2009

MODELLGETRIEBENE ENTWICKLUNG VON SERVICEORIENTIERTEN ARCHITEKTUREN

Rainer Bernhard
Siemens AG Industry Sector

Bernd U. Jahn
Otto-Friedrich-Universität Bamberg

MODELLGETRIEBENE ENTWICKLUNG VON SERVICEORIENTIERTEN ARCHITEKTUREN

Rainer Bernhard[1], Bernd U. Jahn[2]

Kurzfassung
Die Analyse von Geschäftsprozessen gilt als wichtige Voraussetzung für die Entwicklung einer serviceorientierten Architektur. Der Übergang vom Geschäftsprozess zum Entwurf von Diensten und deren Orchestrierung in Workflows stellt auf Grund der unterschiedlichen Zielsetzungen der jeweiligen Betrachtungsebenen eine große Herausforderung dar. Um diese semantische Lücke zwischen der fachlichen und softwaretechnischen Ebene zu schließen, muss eine durchgängige Modellierung über mehrere Abstraktionsebenen hinweg durchgeführt werden. Der in diesem Artikel beschriebene modellgetriebene Ansatz, der auf dem semantischen Objektmodell beruht, soll diese Durchgängigkeit gewährleisten.

1. Einleitung

Nach den technologiegetriebenen Diskussionen über serviceorientierte Architekturen (SOA) der vergangenen Jahre befassen sich immer mehr Autoren mit dem gesamten Entwicklungsprozess einer SOA ([2], [4], [16] und [17]). Die Methoden stützen sich dabei auf die Konzepte und Werkzeuge der modellgetriebenen Softwareentwicklung (*Model Driven Software Development*, MDSD). Durch die Unterteilung des Modellsystems in verschiedene Teilmodellsysteme trägt die MDSD zur Komplexitätsbewältigung bei und gewährleistet die durchgängige Modellierung über alle Abstraktionsebenen hinweg. Dieser Artikel stellt einen modellgetriebenen Ansatz für die durchgängige Analyse und das Design serviceorientierter Architekturen vor. Der Ansatz integriert dazu etablierte geschäfts- und objektorientierte Modellierungsansätze mit serviceorientierten Konzepten.
SOA bauen auf die Erfahrungen mehrerer Jahrzehnte der Softwareentwicklung auf [9, S. 13 - 25] und stützen sich dabei auf bewährte Konzepte aus der objektorientierten Programmierung, der Komponentenorientierung, sowie der Entwicklung von verteilten Systemen. Diese Konzepte stellen einen geeigneten Ausgangspunkt für die Analyse und den Entwurf serviceorientierter Architekturen dar. Deshalb wird die objekt- und geschäftsprozessorientierte SOM-Methodik [7], die durch ihre formalen Modelleigenschaften und durch die Integration der Modelle über Sichten und Ebenen hinweg eine durchgängige Modellierung betrieblicher Systeme gewährleistet, als Grundlage für diese Arbeit verwendet. Als Notation für die Verhaltenssicht einer SOA kommt die Business Process Modeling Notation (BPMN) zum Einsatz. Die Struktursicht wird mit der Unified Modeling

[1] Siemens AG Industry Sector, Lina-Ammon-Str. 9, D-90471 Nürnberg
[2] Otto-Friedrich-Universität Bamberg, Feldkirchenstr. 21, D-96045 Bamberg

Language (UML) modelliert. Die Dienste werden als Web Services implementiert und mit der Business Process Execution Language (BPEL) orchestriert.
Nach einer kurzen Einführung in die MDSD und SOM-Methodik stellt Kapitel 3 die verschiedenen Modellebenen und deren Beziehungen zueinander vor. Die Modellbildung wird in Kapitel 4 mit Hilfe eines Fallbeispiels demonstriert. Der Artikel schließt mit einem Verweis auf vergleichbare Ansätze und einer Zusammenfassung der Ergebnisse.

