Modellierung selbststeuernder produktionslogistischer Prozesse – die Modellierungsmethode ALEM

Bernd Scholz-Reiter, Torsten Hildebrandt, Jan Kolditz

Planung und Steuerung produktionstechnischer Systeme

Universität Bremen

Hochschulring 20, 28359 Bremen

{bsr, hil, kol}@biba.uni-bremen.de

Abstract

Die Beherrschung von Dynamik und Komplexität logistischer Systeme wird auch in Zukunft für Unternehmen weiter an Bedeutung gewinnen. Eine Möglichkeit dieser Herausforderung zu begegnen, stellt die Selbststeuerung logistischer Prozesse dar. Dieser Artikel entwirft auf Basis des Systems Engineering einen Rahmen für die Entwicklung eines selbststeuernden Produktionssystems und stellt die hierfür entwickelte Modellierungsmethode ALEM (Autonomous Logistics Engineering Methodology) unter Beschreibung der verwendeten Notation, des Modellierungsvorgehens und der spezifischen Modellierungssoftware vor. Ziel ist eine für den Logistikexperten handhabbare Modellierungsmethode zum Entwurf derartiger Systeme.

1 Selbststeuerung logistischer Prozesse

Eine möglichst gute Beherrschung inter- und intraorganisationaler Geschäftsprozesse wird angesichts immer stärkeren Wettbewerbs für Unternehmen zunehmend wichtig. Gründe hierfür liegen etwa in der Globalisierung und der Tatsache, dass Unternehmen generell in einem zunehmend dynamischer werdenden Umfeld agieren müssen. Eine Möglichkeit dieser zunehmenden Dynamik zu begegnen ist die Selbststeuerung logistischer Prozesse. Diese zielt darauf ab, trotz steigender externer und interner Komplexität robustere Prozesse zu ermöglichen.

Der Begriff Selbststeuerung beschreibt Prozesse dezentraler Entscheidungsfindung in heterarchischen Strukturen. Er setzt voraus, dass interagierende Elemente in nicht-deterministischen Systemen die Fähigkeit und Möglichkeit besitzen, autonom Entscheidungen zu treffen. Das Ziel ist hierbei, eine erhöhte Robustheit und ein positiv emergentes Verhalten des Gesamtsystems aufgrund eines verteilten und flexiblen Umgangs mit Dynamik und Komplexität zu erreichen (vgl. Hülsmann und Windt (2007)). Da der Fokus der Arbeiten hierbei in den Bereichen der Produktions- und Transportlogistik liegt, handelt es sich hier bei den autonom Entscheidungen treffenden Systemelementen um die logistischen Objekte selbst, wie z.B. Güter, Maschinen, Lager, Fördermittel etc.

Um diese „Intelligenz“ logistischer Objekte zu erreichen, müssen diese mit sog. Smart Labels ausgestattet sein. Aktuell verfügbare RFID (radio frequency identification)-Tags verfügen jedoch nur über sehr begrenzte Fähigkeiten hinsichtlich Energiereserven, Reichweite sowie Speicher- und insb. Verarbeitungskapazitäten (vgl. Finkenzeller (2003)). Trotzdem werden sie bereits auf breiter Basis in der Industrie für Identifikationsaufgaben eingesetzt und es existieren eine Reihe von Visionen für zukünftige Anwendungen (vgl. Das und Harrop (2001), Fleisch (2005) sowie Heinrich (2005)). In naher Zukunft sollte es möglich sein mit weiterentwickelten Smart Labels, die logistischen Objekten die Möglichkeiten von Miniatur-Computern erschließen, die „reine“ Vision der Selbststeuerung logistischer Prozesse umzusetzen. Im Gegensatz zu Manufacturing Execution Systems, die ebenfalls eine bessere Fertigungssteuerung als ERP-Systeme ermöglichen sollen, setzt Selbststeuerung heterarchische Systeme voraus. Mit der Verlagerung der „Intelligenz“ hin zum logistischen Objekt erfolgt sowohl eine logische wie physische Verteilung der Entscheidungsfindung. Neben dieser Ausprägung sind jedoch je nach den Möglichkeiten der verfügbaren Hardware auch Konfigurationen möglich, bei denen physische und logische Verteiltheit nicht übereinstimmen.

Die Entwicklung eines derartigen Systems erfordert eine geeignete Entwicklungsmethode, um alle notwendigen Aspekte des Systems angemessen entwerfen zu können, also etwa: wie sieht das Szenario aus, welche logistischen Objekte gibt es, welche „Intelligenz“ besitzen diese? Derartige Informationen sind die Basis, um eine adäquate Steuerungsstrategie modellieren zu können, inkl. z.B. der Prozesse auf den selbststeuernden logistischen Objekten, dem nötigen Wissen für diese Prozesse und die geeignete Kommunikation oder einer anderen Form der Koordination zwischen den Objekten.

