von tim weilkiens und
claudia schröder

mehr zum thema:
www.oose.de
www.port-of-kiel.de/

PRAXISBERICHT: ERSTE PROJEKTERFAHRUNGEN MIT DER UML 2.0

Das in dem Artikel beschriebene Projekt hat das Ziel, ein ausschreibungsfähiges Pflichtenheft für das Hafenlogistik- und Informationssystem HALIS des Seehafens Kiel zu erstellen. Neben der Vorgehensweise im Projekt wird die Gliederung des zu erstellenden Pflichtenhefts vorgestellt. Dabei wird jeweils erläutert, welche Konzepte der UML dafür verwendet wurden und welche Erfahrungen dabei gesammelt werden konnten.

Das im November 2003 gestartete Projekt bestand auftragnehmerseitig aus drei Personen und war zum Festpreis und mit festem Endtermin durchzuführen. Die drei Projektmitarbeiter waren bereits vor dem Projekt mit der UML 2.0 und mit der in [Oes03] und [Oes04] beschriebenen Methodik vertraut.

Neben Einleitung, Zieldefinition und dem üblichen Drumherum besteht das Pflichtenheft unter anderem aus folgenden Elementen:

- Systemkontext (Abb. 1),
- fachliche Architektur,
- fachliche Komponenten (Abb. 2 und 3),
- Fachklassenmodell,
- Schnittstellen und externe Systeme,
- Akteurmodell,
- Beschreibung externer Schnittstellen,
- technische Architektur,
- Kern-Geschäftsprozesse,
- Geschäftsanwendungsfälle (Abb. 4),
- Systemanwendungsfälle, gegliedert nach fachlichen Subsystemen,
- Abhängigkeiten funktionaler von nicht-funktionalen Anforderungen,
- Beschreibung nicht-funktionaler Anforderungen und
- Ausbaustufenplan.

Vorgehensweise

Nachdem alle Interessenshalter (*Stakeholder*) gefunden waren, analysierten wir vorhandene Unterlagen und Informationen. Mit Hilfe von Interviewreihen und kurzen Workshops wurde die Systemgrenze skizziert und die Soll-Geschäftsprozesse analysiert und modelliert. Diese intensive Vorgehensweise nahm ein gutes Drittel unseres Budgets in Anspruch – Arbeit, die sich lohnte: Sowohl knapp 60% der Systemanwendungsfall-Kandidaten, als auch nicht-funktionale Anforderungen wurden dabei identifiziert und dokumentiert. Im Anschluss konnten diese Kandidaten in Einzelgesprächen mit dem Auftraggeber detailliert werden. Die Dokumentation des Pflichtenheftes geschah fortlaufend und wurde schrittweise mit dem Auftraggeber abgestimmt.

Alle Abbildungen des Pflichtenheftes wurden durchgängig mit der UML 2.0 modelliert. Im Folgenden stellen wir einige interessante Teilbereiche vor.

HALIS ist das Hafenlogistik- und Informationssystem für den RoRo-Verkehr im Ostuferhafen und beinhaltet die Optimierung der Stellplatzverwaltung sowie der Datenschnittstellen zu den Hafenkunden und Behörden.

Systemkontext

In nahezu allen Projekten wird von dem „System" gesprochen, das entwickelt werden soll. Auch in Vorgehensmodellen oder in der UML-Spezifikation taucht immer wieder der Begriff „System" auf. Die Antwort auf die Frage „Was ist das System?" sollte von jedem Projektmitarbeiter eindeutig beantwortet werden können. Dem ist in der Praxis jedoch leider selten so.

die autoren

Tim Weilkiens (E-Mail: tim.weilkiens@oose.de) arbeitet bei der Firma oose.de als Trainer und Berater. Seine Schwerpunkte sind die objektorientierte Analyse und Geschäftsprozessmodellierung sowie die UML.

Claudia Schröder (E-Mail: claudia.schroeder@system-analyst.de) arbeitet als selbstständige Systemanlytikerin. Ihre Schwerpunkte liegen im Bereich Anforderungs-, Geschäftsprozessmodellierung und Entwicklungsmethodik.