2. Grundlagen

Ein zentrales Hilfsmittel zur Komplexitätsbewältigung bei der Softwareentwicklung stellt das Prinzip der Abstraktion und Strukturierung dar [11, S. 263]. Modelle folgen diesem Prinzip: Durch zielgerichtete Abbildung realer Sachverhalte enthalten Modelle lediglich die zur Zielerreichung notwendigen Merkmale des realen Systems. Darüber hinaus werden Modelle auch als Kommunikationsmittel zwischen Auftraggeber und Auftragnehmer, innerhalb des Entwicklungsteams oder zu Dokumentationszwecken eingesetzt. Die modellgetriebene Softwareentwicklung als spezieller Ansatz erweitert die Modellnutzung: Neben reinen Dokumentations- und Kommunikationsaufgaben werden Modelle im gesamten Entwicklungsprozess zum Generieren von Programmcode, Testen sowie für das Deployment von Anwendungssystemen eingesetzt [8, S. 1f]. Die Object Management Group (OMG) hat in diesem Zusammenhang durch die Veröffentlichung zahlreicher Standards wie der Unified Modeling Language (UML) oder der Meta Object Facility (MOF) einen großen Anteil an der Verbreitung des modellgetriebenen Entwicklungsansatzes. Seit 2001 arbeitet die OMG zudem an der Model Driven Architecture (MDA), einem Rahmenwerk, das diese Standards zu einer modellgetriebenen Entwicklungsmethodik zusammenführt.
Grundlage für den in dieser Arbeit vorgestellten Ansatz ist die modellgetriebene SOM-Methodik für die Modellierung betrieblicher Systeme ([6] und [7]). Diese setzt sich aus dem objekt- und geschäftsprozessorientierten Modellierungsansatz SOM (Semantisches Objektmodell), der SOM-Unternehmensarchitektur und dem SOM-Vorgehensmodell (V-Modell) zusammen. Die Unternehmensarchitektur unterscheidet drei Modellebenen (vgl. [7, S. 192f.]):
Der *Unternehmensplan* spezifiziert die Aufgaben- und Aufgabenträgerebene (A- und AT-Ebene) des betrieblichen Systems aus der Außenperspektive. Die Modellierung stützt sich hierbei auf die Metapher einer globalen Unternehmensaufgabe.
Das *Geschäftsprozessmodell* (GP-Modell) spezifiziert die A-Ebene des betrieblichen Systems aus Innenperspektive und beschreibt das Lösungsverfahren für die im Unternehmensplan definierte globale Unternehmensaufgabe. Die Modellbildung beruht auf der Metapher eines verteilten Systems.
Das *Ressourcenmodell* spezifiziert die AT-Ebene des betrieblichen Systems aus Innenperspektive. Als Aufgabenträger für die Durchführung der Aufgaben stehen Personal, Anwendungssysteme (AwS) sowie Anlagen und Maschinen zur Verfügung. Als zugrunde gelegte Metapher dient der Systemtyp eines sozio-technischen Systems: Mensch und Maschine führen betriebliche Aufgaben nach dem Partner-Partner-Modell durch.
Gemäß V-Modell [7, S. 195f.] werden die drei Ebenen von oben nach unten durchlaufen. In zwei Sichten werden auf jeder Ebene die Systemmerkmale Struktur und Verhalten erfasst. Innerhalb jeder Ebene müssen die Sichten aufeinander abgestimmt werden. Die Modelle der verschiedenen Ebenen müssen miteinander korrespondieren (vgl. Abbildung 1), sodass ein konsistentes Modellsystem entsteht. Ausgangspunkt der Modellierung ist der Unternehmensplan, der *Diskurswelt* und *Umwelt* voneinander abgrenzt, sowie die zugehörigen Leistungsbeziehungen definiert. Die Diskurswelt bezeichnet dabei den relevanten Ausschnitt der betrieblichen Realität. Aufgabenziele, Strategien und Rahmenbedingungen für die Umsetzung der Leistungsbeziehungen sind im Zielsystem zu definieren, welches ebenfalls Bestandteil des Unternehmensplans ist. Die Struktursicht des Geschäftsprozessmodells wird im *Interaktionsschema* (IAS) modelliert. Die im Unternehmensplan

identifizierten Diskurswelt- und Umweltobjekte stellen betriebliche Objekte im IAS dar. Transaktionen verbinden betriebliche Objekte und repräsentieren die zugehörigen Leistungsbeziehungen aus dem Unternehmensplan. Das *Vorgangs-Ereignis-Schema* (VES) beschreibt die korrespondierende Verhaltenssicht in Form eines ereignisgesteuerten Ablaufs von Aufgaben. Die Detaillierung des Modells erfolgt durch sukzessive Zerlegung des initialen Geschäftsprozessmodells. Anwendungssysteme repräsentieren maschinelle Aufgabenträger zur Durchführung automatisierter (Teil-) Aufgaben von Geschäftsprozessen. Die Spezifikation der AwS erfolgt mit Hilfe des *konzeptuellen Objektschemas* (KOS) und des *Vorgangsobjektschemas* (VOS).

