23

UML AS A BASIS TO MODEL AUTONOMOUS PRODUCTION SYSTEMS

Bernd Scholz-Reiter, Jan Kolditz, Torsten Hildebrandt
Department of Planning and Control of Production Systems
University of Bremen, Bremen, Germany
{bsr,kol,hil}@biba.uni-bremen.de,

This article will investigate the suitability of the Unified Modelling Language (UML) for requirements analysis of autonomous logistic processes by a logistics domain expert. Such a model is the basis for subsequent implementation of the system consisting of software engineering and hardware configuration. Relevant parts of UML will be used to model an exemplary scenario which will form the basis to derive benefits and drawbacks of using the UML in this context. Suggestions on how the identified gaps can be filled will be presented in the paper.

1. INTRODUCTION

Enterprises are exposed to an increasingly dynamic environment today. Furthermore increasing competition caused by globalisation more and more requires gaining competitive advantages by improved process control, within and beyond the borders of producing enterprises. One possibility to face increasing dynamics is autonomous control of logistic processes, which is the main research topic of the SFB 637, the interdisciplinary research effort this work is based on. Autonomous control in this context means processes of decentralized decision making in heterarchical structures. It requires the ability and possibility of interacting system elements to autonomously make goal-oriented decisions. The use of autonomous control aims at a higher robustness of systems and simplified processes achieved by distributed handling of dynamics and complexity due to greater flexibility and autonomy of decision making. Focus of the SFB lies in the areas of production and transport logistics, so the system elements, making their decisions autonomously, are the logistic objects themselves (Scholz-Reiter, 2004).

In order to enable logistic objects to be intelligent they have to be provided with smart labels. While today's RFID (radio frequency identification)-chips have very limited capabilities with respect to energy, range, storage capacity and especially information processing (Finkenzeller, 2003), near future shall bring highly evolved smart labels that can provide resources alike micro computers to logistic objects. Nowadays RFID is already widely used in industry for identification matters and several visions for future applications exist (Fleisch, 2005), (Heinrich, 2005).

This paper presents a concept for modelling autonomous cooperating logistic processes. First section 2 gives a short overview of important aspects of the modelling method, especially the view concept to structure the model. Main section 3 discusses the notation for modelling the different aspects addressed in their respective views. It presents the appropriate notational elements of UML to be used in these contexts. Shortcomings of the UML notation to depict certain aspects of the autonomous logistics system are also presented in the context of the relevant view. The paper is concluded by a short summary and an outlook of future work.

2. MODELLING CONCEPT FOR AUTONOMOUS COOPERATING LOGISTIC PROCESSES

Creating models of autonomous production systems usually leads to a high degree of complexity, as does the creation of any comprehensive process model. Hence a view concept serves as a means to reduce the complexity constructing such a model (Scheer 1994). 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, according to the basic classification in UML (OMG, 2006) that is also appropriate here. In our proposed methodology we distinguish five different views, three in the static and two in the dynamic sub-model. In the static model we differentiate between a structure view, knowledge view and ability view. Process view and communication view are the different views of the dynamic model.

The *structure view* that shows the relevant logistic objects is the starting point. 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. 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 problem 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 *communication view* presents the contents and temporal sequence of information exchange between logistic objects. Depicting the communication is especially necessary to show the interaction of autonomously deciding, otherwise only loosely coupled objects to model their interaction (Weiss, 2005).

In addition to the dynamic and static model just described we distinguish a macro and micro perspective. This distinction (largely independent of the distinction between the different views just presented) is also used in methods for software agent development (Weiss, 2000). The macro view describes the interaction between the autonomous logistic objects. To some extend, it shows an external view onto the system, its elements and their relations and interactions. On the contrary the micro

view describes the actions within and composition of the autonomous logistic objects. For the micro-level especially the process, knowledge and ability view are relevant, while all views proposed are relevant for the macro-level. This means that the micro-macro perspective is orthogonal to the views described before. Nevertheless not all views use both perspectives to the same extend.

