ALEM-T: A Modelling Tool for Autonomous Logistic Processes

B. Scholz-Reiter (2), T. Hildebrandt, J. Kolditz
Planning and Control of Production Systems, University of Bremen, Germany

Abstract

Autonomous control of logistic processes is proposed as a means for enterprises to better face dynamics and complexity, caused e.g. by globalization. This paper will first briefly sketch the idea of autonomous control of logistic processes and outline our concept for a modelling method designed for engineering of autonomous logistic processes. The main part will detail the implementation of this modelling concept in a software tool, presenting its architecture and underlying concepts.

1 INTRODUCTION

Autonomous control within the context of the interdiciplinary research effort CRC 637 in general means processes of decentralized decision making of interacting system elements in heterarchical structures. Concretised towards autonomous control of logistic processes it is defined as "[…] characterised by the ability of logistic objects to process information, to render and to execute decisions on their own" [1]. A logistic object fulfilling this definition is called an autonomous logistic object; to support its design implicates an approach focused on these objects. RFID (radio frequency identification)- and smart label-technologies and their successors in the foreseeable future are seen as an enabling technology to realize autonomous control of logistic processes.

The paper will first briefly sketch our concept of the modelling method ALEM (Autonomous Logistics Modelling Methodology) specifically tailored for logistics systems based on this principle. The modelling method is based on the unique integration of approaches from business process modelling, knowledge modelling and agent-oriented modelling. It supports requirements analysis by specification of the system, its entities and their interaction. Furthermore the modelling is part of an overall engineering methodology and therefore the models created are the basis for subsequent software implementation, yet the intended user of the method is a logistics expert (re-)designing a logistics system with only limited knowledge in computer science.

The main part of the paper details software support of this modelling concept in a tool (using e.g. the Eclipse Graphical Modelling Framework, GMF), presenting its architecture and underlying concepts. To create this modelling tool recent developments like model driven architecture (MDA; [2], [3]) and the usage of Eclipse [4] as a Rich Client Platform (RCP; [5]) are applied to the domain of logistics.

2 MODELLING AUTONOMOUS CONTROL

As already mentioned ALEM combines and integrates approaches from business process modelling, knowledge modelling and agent-oriented modelling to create a modelling method suitable for a logistic expert yet covering the specifics of autonomous control. Each of the proposed modelling methods within these fields covers certain required aspects for our intended use, but only their combination in our opinion offers the most promising solution (e.g. the concepts of software agents seems well suited to implement autonomous logistics processes but requires usually more computer science background than our intended user; business process modelling usually offers models well suited to be created by a domain expert but does usually not cover autonomous logistic entities).

To support the semi-formal specification of the autonomous logistic system for the logistics expert the modelling methodology as part of ALEM, with its components ALEM-N (ALEM-Notation), ALEM-P (ALEM-Procedure) and ALEM-T (ALEM-Tool), is proposed. ALEM-N consists of a view concept comprised of views each showing specific aspects of the logistic system as well as the notational elements to be used in each view and their intended meaning. ALEM-P is a procedure model describing the steps to be followed in generating a model and is intended to guide the user through analysis and specification of an autonomously controlled logistic system. Each step defines the activities to perform and its expected results. ALEM-T is a software tool supporting the notation and the procedure model, as described in section 3. Furthermore a reference model is also part of ALEM and offered by ALEM-T to ease the construction of a new model by reusing existing work. Because of space constraints of the paper, only the view concept as a fundamental part of ALEM-N and basic structuring instrument can be described further to give an overview of the method and the scope of modelling allowed by it. For a more in-depth discussion of ALEM-N and the modelling concept in general see [6]. The procedure model ALEM-P is sketched in [7].

3 VIEW CONCEPT FOR MODELLING

Creating process models usually leads to a high degree of complexity. A view concept serves as a means to reduce the complexity constructing a model [8]. The view concept distinguishing five different views, backed by a view-spanning meta-model. A fundamental distinction can be made between a static and dynamic (sub-)model. The static model describes the structure, the dynamic model the behaviour of the modelled system, following the basic distinction in UML (Unified Modelling Language, [9]) that is also appropriate here.

The *Structure View* showing the relevant logistic objects is the starting point. The basic elements for this view are UML class diagrams. Besides objects and classes the structure view can show relationships between them, for instance in the form of associations or inheritance relationships.

The *Knowledge View* describes the knowledge, which has to be present in the logistic objects to allow a decentralized decision making. This view focuses on composition and static distribution of the knowledge while not addressing temporal aspects. For this purpose UML class diagrams and Knowledge Maps are sufficient.

The *Ability View* depicts the abilities of the individual logistic objects. Processes of a logistic system need certain abilities, which have to be provided by the logistic objects. These abilities are supposed to be seen as abstractions of problem types and their solving capabilities occurring in reality.

The *Process View* depicts the logic-temporal sequence of activities and states of the logistic objects. Here the objects' decision processes can be modelled. The process view plays a central role connecting the views of the static model and depicting the behaviour of logistic objects, so far only viewed statically. The notation elements used for this are activity diagrams as well as state diagrams as they are also proposed in business process modelling using the UML [10].

