Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

METHOD TO MODEL ACTIONS FOR DISCRETE-EVENT SIMULATIONS OF LOGISTICS NETWORKS

Markus Rabe

IT in Production and Logistics
TU Dortmund
Leonhard-Euler-Str. 5
44227 Dortmund, GERMANY

Dominik Schmitt
Felix Dross

Graduate School of Logistics
TU Dortmund
Leonhard-Euler-Str. 5
44227 Dortmund, GERMANY

ABSTRACT

Managers of logistics networks have the complex task of continuously maintaining their network in good operating conditions under a changing environment. Thus, they need to identify promising actions to adapt and improve the logistics network. Such actions could be the relocation of stock or the adjustment of transport relations. In order to support the managers, the authors have previously proposed a decision support system (DSS) based on discrete-event simulation (DES). The DSS automatically examines possible actions and suggests the best actions found to the managers. Since a data-driven DES approach is used for this DSS, all actions can be described as changes to a database. In this paper, an approach for modeling, integrating and executing user-generated actions into the DSS is described, in order to increase its flexibility and usability. In conclusion, the authors propose to develop a domain-specific modeling language for modeling actions for DES models.

1 INTRODUCTION

Large logistics networks are complex systems that are very hard to manage. To cope with the complexity of logistics networks, several specific performance measurement systems (PMS) for logistics networks have been developed in the past (Gleich and Quitt 2011; Bruns and Hegmanns 2013; Biesen et al. 2013; Gladen 2014). PMS are typically realized with data warehouse systems (Bauer and Günzel 2013). Within a PMS, different performance measures are hierarchically accumulated to result in a key performance indicator (KPI). The purpose of PMS is to provide managers with the background information they need to improve their area of accountability.

Some PMS also provide catalogs of possible corrective actions in the logistics network for different KPI situations. These catalogs are designed to help the managers with the decisions about the consequences of KPI situations. Examples of such actions in a logistics network could be the relocation of stock from one site to another or the adjustment of transport relations within the network. To assist the managers even more and to automate the process of reacting to KPI reports, the concept of KPI monitoring systems (KPIMS) has been developed and previously presented in Dross and Rabe (2014). KPIMS perform an automated analysis of KPI reports on a regular basis. Each KPIMS constantly monitors one KPI and sends an individually composed KPI alert to a responsible manager whenever the KPI exceeds certain predefined limits. An alert consists of two parts: The performance measure that caused the KPI to deteriorate and a set of possible actions that could be performed by the addressed manager in order to improve the KPI.

While the automated surveillance of the different KPIs seems to be a sensible approach, a problem with this setup has already been identified in the past (Dross and Rabe 2014). The different KPIMS are not connected to each other, and each alert is sent to the managers of the company independently. As a result, the actions suggested by one KPIMS could improve its own KPI while possibly worsen one or more other KPIs. Therefore, the initial setup could lead to a decline in the overall network performance instead of an improvement. In order to solve this problem, Rabe and Dross (2014; 2015; 2015; 2016; 2017b) have proposed to develop a decision support system (DSS) for logistics networks in materials trading which uses a simheuristic approach to support the managers with suggestions for sensible action combinations. A simheuristic approach combines a simulation model with a metaheuristic to solve stochastic optimization problems (Juan and Rabe 2013). It is a simulation-based optimization approach with the simulation as an evaluation function of the optimization algorithm as described by März et al. (2011). In the case of the DSS proposed by Rabe and Dross, which is briefly described in section 2, a data-driven discrete-event simulation (DES) model of a logistics network is used to predict the KPI consequences of different actions in the logistics network. A metaheuristic is then used to search for sensible action combinations within the very large search space of all possible action combinations. Effectively, the metaheuristic uses the simulation model as the evaluation function. Action sets that can improve one or more KPIs while not significantly worsening other KPIs have been described as integrated action sets (Dross and Rabe 2014). The overall purpose of the DSS is to find those integrated action sets automatically and suggest them to the managers in a suitable way.

The research presented in this paper is conducted in cooperation with an international materials trading company. The company operates a large, complex and heterogeneous logistics network with almost 500 warehouses and an inventory of around 150,000 items in permanent stock. The DES model for the research has been built with the simulation tool SimChain from SimPlan AG (Gutenschwager and Alicke 2004; SimPlan AG 2017). The software has been chosen because the authors previously had good experiences of using the software in the E-SAVE project (Rabe et al. 2013; e-SAVE 2015) and because of its beneficial design as a data-driven simulation tool. The design principle of SimChain provided the necessary flexibility for the development of the DSS, as is further outlined in section 2.

The DSS is currently under further development to increase its usability and flexibility. Therefore, a concept for extending the system with user-generated action types, a generic pattern for actions, is introduced in this paper. To ease the process of creating action types, the authors propose the development of a domain-specific modeling language (DSML) for the description of actions in logistics networks. An approach for integrating this new modeling language into the architecture of the DSS is presented. Furthermore, an initial set of requirements for the modeling language is presented.

Overall, the paper is structured as follows: Section 2 gives an overview of related work. In section 3, the authors present some extensions recently made to the DSS. These extensions can be seen as the basis for the work presented in sections 4 and 5. Section 4 illustrates the author's approach for creating new action types and describes how to integrate them into the DSS as generic patterns for all evaluated actions. Section 5 presents the requirements for a DSML that can be used to support the described approach. Section 6 closes the paper with a conclusion and an outlook.