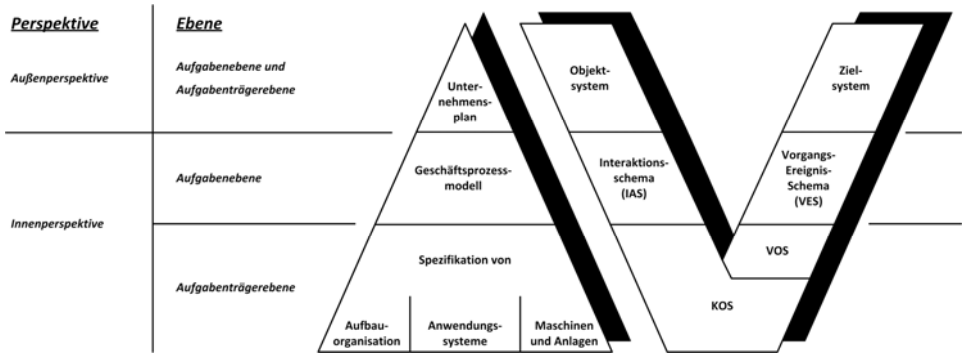

Abbildung 1: SOM-Unternehmensarchitektur und V-Modell [7, S. 193 und 195]

Die fachliche Spezifikation von AwS der SOM-Methodik im KOS und VOS bildet den Ausgangspunkt für den softwaretechnischen Entwurf [6, S. 219]. Amberg [1] und Malischewski [12] haben ein mit der SOM-Methodik abgestimmtes Softwarearchitekturmodell konzipiert (vgl. linke Hälfte von Abbildung 2), das die vollständige Übernahme der Modellierungsergebnisse in den softwaretechnischen Entwurf ermöglicht:
Das *konzeptuelle Objektschema* (KOS) besteht aus einer Menge von konzeptuellen Objekttypen (KOT), die miteinander in Beziehung stehen. Durch die Modellierung von Daten und Funktionen in Form von KOTs stellt das KOS eine objektorientierte Erweiterung der konzeptuellen Datenmodellierung dar [6, S. 212].
Das *Vorgangsobjektschema* (VOS) beinhaltet eine Menge von Vorgangsobjekttypen (VOT), die miteinander in Beziehung stehen, und gemeinsam die zweite Sicht der Spezifikation bilden. Ein VOT beschreibt das Lösungsverfahren für die Durchführung einer Aufgabe, die auch als Vorgang bezeichnet wird und aus einer Folge von Aktionen besteht. Automatisierbare Aktionen können durch Operatoren eines KOTs realisiert werden. Dadurch lässt sich das Lösungsverfahren einer Aufgabe als eine Abfolge von Methodenaufrufen auf KOTs spezifizieren.
Damit ein betriebliches AwS auch mit Menschen (Mensch-Computer-Kommunikation, MCK) und anderen AwS (Computer-Computer-Kommunikation, CCK) interagieren kann, sind geeignete Schnittstellen zu definieren. Diese Schnittstellen werden im *Interface-Objektschema* (IOS) einheitlich beschrieben. Jede Schnittstelle wird als Interface-Objekttyp (IOT) spezifiziert und kann anschließend im Lösungsverfahren eines VOT verwendet werden. Einerseits kann ein Vorgang Nachrichten an IOTs senden, um die weitere Verarbeitung durch einen Menschen oder ein AwS auszulösen. Andererseits wird die Durchführung von betrieblichen Aufgaben durch eingehende Nachrichten von IOTs ausgelöst. KOS, VOS und IOS definieren die fachliche Funktionalität des AwS. Diese Funktionalität stützt sich auf eine anwendungsneutrale technische Funktionalität, die im *technischen Objektschema* (TOS) beschrieben wird. Das TOS stellt eine einheitliche Basismaschine für das AwS zur Verfügung und baut wiederum auf anderen Basismaschinen auf, wie z.B. Datenbanksystemen oder Klassenbibliotheken. Das TOS wird in dieser Arbeit nicht weiter betrachtet.