Im Kern sind sich alle einig, aber die Grenze zu umliegenden Systemen ist häufig unscharf – insbesondere bei eingebetteten bzw. verteilten Systemen. Dabei ist diese Grenze sehr wichtig. Sollen beispielsweise Anwendungsfälle beschrieben werden, muss die Unterscheidung zwischen „innerhalb" und „außerhalb" klar sein. Eine der ersten Aktivitäten ist somit die Analyse des Systemkontextes. Dieser beschreibt das Umfeld des zu modellierenden Systems und somit die Grenze zwischen drinnen und draußen. Dabei ist es eine reine Projektentscheidung, was zum eigenen System gehört und was als Fremdsystem bzw. Benutzer betrachtet wird. Das Vorgehensmodell kann hier keine Vorgaben machen.

Abbildung 1 zeigt den Systemkontext des *Hafenlogistiksystems (HALIS)*. Das Systemkontextdiagramm ist kein vordefiniertes Diagramm der UML, aber in dieser Form erlaubt. Es zeigt UML-Akteure, die Assoziationen zu einer Komponente (hier das System HALIS) haben. Die Kompo-

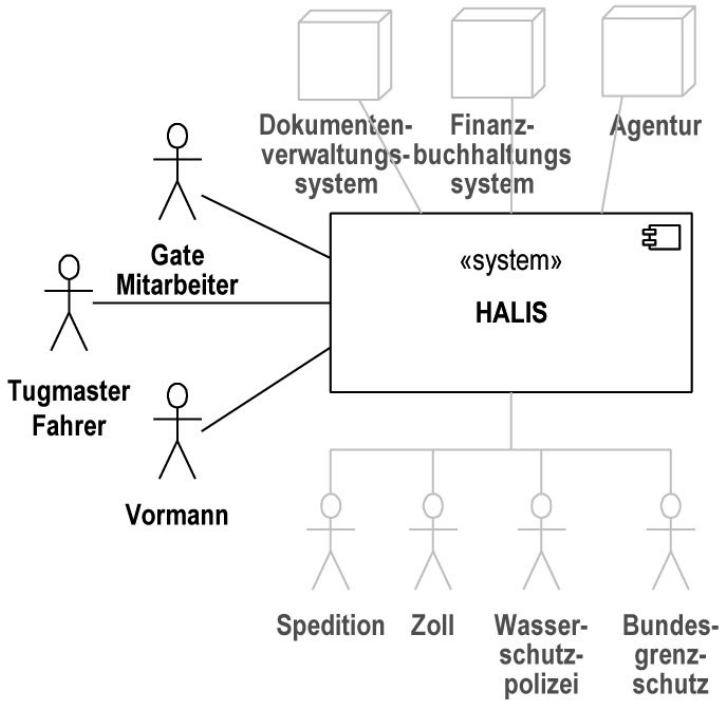

Abb. 1: Systemkontextdiagramm

nente wird mit dem Stereotyp «system» gekennzeichnet, um die Sonderstellung hervorzuheben. «system» ist kein Standardstereotyp der UML, sondern gehört zum UML-Profil, das oose.de in Analyseprojekten einsetzt.

Die grau gezeichneten Akteure wurden identifiziert, sind aber erst für die folgenden Ausbaustufen des Systems von Bedeutung. Auch diese Unterscheidung ist kein UML-Standard.

Fachliche Komponenten

In der Regel findet man bei der Kontextanalyse auch Systeme, die keine Fremdsysteme sind, sondern als Teil des eigenen Systems gesehen werden, sogenannte Subsysteme. Sie werden als Komponenten mit dem Stereotyp «subsystem» modelliert und stellen die oberste logische Struktur des Systems dar. «subsystem» ist ein Standardstereotyp der UML 2.0. In der UML 1.x war das Subsystem ein eigenes Modellelement. Das Gesamtbild wird im Laufe der Analyse zunehmend verfeinert.

