A Concept for Simulation of Autonomous Logistic Processes

Bernd Scholz-Reiter, Daniel Rippel, and Steffen Sowade

Abstract—Today, logistic systems face increasing dynamics and complexity. Autonomous Control faces these challenges, by enabling logistic objects to render their own local decisions. To apply autonomous control to real world applications, it is necessary to model and test autonomous processes before implementing them. The Autonomous Logistic Engineering Methodology provides tools to develop autonomous processes. In order to support testing and validation of these models, the methodology is extended by a simulation component. This article presents a concept, to transform the process models into executable simulation models. This procedure uses concepts and techniques of the Model Driven Architecture. Furthermore, the article presents a procedure that supports in selecting suitable simulation platforms and in integrating them into the methodology's framework.

Keywords—Autonomous Processes, Modeling, Model Driven Architecture, Model Transformation, Simulation

I. Introduction

LOGISTIC systems face growing complexity and the influence of an increasingly dynamic environment. One strategy to cope with this development is the application of autonomous control, as it decentralized decision competencies and therefore reduces the complexity of each local decision. The Collaborative Research Center 637 (CRC 637) investigates the advantages and restrictions of autonomous control in logistic systems. In the course of applying autonomous control to real world systems, it is necessary to model and simulate autonomous logistic processes, as well as to evaluate their performance and feasibility before implementing them.

The Autonomous Logistic Engineering Methodology (ALEM) assists logistic experts in modeling autonomously controlled logistic systems [1]. To support an evaluation of the models, ALEM is currently extended by a simulation component. Due to the structure of the ALEM models, they cannot be executed directly within a simulation platform. Consequently, the models have to be preprocessed to enable simulation.

This article presents a concept to transform ALEM models into arbitrary, executable simulation models. It adapts several elements of the Model Driven Architecture (MDA) [2] to achieve this goal, and enables the creation of simulation models out of a provided ALEM model. As the transformation can target different simulation platforms, a procedure to select suitable simulations is sketched. The article shortly introduces autonomous control, the ALEM framework, and the advantages of simulation in general. Then, it proposes the MDA-based approach to transform ALEM models into executable simulation models. Finally it presents the procedure to select and analyze suitable simulation platforms.

A. Autonomous Control

Various scientific disciplines, like physics, biology, artificial intelligence, control theory, and the engineering sciences, use the term *autonomous control* [3]. In the context of logistic systems, Hülsmann and Windt define autonomous control as "processes of decentralized decision-making in heterarchical structures. It presumes interacting elements in non-deterministic systems, which possess the capability and possibility to render decisions independently. The objective of Autonomous Control is the achievement of increased robustness and positive emergence of the total system due to distributed and flexible coping with dynamics and complexity" [4].

Up to now, different decision-making strategies have been developed for manufacturing systems as well as for logistic transport scenarios. Although it is impossible to predict the overall system's behavior, simulation studies, applying the different decision strategies, demonstrate the positive effects of autonomous control on the system's performance, flexibility and robustness (see for example [5], [6], [7], [8], [9], [10]).

The application of autonomous control in manufacturing systems delegates planning capabilities to commodities. Instead of one global master plan, the commodities proceed through production, based on their own local decisions. For example, once they enter a shop floor, they autonomously request manufacturing from suitable machines or

Manuscript received December 29, 2010. This research is funded by the German Research Foundation (DFG) as part of the Subproject B2 of the Collaborative Research Center 637 "Autonomous Cooperation Logistic Processes – A Paradigm Shift and its Limitations" (CRC 637).

B. Scholz-Reiter is with the BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen, Hochschulring 20, 28359 Bremen, Germany.

D. Rippel is with the BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen, Hochschulring 20, 28359 Bremen, Germany (phone: +49 421 218-9793; fax: +49 421 218-5640; mail: rip@biba.uni-bremen.de).

St. Sowade is with the BIBA – Bremer Institut für Produktion und Logistik GmbH at the University of Bremen, Hochschulring 20, 28359 Bremen, Germany.

workstations. The commodities use objectives to select the most preferable resource. For example, objectives can demand that the commodity proceeds through manufacturing quickly or that it selects those resources with minimum costs.

In case of a malfunction, the commodities react dynamically. Once they are aware of the situation, they request manufacturing from another machine with similar characteristics. With regard to their product structure, they can shift the sequence of manufacturing steps. This allows postponing problematic production steps and helps resolving bottleneck situations [11].

To enable autonomous control in manufacturing systems, the involved logistic objects have to be equipped with the necessary logical and technological infrastructure. On the technological side, the logistic objects have to be able to perform communication, data storage, data processing, and decision execution [12]. On the logical side, a suitable decision-making strategy has to be selected and applied to the logistic system. As it is impossible to predict the overall system's behavior, the selections have to be validated and compared to different alternatives. Therefore, simulation provides a tool to experiment with different setups. The next section shortly introduces simulations and their advantages.

B. Simulation in Logistics

According to the VDI Guideline 3633 sheet 1, simulation resembles the process of replicating a system in form of a model. The simulation model covers the system's dynamic behavior. It is used to draw experimental conclusions that can be carried over to the real world [13]. Following this definition, simulation studies allow examining a system, apart from its real world counterpart.

There are two main areas of application. First, simulations assess the impact of modifications to an existing system, for example while upgrading an existing system to make use of autonomous control. Second, simulations evaluate the feasibility of a newly designed logistic system prior to its implementation. In both cases, a simulation study provides insight into the systems behavior and performance. In particular, during the design process, simulation supports the identification of errors in the modeled processes and prevents these from being implemented in the real world system. Due to the comparably low costs of modifying a simulation model, simulations allow comparing different autonomous decision-making strategies and configurations, with the aim of identifying the best settings for one particular logistic system.