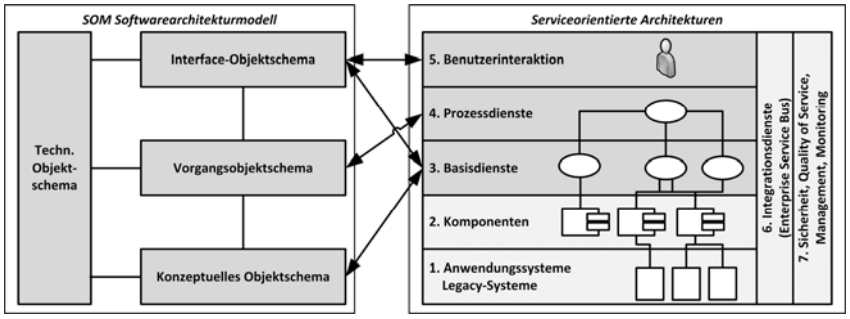

Abbildung 2: SOM-Softwarearchitekturmodell und Serviceorientierte Architektur (in Anlehnung an [12])

Zwischen dem SOM-Softwarearchitekturmodell und dem Ebenenmodell einer SOA ([2], vgl. auch [4], [10] und [14]) existieren strukturelle Ähnlichkeiten (siehe Abbildung 2): Basisdienste der Schicht 3 verwalten betriebliche Entitäten und stellen Operationen zur Manipulation dieser Entitäten zur Verfügung. Diese betrieblichen Entitäten mit ihren Attributen und Operationen finden sich auch in der SOM-Architektur als KOTs wieder. Schnittstellen zur Kommunikation mit AwS oder Menschen werden als Schnittstellen im IOS bzw. als Basisdienste spezifiziert. Objekte für die Vorgangssteuerung haben ihre Entsprechung in den Prozessdiensten einer SOA. Neben diesen strukturellen Ähnlichkeiten gewährleistet die SOM-Methodik eine durchgängige Modellierung betrieblicher Systeme. Damit bildet die SOM-Methodik eine naheliegende Grundlage für die Ableitung von Diensten und deren Orchestrierung in Form von Workflow-Schemata.

3. Modellgetriebene Entwicklung von serviceorientierten Architekturen

Dieser Beitrag konzentriert sich auf die Analyse und das Design der Basis- und Prozessdienste, sowie der Schnittstellen für die Benutzerinteraktion. Sie abstrahiert also von der konkreten Implementierung der Dienste, sowie der technischen Infrastruktur, wie dem Enterprise Service Bus. Kern der Arbeit ist ein Architekturmodell, bestehend aus verschiedenen Abstraktionsebenen und Sichten, die durch Beziehungsmetamodelle miteinander verbunden sind.
Grundlage des Architekturmodells bildet die SOM-Unternehmensarchitektur und das korrespondierende Softwarearchitekturmodell von Amberg und Malischewski. Das GP-Modell beschreibt die Aufgabenebene des betrieblichen Systems. Die betrieblichen Aufgaben werden im verhaltensorientierten VES und im strukturorientierten IAS spezifiziert. Serviceorientierte AwS – als Aufgabenträger für die Durchführung der betrieblichen Aufgaben – wird im KOS, IOS und VOS erfasst. Die zwei strukturorientierten Sichten für die Modellierung von Entitäten (KOS) und für die Schnittstellen zur Kommunikation mit Mensch und Maschine (IOS) werden in UML-Notation modelliert. Durch Stereotypen oder Profile kann die Notation für die Spezifikation von Diensten ertüchtigt werden. Im KOS lassen sich dadurch die Basisdienste für den Zugriff auf Entitäten (*Entitäten-Dienste*) spezifizieren. Im IOS können Dienste für die Benutzerinteraktion und für die Kommunikation mit anderen AwS (*Schnittstellen-Dienste*) erfasst werden. BPMN-Workflows, deren einzelne Aktivitäten aus den im IOS und KOS definierten Entitäten- und Schnittstellen-Diensten bestehen, spezifizieren die Ablaufsicht der Vorgänge (VOS). Durch die klare Trennung zwischen Aufgaben- und Aufgabenträgerebene berücksichtigt das Architekturmodell insbesondere die Unterschiede zwischen einem Geschäftsprozess und dessen Automatisierung durch Workflows [vgl. 15]. Ein mit dem Architekturmodell abgestimmtes Vorgehen leitet den Modellierer durch die verschiedenen Ebenen und Sichten des Entwicklungsprozesses. Für die Modellierung komplexer realer Problemstellungen wird der Modellierer zudem durch ein geeignetes Entwicklungswerkzeug unterstützt. Dieses Kapitel beschreibt die einzelnen Ebenen und den Entwicklungsprozess über die Ebenen hinweg (vgl. Abbildung 3).