Das Komponentendiagramm in Abbildung 2 zeigt die Zusammensetzung des Systems aus den identifizierten Subsystemen.

Mit dem Kompositionsstrukturdiagramm bietet die UML 2.0 eine neue Diagrammform sowie neue Modellelemente, um die interne Struktur einer Klasse oder Komponente detaillierter zu beschreiben. Abbildung 3 zeigt die interne Struktur von HALIS und das Zusammenspiel der Subsysteme. Die Darstellung ist noch sehr grob, aber für den geplanten Umfang der Systemanalyse ausreichend. Die Schnittstellen und ihre Beziehungen werden erst in der detaillierten Analyse bzw. im Design vollständig spezifiziert werden.

Beziehungen innerhalb einer Komponente oder Klasse lassen sich mit den Kompositionsstrukturdiagrammen der UML 2.0 detaillierter modellieren, als dies mit der UML 1.x möglich war. Schnittstellen können sowohl als genutzte als auch als bereitgestellte Schnittstellen dargestellt werden. *Ports* stellen mit den Schnittstellen kombiniert die Interaktionspunkte der jeweiligen Komponenten mit ihrer Umgebung dar. Die auf der Kante des Systems dargestellten *Ports* weisen auf die Interaktionspunkte zu den außerhalb von HALIS liegenden Fremdsystemen hin.

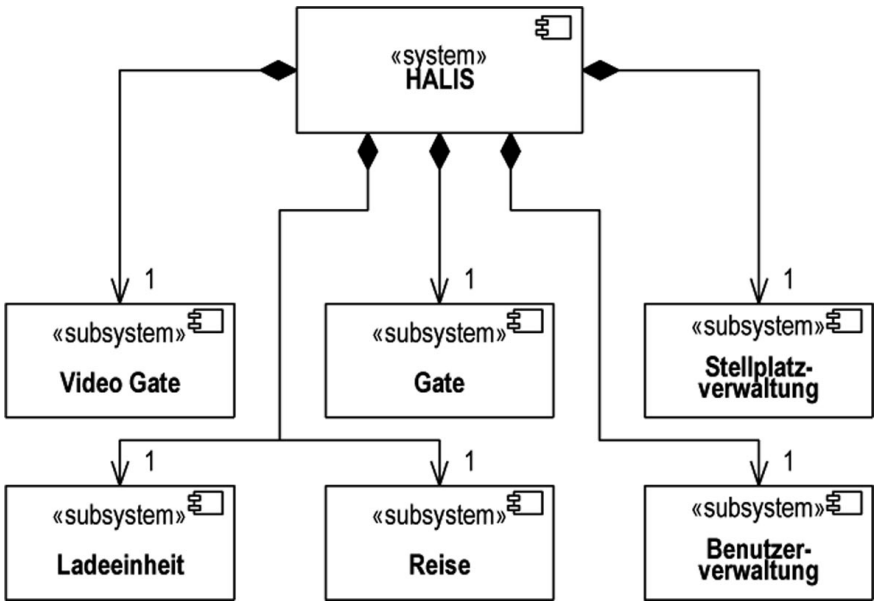

Abb. 2: Komponentendiagramm

Geschäftsprozesse & Co

Die Unterscheidung von Geschäftprozessen, Geschäfts- und Systemanwendungsfällen ist weiterhin kein Standard in der UML. Die UML kennt lediglich Anwendungsfälle. Wir haben die hier üblichen und notwendigen Erweiterungen verwendet, um verschiedene Arten von Anwendungsfällen zu unterscheiden:

- *Geschäftsanwendungsfälle* für Abläufe, die aus rein fachlicher und betriebswirtschaftlicher Sicht – d. h. unabhängig von einer systemtechnischen Umsetzung – beschrieben werden (Stereotyp: «business»). Im Detail haben wir weiterhin Kern- und unterstützende Geschäftsanwendungsfälle unterschieden, um den Beitrag zur betrieblichen Wertschöpfung herauszuarbeiten.
- *Systemanwendungsfälle*, die beschreiben, wie ein Anwendungsfall systemtechnisch umgesetzt wird. Diese Anwendungsfälle bleiben ohne