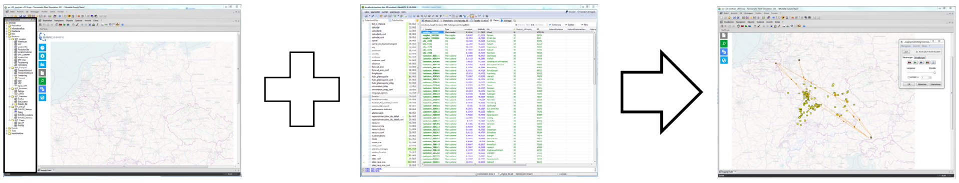
2 RELATED WORK

This section presents the related work of this paper. First, the simulation tool SimChain is introduced in section 2.1. Section 2.2 defines the principles of DSML, as the basis for a requirement framework for the DSML proposed to be developed. Section 2.3 gives an overview of related work regarding DSS for logistics networks that influenced this work. Section 2.4 presents the DSS for logistics networks in materials trading developed by the authors.

2.1 Simulation Tool SimChain

The logistics network simulation tool SimChain (Gutenschwager and Alicke 2004; SimPlan AG 2017) has been used for the prototypical implementation of the simulation part of the DSS. SimChain offers two components: a set of generic building blocks for a logistics network simulation in Siemens Plant Simulation (Siemens PLM Software 2017) and a corresponding data model stored in a MySQL database.

The data model comprises 50 tables that store the complete parameterization for the generic building blocks, including their placement on the map. Therefore, also the structure of the network is stored in the data model. At run time, the actual simulation model is dynamically instantiated from the data model. This principle is illustrated in figure 1.



SimChain model with generic building blocks

MySQL database with model parameters

Dynamically instantiated simulation model

Figure 1: Working principle of the SimChain simulation software (Rabe et al. 2017b).

For the work presented here, two design concepts of SimChain were of importance. First, the modeling of the respective logistics network is done by setting the parameters in the database. This concept offers a complete data-level interface to the simulation model. Desired changes in the simulation model can be implemented by changing entries in the database. It is, therefore, possible to write software which can automatically apply changes to the simulation model. Second, SimChain simulates not only the material flow but also the information flow within the logistics network. The logistics network system load can, therefore, be expressed regarding customer orders. The orders are processed through the system from customers to sites to suppliers, and the material flow is simulated from suppliers to sites to customers. Hence, it is possible to change the logistics system configuration, e.g., the allocation of stock-keeping units to sites, while keeping the customer orders the same as before, and, therefore, to compare the measurements for the two alternative system configurations. The simulation output data and statistics are written to specific tables in the MySQL database after each simulation run. From there, software can read the results. Further descriptions of this mechanism can also be found in Rabe and Dross (2015) and Rabe et al. (2017b).

2.2 Domain-specific Modeling Languages

A modeling language, in general, can be used to express either information, knowledge or systems in a structured way. The structure of the modeling language is defined by a consistent set of rules. The rules can be used to interpret the meaning of the components in the structure. A complete definition of a modeling language consists of the description of its syntax, including well-formedness rules, and its semantics (meaning) (Harel and Rumpe 2004). A modeling language is often used to describe a system in an abstracted way to reduce its complexity. It can be expressed in a textual or graphical way. Two main reasons for using a modeling language are to simplify the modeling process and to ease the understanding of the modeled system. A prominent example for a modeling language is the unified modeling language (UML) (Object Management Group 2015).

In general, languages can be differentiated into general-purpose languages and domain-specific languages (DSL). While UML, which is typically seen as a general-purpose modeling language, is widely used, DSMLs are emerging that allow developers and even customers for expressing solutions to well-defined problems in a concise way (Grönniger et al. 2009). Van Deursen et al. (2000) describe a DSL as a "programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain." Thus, DSML enable higher abstraction levels than general-purpose modeling languages by focusing on a restricted application. Additionally, DSLs, in general, tend to a restricted set of notations and abstractions (van Deursen et al. 2000). As a consequence, fewer low-level details need to be specified. DSLs are also called micro-languages or little languages in the literature (Bentley 1986).

Fowler and Parsons (2011) divided DSLs into three main categories: internal DSLs, external DSLs and language workbenches. Usually, an internal language has the same language as the one from the main application. It is restricted to a small subset of the language`s features using its own particular style. One famous example of an internal DSL is Lisp. However, an external domain-specific language differs from the hosted language. It is defined as a language separate from the one used for the main application. Usually, it has its own syntax and needs to be parsed into a target language. Examples for external DSLs are XML and SQL. A language workbench is described as an integrated development environment for defining and implementing DSLs. It allows the user not only for determining the language`s structure but also for writing DSL scripts which combine the editing environment with the domain-specific language.

In conclusion, a more focused abstraction of the original system can be reached by using a DSML.

2.3 Decision Support for Logistics Networks

DSS for logistics networks have been described as logistics assistance systems (LAS) by Blutner et al. (2007) and Kuhn et al. (2008). According to these authors, LAS are special systems that support logistics managers at critical decisions. LAS should quickly identify critical logistics situations and offer possibilities to objectively evaluate the consequences of possible decision alternatives. Throughout the literature, the terms LAS and DSS for logistics networks are mostly used synonymously. In the international literature, the term DSS is the more widely used term (Shim et al. 2002; Kengpol 2008). Therefore, the authors decided to consistently use the term DSS, although the system described in this paper could also be referred to as a LAS.