A simulation consists of three main components: the simulation platform, the simulation model, and of a set of experiments [13]:

- The simulation platform defines a framework for the simulation and is able to execute the simulation model.
- A simulation model describes a scenario, using the notation provided by the platform. A simulation model usually represents the real world system.
- An experiment describes one certain situation within the system. While the simulation model defines the scenario itself, an experiment defines one definite situation.

Some simulation platforms omit the distinction between simulation models and experiments. These platforms require modeling of the actual systems state in the simulation model itself. They treat different states as distinct models [13].

There exist several simulation technologies, for example material-flow simulation, process-based simulation, multi-agent simulations, or mathematical simulations. Those simulation technologies differ in the selection and focus of simulated elements. For example, material-flow simulations focus on materials, related resources and physical material flows [14], while process-based simulations use activities as primitive simulation elements and focus on their logical and temporal dependencies [15].

In the context of autonomous control, multi-agent simulations (MAS) provide suitable means to simulate the logistic systems. MAS focus on the system's objects and their interactions. They are used to represent and analyze systems that are made up from interacting and communicating entities [16]. The autonomy of intelligent logistic objects and agents constitutes another conceptual similarity between MAS and autonomously controlled systems. Scholz-Reiter et. al. pointed out, that agents are one option to interpret intelligent logistic objects [3]. Due to the high degree of freedom, concerning the implementation of agents, a MAS was selected to simulate the ALEM models.

II. Autonomous Logistic Engineering Methodology

The Autonomous Logistic Engineering Methodology (ALEM) is developed within the CRC 637. It provides tools and methods to develop models of autonomously controlled systems. It offers a notational concept, a view concept, and a procedure to model autonomous systems. The methodology relies on decisions about the desired system infrastructure [17] and the system's architecture [18]. Additionally, ALEM provides a software tool (ALEM-T) which supports the creation, simulation, and evaluation of the model. Fig. 1 depicts the framework's structure.

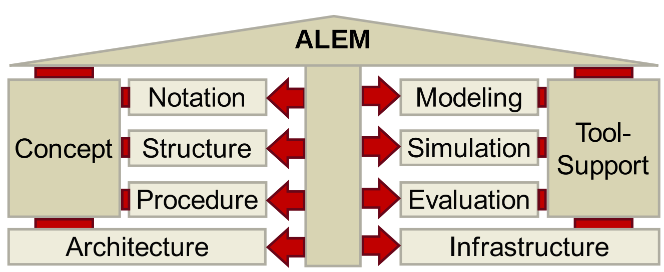

Fig. 1 The ALEM Framework [19]

ALEM's notation bases on the Unified Modeling Language (UML) and extends it by several elements and diagrams specific for this domain of autonomous logistic processes. For example, knowledge maps, a layout diagram, and product structure diagrams have been added [11], [20]. UML provides a well known notation which is widely used in business process modeling (see for example [21], [22]).

Process- and system-models are usually associated with a high degree of complexity [23]. Hence, ALEM applies a view concept (Fig. 2) [24]. Views focus on single aspects of the overall system. They enable editing of lesser complex segments of the model [25].

ALEM's view concept uses five primary views to divide the model into single, semantic aspects. These views are grouped further in static (structure, abilities and knowledge) and dynamic aspects (processes and communication protocols). While static aspects describe unchanging features of the model, dynamic aspects subsume procedures performed by the logistic objects. In addition, the contents of the semantic views are further differentiated into micro aspects, concerning object internal model elements, and macro aspects, which describe for example the overall systems structure.

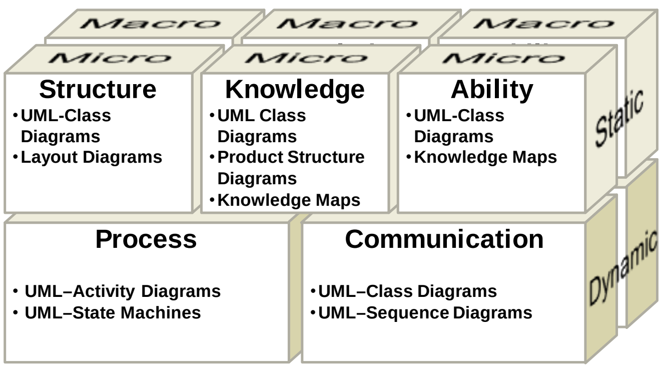

Fig. 2 ALEM View Concept [24]

The semantic views differentiate between the system's structure, knowledge, abilities, processes, and communication. Each view uses multiple diagrams to depict a certain aspect.

The structure view contains the structural features of the system. It defines all logistic objects present in the system and the relationships between them. In addition to the definition, this view includes the spatial layout of the modeled scenario. This semantic view is a static view, primarily containing macro aspects.

The knowledge view covers all aspects concerning knowledge, and the objectives. UML-Class diagrams are used to represent the logistic object's knowledge in form of attributes. In addition, it uses more specialized diagrams, like product structure diagrams and knowledge maps. This semantic view is a part of the static view and mainly contains micro aspects.

The ability view uses a UML-Class diagram to represent abilities, which can be performed by the logistic objects. It applies knowledge maps to assign abilities to specific logistic objects. This semantic view belongs to the static view and covers micro as well as macro aspects.

The process view uses UML-State Machines and UML-Activity diagrams to describe the behavior of logistic objects. It is a part of the dynamic view and incorporates micro and macro aspects.

The communication view contains UML-Class and UML-Sequence diagrams. The class diagram defines messages exchanged by logistic objects, while sequence diagrams represent communication protocols. This view is dynamic and mainly contains macro aspects.

A tool for modeling autonomous logistic systems was proposed as a part of the ALEM framework [26]. Fig. 3 presents a screenshot of the tool and highlights the most important areas. On the left, it displays the model explorer and the model overview. The explorer provides access to different models, while the overview shows the different diagrams of one particular model. These are ordered in accordance to the view concept. The overview allows to create and open the different views' diagrams. In the center, there is the graphical diagram editor, having the drawing palette on its right side and the property sheet at the bottom. The property sheet provides editing capabilities for a selected element's properties like a class' name or an attribute's type. To the right, there is a dynamic view, which gives access to inter diagram relationships. According to the currently edited diagram, it provides different functionalities. For example, while editing the structure view's class diagram, it enables the assignment and creation of life cycles (process view) for logistic objects [10].