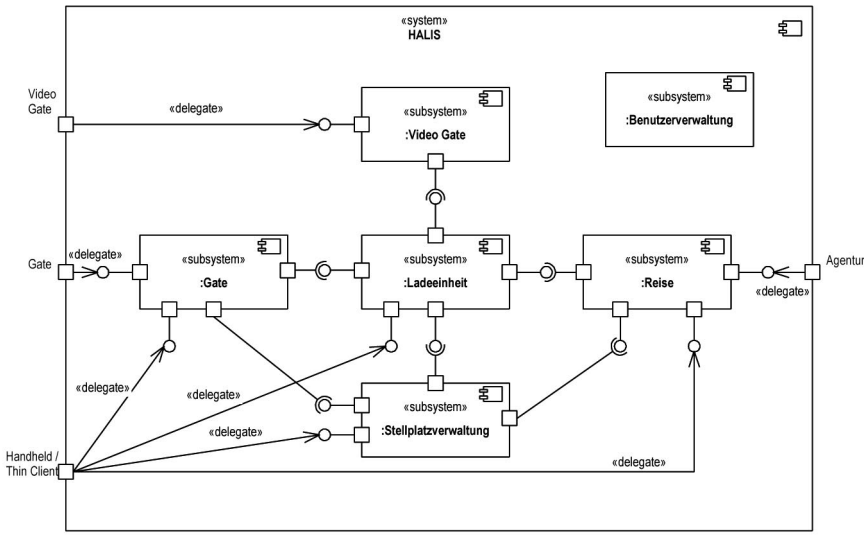

Abb. 3: Kompositionsstrukturdiagramm

Stereotyp, wir betrachten die Normalform der Anwendungsfälle als Systemanwendungsfälle.

- *Sekundäre Anwendungsfälle*, die gewöhnlich einzelne Schritte oder Teilabläufe darstellen, die mehrfach vorkommen und zur redundanzfreien Beschreibung separiert werden. Hier verwenden wir den Stereotyp «secondary».
- *Geschäftsprozesse* sind für uns Gruppierungen von Geschäftsanwendungsfällen und wurden mit dem Stereotyp «business process» versehen.

Zusätzlich haben wir aus methodischen Gründen klare Definitionen und Kriterien für die einzelnen Anwendungsfallarten und ihrer Beschreibung verwendet, beispielsweise die Konzepte „Auslöser", „Ergebnisse", und „zeitliche Kohärenz", wie sie in [Oes04] beschieben sind.

Geschäftsanwendungsfälle

Das Aktivitätsmodell der UML 2.0 ist nicht wie in der UML 1.x vom Zustandsmodell abgeleitet. Die Semantik entspricht mehr dem der Petri-Netze, was es deutlich erleichtert Abläufe zu beschreiben. Es gibt viele neue Elemente für die Aktivitätsmodellierung, die bei der ersten theoretischen Auseinandersetzung mit der UML 2.0 eher überflüssig und übertrieben oder nur für Spezialthemen geeignet scheinen. Bei der praktischen Anwendung zeigt sich aber durchaus, dass diese sehr sinnvoll sind.

Zu den neuen Elementen der UML 2.0-Aktivitäten zählen z. B. die Mengenverarbeitung (*expansion region*), Streams, Signale und unterbrechbare Aktivitätsbereiche. In unserem Projekt half beispielsweise die Mengenverarbeitung dabei den Verlade- bzw. Entladevorgang der einzelnen Ladeeinheiten darzustellen. Beim Import kommt ein Schiff mit vielen Ladeeinheiten an. Im weiteren Verlauf werden die Ladeeinheiten aber einzeln weiterverarbeitet. Ihnen wird ein Stellplatz zugeordnet, wo sie abgestellt und schließlich von einem Transportfahrzeug für den Landtransport wieder abgeholt werden (vgl. Abb. 4).