The *Communication View* presents the contents and temporal sequence of information exchange between logistic objects. Depicting the communication is especially necessary to depict the interaction of autonomously deciding, otherwise only loosely coupled objects to model their interaction [11]. To display the communication, UML sequence diagrams showing the interacting partners, the messages and their temporal progression as well as class diagrams to display communication contents are used here.

4 TOOL SUPPORT FOR MODELLING AUTONOMOUS CONTROL: ALEM-T

As already stated earlier, a software tool is part of the modelling method. To achieve a high platform independence Java was chosen as a programming language. More specifically Eclipse is used as a so-called Rich-Client-Platform [5]. Originally being a Java-development environment, it supports a large variety of platforms and its modular architecture and extendibility by a very flexible plug-in-mechanism makes it possible to easily create customized applications, either by extending existing functionality or just using necessary parts of the functionality to create stand-alone applications. This makes it possible to offer multi-platform “rich” user interfaces (i.e. offering a user interface with much more possibilities e.g. a web-based application could offer), by using the relevant subset of Eclipse’s functionality. Furthermore there are two software-frameworks making Eclipse very suitable for creating our modelling tool. These are the Eclipse Modelling Framework (EMF, [12]) and the Graphical Modelling Framework (GMF, [13]). Both together allow an easy creation of a graphical editor as is required for a user interface in our application. Both frameworks offer code-generators to automatically create large portions of the later modelling application. EMF for instance offers a modelling language similar in expressivity to UML-class diagrams to allow the creation of meta-models (including XML exchange formats for them) for a specific application and creates large fractions of the required source code to handle models of such a meta-model.

Reasons for choosing Eclipse were: (a) Java-based; (b) Open-Source; (c) broad developer community; (d) cross-platform (currently supported platforms are Microsoft Windows, Linux, MacOS X, Sun Solaris, HP UX, IBM AIX), yet native look-and-feel of the user interface; (e) supports model-driven development and generative programming in the two frameworks EMF and GMF leading to increased development efficiency and easier to maintain applications; (f) EMF and GMF allow relatively easy development of graphical editors.

The application architecture of ALEM-T is shown in figure 4.1. The different modules of the application are implemented as Eclipse plug-ins, with its creation especially eased by the two software frameworks EMF and GMF and the ability of Eclipse to create RCP (rich client platform)-applications. RCP describes the possibility to create (largely) platform-independent, stand-alone applications with Eclipse that nevertheless have more possibilities to design graphical user interfaces (GUIs) than for instance web-based applications (as another way to create platform independent applications) can offer.

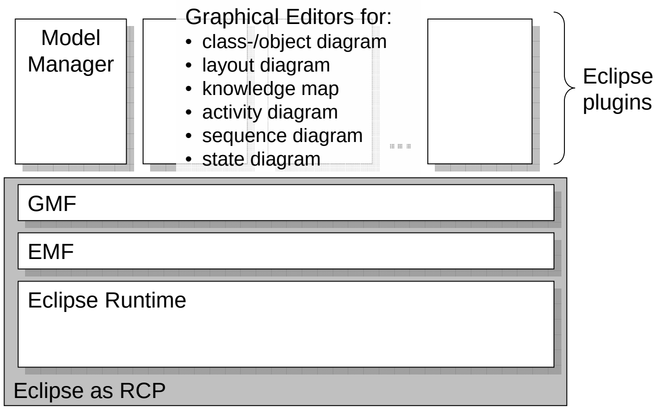

Figure 4.1. Architectural overview of the modelling application.

The two frameworks EMF and GMF of the Eclipse platform mentioned earlier allow an easy creation of graphical editors. Using code-generators the software developer is supported by the creation of a lot of standard source code that can later be customized to more specific needs. Based on a graphical definition of the meta-model of ALEM-N EMF allows the creation of source-code to represent model-instances of it in memory and to read/write such models in an XML-format. On top of this EMF model, GMF on the other hand allows to generate code for the basic functionality of graphical editors (e.g. for a class diagram editor).

Figure 4.2 shows the concept to divide the main application window of the GUI. Besides the standard elements like a tool and menu bar, an area to display messages to the user and a thumbnail view of the complete diagram currently opened a division of the into three parts can be recognized. In the main part a graphical editor for the currently opened diagram is displayed. The main drawing area shows this diagram. Using the “Palette” new diagram elements can be created. Model elements created using the graphical editor immediately appear in the “Diagram Explorer”. Besides the elements of the current diagram it shows further “matching” model elements, i.e. elements already defined in other diagrams or views but semantically and syntactically fitting into the current diagram. Using Drag&Drop these elements can be added to the current diagram, is deleted from “matching model elements” and added to the list of “elements in diagram”. Additionally in the upper part of the diagram explorer an input field with two adjacent buttons can be seen. These GUI-elements serve as a means to filter the model elements shown in the GUI-parts below if their name or a part of it is known. This allows more efficient work even with larger models.