Dieser Beitrag beschreibt die Entwicklungsmethode ALEM (Autonomous Logistics Engineering Methodology) und fokussiert dabei auf das Modellierungskonzept als einem Teil hiervon. Hierzu wird zunächst ein Überblick über Modellierung unter dem Paradigma der Selbststeuerung gegeben. Danach erfolgt in den Abschnitten 3 und 4 eine detaillierte Darstellung des Modellierungsvorgehens und der entwickelten Modellierungssoftware. Den Beitrag beschließt eine kurze Zusammenfassung und ein Ausblick auf zukünftige Forschungsarbeiten.

2 Modellierung von Selbststeuerung

2.1 Der Entwurfsprozess

Die Entwicklung eines selbststeuernden logistischen Systems lässt sich, wie in Abbildung 1 dargestellt, auf Basis des Vorgehensmodells des Systems Engineering (vgl. Haberfellner et al. (2002)) konkretisieren. Den methodischen Kern bildet dabei ein Iterationszyklus während der Haupt- und Detailstudienphase. Im Folgenden werden die einzelnen Phasen der Entwicklung eines selbststeuernden Systems kurz erläutert.

Anstoß: Diese erste eher unstrukturierte Phase wird durch die Wahrnehmung eines Problems initiiert und mit dem Entschluss zur Einleitung einer geordneten Vorstudie beendet. Der Anstoß dürfte in erster Linie in der Erwartung verbesserter Leistungsdaten des Produktionssystems durch die Einführung selbststeuernder logistischer Prozesse liegen.

Vorstudie: Während der Vorstudie ist eine Definition des Problemfelds bzw. des zu betrachtenden logistischen Systems vorzunehmen. Damit zusammenhängend ist auch die Frage zu klären, welche konkreten Ziele mit einer Einführung selbststeuernder Prozesse im vorliegenden Fall verfolgt werden, was vor allem durch die Auswahl und Quantifizierung logistischer Zielgrößen umgesetzt werden kann. Für die Entscheidung über die Art und Weise der Fortführung des Projekts wird mit vertretbarem Aufwand hinreichend gut überschlagen, inwiefern eine Anwendung der Selbststeuerung auf das betrachtete Szenario sinnvoll und vielversprechend ist. Diese Beurteilung der Vorteilhaftigkeit der Selbststeuerung unter bestimmten Bedingungen stellt eine umfassende in Bearbeitung befindliche Forschungsfrage dar (vgl. Philipp et al. (2007)), welche hier nicht weiter vertieft werden soll.

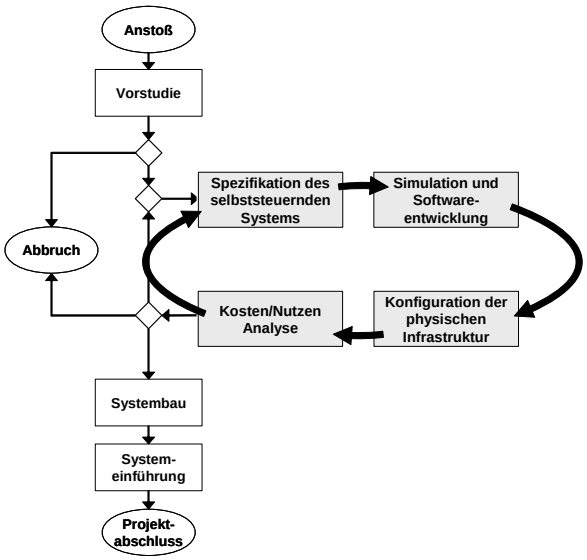

Abbildung 1: Entwicklung eines selbststeuernden logistischen Systems (in Erweiterung zu Haberfellner et al. (2002))

Haupt- und Detailstudien: Der in Abbildung 1 zu erkennende Iterationszyklus als Kern des Entwicklungsprozesses setzt sich aus vier aufeinander aufbauenden Teilphasen zusammen.

Die erste Teilphase im Iterationszyklus besteht in der *Spezifikation* des selbststeuernden Systems. Es wird eine semi-formale Spezifikation der selbststeuernden Objekte durchgeführt sowie die Auslegung und Zuordnung der Entscheidungsprozesse. Um die Funktionsfähigkeit des Systems zu gewährleisten, sind sämtliche Elemente und Prozesse aufeinander abzustimmen, wodurch diese Teilphase das Fundament der Entwicklung des Systems darstellt.

In der Teilphase *Simulation* wird das zuvor erstellte Konzept getestet. Dabei steht insbesondere die durch die Auslegung der Systemelemente bestimmte Lauf- und Leistungsfähigkeit des Gesamtsystems im Mittelpunkt. Die zentrale Aufgabe der Verifikation besteht auch darin, ein gewünschtes Systemverhalten nachzuweisen und die verbesserte Erreichung logistischer Ziele zuverlässig vorherzusagen. Nur so können auf der Selbststeuerung basierende emergente logistische Systeme für industrielle Anwendungen interessant sein.

In der Teilphase der *Konfiguration* der physischen Infrastruktur wird auf Basis der zuvor gewonnenen Erkenntnisse eine Abschätzung der notwenigen technischen Ausstattung des selbststeuernden Systems durchgeführt, welche mit jeder Iteration detaillierter wird. Schlussfolgerungen hierzu können sowohl aus dem erstellten Prozessmodell als auch aus der Simulation gezogen werden. Beispielsweise können voraussichtlich aus der im Prozessmodell durchgeführten Zuordnung von Steuerungsprozessen und Daten zu logistischen Objekten des Systems notwendige Speicher und Rechenkapazitäten abgeleitet werden.