3. UML AS A MODELLING BASIS

3.1 Structure View

In the structure view class diagrams (see figure 1) are the most important diagrams. In general class diagrams support the modelling of system structure. Class diagrams describe the static elements, their fundamental characteristic and the relations between each other. Classes themselves are specified by attributes and operations. An attribute is a characteristic of a class that takes a concrete value in an object, which is an instance of a class. One kind of relationship between classes is generalisation. The central point there is the inheritance of the generalised class' structure, e.g. its attributes, to the specialized one. Associations describe the static relations between classes and are mostly explained by a name, association specific roles of the classes and multiplicities. Special associations are aggregation and composition. Both describe a part-of-relationship between two classes while the composition represents an existential dependency between a part and the whole it belongs to.
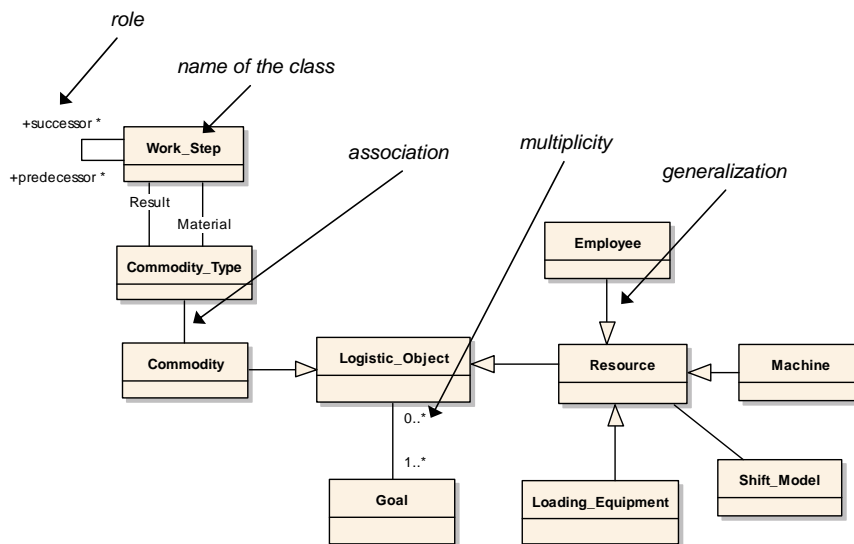
Figure 1 –   Class diagram in the structure view

For modelling autonomous logistics processes class diagrams provide the opportunity to picture structure and organization of the logistic system. Accordingly

aggregation and composition can be used to model organisational units and generalizations to model the hierarchy of classes. Roles and names of associations support a better understandability of a model. The same applies for multiplicities which also provide a possibility of an early explication of structural constraints.

The exemplary class diagram in figure 1 shows an extract of the system structure. The central class *Logistic Object* is a generalisation of *Commodity* and *Resource*, whereas *Resource* again has specialisations *Employee*, *Machine* and *Loading Equipment*. The class *Shift Model* determines resource availability and therefore is an important factor for its capacity. Logistic objects are associated with goals and it is determined, that a logistic object may have multiple goals but have to have at least one.

3.2 Knowledge View

In the knowledge view class and object diagrams are the most important diagrams. These have already been sketched in the previous section. In the structure view the main modelling focus lies in the structure of the system and thus in the possible relationships of one element to another. The knowledge view addresses the content of elements already appearing in the structure view as well as of elements not specified yet. Furthermore this view provides information about the distribution of knowledge in the system. The knowledge distribution only shows a point in time, because no changes in time are described. The changes of knowledge distribution depending on time are looked at on an abstract level in the communication and process view but the concrete value of a system parameter arises during run time and is not specified beforehand. Modelling the knowledge distribution rather serves the purpose of assuring that the knowledge necessary for running the processes is present in the system and it is specified where it can be found.