Regarding further previous research, Deiseroth et al. (2008) describe a simulation-based DSS for the disposition of global supply chains. Bockholt et al. (2011) describe a DSS for collaborative supply chain planning. Both approaches mainly target supply chains in the automotive sector. Liebler et al. (2013) give a detailed introduction to OTD-NET, an order-to-delivery network simulation and DSS for complex production and logistics networks. As in Deiseroth et al. (2008) and Bockholt et al. (2011), the focus lies on supply chains in the automotive sector.

2.4 Decision Support System for Logistics Networks in Materials Trading

The architecture of a DSS for logistics networks in materials trading developed by the authors is shown in figure 2. This section provides a brief description of this system. For an extended overview of the initial working principles of the DSS and the related work regarding the DSS, the reader is kindly referred to previous publications of the authors regarding the DSS (Dross and Rabe 2014; Rabe et al. 2015; Rabe and Dross 2015; Rabe and Dross 2016; Rabe et al. 2017b).

Basis systems in every large logistics company, such as the transactional systems (e.g., SAP R/3), the data warehouse (DWH), and the components of the DSS are displayed as rectangles in figure 2. As briefly described in the introduction, the company's data is regularly collected and transferred to the DWH to calculate KPIs on a regular basis. Each KPI is monitored by one KPIMS, which is essentially a program that can execute queries on the DWH. The KPIMS periodically generate KPI reports and examine those KPI reports regarding predefined criteria. Certain conditions can then trigger a KPI alert, e.g., if a KPI leaves a specific predefined corridor. The KPI alert consists of two main pieces of information, the fact triggering the alert and a list of possible actions to return the KPI to its predefined optimum. The action proposals generated by the KPIMS are routed to the DSS for further evaluation.

Since the DSS uses a simheuristic approach, it needs a simulation model of the logistics network as an evaluation function. To always have a current simulation model, the company's data is automatically extracted, enriched and transformed into the data model of a data-driven DES tool. This is done by a component called model builder. The model builder is a program that can automatically perform the generation of the data model from the raw data. The rules for the data model generation are programmed into the model builder. As long as the structure of the raw data stays the same, it is, therefore, possible to update the data model with new data at any time. In case of the planned usage, this would be once at the end of each month.