Den Abschluss eines Iterationszyklus bildet die *Kosten-/Nutzen-Analyse*. Auf Basis der hier zu erfolgenden Bewertung kann das während der Spezifikation entworfene Modell entsprechend den neuen Erkenntnissen angepasst werden. Bei wiederholt oder generell negativem Ausgang der Bewertung ist von einem Selbststeuerungsansatz für das betrachtete Szenario abzusehen.

Systembau: Der Inhalt dieser Phase besteht in der Realisierung des auf selbststeuernden Prozessen beruhenden Systems. Die beiden Hauptkomponenten sind dabei die Implementierung der Software sowie die Herstellung und Integration von Anlagen und Geräten. Die Implementierung der Software sollte zur Vermeidung zusätzlicher Arbeit nach Möglichkeit unter Verwendung von bereits im Rahmen der Simulation erstellter Komponenten durchgeführt werden.

Systemeinführung und Projektabschluss: Die Einführung des Systems ist nach Möglichkeit stufenweise vorzunehmen. In der Regel wird es sich um große und komplizierte Systeme handeln, deren Einführung mit schwer oder nicht kalkulierbaren Nebenerscheinungen und deshalb mit hohen Risiken verbunden ist. Nach Prüfung der Zielerfüllung kann die Übergabe des Systems durch den Ersteller bzw. das Projektteam an den Betreiber erfolgen.

2.2 Das Modellierungskonzept von ALEM im Überblick

Dieser Abschnitt konzentriert sich auf den ersten Schritt des eben dargelegten Entwicklungszyklus, der semi-formalen Spezifikation des logistischen Systems. Um diese Spezifikation zu unterstützen, schlagen wir eine Modellierungsmethode als Teil der Autonomous Logistics Engineering Methodology (ALEM) vor, die aus den drei Teilen ALEM-N (ALEM-Notation), ALEM-P (ALEM-Procedure) und ALEM-T (ALEM-Tool) besteht.

ALEM-N enthält ein spezifisches Sichtenkonzept. In jeder der darin definierten fünf Sichten (im statischen Teilmodell wird unterschieden zwischen einer Struktur-, Wissens- sowie Fähigkeitensicht; eine Modellierung dynamischer Aspekte erfolgt in der Prozess- und Kommunikationssicht) wird festgelegt, welche Aspekte des Systems hierin modelliert werden, welche Notationselemente zum Einsatz kommen und deren Bedeutung. Die Notation greift soweit als möglich auf die relevante Teilmenge der Unified Modelling Language (UML, vgl. OMG (2007)) zurück. Eine nähere Beschreibung von ALEM-N findet sich in Scholz-Reiter et al. (2006).

Das Vorgehensmodell ALEM-P beschreibt die Schritte, die zu einer erfolgreichen Modellerstellung zu erfolgen haben und leitet einen Modellierer bei der Analyse eines selbststeuernden logistischen Systems an. ALEM-T ist eine Modellierungssoftware, die entwickelt wurde, um Notation und Vorgehensmodell möglichst gut zu unterstützen. Darüber hinaus ist ein Referenzmodell ebenfalls Teil von ALEM und wird von ALEM-T bereitgestellt,

um die Konstruktion neuer Modelle unter Wiederverwendung bestehender Arbeiten zu erleichtern. ALEM-P bzw. ALEM-N sind Gegenstand der folgenden beiden Abschnitte. Eine detaillierte Diskussion der Anforderungen an eine Modellierungsmethode für Selbststeuerung und inwieweit diese von ALEM erfüllt werden, kann hier aus Platzgründen leider nicht erfolgen, der interessierte Leser sei hierzu auf Scholz-Reiter et al. (2007) verwiesen.

2.3 Einordnung in bestehende Arbeiten

Besondere Relevanz für die hier vorgestellte Modellierungsmethode weisen Arbeiten zur Anwendung von Multiagentensystemen im Bereich der Produktionsplanung und –steuerung auf (vgl. Monostori et al. (2006)). Zahlreiche Arbeiten hierzu existieren insbesondere im Kontext der Arbeiten zu Holonischen Fertigungssystemen (HMS, Holonic Manufacturing Systems, vgl. Babiceanu und Chen (2006)).

Babiceanu und Chen (2006) weisen darauf hin, dass holonische Systeme bisher in der Praxis nur wenig verbreitet sind. Als Ursachen werden neben den Geschäftsstrategien der Unternehmen ein schwieriger Entwicklungsprozess für derartige Systeme genannt. Ziel der hier vorgestellten Methodik ist es, einen Beitrag zu liefern, um genau diesen letztgenannten Punkt durch eine für den Logistikexperten handhabbaren Entwurfsmethodik abzuschwächen. Ziel ist also nicht eine allgemeine Modellierungsmethodik für Multi-Agentensysteme zu entwickeln, wie es etwa Tropos (vgl. Bresciani et al. (2004)) oder GAIA (vgl. Zambonelli et al. 2003) darstellen.