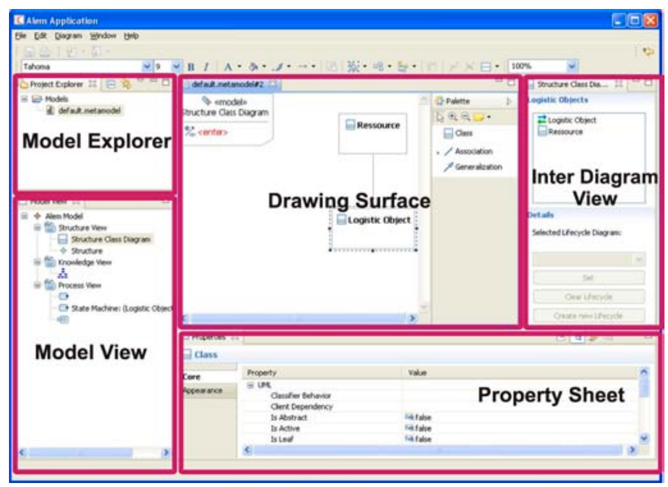

Fig. 3 Screenshot of the ALEM-Tool

The ALEM-Tool is implemented as a set of plug-ins within the Eclipse Rich Client Platform (RCP). The ALEM-Tool relies on several open source frameworks, like the Eclipse Modeling Framework (EMF) to realize the ALEM models [27], [28]. Additionally, EMF-based implementations of the Unified Modeling Language are used to cover default diagrams. Graphical editors were generated for all diagrams using the Eclipse Graphical Modeling Framework [29].

By linking the tool to an existing simulation platform, it will be possible to validate and to iteratively enhance the models. The goal is to enable a user of ALEM-T to directly execute the models within or from the application.

III. Simulation of ALEM Models

ALEM models use a variety of standard diagrams. According to the ALEM view concept, several types of diagrams apply in different contexts. For example, UML-Class diagrams depict the systems structure, the logistic objects' knowledge, as well as their abilities. Therefore, the semantic meaning of syntactically equal elements differs. To reflect the meaning of an element, the model's structure closely conforms to the ALEM view concept. Structural elements are stored in one part of the diagram, while knowledge related elements are stored in another segment. In contrast, simulation models focus on the objects or the processes. They store all information regarding one entity (e.g. agent, object, activity) at the entity itself. Consequently, the syntactic and semantic structure of ALEM models differs from simulation models. For this reason, ALEM models have to be preprocessed and transformed to be executable within a simulation platform.

A. Model Driven Architecture

This section proposes a general transformation procedure, based on concepts from the Model Driven Architecture (MDA) [2], to transform ALEM models into models of an arbitrary simulation platform. The procedure takes an ALEM model as input and creates an executable simulation model for the selected target platform. For each target simulation platform, a RCP plug-in will be implemented which creates the necessary models and files.

An MDA-based approach was selected, as MDA proposes the paradigm to implement programs apart from platform specific requirements as models. In this process, MDA applies transformations to specialize generic models to comply with a specific target structure, like source code or equal highly specific models. Mellor et al. [30] provide an overview over the MDA's basic concepts and the relationships between them. MDA's primitive types are models and meta-models. A model is an instance of a meta-model. If a meta-model describes elements specific to a certain platform, its implementations are called platform specific models (PSM). If the meta-model is more abstract, the models are called platform independent models (PIM). The structure of each formal modeling language can be expressed using a meta-model, describing which elements are allowed in which context.

B. Transformation Process

To enable simulation, ALEM models will be preprocessed and transformed on both the semantic and the syntactical level. Therefore, the transformation procedure covers three major steps: first, it semantically restructures information and thereby identifies ambiguous or missing information. Second, it obtains all information necessary to simulate the model and resolves ambiguities by instantiation. Finally, it refines the extended, restructured model into an executable simulation model (Fig. 4).

On the semantic level, the first transformation step collects and restructures information that is present in an ALEM model. The restructuring process can identify missing or ambiguous information and point these out to the user. Moreover, it converts semantic elements of ALEM into respective representations of the target platform. For example, the transformation matches an intelligent logistic object, represented as a class in ALEM, to the simulation model's representation of a transportation device. The second transformation step, the instantiation, acquires missing information from the user. By instantiating the simulation elements, the user creates the simulation model, including one particular simulation experiment. He assigns initial values and setups to the simulation elements. On the syntactical level, the third transformation step translates between different model formats, like EMF, XML-Schemes, modeling languages or program code.

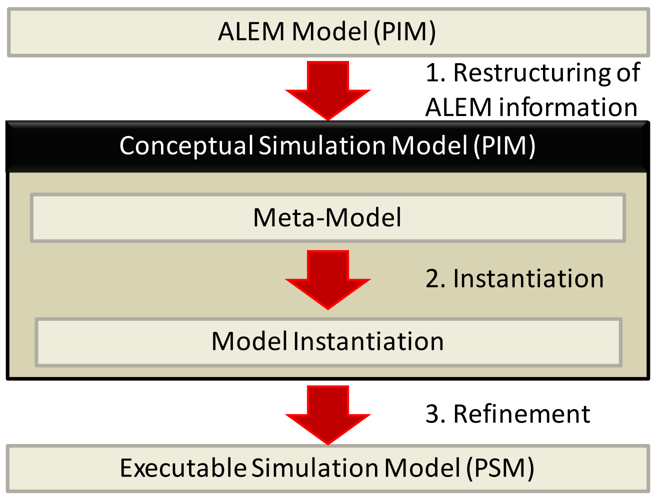

Fig. 4 Model Transformation Concept [31]

The first two steps require the assistance of an intermediate model. This model conforms to the executable simulation model with respect to semantic aspects, but omits syntactic aspects. It operates on conceptual levels without regarding characteristics of the target simulation platform. It is called a conceptual simulation model (CSM). The first transformation step collects and restructures present information. Thereby, it creates a meta-model for the later insanitation (CSM-Meta-Model). By instantiating this meta-model, a human expert adds and embeds missing information into the CSM. Finally, the CSM's instance is refined to be executable on one particular simulation platform. This last step executes the syntactical conversion into platform specific languages.