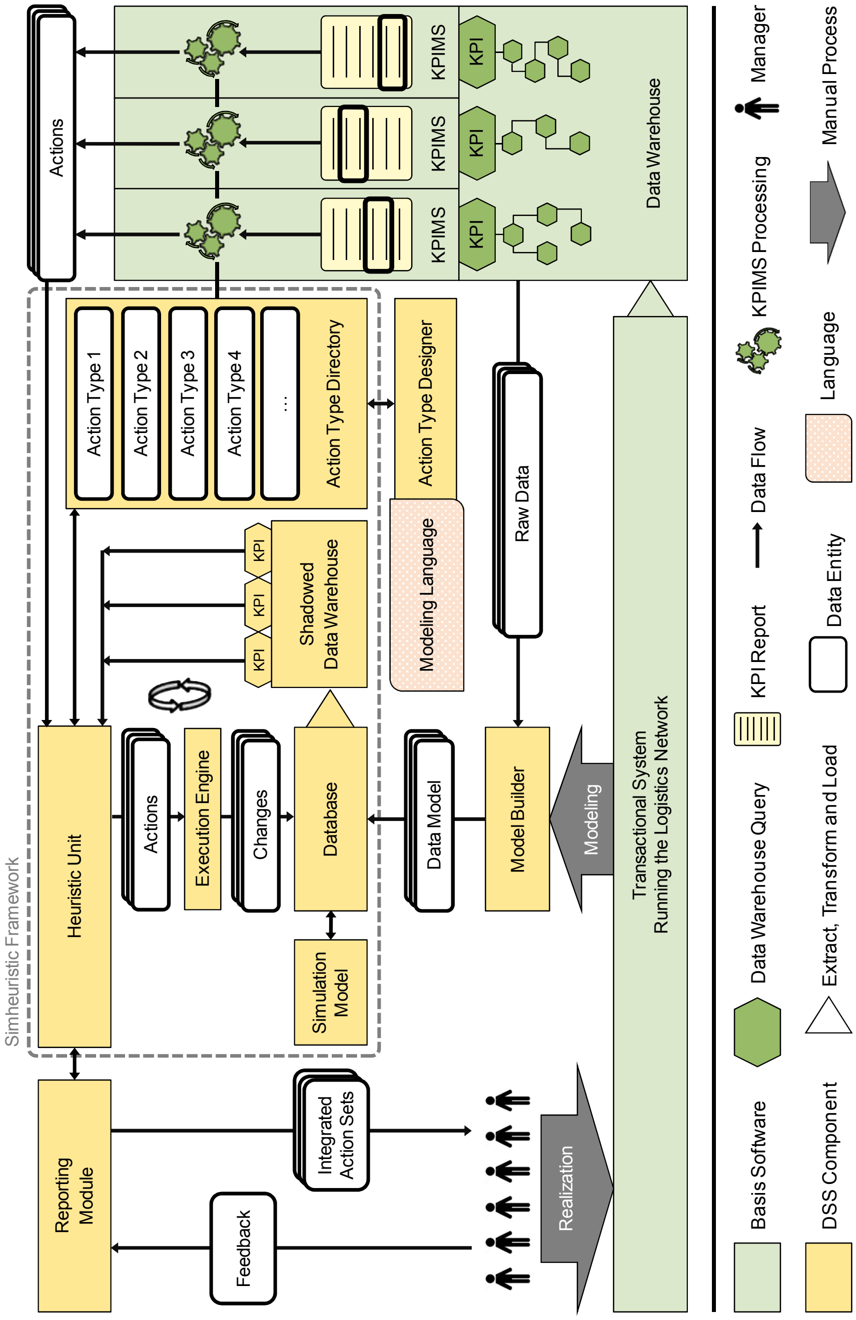


Figure 2: Architecture of the decision support system, based on Rabe and Dross (2014, 2015, 2017a).

In the implemented prototype, the simulation unit is realized with SimChain. As described in Section 2.1, SimChain consists of a simulation model with generic building blocks and the corresponding tables for the parameterization of the building blocks in a database. The simulation output data are also written to special output tables in the database. From there, these simulation results data can be transferred to the shadowed DWH (SDWH). The KPI calculation logics are copied to the SDWH within the DSS for KPI evaluation purposes. This separation ensures that no simulation output data are mixed with real world data from the transactional systems in the original data warehouse.

A heuristic unit (HU) can automatically analyze possible interdependencies of the different actions suggested by the KPIMS by testing them on the DES model. The overall goal is to find the best sets of actions regarding the overall performance of the logistics network. Possible actions are executed on the data model through an execution engine which decomposes actions into changes and applies those changes to the database. This process is described in more detail in the remainder of this paper.

Once one or more actions are applied and it is necessary to evaluate the action set's effects, the simulation model is instantiated, and a simulation experiment is conducted. The results of the simulation experiment are loaded into the SDWH to be analyzed with the KPI logic. This process can be repeated iteratively. Since an exhaustive search of the complete solution space of possible action sets is very time-consuming and unfeasible for larger simulation models, the authors experiment with different algorithms for the HU. Currently, two approaches have been implemented. The first approach is an evolutionary algorithm (EA) which models action sets as individuals. The second approach uses deep reinforcement learning (DRL) to model an agent that iteratively learns to construct promising action sets. Both approaches have advantages and disadvantages, which are discussed in detail in other papers of the authors (Rabe and Dross 2015; Rabe et al. 2017a; Rabe et al. 2017b).

Using this simheuristic framework, the HU can test different action sets before suggesting integrated action sets to the users via the reporting module. A feedback mechanism can provide the HU with all information that cannot be automatically extracted from the SDWH. This information could include actual effects of actions in the logistics network or feedback from users reporting unfeasible action combinations.

Summarizing, the authors have developed a concerted reference model for the implementation of a DSS for logistics networks and created a working prototype of the described system with SimChain and the python programming language. Currently, only small simulation models are tested, but both, the EA approach and the DRL approach produce promising results. Furthermore, a web-based implementation of the reporting module provides a client-independent access to the system from a browser.

Currently, the actions provided to the system need to be hard-coded into the execution engine to be executable by the HU. This requires too much manual work whenever new action types are introduced. Therefore, an extension is necessary which makes the system more flexible. The recently developed concept for this extension is presented in the remainder.

3 EXTENSIONS MADE TO THE ARCHITECTURE OF THE DSS

A single action, e.g., shifting the assortment of several locations to a central warehouse, can lead to hundreds of SQL statements, which may result in thousands of changes in the database. As mentioned in the introduction, the authors propose to develop a DSML to describe actions on DES models of logistics networks to abstract a large amount of changes to the DES data model. Therefore, it should be easier for the modeling person to handle the complexity of changes to the DES data model. The DSML is depicted under the simheuristic framework in figure 2. There are no connections from or to this icon since the new language should be seen as an aspect which is going through the complete system. Therefore, it is affecting multiple parts of the DSS, not just a few components. Details regarding the proposed DSML and its requirements are presented in section 5.

New action types are created in the action type designer, which is shown below the simheuristic framework in figure 2. As a concept for further extensions, any action type may be saved in the action type directory. The action type directory can be accessed from within the action type designer, e.g., for editing or concatenating existing action types. It holds all possible actions types for the HU and is, therefore, a new part of the simheuristic framework.

Besides the action type designer, further components of the DSS can access the stored action types as also displayed in figure 2. Alongside the HU, the KPIMS needs to take all possible actions for the logistics network into account. Whenever new action types are added to the system, these new options are provided to these components. As briefly mentioned above, the HU should be able to integrate the newly available action types into its search. The action types effectively display decision variables for the heuristic and, therefore, changing the available action types will affect the search space for the heuristic. How the system could handle this circumstance will be subject to further research. The authors also aim to extend the possibilities of the HU in another way. The HU should be able to learn with any simulation experiment. After each iteration of the heuristic, the HU could save information about the action type, e.g., the success of an action type regarding the overall performance of the system. Even though the system changes over time by applying actions, the information might be used as an indicator for promising actions for later runs of the heuristic.