Eine ähnliche Zielstellung verfolgen die Arbeiten von Bussmann et al. (2004) zur DACS-Methodik (Designing Agent Based Production Control Systems). Die Methode zielt im Gegensatz zu den eben genannten Methoden zur agentenorientierten Softwareentwicklung auf die Anwendung durch Steuerungsexperten für Produktionssysteme ohne Erfahrung im Bereich agentenbasierter Technologie ab. Sie ist vor allem auf die Automatisierung einzelner Arbeitssysteme ausgerichtet und besteht aus drei Hauptschritten *Analyse der Steuerungsentscheidungen*, *Identifizierung von Agenten* und *Auswahl der Interaktionsprotokolle*. Unterschiede zu ALEM bestehen etwa in einer im stärkeren Maße auf UML basierende Notation, also einem Standard, der in verschiedenen Bereichen weite Verbreitung findet und eine entsprechend breite Softwareunterstützung aufweist. Für DACS besteht aktuell keine spezifische Softwareunterstützung. Für ALEM wurde der Weg gegangen eine eigene Modellierungssoftware zu entwickeln, um eine möglichst nahtlose Unterstützung der Methodik und auch der nicht UML-basierten Notationselemente zu erlauben.

3 Das Vorgehensmodell ALEM-P

Das in Abbildung 3 skizzierte Vorgehensmodell stellt einen Leitfaden zur Modellierung dar, welches einerseits zur Sicherung der Modellqualität und andererseits zur Verringerung des Konstruktionsaufwands während der Modellierung beiträgt. Es handelt sich um ein spezifisches Vorgehensmodell, welches operationale Handlungsempfehlungen unter Verwendung der Notation ALEM-N zur Verfügung stellt. Dadurch wird einem Nutzer mit vertieften logistischen Kenntnissen eine Möglichkeit zur Konstruktion von visuellen Modellen zur Unterstützung von Analyse, Entwurf und Optimierung selbststeuernder logistischer Systeme zur Verfügung gestellt. Im Folgenden erfolgt eine Beschreibung der Hauptschritte.

3.1 Ziele

Der erste Schritt im Entwurfsprozess des selbststeuernden Systems besteht in der Betrachtung der im System verfolgten Ziele. Startpunkt ist die Klärung der globalen Zielsetzung der Steuerungsprozesse. Die Ziele und ihre Priorisierungen sind von dem Modellierer in Abstimmung mit einer für die Strategie verantwortlichen Person zu klären und natürlichsprachlich festzuhalten. Dabei liegt es nahe, die klassischen Ziele der Produktionslogistik (vgl. Wiendahl (2005)) als übergeordnete Globalziele zu verwenden, von welchen detaillierte und konkretisierte lokale Ziele abgeleitet werden.

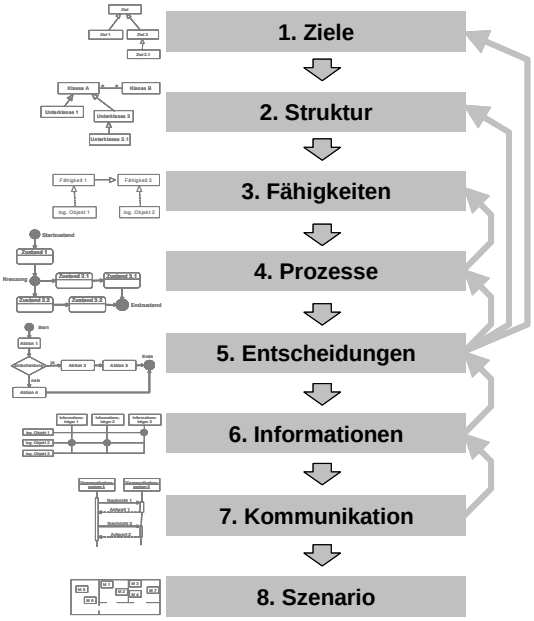

Abbildung 2: Das Vorgehensmodell im Überblick

3.2 Struktur

Der zweite Schritt im Entwurfsprozess besteht in der Darstellung der Struktur des Systems und damit in der Zusammenstellung und Dokumentation der das System bildenden Elemente sowie ihrer statischen Beziehungen. Im Mittelpunkt dieses Schrittes stehen die selbststeuernden logistischen Objekte, weshalb der Modellierer bereits zu diesem frühen Zeitpunkt der Modellkonstruktion eine grobe Vorstellung zu entwickeln hat, welche Systemelemente als selbststeuernd vorgesehen werden und welche nicht. So findet der Aufbau des Systems nicht rein Top-Down statt, sondern beginnt bei den ausgewählten selbststeuernden logistischen Objekten, welche mit anderen identifizierten Systemelementen in Beziehung gesetzt werden, wodurch allmählich ein Gesamtmodell entsteht.