In dem Bild ist in der oberen Mengenverarbeitung mit dem Schlüsselwort parallel gekennzeichnet, dass alle Transporteinheiten einer Reise prinzipiell nebenläufig entladen und abgestellt werden können. Hingegen wird im unteren, mit iterative gekennzeichneten Bereich so vorgegangen, dass die Transporteinheiten nacheinander ausgecheckt werden müssen.

Die Darstellung als Streams (ausgefüllte Pins an den einzelnen Aktionen) drückt auf einfache Weise die fachliche Gegebenheit aus, dass die Transporteinheiten als kontinuierlicher Einheitenfluss von Bord gehen, dessen Gesamtkapazität von der Anzahl der Brücken, Transportfahrzeuge und Ähnlichem abhängt.

Mit den Aktivitätsdiagrammen der UML 2.0 können fachliche Abläufe sehr direkt abgebildet werden. Die Modelle werden nicht aus modellierungstechnischen Gründen verkompliziert, wie es oft in der UML 1.x der Fall war. Mengenverarbeitungen oder unterbrechbare Aktivitätsbereiche mit der UML 1.x zu modellieren führt beispielsweise zu sehr komplexen Diagrammen, obwohl die Fachlichkeit dahinter eher einfacher Natur ist.

Durch die Änderung der Semantik der ein- und ausgehenden Kanten (früher Transitionen) an den Aktionen sind nun vor und nach Aktionen häufig explizite Entscheidungen und Zusammenführungen notwendig, was die Diagramme etwas umfangreicher macht. Zudem erfordert die neue Semantik eine neue Denk- und Herangehensweise: Wer versucht, mit den gewohnten Entwurfsmustern Abläufe zu modellieren, beißt sich leicht die Zähne aus.

