as a parameter function of a respective building block. In our approach, it is possible to copy Java functions to the value of a property instance in the Simulation Ontology, and the Parser as well as AnyLogic are able to compile them. Despite being feasible from a technical point of view, the practicability of such a procedure is a subject to test in actual applications. At the current stage, mapping support for parameters is provided by the semantic properties within the Simulation Ontology. In case of such complexity, a further development of the static Mapping Rules to a dynamic support system for model conceptualization is expected to be promising. With regard to model generation, the proposed automation procedure contributed to an efficient use case application. It did not reduce the complexity of system knowledge but led to a low resource expenditure for model encoding. A large-scale case study with a multitude of required building blocks and parameters is expected to further underline this.

Besides this efficiency, the overall effectiveness in correctly building the most appropriate model is a subject best to be evaluated by an application to other use cases. As shown in Fig. 1, the model has to be verified (syntax errors) and validated (semantic errors) to enable meaningful experiments and thereby decision support. The impact of potentially required modeling iterations is highly case-dependent. In case of minor modeling mistakes, such as single missing or wrongly placed parameters, the user might either directly correct them within the simulation software or adjust the already instantiated Simulation Ontology correspondingly and execute the Parser again. In case of rather structural issues, such as wrong model behavior, the simulation engineer might either study the Use Case Ontology again or discuss the instantiated Simulation Ontology with a local domain expert by use of the ontology language and the associated Mapping Rules. Hence, the presented framework is overall assumed to be quite resilient against potential dynamics.

In the particular view from industrial practice, we assume that simulation engineers might be sceptical whether they should use the Simulation Ontology and the Mapping Rules instead of simply one of their simulation applications. However, it can be observed that they are usually only familiar with a single modeling technique according to their personal preference and experience. This implies that an expert in the field of DES can benefit from our work by gaining additional knowledge on additional model types, that might be more suitable for specific problem settings. Furthermore, the ontology-based approach is expected to improve the general communication along the entire process, since it enables all kinds of stakeholders to better understand the world of simulations and their mostly complex models. Regarding the Parser for automated model generation, one has to ensure that the script is capable of all potentially required building blocks and parameters. But this initial coding effort is negligible since the savings in time accumulate with every application and the algorithm itself showed to be highly efficient at a runtime in the range of seconds — and not days or even weeks as for manual model generation procedures.

To assess our framework as fully suitable for the complexity of industrial applications, three key enablers are required beforehand. (1) Since the framework builds up on the existence of a certain domain ontology, a sufficient web of linked data is required. The more an organization uses semantic web technologies for their general knowledge management, the more useful our approach is expected to be. (2) People also need to be trained on and willing to work with these technologies. Our process requires more tools and interfaces than the current best practice in simulation modeling. However, the enhanced comprehensibility of the system, of the general constructs behind a simulation and of the generated model as well as the overall potential reduction in resource expenditure can justify our framework and thus the necessity for an adequate change management. (3) A related but already prior existing weak point is the general acceptance of model-based decision support. Practitioners still report that upper level decision makers are often not willing to rely on results provided by a simulation study. However, we expect the continuously increasing complexity of systems as well as external disruptions such as COVID-19 to rise the need for tools that are able to provide insights into such complex, dynamic and uncertain environments. The enhanced comprehensibility of simulation models defined in the language of an ontology is further expected to make the models themselves and thereby also their results more tangible due to an increased level of felt truth.

6. Conclusion and outlook

Our framework creates an opportunity to use the system knowledge stored in an ontology for the efficient development of a digital replica for simulations. This enables fast and profound model-based decisions, which is particularly beneficial in a dynamic and uncertain environment. Our framework consists of three components: (1) the Simulation Ontology, a semantic model for the basic building blocks and interrelationships of a simulation, (2) Mapping Rules that support the transfer of knowledge from an existing domain ontology into instances of the Simulation Ontology and thereby present a novel approach for model conceptualization, as well as (3) a Parser, which automatically generates an executable simulation model from the instantiated Simulation Ontology. Our approach can be classified as semi-automatic, whereby the steps of model development are supported and model generation is automated. Our approach is not limited to a particular simulation type, but supports DES, ABM, SD as well as hybrid modeling.

We applied the framework in the exemplary supply chain of a semiconductor manufacturer. By exploiting an industrial knowledge base about the COVID-19 pandemic and its impacts on the company's order-fulfillment process, we were able to efficiently create a simulation model in which SD and DES components interact for the prediction of new lockdowns and thus appropriate countermeasures.