Following the concepts of the MDA, the CSM, as well as the ALEM-Model itself, are considered to be platform independent. Although the CSM conforms to one particular simulation technology (e.g. MAS), it omits platform specific characteristics.

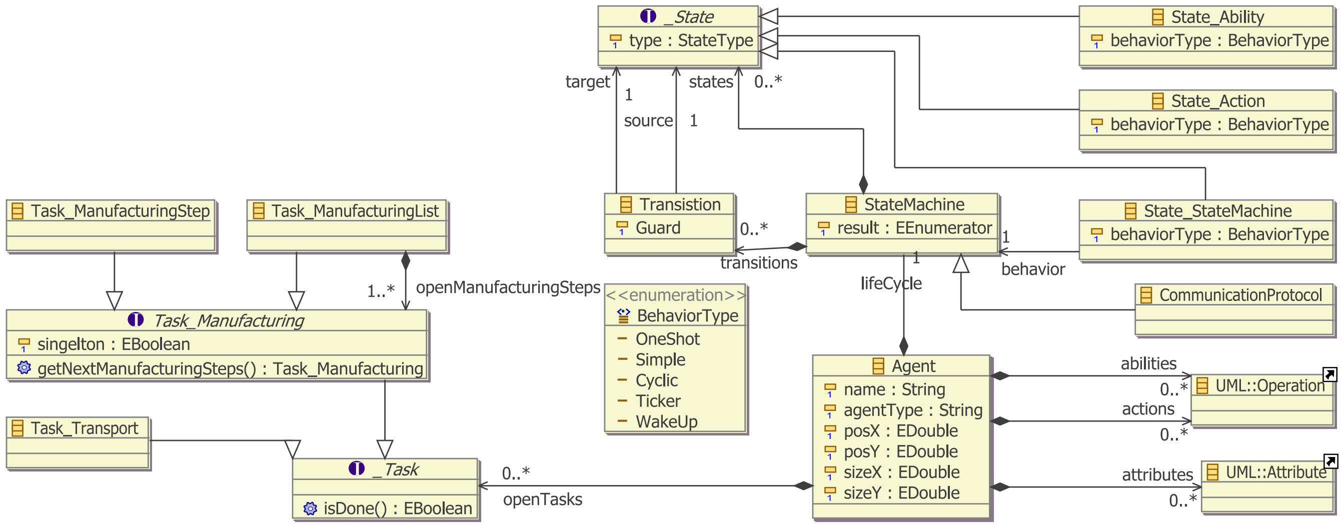

Fig. 5 Sample CSM 2nd level meta-model for a multi agent simulation (EMF/eCore Notation) [31]

To assure compatibility with the ALEM-Tool, the CSM models have to be implemented using EMF. Therefore, it is necessary to evaluate the semantic structure of the target simulation platform and to formalize a description of the CSM-Meta-Model's structure. Exemplary, Fig. 5 depicts an EMF description of the structure of a CSM-Meta-Model for a multi agent simulation platform. All CSM-Meta-Models derived by the first transformation step, comply with this structure. Therefore, this description is the CSM-Meta-Models' meta-model.

This CSM's main simulation elements are agents. Those consist of a set of attributes, different kinds of operations (actions and abilities), and a state machine, describing the agents' behaviors. Actions and abilities differ in their scope. Actions affect the simulation's world model. For example, actions describe movement or the loading or unloading of cargo. In contrast, abilities only affect the agent internal states, like the calculation of its objectives or the planning of a route. The state machines consist of states and conditional transition. Each state can either be a state machine on its own, or it refers to an ability or action. This structure enables reusability of the state machines. Tasks are default data types, which describe an agent's primary goals, like being manufactured or transported. Using this description of the CSM-Meta-Model's structure, the first transformation step can derive a valid CSM-Meta-Model from an ALEM-Model.

1) Restructuring

The first transformation step instantiates the aforementioned description of the CSM-Meta-Model's structure (e.g. Fig. 5), to create so called agent templates. These templates form the CSM-Meta-Model. Therefore, the step gathers information from the different ALEM diagrams and combines the information. All elements of the ALEM structure view's class diagram are converted either to agents or to data types, depending on the existence of an associated life cycle. In both cases, the transformation copies all attributes and operations, defined in the respective views, into the templates. The process view's UML-State-Machines and UML-Activity-Diagrams are transformed into the CSM's state machines and are associated to the respective agents. Therefore, the transformation introduces empty pseudo-states into the activity diagrams, to convert them into state machine.

2) Instantiation

The second transformation step is the instantiation of the CSM-Meta-Model. The user creates the simulation experiment by instantiating the agent templates. This includes the definition of the scenario's spatial layout as well as of the agents' initial attribute values.

To enable this task, ALEM's structure view includes a layout diagram. The corresponding editor is generated using the CSM-Meta-Model's structural description to handle arbitrary CSMs. It provides modified a palette and property sheets to access the agent templates instead of the generic agent type described in Fig. 5. As a result, the user can edit and spatially position the agents' instances.

3) Refinement

The refinement transformation step converts the scenario's formal EMF model (the CSM instance) into an executable simulation model. Depending on the target simulation platform, different technologies must be applied to perform this step. Target platforms can require models in a textual form (e.g. XML or source code) or in form of formal models like EMF or different internal model formats.