4 APPROACH FOR CREATING AND APPLYING ACTION TYPES

In order to increase the flexibility and usability of the DSS, the authors propose an approach for describing complex adjustments in logistics networks as changes to the DES model. These changes can be derived from user-generated actions. The authors pursue a general approach by abstracting similar actions in the form of action types, a generic pattern for specific actions. To realize the approach, a method to generate, integrate and execute user-generated action types is needed. The correspondingly developed process is shown in figure 3.

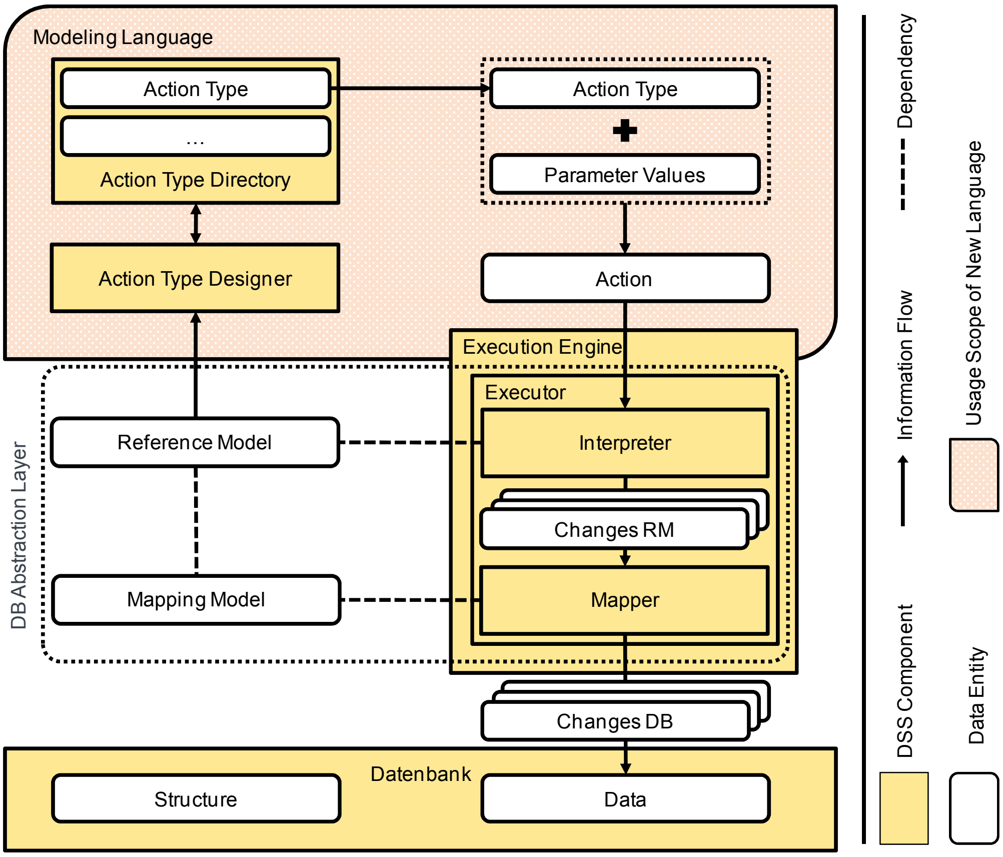


Figure 3: Process of creating an action type and applying changes to the database.

A key principle of the proposed DSS is that the simulation model gets instantiated from the data model, which is based on a SQL database. Thus, actions are not implemented by changing the simulation model directly, but by applying changes to the data model. The database consists of two parts. The data structure, including structural information, e.g., tables and relations and the data, containing the information that represents the modeled logistics network.

Depending on the structure of the data model, an action type is related to specific parts of the data structure, e.g., when creating a new transportation route, a new entry in the corresponding table needs to be added. Thus, the authors propose to integrate the data models' structure into the action types.

Action types are not related directly to the underlying data base, but instead they are related to a persistent and predefined reference model. A mapping model connects the refence model with the underlying database by linking their data structures. This abstraction can be interpreted as a database abstraction layer. Thus, the method for creating and applying action types can work on different data models without adapting the action types. Instead, only the link between the structure of the reference model and the structure of the underlying database needs to be adjusted. By using a database abstraction layer, a generalization and increase in flexibility of the process is achieved.

The main components of each action type are the relations to the corresponding tables in the database where possible changes are applied. An action type can be related to multiple parts of the data structure. Thus, defining an action type with its complex changes and further attributes can be very difficult, e.g., defining all necessary SQL statements to apply the corresponding changes to the data. To ease this process, an abstraction layer is needed. Therefore, the authors are proposing a DSML for describing action types in DES models. The aim of integrating such a language is to simplify the process of creating new action types and managing its complexity. A modeling language can provide an integrated and consistent way of describing new action types. The syntax for those action types and, therefore, the syntax of the actions, is described by the modeling language.

An action type can be related to one or more parts of the data structure. The modeling language is used to define the matching between the action type and corresponding tables, e.g., for an action type that adds an SKU to a site, a relation between the action type and the corresponding SKU table is modeled. To utilize all possibilities of creating and configuring new action types, a feasible user interface is needed, depicted as the action type designer in figure 3. The designer may be implemented as a text-based user interface or a graphical-based user interface. The converted data structure is provided to the action type designer. Within the editor, the user can access the constructs of the modeling language in close conjunction with the data model to create new action types.