3.3 Fähigkeiten

Der dritte Schritt des Vorgehens dient dem Entwurf einer Strukturierung von Fähigkeiten und Ihrer Zuordnung zu selbststeuernden logistischen Objekten. Fähigkeiten stellen dabei abstrakte

Konstrukte dar, welche sich aus einer Menge von zu ihrer Umsetzung notwendigen Prozessen zusammensetzen. Die Fähigkeiten und damit auch die sie realisierenden Prozesse können darüber hinaus selbst strukturiert und zueinander in Beziehung gesetzt werden, da beispielsweise mehrere untergeordnete Fähigkeiten eine übergeordnete Fähigkeit ausbilden können.

Zu Beginn der Modellkonstruktion handelt es sich um einen ersten Entwurf der Fähigkeiten und ihrer Zuordnung zu den logistischen Objekten. Beim Durchlaufen weiterer Schritte der Modellkonstruktion wird eine zunehmend vollständige Erfassung der im System relevanten Prozesse angestrebt, wodurch die hier vorgenommene Fähigkeitenzuordnung ständig zu aktualisieren und zu ergänzen ist. Dadurch ist eine einfachere Identifizierung von Funktionshäufungen möglich, welche bei absehbaren Verletzungen von Restriktionen, beispielsweise während der sich anschließenden Softwareentwicklung oder durch begrenzte Kapazitäten der für die Umsetzung vorgesehenen physischen Infrastruktur, eine Anpassung und Neuverteilung der Prozesse nach sich zieht. Somit beinhaltet dieser Schritt einerseits entscheidende Weichenstellungen für die nachfolgende Prozessauslegung, andererseits können und müssen auf Basis späterer Erkenntnisse die Strukturierungen und Zuordnungen der Fähigkeiten überarbeitet werden.

3.4 Prozesse

Der vierte Entwurfsschritt hat die Modellierung der Prozesse, insbesondere der Steuerungsprozesse zum Inhalt. Der Prozessentwurf ist in zwei Teilschritte untergliedert. Zunächst werden von einem störungsfreien Betrieb ausgehende Standardprozesse dargestellt, um diese anschließend systematisch durch Prozesse zum Abfangen von ungeplanten Ereignissen zu ergänzen.

Die Modellbildung erfolgt gemäß der im logistischen System ablaufenden physischen Prozesse der einzelnen selbststeuernden logistischen Objekte. So sind in einem Produktionssystem beispielsweise der Weg eines Gutes vom Wareneingangslager durch die Produktion bis zum Warenausgangslager, die im Laufe der Zeit durchzuführenden Prozesse eines Arbeitssystems oder die Aktivitäten und Zustände eines Transportmittels von der Aufnahme einzelner Güter bis zur Planung seiner Route zu betrachten. Eine detaillierte Ausführung der einzelnen zu bewältigenden Entscheidungssituationen wird grundsätzlich noch nicht angestrebt, sondern lediglich deren Einbindung in die umgebenden Prozesse.

Im zweiten Teilschritt ist nun die Annahme eines idealen Ablaufs mit Vernachlässigung möglicher Störungen durch die Berücksichtigung von in einem logistischen System auftretenden Unsicherheitsfaktoren zu ersetzen. Für den Entwurf eines selbststeuernden logistischen Systems sind daher zusätzlich zu den bisher entworfenen, von einem idealen Ablauf ausgehenden Steuerungsprozessen weitere für den Umgang mit auftretenden Störungen zu ergänzen. Um eine systematische Einbindung relevanter Prozesse zu gewährleisten, erfolgt eine Orientierung an bestehenden Strukturierungen von Ursachen, Störungen und Auswirkungen sowie entsprechenden Aufgaben im Störungsmanagement (vgl. Patig (2001)).

3.5 Entscheidungen

In diesem Schritt des Modellierungsprozesses wird auf die Entscheidungen fokussiert. Für die Identifikation aller Entscheidungssituationen in den selbststeuernden Prozessen ist eine Betrachtung aller selbststeuernder logistischer Objekte und der in Schritt 4 erstellten Prozessmodelle notwendig. Grundlage der Identifikation und Beschreibung bildet die Struktur einer Entscheidung (vgl. Laux (2005)). Darauf aufbauend wird eine Steuerungsentscheidung durch einen Entscheider, ein Ziel und eine dieses Ziel abbildende Entscheidungsregel, einen Auslöser sowie ein Entscheidungsfeld charakterisiert. Das Entscheidungsfeld wird insbesondere durch die möglichen Aktionen und die mit diesen Aktionen verbundenen Ergebnisse in einer Situation beschrieben.

3.6 Informationen

Nach der Beschreibung der Entscheidungssituationen findet eine Fokussierung auf das Entscheidungsfeld und damit die notwendigen Informationen statt. Dazu ist jede einzelne Entscheidungssituation zu analysieren, inwieweit dort welche Informationen benötigt werden. Nach Klärung dieser Frage ist zu spezifizieren, woher die Informationen kommen. Dies geschieht durch die Zuweisung der Informationen zu Speicherorten. Demnach ist der wichtige Aspekt an dieser Stelle nicht der Ort bzw. die Situation der Informationsverwendung, was zuvor bei dem Einsatz in den einzelnen Entscheidungsprozessen relevant war. Dagegen ist festzulegen, wo die Informationsobjekte in ständig aktualisierter Form vorliegen und von den nachfragenden selbststeuernden logistischen Objekten des Systems abgerufen werden können. Beispiele für solche Speicherorte sind wiederum selbststeuernde logistische Objekte oder auch Altsysteme, die über Schnittstellen angebunden sind.