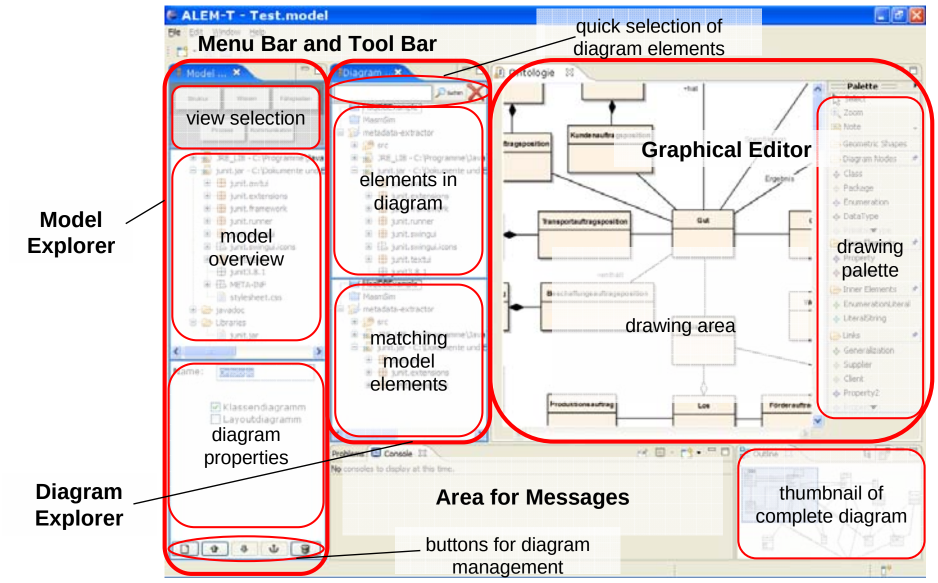

Figure 4.2. GUI-concept of the main application window.

Eventually the “Model Explorer” is divided into four parts. In the model overview a tree view of the diagrams existing in the model is shown, clearly structured by the views of the view concept. The “view selection”-part above it is used to confine the diagrams shown. Each view behaves like a button snapping in, confining the content shown to the diagrams of the respective view. Below the model overview the GUI-parts “diagram properties” and a toolbar with buttons to manage the diagrams can be found. They can be used to create new diagrams (the diagram types possible are sensitive to the active view of the view concept), to delete diagrams not used any more and change diagram properties like e.g. the name as a simple example.

5 CONCLUSION AND OUTLOOK

This paper addressed the topic of modelling autonomous logistic processes focussing on the tool support for the modelling method of ALEM after motivating and sketching the overall modelling concept. The implementation of the software tool ALEM-T supporting the notation and procedure model is currently ongoing. With its help a process expert (e.g. a logistics expert with only little background in computer science) will be supported in modelling and designing autonomous logistic processes. Our plans for the near future are to validate methodology and tool on real production logistics systems.

Acknowledgement

This research is funded by the German Research Foundation (DFG) as part of the Collaborative Research Centre 637 *Autonomous Cooperating Logistic Processes - A Paradigm Shift and its Limitations* (SFB 637).

6 REFERENCES

[1] Hülsmann, M., Windt, K. (eds), 2007, Understanding Autonomous Cooperation & Control in Logistics – The Impact on Management, Information and Communication and Material Flow, Springer, Berlin, Germany

[2] Object Management Group: Homepage of the OMG Model Driven Architecture Initiative. URL: http://www.omg.org/mda/, last access: 2007-01-29

[3] Assmann, U., Rensink, A., Aksit, M. (Eds), 2005, Model Driven Architecture: Foundations and Applications, LNCS 3599, Springer, Berlin, Germany

[4] Eclipse Foundation: Homepage. URL: http://www.eclipse.org/, last access: 2007-01-29

[5] McAffer, J., Lemieux, J.-M., 2005, Eclipse Rich Client Platform: designing, coding, and packaging Java applications, Addison-Wesley, Upper Saddle River, NJ, USA

[6] Scholz-Reiter, B., Kolditz, J., Hildebrandt, T., 2007, Specifying adaptive business processes within the production logistics domain – a new modelling concept and its challenges. In: [1]

[7] Scholz-Reiter, B., Hildebrandt, T., Kolditz, J., 2006, A modelling methodology for control processes of autonomous production systems. In: Monostori, L., Ilie-Zudor, E. (eds.), Proceedings of MITIP 2006, Budapest, Hungary, pp. 317-322

[8] Scheer, A.-W., 1994, Business Process Engineering – Reference Models for Industrial Companies, 2nd ed., Springer, Berlin et al., Germany

[9] OMG: Unified Modelling Language Specification (version 2.0). URL: http://www.uml.org 2005, last access 2007-01-29

[10] Oestereich, B, 2003, Objektorientierte Geschäftsprozessmodellierung mit der UML, dpunkt Verlag, Heidelberg, Germany

[11] Weiß, G., Jakob, R., 2005, Agentenorientierte Softwareentwicklung, Springer, Berlin, Germany

[12] Budinsky, F. et al., 2003, Eclipse Modeling Framework: a developer's guide, Addison Wesley, Boston, USA.

[13] Eclipse Graphical Modeling Framework (GMF), Homepage, URL: http://www.eclipse.org/gmf/, last access: 2007-01-29