One example for the knowledge view is the description and assignment of goals. In the structure view it is specified that a logistic object is supposed to have goals. In the knowledge view dedicated goals are assigned to the different logistic objects. This is possible on the class level for example when assigning the goal class *maximise utilisation* to the class *machine*, as well as on the object level when assigning goals with concrete parameter values to single machine objects.

For a clearly arranged overview of the durable aspects of knowledge distribution of the system, particularly the distribution existent in the initial state before run time, UML diagrams do not provide adequate notations. Therefore we use so called *knowledge maps* from business process modelling that have been enriched with elements from knowledge management (Allweyer, 1998).

3.3 Ability View

The ability view offers to model the autonomous system elements' abilities as abstract collections of different operations they are able to perform. This concept can be represented using UML-interfaces, which also allow generalisations between abilities and therefore creating inheritance hierarchies between the abilities. Interfaces and thus abilities can be shown in UML by class diagrams, already described earlier in this paper. The ability view therefore gives an overview of the abilities present in the system and their distribution to various objects.

A second aspect of abilities is the mapping between them and logistic objects, i.e. specifying which logistic objects have certain abilities. This is again shown in class diagrams and indicated by a special kind of relationship: the realizes-relationship between an interface and a class, graphically shown by a dashed line (with an arrow-head like a generalisation) from the interface to the class that offers the interface. How a class that implements a certain ability provides the necessary operations is not specified in the ability view, but the views of the dynamic model (process and communication view) instead as described in the following.

3.4 Process View

In the process view activity diagrams and state machines are used. Activity diagrams are constructed similar to Petri nets and provide a basis to model the processes possible at run time. They allow to specify particular operations or, on a higher level of abstraction, complete business processes. Activity diagrams are the diagram type most often used when applying the UML to business process modelling (Oesterreich, 2003).
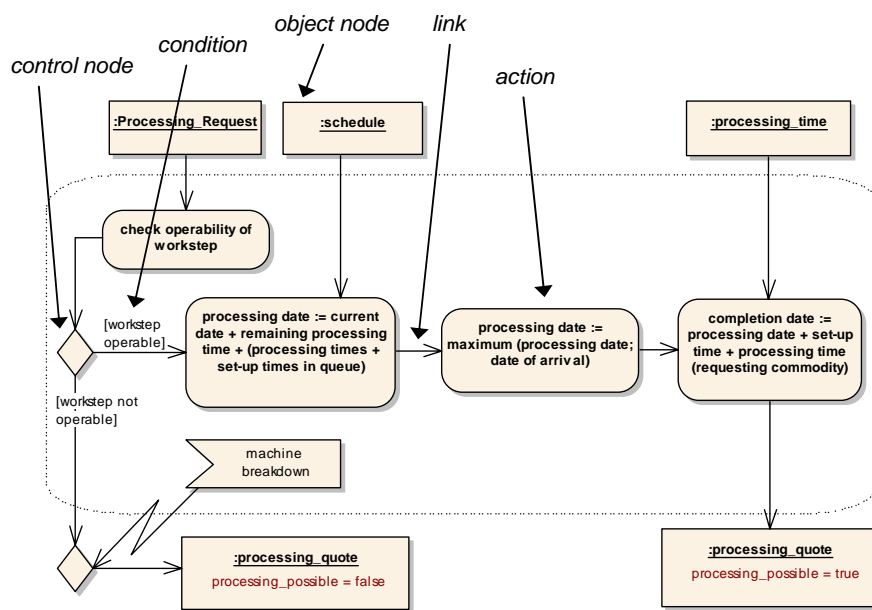

Figure 2 – Activity diagram in the process view

First of all a number of actions characterising an activity are elements included in an activity diagram. These actions are connected by directed links which in combination with control nodes determine the control flow within a process and therefore possible sequences in which the action nodes are executed. Among the control nodes there are initial and final nodes, decision and merge nodes as well as fork and join nodes.