3.7 Kommunikation

Auf Basis der Prozesse, Entscheidungen, gegenseitiger Verknüpfungen und der Festlegung der Informationsquellen ist nun die Kommunikation zu modellieren. Dabei sind zwei Hauptaspekte zu unterscheiden, zum einen die Kommunikationsprozesse und zum anderen die ausgetauschten Nachrichten. Die notwendigen Kommunikationsprozesse werden aus den bestehenden Modellen abgeleitet. Für jede Entscheidungssituation sind der Entscheider, die notwendigen Informationsobjekte sowie die Informationsträger identifiziert. Aus dem logisch-zeitlichen Aufbau der einzelnen Prozesse ergibt sich dann auch der Aufbau der Interaktionen. Im Fall einer simplen Abfrage kann ein Interaktionsprotokoll lediglich bestehend aus einer Anfrage und der zugehörigen Antwort, gegebenenfalls mit entsprechenden Bestätigungen, definiert werden. Aufwändigere Protokolle sind für Verhandlungen zwischen Systemelementen notwendig, welche sich aus den Verknüpfungen und Abhängigkeiten der bei den einzelnen Elementen ablaufenden Prozesse ergeben.

3.8 Szenario

Im abschließenden Schritt der Modellierung werden die konkreten Szenariendaten erfasst. Zu den während der vorhergehenden Schritte definierten Klassen sind sämtliche vorhandenen Instanzen zu sammeln und zu dokumentieren, um die Voraussetzungen für die nachfolgende Teilphase der Simulation und in letzter Konsequenz für die Lauffähigkeit des Systems zu schaffen.

Das vollständige Vorgehensmodell definiert zu diesen hier skizzierten Schritten durchzuführende Aktivitäten und dabei zu erzielende Ergebnisse. Darüber hinaus werden zur Unterstützung der Aktivitäten Methoden und Werkzeuge vorgeschlagen. Zu diesen gehören erstens die erwähnte Notation, zweitens Modellierungskonventionen in Form von Konstruktions- und Konsistenzregeln sowie drittens ergänzende für die einzelnen Schritte ausgewählte Hilfsmittel wie beispielsweise das Entscheidungsmodell. Komplettiert werden die einzelnen Schritte noch durch Hinweise auf die möglicherweise in einem Schritt initiierten und in Bild 2 angedeuteten Iterationen, welche in Verweisen zu vorhergehende Schritten des Vorgehensmodells bestehen.

4 Werkzeug-Unterstützung: ALEM-T

Für die in den vorherigen Abschnitten dargestellte Modellierungsmethode wurde eine Modellierungssoftware unter Nutzung der Model Driven Architecture (MDA, vgl. OMG (2007a) und Assmann et al. (2005)) entwickelt.

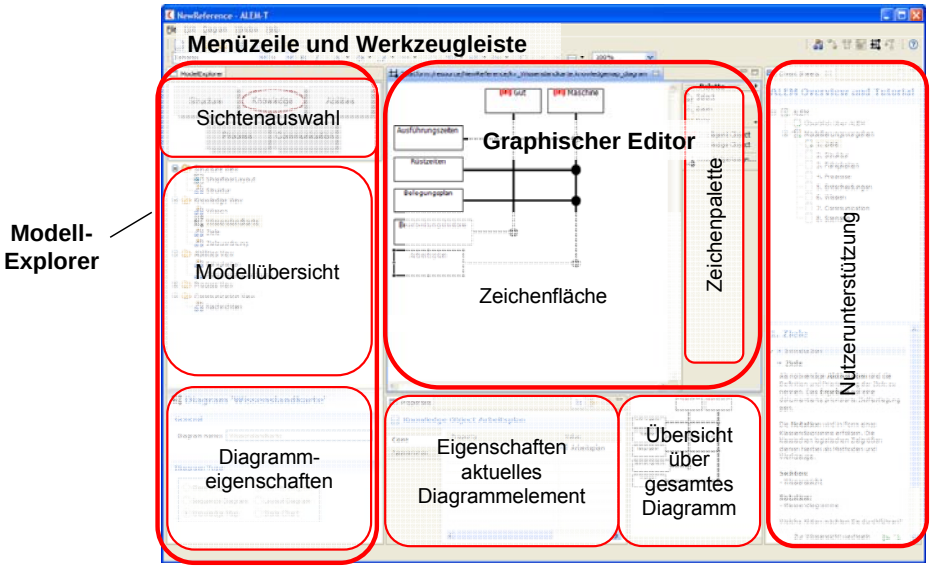

Abbildung 3: Screenshot der Modellierungssoftware ALEM-T mit Darstellung der wichtigsten Oberflächenbestandteile.