In case of textual models, the Eclipse Model-To-Text (M2T) project provides several script-based languages to convert EMF models into specified texts. For example, EMF itself uses the Java Emitter Templates (JET) to generate executable Java code out of its models [28].

The Eclipse Model-To-Model (M2M) project provides different standardized model transformation languages. All of these focus on EMF, which enables an efficient transformation within the ALEM context. Nevertheless, although the source models (instances of the CSM) are created using EMF, there is no guarantee that the target models conform to EMF. In this case, a direct transformation may be impossible or has to make use of import functions provided by the simulation platform (e.g. XML Import). This directly influences the selection of the target simulation platform, as appropriate formats or import functions must be available.

The proposed transformation procedure can be implemented for several target simulation platforms. Nevertheless, each platform requires the implementation of a suitable CSM, as well as an implementation of the required transformations.

Once the process is implemented, a majority of the transformation executes automatically. Commonly, the logistic expert, using ALEM, has to define the scenario/experiment (instantiation) as well as the simulation elements' basic functionalities (e.g. operations or abilities). The use of templates for common basic functions (e.g. default decision strategies, or operations like loading or unloading cargo) eases the instantiation for the logistic expert.

IV. Application of the Transformation Concept

The proposed transformation concept includes several steps, which have to be conducted or implemented, before a logistic process expert can make use of the simulation. These are the selection of a simulation platform, the implementation of the CSM's meta description, the design and implementation of the semantic transformation (first transformation step), and the design and implementation of default templates for the syntactic conversion. In addition, it may become necessary to adopt the ALEM – Layout editor to conform to the new CSM's structure. The majority of these steps require a detailed analysis of the simulation platform's models and structure. Therefore, this section proposes a procedure, which guides through these steps (Fig. 6).

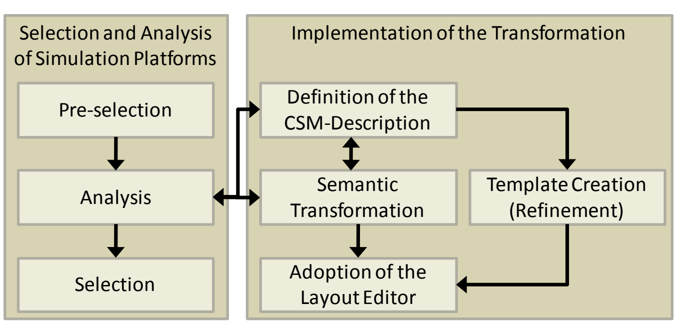

Fig. 6 Application Procedure

In order to apply the transformation concept, a suitable simulation platform has to be selected. It has to satisfy several criteria, described later on. The pre-selection should result in a small set of possible simulation platforms. With regard to the selected criteria, these may be equally capable to comply with a user's requests. Afterwards, each candidate platform is analyzed in more detail. Thereby, the simulation model's semantic structure, as well as its compliance to an ALEM-Model's structure is mapped. This mapping delivers primary inputs to the creation of the CSM's descriptor model as well as to the design of the semantic transformation step. The other way around, both development steps provide experiences, which influence the decision for a simulation platform. Some simulation models may not be as expressive as necessary, or the transformation may become to complex in contrast to equally capable simulation platforms. Using these experiences, one simulation platform is selected, and the in-depth development of the transformation steps as well as the adaption of the layout editor can be conducted. The following subsections focus on the process of selecting a suitable simulation platform. Thereby, specific requirements, originating from the use of the ALEM-Framework will be regarded. Additional constraints could occur, if specific functionality is requested by a user.

A. Pre-Selection

In 1993, the "Verein Deutscher Ingenieure" (VDI) released a catalogue of criteria, which specifically covers the selection of simulation platforms for logistic applications [32]. This catalogue provides several qualitative and quantitative criteria that refer to the application of software in general, to simulation platforms in general, as well as criteria that particularly target simulation of logistic systems. It covers criteria like the platforms software-license, number and type of allowed simulation elements, type of the simulation (discrete/continuous) or its application segment (logistics in general, material flow, plant layout, etc.). The application of a catalogue of criteria provides an overview about a simulation platform's main characteristics. Therefore, it is a suitable tool to compare possible simulation platforms. The application of such a catalogue results in a pre-selection of candidates that satisfy basic requirements, like costs, range of functions and area of application.

Although the catalogue proposed by the VDI covers several criteria, it does not include criteria that arise from the requirements of autonomous control or from the intended connection with the ALEM-Tool. These additional criteria deal with the compatibility of the tools, as well as of their models. Furthermore, the concepts of autonomous control require the simulation platform to provide a high degree of freedom, regarding the simulation entities and their abilities. Other criteria arise from ALEM's focus on process experts. These cannot be expected to be familiar with a simulation platform. Therefore, the coupling between the ALEM-Tool and the simulation platform should be automated as far as possible. In the following, some additional criteria are provided.

Criteria, referring to the compatibility of a simulation platform, concern with the accessibility of the simulation models and the coupling between the ALEM-Tool and the platform.

Accessibility of model files/import formats: One major criterion is the accessibility of the simulation platform's models. To enable the automatic creation, the platform is required to provide clear textual models, accessible data structures or at least import formats, which can be created by external applications. A typical example for textual models or an open import format is a XML file.

Integration: The coupling between the ALEM-Tool and the Simulation platform can be of different degrees. The most desirable form of integration would be a direct coupling. Thereby, the simulation model is created in memory and directly passed to the simulation platform for execution. This kind of coupling would require the platform to be implemented in Java and to provide a suitable interface to pass the model and to execute simulation. A more common kind of coupling is based on the creation of a simulation model file. Thereby, the ALEM-Tool creates the simulation model as a file, and executes the simulation platform. The platform is at least required to provide a command line client to run and execute the simulation model file, and to record simulation feedbacks into an ALEM-readable file. The most undesirable kind of coupling is an external simulation platform. In this case, ALEM would create the simulation model files. The user has to execute the simulation platform, load the file, configure simulation runs, execute the simulation and pass back the simulation results to ALEM.