Based on the insights gained throughout the development and application of our approach, several potentials remain for future investigations. Our particular use case application resulted in a hybrid simulation with SD and DES submodels. Further use cases, which might also include ABM components, represent a promising approach to further test the overall practicability. Overall, a thorough case study where a traditional modeling approach is compared with an ontology-based procedure might be able to quantify the expected savings in time. In addition to our general path, i.e. the development and implementation of a simulation model based on knowledge stored in an existing domain ontology, the investigation and integration of the reverse direction, i.e. the transfer of gained knowledge from a simulation study back to the ontology, represents another interesting subject for research. Especially under volatile, uncertain, complex and ambiguous business

conditions, the efficient gathering of knowledge in ontologies, fast findings from simulation studies and their feedback into the semantic knowledge base could lead to a better understanding of the situation and more agility as well as sustainability in decision making. We acknowledge similar research efforts in adjacent domains that deal with similar problem settings (see Wang et al. [35], e.g.) that may benefit from our findings and can be an inspirational perspective for our future research endeavors.

CRediT authorship contribution statement

Wiking Jurasky: Conceptualization, Methodology, Software, Writing - original draft , Writing - reviewing and editing. **Patrick Moder:** Conceptualization, Validation, Writing - original draft, Writing - reviewing and editing. **Michael Milde:** Conceptualization, Methodology, Writing - original draft , Writing - reviewing and editing. **Hans Ehm:** Writing - reviewing and editing, Supervision, Funding acquisition. **Gunther Reinhart:** Supervision, Funding acquisition.

Declaration of competing interest

One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. For full disclosure statements refer to https://doi.org/10.1016/j.rcim.2021.102174. We declare that Wiking Jurasky, Patrick Moder and Hans Ehm are employed with Infneon Technologies AG.

Acknowledgments

The research has received funding from the EU ECSEL JU under the H2020 Framework Programme, JU grant no. 783163 (project iDev40), no. 826452 (project Arrowhead Tools), and from the partner's national programs/funding authorities.