Diese Software wurde in Java entwickelt und basiert auf der Nutzung von Eclipse (http://www.eclipse.org/) als sog. Rich Client Platform (RCP, vgl. McAffer und Lemieux (2005)). Diese technologische Basis ermöglicht eine modulare, weitgehend plattformunabhängig ablauffähige Software mit einer „reichhaltigen Oberfläche“ (rich client). Der Begriff Rich Client ist hierbei so zu verstehen, dass man in plattformunabhängiger Art und Weise Programme erstellen kann, die sehr viel umfangreichere Möglichkeiten zur Programm- und insbesondere Oberflächengestaltung bieten als z.B. web-basierte Lösungen, mit denen sich zwar ebenfalls eine Plattformunabhängigkeit erreichen lässt, jedoch nur unter Beschränkung

auf sehr reduzierte Möglichkeiten insbesondere zur Oberflächengestaltung.

Darüber erleichtert die Verfügbarkeit insbesondere der beiden Frameworks EMF (Eclipse Modelling Framework, vgl. Budinsky et al. (2003)) und GMF (Graphical Modelling Framework, vgl. Eclipse Foundation (2007)) die Entwicklung deutlich. Die beiden Frameworks ermöglichen die modellgetriebene Entwicklung graphischer Editoren für domänenspezifische Modelle und bieten Code-Generatoren, mit deren Hilfe ein Großteil der Routine-Programmiertätigkeiten zur Erstellung derartiger Editoren automatisiert werden kann. EMF stellt Möglichkeiten bereit, aus einem graphisch modellierbaren, konzeptionellen (Meta-) Modell weitestgehend automatisiert Java-Quellcode zu erzeugen, mit dem dieses Modell bearbeitet und in einem XML-Format gespeichert werden kann. Mit GMF lassen sich auf Basis eines solchen EMF-Modells weitere Modelle erstellen, mit denen u.a. die graphische Darstellung der einzelnen Modellelemente graphischen festgelegt werden kann. Ein Code-Generator erzeugt aus diesen verschiedenen Teilmodellen automatisch den Quellcode für ein Eclipse-Plugin, das einen graphischen Editor für das konzeptionelle EMF-Modell bietet. Sowohl Eclipse als auch EMF und GMF sind als Open Source-Software verfügbar, was eine sehr große Flexibilität ermöglicht.

Abbildung 3 zeigt einen Screenshot der entwickelten Modellierungssoftware mit Darstellung der grundlegenden Bildschirmaufteilung. Im linken oberen Teil findet sich in der Sichtenauswahl das in Abschnitt 2 kurz vorgestellte und in Scholz-Reiter et al. (2007) näher beschriebene Sichtenkonzept wieder. Durch Auswahl der jeweils relevanten Sicht erfolgt eine Einschränkung zur Darstellung der für die jeweils relevanten Modellaspekte nötigen Funktionalitäten. Im Hauptteil dargestellt, findet sich ein graphischer Editor für das gerade geöffnete Diagramm. Diese Editoren werden als separate Module, das heißt als Eclipse-Plugins, implementiert und stellen jeweils die Funktionen zur Modellierung eines der benötigten Diagrammtypen zur Verfügung. Das gerade in Abbildung 3 geöffnete Diagramm zeigt ein einfaches Beispiel einer Wissenslandkarte (vgl. Allweyer und Scheer (1998)) zur Visualisierung der statischen Wissensverteilung im System. Der rechte, mit „Nutzerunterstützung“ bezeichnete, Teil der Programmoberfläche leitet den Modellierer in Form eines Tutorials an und stellt das Vorgehensmodell (siehe Abschnitt 3) im Überblick dar. Neben der Möglichkeit, sich kurz über die im jeweiligen Schritt des Vorgehens durchzuführenden Aktivitäten und die zu verwende Notation zu informierten, enthält das Tutorial auch interaktive Elemente, etwa um bestimmte Aktivitäten direkt aus dem Tutorial heraus anzustoßen (z.B. das Aktivieren eine bestimmten Sicht) oder auch eine detailliertere Beschreibung der Schritte des Vorgehensmodells aufzurufen.

5 Fazit

Der Beitrag untersuchte die Modellierung selbststeuernder logistischer Prozesse und stellte nach einer kurzen Problembeschreibung die Modellierungsmethode ALEM vor, wobei insbesondere das Vorgehensmodell und die entwickelte Modellierungssoftware näher beschrieben wurden.

Die zukünftigen Forschungsaufgaben bestehen neben einem Ausbau des dem Modellierers unterstützenden Referenzmodells u.a. in einer Erweiterung der unterstützten Phasen im Systementwicklungszyklus (Abbildung 1) und hierbei insbesondere in der Automatisierung der

Phasenübergänge. Hierzu sollen zur Entwicklung der Modellierungssoftware bereits angewandten Prinzipien zur modellgetriebenen Softwareentwicklung ebenfalls auf die Entwicklung des logistischen Steuerungssystems übertragen und dem Modellierer möglichst automatisiert z.B. eine Transformation des Prozessmodells aus der Phase „Spezifikation“ in ein Simulationsmodell ermöglicht werden.

Darüber hinaus soll die Modellierungsmethode und das entstandene -werkzeug in der betrieblichen Praxis erprobt werden.