Due to the application in the field of autonomous logistics, the simulation platform is required to satisfy additional functional requirements. These in particular deal with the execution of simulation runs, with the degree of freedom in creating simulation models, and the platform's ability to record key values for a later evaluation.

Evaluation: ALEM does not restrict a user in the design of the logistic system. This in particular refers to the distribution of abilities and knowledge amongst the intelligent logistic objects. Therefore, the measuring of key data (e.g. throughput-times or capacities) can take place at different logistic objects. A user could require the evaluation to record key values differently for distinct logistic objects. Consequently, the simulation platform should allow recording of various key data in different simulation entities. Optimally, this information could be specified as a part of the simulation model.

Degree of Freedom: ALEM allows a user to distribute abilities and knowledge freely amongst the intelligent logistic objects. The simulation platform has to reflect this degree of freedom. Several platforms provide a set of predefined simulation objects. In such a case, these predefined objects have to be modifiable to alter provided functionality, or to implement additional abilities.

Execution Modes: Different simulation modes increase the usability of a simulation platform. In case of testing a modeled behavior, the platform should provide a graphical interface that allows a step-by-step execution. In contrast, while assigning the logistic performance of a modeled system, the user might not require a graphical user interface. In this case, it is important that the simulation executes automatically and delivers its results quickly.

Execution Times: The execution times of the transformation as well as of the simulation provide additional criteria. Complex simulation models could prolong the transformation, while a slow simulation platform, or long startup times impede with a quick evaluation and with testing.

A third set of additional criteria originate from ALEM's focus on logistic process experts. These criteria deal with the use of information, and the mapping between ALEM-Models and the simulation platform's models. In particular, these focus on the use of already specified information, the necessity of providing additional information.

Utilization: ALEM-Models require logistic process experts to specify the logistic system with a high degree of detail. This enables a very detailed description of the desired processes, and the single objects' behaviors. A simulation platform should be able to exploit as much of the specified information as possible.

Additional Information: Depending on the simulation platform, it may become necessary to specify additional information. This task may become difficult for logistic process experts, if the additional information is very specific to a certain simulation platform.

Coherence of Definitions: ALEM makes use of default UML diagrams to define the intelligent logistic objects' abilities, knowledge and behavior. To enable simulation, the transformation converts these diagrams into simulation specific representations. One important criterion is the coherence between the different representations. As an example, a conversion between different state machines is an easy task compared to a conversion into a mathematical representation.

By applying the catalogue of criteria to different simulation platforms in question, a pre-selection is achieved. For each of the remaining candidates, a more detailed analysis will be conducted. While the pre-selection focused on functional requirements, the following analysis provides a deeper insight into the simulation models structure. Thereby, it helps to identify those simulation platforms, which facilitate the development of the model transformations.

B. Analysis

The analysis aims to investigate the simulation models in terms of their syntactic and semantic structure. Furthermore, it results in a preliminary mapping between the ALEM-Model's elements and those of the simulation model. The structural analysis' first step is the identification of primary simulation elements. These depend on the type of simulation (e.g. multi-agent, material-flow, etc.) as well as on the specific simulation platform. Some platforms may provide a simulation element "Machine", which has to be instantiated with a specific purpose, while others may provide a set of predefined machines. In a second step, the simulation platform's data structures will be identified. This includes predefined data structures, e.g. for machine schedules, as well as data structures which are used to manage a simulation element's behavior, state, or arbitrary information. The third step covers the identification of relations between these elements. Due to ALEM's focus on the single logistic objects behavior, the fourth step covers a detailed analysis of the simulation's behavioral definition. Using the identified elements, a formal definition of the simulation model's structure can be created. An example for formal description of a multi-agent simulation platform was already given in Fig. 5 on page 5.

Using the (preliminary) description of the simulation model's structure, a mapping between ALEM's simulation elements, and those of the simulation model can be developed. This mapping should be refined iteratively. In a first step, a semantic matching between top level simulation elements and the general contents of ALEM diagrams should be created. Fig. 7 presents a very rough sketch between the ALEM model components (e.g Classes in the structure view's class diagram, the knowledge views objectives or the process view's state machines and activity diagrams) and their respective counter-parts within a multi-agent simulation model. The data type of the respective simulation model element is given in brackets for a later syntactic conversion. By refining this mapping, a set of relations between ALEM model primitives, and those primitives, identified during the structural analysis is developed.

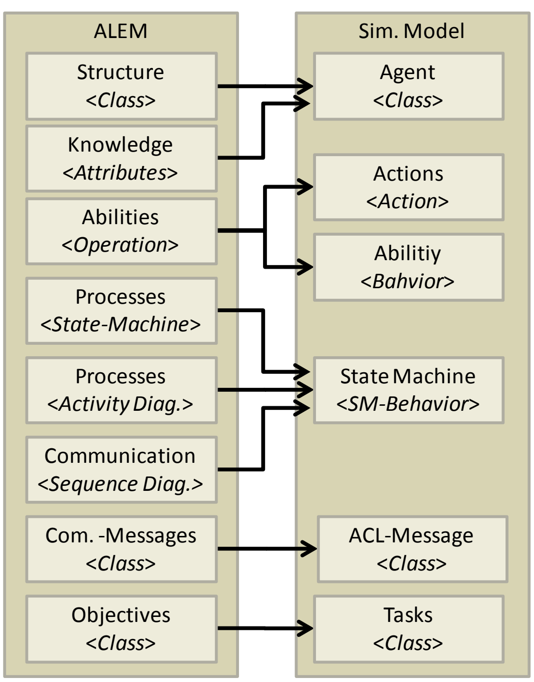

Fig. 7 Rough Mapping from ALEM to a Simulation Model

This mapping provides insight into the semantic transformation process. It supports the decision between candidate simulation platforms, as it enables an estimation of the necessary effort in implementing the transformation. Furthermore, the mappings and descriptions of different simulation platforms can be compared easily, and issues in transforming between ALEM and one of the candidates can be uncovered early. The structural analysis provides inputs for the development of the CSM's descriptor model and thus strongly facilitates this process later on. The information mapping provides relations between both types of models and therefore acts as a template for the development of the first, semantic transformation step. Additionally, while designing those relations, errors within the structural mapping can be identified before formalizing the descriptor model.