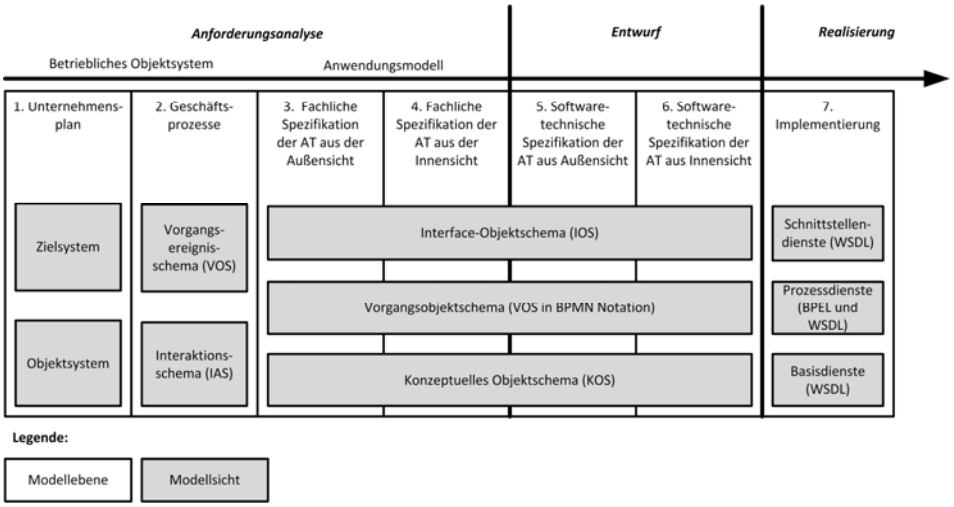

Abbildung 3: Modellarchitektur

Ebene 1 – Unternehmensplan: Der SOM-Unternehmensplan ist Teil der Anforderungsanalyse und bildet die oberste Ebene der Modellarchitektur. Auf dieser Ebene wird die globale Unternehmensaufgabe durch das Objekt- und Zielsystem verbal beschrieben.

Ebene 2 – Geschäftsprozess-Modell: Mit dem GP-Modell beginnt die Spezifikation der Innenperspektive des betrieblichen Systems. Als Fortsetzung der Anforderungsanalyse wird das Lösungsverfahren für die globale Unternehmensaufgabe im IAS und VES beschrieben. Das initiale GP-Modell wird anschließend gemäß den Zerlegungsregeln für Objekte und Transaktionen [6, S. 196] sukzessive zerlegt.

Ebene 3 – Fachliche Spezifikation der Aufgabenträger aus Außensicht: Ausgehend vom GP-Modell, beginnt die fachliche Spezifikation der (teil-)automatisierbaren AT im VOS, KOS und IOS. In der Außensicht werden das Aufgabenobjekt, die Ziele und die Ereignisse für die Durchführung der einzelnen betrieblichen Aufgaben festgelegt. Dabei abstrahiert der Modellierer auf dieser Ebene von technischen Aspekten und konzentriert sich auf die fachlichen Anforderungen an die zu entwickelnde SOA. Aus der Verhaltenssicht des GP-Modells wird der BPMN-Workflow abgeleitet, der den Ablauf zwischen den Vorgängen definiert Gleichzeitig können aus der Struktur- und Verhaltenssicht des GP-Modells die initialen, fachlichen Entitäten abgeleitet und in einem UML-Klassendiagramm spezifiziert werden. Mehrere Entitäten werden anschließend zu Entitäts-Diensten zusammengefasst.
Neben Entitäten können sich bereits aus dem BPMN-Workflow heraus weitere Dienste der SOA ergeben. Jedem Vorgang, der an einer nicht-automatisierbaren Transaktion beteiligt ist, muss eine Schnittstelle für die Mensch-Computer-Kommunikation zur Verfügung gestellt werden. Hingegen sind Anforderungen an Schnittstellen für die Computer-Computer-Kommunikation (z. B. Adapter) auf der fachlichen Ebene noch nicht zu berücksichtigen, da sie sich erst aus den einzusetzenden Technologien ergeben.