The exemplary activity diagram shown in figure 2 describes the internal handling process of a machine for a processing request. First the incoming request is checked whether the machine is able to perform the asked workstep. If the workstep is operable the machine calculates the earliest completion date for the commodity on the basis of its current schedule, the estimated arrival date of the commodity and the processing times. The process is completed by an outgoing message *processing quote* containing the estimated completion date for the requesting commodity. If the machine breaks down, the whole process is aborted and a negative processing quote is sent.

The state machine is another possibility to model the behaviour of a part of a system. There the behaviour is specified by possible sequences of discrete system states as well as internal or external triggers that cause a change from one state to another. States are connected by transitions in form of directed links. To model more complex relations between states the UML also provides pseudo states like an initial state, entry and exit point. Furthermore alternatives can be described by junction pseudo states or choice pseudo states. Parallelisation can also be expressed and is shown by forks at the beginning and joins at the end of parallel sections. These pseudo states can be connected with states and with each other.

3.5 Communication View

The communication view consists of two main components, the description of the message exchange between logistic objects and the description of the messages themselves concerning their structure and contents. For description of message exchange the sequence diagram is primarily applicable.

Sequence diagrams (see example in figure 3) provide a basis to illustrate communication processes within a system. They allow to model messages exchanged by communication partners in respect to their temporal and logical order. The communication partners are listed horizontally, the messages are ordered vertically. The messages themselves are symbolised as a directed link from the lifeline of the sender to the one of the receiver. Using so called combined fragments the message exchange process can be combined with rules to model for example alternative, parallel or optional message sequences.

Sequence diagrams provide vital instruments for modelling relevant aspects of autonomous logistic objects. So the information exchange necessary for solving a problem can be analysed and specified. That assures the availability of information needed during the runtime of the system and helps to allow an estimation of the communication volume.

The exemplary sequence diagram in figure 3 shows the communication between a commodity and a machine. The commodity executes a *processing request* whereon the machine answers with a *processing quote*. After the commodity has compared the quotes and selected one machine, the machine gets either a *booking request* or a *cancellation of the processing quote*.

One aspect that is not covered by the existing UML 2.0 specification concerns the exact determination of the number of messages in case of multiple communication partners. So the example in figure 3 does not determine, that the sum of the messages *booking request* and *cancel processing quote* is not allowed to exceed the number of requested machines. However such a determination would be

useful to increase clearness and to allow enhanced plausibility checks at run time. A similar problem arises in the context of efforts to provide a better support of UML for concepts needed for agent-oriented software engineering (Bauer, 2005). There it is recommended to integrate notational elements in sequence diagrams similar to multiplicities in class diagrams. A similar mechanism is useful in our context of modelling autonomous logistics processes as well.
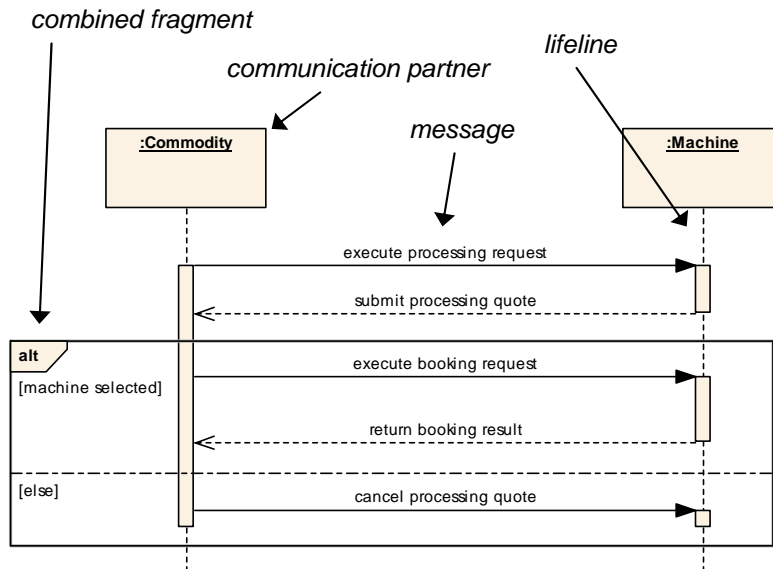