V. Conclusion and Future Work

Simulation provides a suitable technique to validate and test autonomous business processes. In particular, during the development of such processes, simulation supports an iterative enhancement of the modeled processes. As ALEM models cannot execute directly in an arbitrary simulation platform, ALEM will apply an MDA-based transformation process to convert its models into simulation models.

The possibility to simulate ALEM models will support the application of autonomous control in different ways. First, it will provide logistic experts with a tool to check the correctness and feasibility of the modeled autonomous processes. Second, simulation results can be compared with a real world logistic system to assess the benefits and drawbacks of an application of autonomous control to that system. Furthermore, the logistic expert can experiment with different autonomous setups to determine the most suitable alternative for his system.

The proposed transformation can be applied to different simulation platforms. Therefore, the article proposed a procedure to support in selecting a suitable simulation platform. In particular, the procedure focuses on the pre-selection of candidates, and covers an analysis of the simulation models semantic and syntactic structure.

As a next step, the transformation will be implemented exemplarily for a specific simulation platform. Thereby, a library of default abilities will be created, to ease the use of the transformation. Afterwards, the prototypical transformation will be tested to assess the limitations of ALEM-Models regarding their qualification to provide executable simulation models.

References

[1] B. Scholz-Reiter, J. Kolditz, and T. Hildebrandt, "Engineering autonomously controlled logistic systems," *International Journal of Production Research*, vol. 47, no. 6, pp. 1449–1468, 2009.
[2] *OMG Model Driven Architecture*, Object Management Group Std. 1, Rev. 1. [Online]. Available: http://www.omg.org/mda/
[3] B. Scholz-Reiter and H. Höhns, "Selbststeuerung logistischer Prozesse mit Agentensystemen," in *Produktionsplanung und -steuerung:*

Grundlagen, Gestaltung, Konzepte, G. Schuh, Ed. Berlin: Springer Verlag, 2006, pp. 745–780.

[4] M. Hülsmann and K. Windt, Eds., *Understanding of Autonomous Cooperation and Control in Logistics – The Impact of Autonomy on Management, Information, Communication and Material Flow*. Berlin: Springer Verlag, 2007.

[5] B. Scholz-Reiter, F. Boese, T. Jagalski, and K. Windt, "Selbststeuerung in der betrieblichen Praxis: Ein Framework zur Auswahl der passenden Selbststeuerungsstrategie," 2007.

[6] B. Scholz-Reiter, M. Teucke, S. Sowade, D. Rippel, M.-E. Özsahin, and T. Hildebrandt, "Applying autonomous control in apparel manufacturing," in *Advanced robotics, control and advanced manufacturing systems. Proceedings of the 9th WSEAS International Conference on Robotics, Control and Manufacturing Technology (ROCOM'09)*, C. Shengyoung and L. Qing, Eds. Hangzhou, China: WSEAS Press, 2009, p. 6.

[7] B. Scholz-Reiter, M. Görges, T. Jagalski, and A. Mehrsai, "Modelling and analysis of autonomously controlled production networks," in *Proceedings of the 13th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 09)*, Moscow, Russia, 2009, pp. 850–855.

[8] S. Dashkovskiy, F. Wirth, and T. Jagalski, "Autonomous control in shop floor logistics: Analytic models," in *Manufacturing, Modelling, Management and Control 2004*, G. Chryssolouris and D. Mourtzis, Eds. Amsterdam, NL: Elsevier Science Ltd, 2005.

[9] H. Rekersbrink, T. Makuschewitz, and B. Scholz-Reiter, "A distributed routing concept for vehicle routing problems," *Logistics Research*, vol. 1, no. 1, pp. 45–52, 2009.

[10] B. Scholz-Reiter, S. Sowade, D. Rippel, M. Teucke, M. Özsahin, and T. Hildebrandt, "A Contribution to the Application of Autonomous Control in Manufacturing," *NAUN International Journal of Computers*, vol. 3, no. 3, pp. 279–291, 2009.

[11] B. Scholz-Reiter, S. Sowade, T. Hildebrandt, and D. Rippel, "Modeling of orders in autonomously controlled logistic systems," *Production Engineering Research & Development*, vol. 4, no. 4, pp. 319–325, 2010.

[12] K. Windt, F. Böse, and T. Philipp, "Criteria and Application of Autonomous Cooperating Logistic Processes," in *Proceedings of the 3rd International Conference on Manufacturing Research . Advances in Manufacturing Technology and Management*, J. Gao, D. Baxter, and P. Sackett, Eds., 2005.

[13] *VDI-Richtlinie 3633 Blatt 1: Simulation von Logistik-, Materialfluß- und Produktionssystemen*, Verein Deutscher Ingenieure, Berlin: Beuth, 1993.

[14] A. Kuhn and M. Rabe, Eds., *Simulation in Produktion und Logistik*. Berlin: Springer, 1993.

[15] K. Tumay, "Business process simulation," in *WSC '96: Proceedings of the 28th conference on Winter simulation*. Washington, DC, USA: IEEE Computer Society, 1996, pp. 93–98.

[16] A. Karageorgos, N. Mehandjiev, G. Weichhart, and A. Hämmerle, "Agent-based optimisation of logistics and production planning," *Engineering Applications of Artificial Intelligence*, vol. 16, no. 4, pp. 335 – 348, 2003, intelligent Manufacturing.