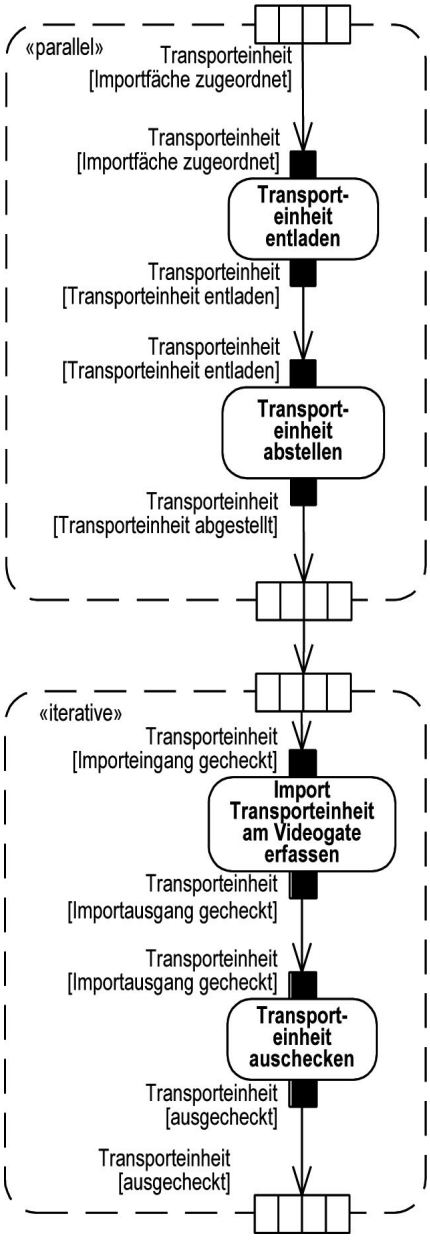

Abb. 4: Aktivitätsdiagramm mit Mengenverarbeitungsbereichen und Streams

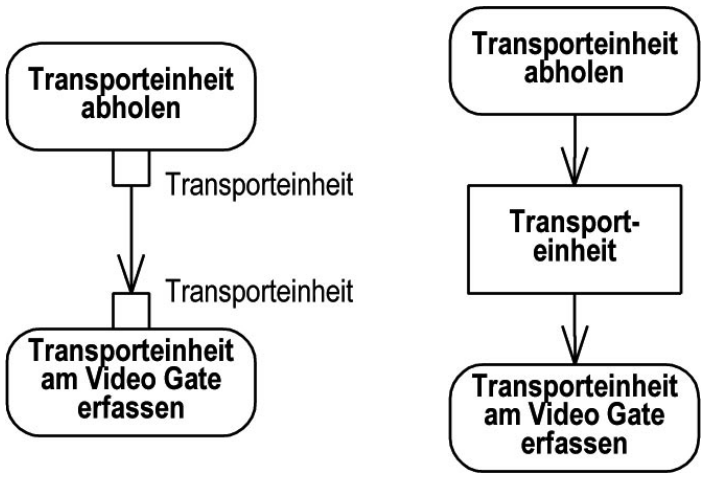

Abb. 5: Varianten für Kanten-Zusammenführung

Lässt man sich hingegen auf die neue, objekt- statt prozessorientierte Semantik ein, geht die Modellierung nach einer kurzen Umgewöhnungszeit leichter von der Hand.

Während der Aktivitätsmodellierung mit der UML 2.0 sind uns in der Praxis einige Stolpersteine aufgefallen – Konstrukte, die theoretisch sehr praktikabel erscheinen, aber in der Praxis dann doch zu Problemen führen. Hierzu zählen z.B. die Varianten der Objektfluss-Notation und Entscheidungs- bzw. Zusammenführungsnotation, die wir im Folgenden darstellen, oder die neue Semantik, nach der von einem Aktivitätsschritt ausgehende Kanten ein implizites Splitting, eingehende Kanten eine implizite Synchronisation sind.

Objektfluss-Notation

Abbildung 5 zeigt zwei alternative Notationen für den Objektfluss. Die linke Form zeigt die Pin-Notation, die neu in der UML 2.0 eingeführt wurde. Die rechte Form entspricht fast der Objektflussdarstellung der UML 1.x mit dem Unterschied, dass die Pfeile durchgezogen und nicht gestrichelt sind. Beide Darstellungen basieren auf demselben Modell. In der praktischen Arbeit am Pflichtenheft hat sich gezeigt, dass die rechte Form ungeeignet ist, da sie leicht zu formalen Fehlern in der Modellierung führt. Zudem führt es zu einer Mischform in den Diagrammen, da man nicht ausschließlich mit der rechten Darstellung auskommt und auch die linke Pin-Notation benötigt. Umgekehrt kann aber auf die rechte Form vollständig verzichtet werden – dies führt dann zu einer einheitlicheren Darstellung und erleichtert das Lesen der Abläufe.

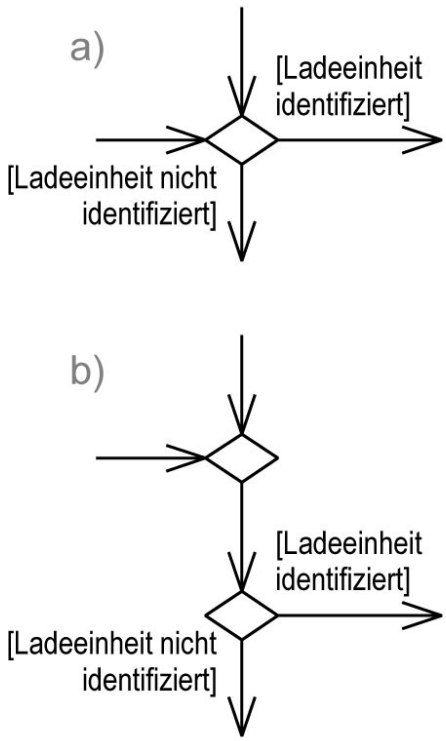

Abb. 6: Ablauf-Ende in Aktivitäten

Entscheidung und Zusammenführung

Eine andere Notationsform, die in der UML 2.0 als praktische Kurzschreibweise erlaubt ist, hat sich in der Praxis auch nicht bewährt. Folgt auf eine Zusammenführung unmittelbar eine Entscheidung, dürfen die Symbole auch verschmolzen werden (vgl. Abb. 6).