Figure 3 –   Sequence diagram in the communication view

The second main component of the communication view addresses the structure and content of the messages. For modelling the messages themselves class diagrams and object diagrams are adequate. An important part of specifying classes are attributes. In the example shown in figure 4 the message is described by the class *processing request*. There are four characterising attributes. The first attribute *sender* is of the type *commodity*. That way it is determined that processing requests can only be sent by a commodity or more precisely only a commodity can be put in as a sender.
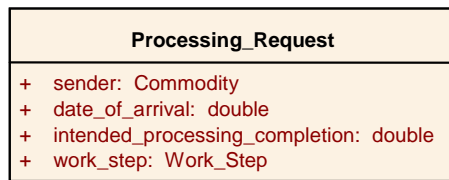

Figure 4 –   class diagram in the communication view

4. CONCLUSION

This paper investigates the use of UML to model autonomous logistic processes. Structured by a view model consisting of five different views, three static and two dynamic views, suitable UML notation to represent the elements necessary for requirement analysis to engineer such a system is presented as well as problems in using UML to show certain aspects are highlighted and solutions to adequately solve the problems are presented.

The work presented here is the foundation of a more comprehensive modelling methodology an overview of which is presented in (Scholz-Reiter, 2006). Further research will use the notation presented here to create a reference model of autonomous (production-)logistic processes, a procedure model to guide a modeller to efficiently engineer such systems. Furthermore a software tool is currently in development that will be specifically tailored to support our modelling methodology.

5. ACKNOWLEDGEMENTS

6. REFERENCES

Allweyer, Th.: Modellbasiertes Wissensmanagement. In: Information Management 13 (1998) 1, pp. 37–45.

Bauer, B., Odell, J.: UML 2.0 and Agents: How to Build Agent-based Systems with the new UML Standard. In: Journal of Engineering Applications of Artificial Intelligence 18 (2005) 2, pp. 141–157.

Heinrich, C.: RFID and Beyond: Growing Your Business through Real World Awareness. Wiley Publishing, Indianapolis, Indiana, 2005.

Finkenzeller, K.: RFID-Handbook - Fundamentals and Applications in Contactless Smart Cards Identification, 2. edn. Wiley & Sons LTD, Swadlincote, UK, 2003.

Fleisch, E., Mattern, F. (eds.): Das Internet der Dinge - Ubiquitous Computing und RFID in der Praxis: Visionen, Technologien, Anwendungen, Handlungsanleitungen. Springer, Berlin, Germany, 2005.

OMG: Unified Modelling Language Specification (version 2.0). http://www.uml.org. Retrieved April 6th, 2006.

Scheer, A.-W.: Business Process Engineering – Reference Models for Industrial Companies, 2. edn. Springer, Berlin et al., 1994.

Scholz-Reiter, B., Windt, K., Freitag, M.: Autonomous Logistic Processes - New Demands and First Approaches. In: Proceedings of the 37th CIRP-International Seminar on Manufacturing Systems 2004, Budapest, Hungary, 19.-21.5.2004, pp. 357-362.

Scholz-Reiter, B., Kolditz, J., Hildebrandt, T.: Analysis and Design of Autonomous Logistic Processes. In: Proceedings of the 39th CIRP International Seminar on Manufacturing Systems, Ljubljana, Slovenia, 7.–9.6.2006, to appear.

Weiss, G. (ed): Multiagent Systems – A modern approach to distributed artificial intelligence. The MIT Press, Cambridge, USA, 2000.

Weiss, G., Jakob, R.: Agentenorientierte Softwareentwicklung. Springer, Berlin Heidelberg, Germany, 2005.