[17] B. Scholz-Reiter, S. Sowade, and D. Rippel, "Modeling the infrastructure of autonomous logistic control systems," in *Advances in Communications, Computers, Systems, Circuits and Devices. European Conference of Systems (ECS'10)*, V. Mladenov, K. Psarris, N. Matorakis, A. Caballero, and G. Vachtsevanos, Eds. Tenerife: WSEAS Press, 2010, pp. 295–300.

[18] B. Scholz-Reiter and S. Sowade, "Der Beitrag der Selbststeuerung zur Wandlungsfähigkeit," in *Schriftenreihe der Hochschularbeitsgruppe für Arbeits- und Betriebsorganisation e.V. (HAB). Wandlungsfähige Produktionssysteme*. Berlin: Gito, 2010, pp. 303–322.

[19] B. Scholz-Reiter, D. Rippel, S. Sowade, and M. Teucke, "Selbststeuerung als Ansatz in der Praxis manuell getriebener Logistik," in *Deutscher Logistik Kongress*, 2009.

[20] B. Scholz-Reiter, J. Kolditz, and T. Hildebrandt, "UML as a Basis to Model Autonomous Production Systems," in *Digital Enterprise Technology: Perspectives and Future Challenges*, P. F. Cunha and P. Maropoulos, Eds. Berlin: Springer Verlag, 2007, pp. 553–560.

[21] D. Donko and S. Sabeta, "Specific Modeling of the Business Processes," in *Advances in Data Networks, Communications, Computers - 9th WSEAS International Conference on Data Networks, Communications, Computers (DNCOCO'10)*, N. Mastorakis and V. Mladenov, Eds. Faro, Portugal: WSEAS Press, 2010.

[22] T. Gardner, "UML Modelling of Automated Business Processes with a Mapping to BPEL4WS," in *Proceedings of the First European Workshop on Object Orientation and Web Services at ECOOP'03*, 2003.

[23] A.-W. Scheer, *Business Process Engineering - Reference Models for Industrial Enterprises*. Telos: Springer, 1994.

[24] B. Scholz-Reiter, H. Höhns, J. Kolditz, and T. Hildebrandt, "Autonomous Supply Net Coordination," in *Proceedings of 38th CIRP Manufacturing Systems Seminar*, Florianopolis, Brazil, 2005, CD-ROM, 7 pages.

[25] A.-W. Scheer, *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen*, 4th, Ed. Berlin: Springer Verlag, 2001.

[26] B. Scholz-Reiter, T. Hildebrandt, and J. Kolditz, "Modellierung selbststeuernder produktionslogistischer Prozesse - die Modellierungsmethode ALEM," in *Informations- und Kommunikationssysteme in SCM, Logistik und Transport. Teilkonferenz der Multikonferenz Wirtschaftsinformatik 2008*, D. Mattfeld, H.-O. Günther, L. Suhl, and S. Voß, Eds. Paderborn: Universität Paderborn, 2008, pp. 173–185.

[27] F. Budinsky, D. Steinberg, E. Merks, R. Ellersick, and T. Grose, *Eclipse Modeling Frame-work: a developer's guide*. Boston: Addison Wesley, 2003.

[28] Eclipse Foundation. (2009, September) Eclipse Modeling Framework (EMF). [Online]. Available: www.eclipse.org/emf/

[29] Eclipse Foundation. (2009, September) Eclipse Graphical Modeling Framework (GMF). [Online]. Available: www.eclipse.org/gmf/

[30] S. J. Mellor, K. Scott, A. Uhl, and D. Weise, "Model-Driven Architecture," in *Advances in Object-Oriented Information Systems*, ser. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Verlag, 2002, vol. 2426/2002, pp. 233–239.

[31] B. Scholz-Reiter, D. Rippel, and S. Sowade, "Modeling and simulation of autonomous logistic processes," in *Advances in Communications, Computers, Systems, Circuits and Devices. European Conference of Control (ECC'10)*, V. Mladenov, K. Psarris, N. Matorakis, A. Caballero, and G. Vachtsevanos, Eds. Tenerife: WSEAS Press, 2010, pp. 148–153.

[32] *VDI-Richtlinie 3633 Blatt 4: Auswahl von Simulationswerkzeugen - Leistungsumfang und Unterscheidungskriterien*, Verein Deutscher Ingenieure, Berlin: Beuth, 1993.

.

Bernd Scholz-Reiter heads the chair of Planning and Control of Production Systems in the Department of Manufacturing Engineering at the University of Bremen and is managing director of the BIBA - Bremer Institut für Produktion und Logistik GmbH.

Amongst others, he is Associate Member of International Academy for Production Engineering (C.I.R.P.), full member of the German Academy of Engineering Sciences, full member of the Berlin-Brandenburg Academy of Sciences, and member of the Scientific Society of Manufacturing Engineering. He is Vice President of the German Research Foundation and is the speaker of the Collaborative Research Centre 637 "Autonomous Cooperating Logistic Processes - A Paradigm Shift and its Limitations". He is editor of the professional journals Industry Management and PPS Management, and a member of editorial committees of several international journals.

Prof. Scholz-Reiter's research aims to develop and apply new concepts, methods and IT-solutions, which contributes to the sustainable improvement in production and logistics systems.

Daniel Rippel works as research scientist at the BIBA – Bremer Institut für Produktion und Logistik GmbH. His educational background is located in computer sciences, in particular in artificial intelligence and soft computing. Currently, his research interests are in the field of autonomous control, model transformation, and collaborative decision management in non hierarchical supply chains.

Steffen Sowade works as research scientist at the BIBA – Bremer Institut für Produktion und Logistik GmbH. His educational background is located in communication networks, system analysis and management of innovation. His current research interests are in the field of autonomous control, service-oriented architectures and protocols for manufacturing planning and control systems.