Auf den ersten Blick wirkt dies sinnvoll, da die rechte Darstellung kompakter ist und das Diagramm insgesamt übersichtlicher wird. Was der Leser aber wissen muss, ist, dass fachlich zuerst eine Zusammenführung der Abläufe stattfindet und dann die Entscheidung. Dies kann allerdings auch anders gelesen werden, nämlich so, dass zuerst die Entscheidung durchgeführt wird und dann die Zusammenführung erfolgt – diese Interpretation ist aber falsch! Genau dieses Missverständnis, das wir nicht erwartet hatten, ist in der praktischen Arbeit häufiger aufgetaucht und führte zu versteckten Fehlern und Missverständnissen.

Ablauf-Ende

Ein dringend notwendiges neues Konstrukt der Aktivitätsmodellierung ist das Ablauf-Ende. Es ermöglicht die Beendigung eines einzelnen Ablaufes, ohne die gesamte Aktivität zu beenden, wie es beim Aktivitätsendknoten der Fall ist. Die neue Semantik impliziert, dass es zu einem Splitting nicht zwingend eine zugehörige Synchronisation geben muss, wie es in der UML 1.x der Fall war. Das führte bei der Anwendung der UML 1.x häufig zu komplizierten Modellen (Abb. 7).

Sonstige Erfahrungen

Die Systemanwendungsfälle haben wir nach den fachlichen Subsystemen gegliedert. Sie wurden in Geschäftsanwendungs- und Systemanwendungsfälle eingeteilt und

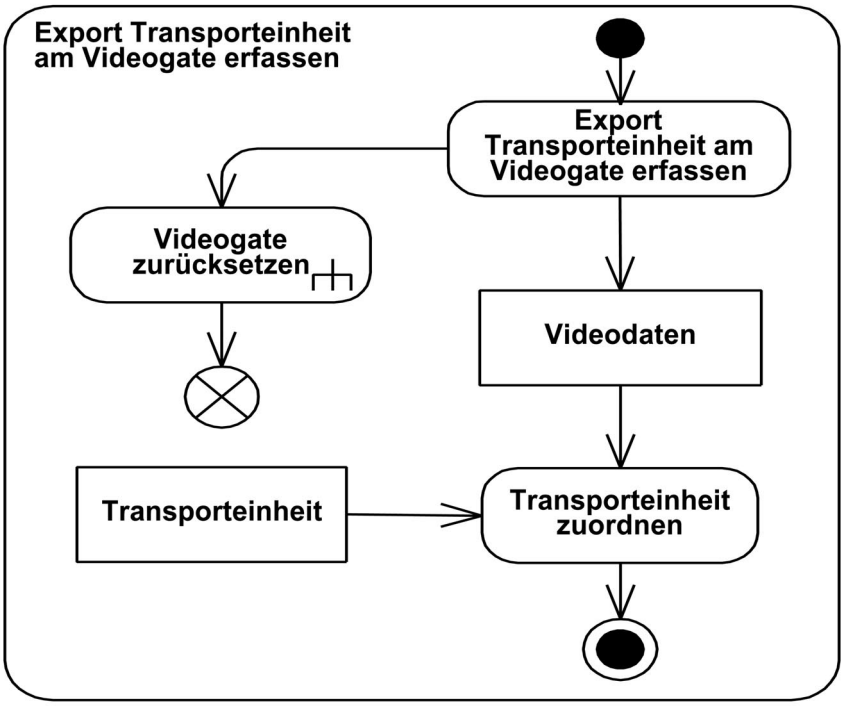

Abb. 7: Beispiel für ein Ablaufende.

mit Hilfe der oose.de-Templates (tabellarische Textschablonen) essenziell inklusive aller Aktionen und Kanten beschrieben.