Action types can consist of different information. First, the user can add a description to the action type containing information about the modeled changes to the underlying database. Additionally, required input values for the action type may be added to the description. Thus, an action type needs some placeholders for these values (parameters). Instantiating the action type into an action, these parameters determine which concrete changes are applied to the data. For example, when adding a transport relation (action type "add_transport_relation"), the parameter values for "start" and "destination" define the specific starting location and the destination. Depending on the number of locations in the DES model, the number of possible actions from the action type can be calculated. For example, if there would be five locations in the DES model, the action type "add_transport_relation" would result in 20 possible actions (one action for each possible route from each location to each other location).

As displayed in figure 3, the user can access the data structure from the editor. Using a modeling language, the user is supported with an integrated method to model the relations between an action type and the database. By providing a list of database tables to the user, false specifications of these relations may be reduced.

Additional domain-specific information may be added to the action types. This knowledge may be used by the HU to improve the search for integrated action sets. A list of correlations between action types may be added, e.g., when shifting an assortment to a central warehouse, adjusting the minimum order amount or the minimum stock of corresponding SKU may be promising candidates of further changes to the system. The additional information could increase the search speed by finding good action sets in a shorter period. The user may be able to set a value for the expected impact of an action type regarding its influence on the overall performance of the logistics network. Analyzing this parameter, the HU can evaluate whether or not it is promising to start a new simulation run directly after applying this action. Adding a transport relation between two places may have a lower impact on the performance of the modeled logistics network than shifting hundreds of SKUs from several sites to a central warehouse. Thus, it might be better to wait for additional actions to be applied before running a simulation experiment to evaluate the effects of the adjustments. Simulation runs are very time-consuming compared with other processes of the DSS. Using the impact factor of action types to reduce the total amount of simulation runs may have a big influence on the performance of the DSS. Further domain-specific parameters may be constraints, restricting the execution of action types, e.g., when adding an SKU to a site a suitable supplier for the SKU must be available. Also, the SKU may require certain storage techniques, such as a lifting platform or a hall crane. The modeling language has to offer constructs to model this domain- specific information.

Besides common operations, such as conditional constructs or loops, the modeling language has to provide domain-specific modeling constructs to describe actions for logistics networks. Such constructs may be used to model the relations between an action type and the integrated data structure. Additionally, filters may be used to apply changes to one or more objects. An action type for adjusting the minimum stock of an assortment may use a filter to extract corresponding SKUs. A filter models the relation between an action type and the related objects, e.g., SKUs, sites, transport relations or suppliers. Executing such an action type results in changes to one or more related objects.

A complex action type, such as shifting an assortment to a central warehouse, may consist of several sub-action types, e.g., adding or removing transport relations, adjusting the replenishment strategy for corresponding SKUs, changing the minimum order quantity or stock quantity. In order to model hierarchical relations between action types, the modeling language must provide adequate constructs to concatenate action types. To ease this process, the editor is connected to the action type directory. When accessing the action type directory, the user will be provided with a list of existing action types and detailed information describing these action types, e.g., required parameters, the relations to the data structure or the description. To model complex action types, the user can access and combine existing action types.

Instantiating an action type with parameter values leads to an action. When instantiating complex action types, many parameters for each sub-action type might be required. To ease this process, related parameters will be forwarded to the corresponding components of the action type for parameterization. Thus, the amount of required input might be reduced, e.g., when shifting an assortment to a specific central warehouse X, additional transport relations towards site X might be needed, which can be automatically predefined.

Actions can be applied by the execution engine. An interpreter analyzes the action to identify all changes to the reference model that are represented by the action. By using the mapping model changes to the reference model can be transformed into changes to the underlying database. Subsequently, these changes are applied to manipulate the data of the underlying database.

5 REQUIREMENTS FOR THE PROPOSED DSML

To ensure the usability of the proposed approach for creating, integrating and applying possible changes in DES models, the authors propose to use a DSML. In order to provide a feasible and accessible way of modeling changes, the modeling language needs to fulfill certain requirements.

First, the DSML has to be intuitive and easy to utilize for non-technical users. Reducing the constructs to a plain set and focusing on the particular problem domain, the overall complexity of the modeling language can be reduced. These constructs may only describe domain relevant aspects of the modeled system. Thus, the learnability can be increased. Furthermore, the maintenance and possible extension of the DSML must be ensured.

The authors propose to use an external domain-specific language for modeling action types in order to reduce the complexity of the modeling process. By choosing an external DSL, the modeling language can have its own language and syntax instead of using the language of the main application. This increases the domain-specific aspect of the modeling language and the overall flexibility of the approach. Using an external DSL requires the derived actions to be transformed into the applications language, e.g., into SQL-Statements. However, when changing the language of the underlying database, only the process of transforming actions into changes to the database needs to be adjusted, whereas the definition of the action types may stay the same. Combining this aspect with a database abstraction layer, the underlying database can be exchanged without adjusting the implementation of the modeled action types.

An additional requirement for the DSML is an integrated interface to the underlying data structure. Accessing the data structure through constructs of the modeling language may ease the matching between actions for the logistics network and changes to the database. To describe changes to DES models effectively, the modeling language should contain general constructs, such as conditional constructs and loops, as well as domain-specific constructs, e.g., filter or relevant constraints. Typically, actions in the modeled system effect one or more components of the logistics network. Thus, the modeling language may provide a mechanism to describe generic data objects representing these components. Also, the user should be able to model required input data for specifying the generic objects, e.g., when adding a specific transport relation the starting and the destination of the relation must be specified by concrete values. To create complex action types the DSML has to provide constructs to model the hierarchical structure of these action types, a possibility for concatenating existing action types.