Danksagung

Dieser Beitrag entstand im Rahmen des Teilprojekts „Adaptive Geschäftsprozesse - Modellierung und Methodologie“ des von der Deutschen Forschungsgemeinschaft (DFG) geförderten Sonderforschungsbereichs 637 „Selbststeuerung logistischer Prozesse – Ein Paradigmenwechsel und seine Grenzen“ (SFB 637).

Literatur (Literatur)

Assmann, U.; Rensink, A.; Aksit, M. (Hrsg.) (2005) Model Driven Architecture: Foundations and Applications, LNCS 3599. Berlin: Springer.

Allweyer, Th.; Scheer, A.-W. (Hrsg.) (1998) Adaptive Geschäftsprozesse: Rahmenkonzept und Informationssysteme. Wiesbaden: Gabler.

Babiceanu, R.F.; Chen, F.F. (2006) Development and Applications of Holonic Manufacturing Systems: A Survey. In: Journal of Intelligent Manufacturing, 17 [1], S. 111-131.

Bresciani, P.; Perini, A.; Giorgini, P.; Giunchiglia, F., Mylopoulos, J. (2004) Tropos: An Agent-Oriented Software Development Methodology, Autonomous Agents and Multi-Agent Systems. In: Autonomous Agents and Multi-Agent Systems 8 [3], S. 203-236.

Budinsky, F. et al. (2003) Eclipse Modeling Framework: a developer’s guide. Boston, USA: Addison Wesley.

Bussmann, S.; Jennings, N.R.;Wooldridge, M. (2004) Multiagent Systems for Manufacturing Control. A Design Methodology. Berlin: Springer.

Das, R.; Harrop, P. (2001) The Internet of Things: Massive new markets for automated location, tracking, authentication and barcode replacement. Total Asset Visibility. Cambridge, UK: IDTechEx.

Eclipse Foundation (2007) Eclipse Graphical Modeling Framework (GMF) http://www.eclipse.org/gmf/. Letzter Abruf: 2007-11-10.

Finkenzeller, K. (2003) RFID Handbook, 2nd edn. West Sussex, England: Wiley.

Fleisch, E. (2005) Die betriebswirtschaftliche Vision des Internets der Dinge. In: Fleisch, E.; Mattern, F. (Hrsg.): Das Internet der Dinge – Ubiquitous Computing und RFID in der Praxis, S. 3–38. Berlin: Springer.

Heinrich, C. (2005) RFID and beyond: growing your business through real world awareness. Indianapolis, USA: Wiley Publishing.

Haberfellner, R.; Daenzer, W.F.; Huber, F. (Hrsg.) (2002) Systems Engineering, 11. Aufl. Zürich: Orell Füssli.

Hülsmann, M.; Windt, K. (Hrsg.) (2007) Understanding Autonomous Cooperation & Control in Logistics – The Impact on Management, Information, Communication and Material Flow. Berlin: Springer.

Laux, H. (2005) Entscheidungstheorie, 6. Aufl. Berlin: Springer.

McAffer, J.; Lemieux, J.-M. (2005) Eclipse Rich Client Platform: designing, coding, and packaging Java applications. Upper Saddle River, USA: Addison-Wesley.

Monostori, L.;Váncza, J.; Kumara, S.R.T. (2006) Agent-Based Systems for Manufacturing. In: CIRP Annals 55 [2], S. 697-720.

OMG (Object Management Group) (2007) Unified Modeling Language Specification, Version 2.1.1. http://www.uml.org/. Letzter Abruf: 2007-09-10.

OMG (Object Management Group) (2007a) Homepage of the OMG Model Driven Architecture Initiative. http://www.omg.org/mda/. Letzter Abruf: 2007-09-10.

Patig, S. (2001) Flexible Produktionsfeinplanung mit Hilfe von Planungsschritten: ein Planungsansatz zum Umgang mit Störungen bei der Produktion. Frankfurt a.M.: Lang-Verlag.

Philipp, T.; de Beer, C.; Windt, K.; Scholz-Reiter, B. (2007) Evaluation of Autonomous Logistic Processes – Analysis of the Influence of Structural Complexity. In: Hülsmann, M.; Windt, K. (Hrsg.): Understanding Autonomous Cooperation and Control in Logistics – The Impact on Management, Information, Communication and Material Flow, S. 303-324. Berlin: Springer.

Scholz-Reiter, B.; Kolditz, J.; Hildebrandt, T. (2006) UML as a Basis to Model Autonomous Production Systems. In: Cunha, P.F.; Maropoulos, P. (Hrsg.): Digital Enterprise Technology. Perspectives and Future Challenges. Proceedings of the 3rd CIRP Sponsored Conference on DET, S. 553-560. Berlin: Springer.

Scholz-Reiter, B.; Kolditz, J.; Hildebrandt, T. (2007) Engineering autonomously controlled logistic systems. In: International Journal of Production Research 2007, accepted for publication, to appear.

Wiendahl, H.-P. (2005) Betriebsorganisation für Ingenieure, 5. Aufl. München, Hanser.

Zambonelli, F.; Jennings, N.R.; Wooldridge, M. (2003) Developing Multiagent Systems: The GAIA Methodology. In: ACM Trans. on Software Engineering and Methodology 12 [3], S. 317–370.