Zwischen funktionalen und nicht-funktionalen Anforderungen existieren gewöhnlich verschiedene Abhängigkeiten, die wir durch eine einfache Beschreibungsmatrix anschaulich gelöst haben. Die gleichen Sachverhalte hätten wir auch in der UML mit Hilfe entsprechender Abhängigkeitsbeziehungen abbilden können. Die Darstellung in einer einfachen Tabelle erschien aber deutlich einfacher und zumindest für die gegenwärtige Phase des Projektes völlig ausreichend.

Bei der Modellierung der hier nicht näher beschriebenen Teile des Pflichtenheftes – wie das Fachklassenmodell, das Akteurmodell sowie die technische Architektur – haben wir keine besonderen Veränderungen der UML 2.0 genutzt. Wir verzichten hier daher auf die entsprechenden Details.

Fazit

Die neue Version der UML macht insgesamt einen sehr guten und praktikablen Eindruck. Mit den geänderten und neuen Modellierungselementen lässt sich die Fachlichkeit viel leichter und intuitiver abbilden bzw. lesen als in der Vorgängerversion.

Im Bereich der Klassenmodellierung gibt es wenige Änderungen – hier waren die früheren UML-Versionen unproblematisch. Kleinigkeiten, wie die erweiterte Darstellung der Navigationsrichtungen, sind hilfreich, allenfalls hätte die Vielzahl der verschiedenen Assoziationsvarianten vielleicht reduziert werden können, denn in der Praxis werden sie kaum verwendet. Das Kompositionsstrukturdiagramm ist eine Bereicherung für die verständliche Modellierung von Architekturaspekten.

Im Bereich der Dynamikmodellierung gab es die größten und wichtigsten Fortschritte. Neue Möglichkeiten wie Ausnahmekanten, Mengenverarbeitung, Streams und Petrinetz-Semantik haben sich aus unserer Sicht sofort bewährt, wir möchten nicht mehr darauf verzichten.

Rund um die Anwendungsfälle hat sich wenig getan. Die UML steht allerdings auch den gängigen Konventionen und Erweiterungen nicht im Weg.

Die Kritik, dass die UML 2.0 mit Modellierungsmöglichkeiten überfrachtet wurde, können wir nicht nachvollziehen. Die Sprache bietet einen großen Wortschatz an, aber niemand ist verpflichtet, jedes Wort zu verwenden. Für ungeübte UML-Anwender ist der große Umfang in der ersten Zeit eine kleine Hürde.

Die Änderungen gegenüber der Vorgängerversion UML 1.5 sind sehr groß und es ist eine x.0 Version. Entsprechend ist damit zu rechnen, dass die Praxis neben gelösten Problemen auch neue Probleme entdecken wird. Einige davon haben wir Ihnen in diesem Artikel gezeigt. Eine UML 2.1 wird sicherlich schnell angebracht sein.

Spannend ist die Frage, wie die Werkzeughersteller die UML 2.0 umsetzen werden. Gute Konzepte könnten versanden, wenn sie nicht praktikabel in gängigen Tools angewandt werden können. Wir haben für unser Projekt „MS Visio" mit eigens angepassten UML-2.0-Schablonen verwendet. Bei einer beabsichtigen systematischen Weiterverwendung der Modelle wäre aber sicher ein anderes Werkzeug unser Favorit gewesen. ■

Literatur & Links

[Oes03] B. Oestereich, C. Weiss, T. Weilkiens, C. Schröder, A. Lenhard, Objektorientierte Geschäftsprozessmodellierung mit der UML, dpunkt-Verlag, Heidelberg, 2003

[Oes04] B. Oestereich, Objektorientierte Softwareentwicklung, Analyse und Design mit der UML 2.0, 6. Auflage, Oldenbourg Wissenschaftsverlag, München, 2004.

[OMG03] Object Management Group, UML 2.0 Superstructure Specification, August 2003, siehe: www.omg.org/cgi-bin/doc?ptc/03-08-02