Eventually, the DSML should provide an easy-to-learn approach of creating new action types.

6 CONCLUSION AND OUTLOOK

Using the presented approach, the authors aim to develop a solution approach for modeling changes in DES models to describe actions in logistics networks. Therefore, the architecture of an existing DSS for logistics networks was extended with components for creating, storing and applying these changes. The complexity of creating possible actions is very high because complex actions can easily lead to thousands of changes to the underlying database. The authors introduced a concept to integrate a DSML to ease the process of creating new action types. Thus, the proposed modeling language serves as an abstraction layer. Furthermore, the authors identified constructs of the DSML in addition to a set of requirements for the language to fulfill the demands of simplifying this process. The authors laid the foundation for further research by presenting a generic concept of modeling, integrating and executing actions types in DES models for logistics networks.

The next steps regarding the presented research are the development and evaluation of the proposed DSML. Beyond this, on a larger scope, further research regarding DSML to describe changes in data models seem to be very interesting. Additionally, the integration and use of domain-specific information to improve the search for good action sets in the simheuristic framework seem promising.

REFERENCES

Bauer, A., and H. Günzel. 2013. *Data-Warehouse-Systeme*: *Architektur, Entwicklung, Anwendung.* 4th ed. Heidelberg: dpunkt.

Bentley, J. 1986. “Programming pearls”. *Commun. ACM* 29/8:711–721.

Biesen, M., A. Tavakoli, and T. Hegmanns 2013. “Managing Logistics Performance of Numerous Facilities under Consideration of Facility-individual Preconditions”. In *Proceedings of the 6th International Colloquium on Business & Management* edited by C. Jones. Bankok, Thailand.

Blutner, D., S. Cramer, S. Krause et al.. 2007. “Assistenzsysteme für die Entscheidungsunterstützung”. Technical Report No. 06009, Sonderforschungsbereich 559, Dortmund.

Bockholt, F., W. Raabe and M. Toth. 2011. “Logistic assistance systems for collaborative supply chain planning”, *International Journal of Simulation and Process Modelling* 6/4:297–307.

Bruns, C., and T. Hegmanns. 2013. “Performance Measurement System for a Changeable Stocking Strategy in Distribution Systems of Commercial Enterprises”. In *Proceedings of the 6th International Colloquium on Business & Management,* edited by C. Jones. Bankok, Thailand.

Deiseroth, J., D. Weibels, and M. Toth. 2008. “Simulation-based Decision Support System for the Disposition of Global Supply Chains”. In *Advances in Simulation for Production and Logistics Applications*. *Proceedings of the 13th ASIM Conference on Simulation in Production and Logistics,* edited by M. Rabe, 41–50. Stuttgart, Germany: Fraunhofer IRB.

Dross, F., and M. Rabe. 2014. “A SimHeuristic Framework as a Decision Support System for Large Logistics Networks With Complex KPIs”. In *Proceedings of the 22nd Symposium Simulationstechnik (ASIM 2014)* edited by J. Wittmann and C. Deatcu. Vienna, Austria: ARGESIM / ASIM.

e-SAVE. 2015. e-SAVE. Accessed on June 30. http://www.e-save.eu/.

Fowler, M., and R. Parsons. 2011. *Domain-specific languages.* Upper Saddle River, New Jersey: Addison-Wesley.

Gladen, W. 2014. *Performance Measurement*: *Controlling mit Kennzahlen.* 6th ed. Wiesbaden: Springer Gabler.

Gleich, R., and A. Quitt. 2011. *Performance Measurement*: *Konzepte, Fallstudien und Grundschema für die Praxis.* 2nd ed. München: Vahlen.

Grönniger, H., J. O. Ringert, and B. Rumpe. 2009. “System Model-Based Definition of Modeling Language Semantics”. In *Formal techniques for distributed systems*. P*roceedings of the Joint 11th IFIP WG 6.1 International Conference FMOODS 2009 and 29th IFIP WG 6.1 International Conference FORTE 2009,* edited by D. Lee, A. Lopes, and A. Poetzsch-Heffter, 152–166. Berlin: Springer.

Gutenschwager, K., and K. Alicke. 2004. “Supply Chain Simulation mit ICON-SimChain”. In *Logistik Management,* edited by T. Spengler, S. Voß and H. Kopfer, 161–178. Heidelberg, Germany: Physica.

Harel, D., and B. Rumpe. 2004. “Meaningful modeling. What's the semantics of "semantics"?”. *Computer*, 37/10:64–72.

Juan, A. A., and M. Rabe. 2013. “Combining Simulation with Heuristics to Solve Stochastic Routing and Scheduling Problems”. In *Simulation in Produktion und Logistik 2013*. *Proceedings of the 15th ASIM Conference on Simulation in Production and Logistics,* edited by W. Dangelmaier, C. Laroque, and A. Klaas, 641–649. Paderborn, Germany: Heinz-Nixdorf-Institut.

Kengpol, A. 2008. “Design of a decision support system to evaluate logistics distribution network in Greater Mekong Subregion Countries” *International Journal of Production Economics*, 115/2:388–399.