Ebene 4 – Fachliche Spezifikation der Aufgabenträger aus Innensicht: Die identifizierten Vorgänge der Ebene 3 werden auf Ebene 4 aus der Innensicht betrachtet. Das Lösungsverfahren eines Vorgangs besteht aus einer Navigation über Entitäten-Diensten und Diensten für die Benutzer- und Systeminteraktion (Schnittstellen-Dienste). Durch Betrachtung der Innensicht eines Vorgangs werden neue Anforderungen an Entitäten- und Schnittstellen-Dienste aufgedeckt. Ergebnis der Ebene sind somit BPMN-Workflows für jeden Vorgang sowie detaillierte UML-Klassendiagramme für die Entitäten- und Schnittstellen-Dienste. Ein- und ausgehende Transaktionen aus der Außensicht

eines Vorgangs bilden die Start- und End-Ereignisse des BPMN-Workflows aus der Innensicht. Initial enthält das Modell darüber hinaus lediglich Aktivitäten, die Nachrichten von anderen Vorgängen empfangen oder an diese senden. Diese eingehenden und ausgehenden Nachrichten bilden somit den Kontext, in dem der Modellierer das Lösungsverfahren spezifizieren kann. Der abgeleitete BPMN-Workflow wird anschließend derart überarbeitet, dass ein lückenloser Kontrollfluss zwischen den einzelnen Aktivitäten entsteht. Dabei werden neue identifizierte Aktivitäten in Form von Diensten (Schnittstellen oder Entitäts-Dienste) im KOS und IOS ergänzt.

Ebene 5 – Softwaretechnische Spezifikation der Aufgabenträger aus Außensicht: Die Umsetzung der fachlichen Anforderungen durch eine serviceorientierte Architektur beginnt auf Ebene 5. Für den Entwurf werden die Modellierungsergebnisse der Ebene 3 um softwaretechnische Aspekte erweitert. Die betrieblichen Objekte sind unabhängig voneinander und kooperieren im Hinblick auf gemeinsame Zielerreichung [6, S. 187f]. Für die Kooperation betrieblicher Objekte und deren Vorgänge legt der softwaretechnische Entwurf aus Außensicht das Protokoll für den Nachrichtenaustausch fest. Das Protokoll ist für die beteiligten Objekte bindend und ermöglicht dadurch eine dezentrale Koordination. Diese Form nichthierarchischer Steuerung wird auch als Choreographie bezeichnet. Zunächst werden die Dienste des IOS und KOS typisiert. Anschließend müssen die Implementierungsartefakte (siehe Ebene 7 zur Generierung der XSD und WSDL-Dateien) der Dienste generiert werden, so dass diese bei der Überarbeitung des BPMN-Workflows verwendet werden können. Den einzelnen Vorgängen des BPMN-Workflow werden die generierten Schnittstellenbeschreibungen zugeordnet, so dass die Nachrichtenbeziehungen zwischen den Vorgängen eindeutig festgelegt sind. Alle nichtautomatisierbaren Vorgänge müssen in diesem Schritt aus dem Workflow entfernt werden. Für diese Transaktionen übernehmen Dienste für die Mensch-Computer-Interaktion den Austausch der Nachrichten. Auch automatisierbare Transaktionen, die nicht als Web-Service implementiert werden, müssen durch Adapterdienste ersetzt werden, die die Anbindung anderer Systeme ermöglichen. Da sich aus der Überarbeitung des BPMN-Workflows wiederum Anforderungen an die Entitäten- und Schnittstellendienste ergeben können, sind die Ebenen 5 bis 7 in einem iterativen Entwicklungsprozess zu durchlaufen.