References

[1] N. Bennett, G.J. Lemoine, What a difference a word makes: Understanding threats to performance in a VUCA world, Bus. Horiz. 57 (3) (2014) 311–317.
[2] V.R.S. Kumar, A. Khamis, S. Fiorini, J.L. Carbonera, A.O. Alarcos, M. Habib, P. Goncalves, H. Li, J.I. Olszewska, Ontologies for industry 4.0, Knowl. Eng. Rev. 34 (2019).
[3] N. Huang, S. Diao, Ontology-based enterprise knowledge integration, Robot. Comput.-Integr. Manuf. 24 (4) (2008) 562–571.
[4] P. Barlas, C. Heavey, KE tool: an open source software for automated input data in discrete event simulation projects, in: 2016 Winter Simulation Conference (WSC), IEEE, 2016, pp. 472–483.
[5] A. Skoogh, B. Johansson, J. Stahre, Automated input data management: evaluation of a concept for reduced time consumption in discrete event simulation, Simulation 88 (11) (2012) 1279–1293.
[6] M. Milde, G. Reinhart, Automated model development and parametrization of material flow simulations, in: 2019 Winter Simulation Conference (WSC), IEEE, 2019, pp. 2166–2177.
[7] P.A. Fishwick, J.A. Miller, Ontologies for modeling and simulation: issues and approaches, in: Proceedings of the 2004 Winter Simulation Conference, 2004. Vol. 1, IEEE, 2004, pp. 259–264.
[8] G. Antoniou, P. Groth, F. van Harmelen, R. Hoekstra, A Semantic Web Primer, in: Cooperative Information Systems, MIT Press, 2012.
[9] L. Mönch, J.W. Fowler, S. Dauzere-Peres, S.J. Mason, O. Rose, A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations, J. Sched. 14 (6) (2011) 583–599.
[10] C.S. Currie, J.W. Fowler, K. Kotiadis, T. Monks, B.S. Onggo, D.A. Robertson, A.A. Tako, How simulation modelling can help reduce the impact of COVID-19, J. Simul. (2020) 1–15.
[11] T.R. Gruber, A translation approach to portable ontology specifications, Knowl. Acquis. 5 (2) (1993) 199–220.
[12] N.F. Noy, D.L. McGuinness, Ontology Development 101: A Guide to Creating your First Ontology, Stanford knowledge systems laboratory technical report KSL-01-05, 2001.
[13] M. Uschold, M. Grüninger, Ontologies: Principles, methods and applications, Knowl. Eng. Rev. 11 (1996).
[14] VDI, Modellierung und simulation - Modellbildungsprozess, 2016, https://www.vdi.de/richtlinien/details/vdi-4465-blatt-1-modellierung-und-simulation-modellbildungsprozess [accessed 19 June 2020].
[15] D. Mourtzis, M. Doukas, D. Bernidaki, Simulation in manufacturing: Review and challenges, Procedia Cirp 25 (2014) 213–229.
[16] G.S. Fishman, Discrete-Event Simulation: Modeling, Programming, and Analysis, in: Springer Series in Operations Research, Springer, New York, NY, 2001.
[17] S.C. Brailsford, T. Eldabi, M. Kunc, N. Mustafee, A.F. Osorio, Hybrid simulation modelling in operational research: A state-of-the-art review, European J. Oper. Res. 278 (3) (2019) 721–737.
[18] L. McGinnis, E. Huang, K.S. Kwon, V. Ustun, Ontologies and simulation: a practical approach, J. Simul. 5 (3) (2011) 190–201.
[19] P. Benjamin, M. Patki, R. Mayer, Using ontologies for simulation modeling, in: Proceedings of the 2006 Winter Simulation Conference, IEEE, 2006, pp. 1151–1159.
[20] M. Rabe, P. Gocev, Applying semantic web technologies for efficient preparation of simulation studies in manufacturing, in: Proceedings of the 2012 Winter Simulation Conference (WSC), IEEE, 2012, pp. 1–12.
[21] J. Du, H. Jing, K.-K.R. Choo, V. Sugumaran, D. Castro-Lacouture, An ontology and multi-agent based decision support framework for prefabricated component supply chain, Inf. Syst. Front. (2019) 1–19.
[22] M. Fayez, L. Rabelo, M. Mollaghasemi, Ontologies for supply chain simulation modeling, in: Proceedings of the Winter Simulation Conference, 2005, IEEE, 2005, pp. 2364–2370.
[23] D. Cope, Automatic Generation of Supply Chain Simulation Models from SCOR based Ontologies, Vol. 2004–2019 (Electronic Theses and Dissertations), STARS, 2008.
[24] Y.-J. Chen, Y.-M. Chen, An XML-based modular system analysis and design for supply chain simulation, Robot. Comput.-Integr. Manuf. 25 (2) (2009) 289–302.
[25] T. Wagner, A. Gellrich, C. Schwenke, K. Kabitzsch, G. Schneider, Automated planning and creation of simulation experiments with a domain specific ontology for semiconductor manufacturing AMHS, in: Proceedings of the Winter Simulation Conference 2014, IEEE, 2014, pp. 2628–2639.
[26] G.A. Silver, J.A. Miller, M. Hybinette, G. Baramidze, W.S. York, An ontology for discrete-event modeling and simulation, Simulation 87 (9) (2011) 747–773.
[27] O. Batarseh, L. McGinnis, SysML to discrete-event simulation to analyze electronic assembly systems, in: Proceedings of the 2012 Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium, 2012, pp. 1–8.
[28] P.-C. Lin, L. McGinnis, Test problems, reference models and fab simulation, in: 2017 Winter Simulation Conference (WSC), IEEE, 2017, pp. 3624–3635.
[29] H. Ehm, L. McGinnis, O. Rose, Are simulation standards in our future? in: Proceedings of the 2009 Winter Simulation Conference (WSC), IEEE, 2009, pp. 1695–1702.
[30] J. Ashayeri, L. Lemmes, Economic value added of supply chain demand planning: A system dynamics simulation, Robot. Comput.-Integr. Manuf. 22 (5) (2006) 550–556.
[31] X. Yuan, L. Shen, J. Ashayeri, Dynamic simulation assessment of collaboration strategies to manage demand gap in high-tech product diffusion, Robot. Comput.-Integr. Manuf. 26 (6) (2010) 647–657.
[32] P. Hitzler, M. Krotzsch, S. Rudolph, Foundations of Semantic Web Technologies, in: Chapman & Hall/CRC Textbooks in Computing, CRC Press, 2009.
[33] A. Borshchev, S. Brailsford, L. Churilov, B. Dangerfield, Multi-method modelling: Anylogic, in: Discrete-Event Simulation and System Dynamics for Management Decision Making, Wiley Online Library, 2014, pp. 248–279.
[34] W. Jurasky, Simulation ontology & parser, 2021, http://dx.doi.org/10.17632/w86w9yn572.1.
[35] H. Wang, G. Wang, J. Lu, C. Ma, Ontology supporting model-based systems engineering based on a GOPPRR approach, in: World Conference on Information Systems and Technologies, Springer, 2019, pp. 426–436.