Kuhn, A., B. Hellingrath, and H. Hinrichs. 2008. “Logistische Assistenzsysteme”. In *Software in der Logistik*. *Weltweit sichere Supply Chains,* 20-26. München, Germany:Huss.

Liebler, K., U. Beissert, M.Motta et al.. 2013. “Introduction OTD-Net and LAS: Order-to-delivery network simulation and decision support systems in complex production and logistics networks”. In *Proceedings of the 2013 Winter Simulation Conference,* edited by R. Pasupathy, S.-H. Kim, A. Tolk et al, 439-451. Piscataway, New Jersey: IEEE Press.

März, L., W. Krug, O. Rose et al.. 2011. *Simulation und Optimierung in Produktion und Logistik*: *Praxisorientierter Leitfaden mit Fallbeispielen.* Heidelberg: Springer.

Object Management Group. 2015. “Unified Modeling Language Specification”. Accessed on June 30. http://www.omg.org/spec/UML/2.5/PDF/.

Rabe, M., and F. Dross. 2015. “A Reinforcement Learning Approach for a Decision Support System for Logistics Networks”. In *Proceedings of the 2015 Winter Simulation Conference,* edited by L. Yilmaz, W. K. V. Chan, I. Moon et al., 2020–2032. Piscataway, New Jersey: IEEE Press.

Rabe, M., F. Dross, and A. Vennemann. 2015. “A Procedure Model for the Credible Measurability of Data Warehouse Metrics on Discrete-event Simulation Models of Logistics Systems”. In *Simulation in Produktion und Logistik 2015. Proceedings of the 16th ASIM Conference on Simulation in Production and Logistics,* edited by M. Rabe and U. Clausen, 167-176. Stuttgart, Germany: Fraunhofer Verlag.

Rabe, M., and F. Dross. 2016. “Intelligente Entscheidungsunterstützung für Logistiknetzwerke des Werkstoffhandels”. *Werkstoffe in der Fertigung*:24–25.

Rabe, M. et al.. 2017a. “Decision Support for Logistics Networks in Materials Trading Using a Simheuristic Framework and User-generated Action Types”. In *Proceedings of the 17th ASIM Conference on Simulation in Production and Logistics.* To Be Published.

Rabe, M., F. Dross, and A. Wuttke. 2017b. “Combining a Discrete-event Simulation Model of a Logistics Network with Deep Reinforcement Learning”. In *Proceedings of the 12th Metaheuristics International Conference,* edited by A. Duarte, A. A. Juan, and H. R. Lourenço. To Be Published.

Rabe, M., K. Gutenschwager, T. Fechteler, et al.. 2013. “A Data Model for Carbon Footprint Simulation in Consumer Goods Supply Chains”. In *Proceedings of the 2013 Winter Simulation Conference,* edited by R. Pasupathy, S.-H. Kim, A. Tolk et al., 2677–2688. Piscataway, New Jersey: IEEE Press.

Shim, J. P., M. Warkentin, J. F. Courtney, et al.. 2002. “Past, present, and future of decision support technology”. *Decision Support Systems* 33/2:111–126.

Siemens PLM Software. 2017. “Tecnomatix Plant Simulation”. Accessed on June 30. http://www.plm.automation.siemens.com/de_de/products/tecnomatix/plant_design/plant_simulation.shtml.

SimPlan AG. 2017. “SimChain”. Accessed on June 30. http://www.simchain.net/.

van Deursen, A., P. Klint, and J. Visser. 2000. “Domain-specific languages”. *SIGPLAN Not.* 35/6:26–36.

AUTHOR BIOGRAPHIES

MARKUS RABE is full professor for IT in Production and Logistics at the Technical University Dortmund. Until 2010 he had been with Fraunhofer IPK in Berlin as head of the corporate logistics and processes department, head of the central IT department, and a member of the institute direction circle. His research focus is on information systems for supply chains, production planning, and simulation. Markus Rabe is vice chair of the “Simulation in Production and Logistics” group of the simulation society ASIM, member of the editorial board of the Journal of Simulation, member of several conference program committees, has chaired the ASIM SPL conference in 1998, 2000, 2004, 2008, and 2015, and was local chair of the WSC’2012 in Berlin. More than 180 publications and editions report from his work. His e-mail address is markus.rabe@tu-dortmund.de.

DOMINIK SCHMITT works as a researcher at the Graduate School of Logistics at the Technical University Dortmund. He holds a diploma in Computer Science from the Technical University Dortmund. He graduated with a diploma thesis on the development of a computer-based model for the representation of transformable production systems for scheduling algorithms. Since 2016 he focuses his research on how to make the modeling of changes in discrete-event simulation models more accessible for non-simulation experts. His e-mail address is dominik.schmitt@tu-dortmund.de.

FELIX DROSS works as a researcher at the Graduate School of Logistics at the Technical University Dortmund. He holds a B.Sc. in Applied Computer Science from the Ruhr-University Bochum and an M.Sc. in Software Engineering from the University of Augsburg, Technical University Munich and Ludwig-Maximilians-University Munich. He graduated with a master thesis on a hybrid heuristic approach for solving the inventory routing problem. Since 2013 his research is focused on simulation-based optimization techniques for logistics networks. His e-mail address is felix.dross@tu-dortmund.de.