Ebene 6 – Softwaretechnische Spezifikation der Aufgabenträger aus Innensicht: Das fachliche Modell der Innensicht (Ebene 4) bildet die Grundlage für diese Ebene. Zunächst werden die Nachrichten, die in der softwaretechnischen Außensicht bereits typisiert wurden, in den BPMN-Workflow aus Ebene 4 integriert. Der Modellierer kann nun mit der Typisierung und Erweiterung des fachlichen Modells fortfahren, so dass eine geeignete Basis für die Generierung von BPEL-Workflows möglich ist. So müssen z. B. PartnerLinkTypes und PartnerLinks definiert, Web-Service-Operationen den Aktivitäten zugeordnet, Kompensationsaktivitäten erstellt und alternative Kontrollflüsse für die Ausnahmebehandlung spezifiziert werden.

Ebene 7 – Implementierung: Die Datenbasis der serviceorientierten Architektur leitet sich aus dem KOS ab und wird durch ein XML-Schema definiert. Web-Services beschreiben die Entitäten- und Schnittstellen-Dienste. Aus der Ablaufsicht können BPEL-Workflows generiert werden. Zunächst wird aus dem BPMN-Workflow der softwaretechnischen Außensicht die Schnittstellenbeschreibung des BPEL-Workflows gewonnen. Dadurch verbirgt sich in den einzelnen BPEL-Workflows implizit die in der softwaretechnischen Außensicht definierte Choreographie zwischen den betrieblichen Objekten. Explizit kann die Choreographie aber auch durch geeignete Standards, wie z. B. der Web-Service Choreography Description Language (WSCDL), erfasst werden. Aus der softwaretechnischen Innensicht kann schließlich der eigentliche BPEL-Prozess generiert werden, der die Schnittstellenbeschreibung implementiert. Das abgeleitete XML-Schema, die Web-Service Beschreibungen und BPEL-Workflows sind das Ergebnis der Modellbildung. Aufbauend auf diesen

Artefakten muss die Implementierung der Web-Services durch Komponenten erfolgen. Entwurf und Realisierung dieser Komponenten sind jedoch nicht Bestandteile dieser Arbeit.

4. Erläuterndes Beispiel

Mit Hilfe eines erläuternden Beispiels soll die Modellbildung über alle Ebenen und Sichten hinweg demonstriert werden. Grundlage ist ein produzierendes Unternehmen, wie z. B. die Siemens AG, das Waren von seinen Lieferanten bezieht, Produkte fertigt und diese an Kunden distribuiert. In diesem Kapitel soll der vereinfachte Geschäftsprozess *Warenbereitstellung* (siehe Abbildung 4) durch eine serviceorientierte Architektur automatisiert werden.

Ebene 1: Die Diskurswelt umfasst den Aufgabenbereich *Warenbereitstellung* eines Produktionsunternehmens. Dabei sind bei Lieferanten Waren zu beziehen, zu lagern und für die weitere Verarbeitung bereitzustellen. Sachziel der Beschaffung ist die Bereitstellung der Waren, als Formalziel wird die Kostenminimierung durch optimierte Lager- und Beschaffungskosten verfolgt.

Ebene 2: Der Geschäftsprozess *Warenbereitstellung* besteht aus den betrieblichen Objekten *Einkauf*, *Lager* und *Lieferant*. Die Koordination der betrieblichen Objekte erfolgt über die Transaktionen *Bestellung*, *Annahmeanweisung*, *Warenlieferung* und *Annahmemeldung*, deren zeitlicher Ablauf im VES spezifiziert wird.

Ebene 3: Die Verhaltenssicht lässt sich direkt in einen BPMN-Workflow übertragen. Der Eingang einer Bestellanforderung löst den Workflow aus. Parallel zur Ablaufsicht wurde das initiale KOS abgeleitet und überarbeitet. Die Abbildung zeigt einen Ausschnitt des KOS, bestehend aus verschiedenen Entitäten, die zu den Diensten *Lieferantenverwaltung*, *Produktverwaltung* und *Beschaffungsverwaltung* zusammengefasst wurden. Da der Wareneingang in diesem Beispiel eine nichtautomatisierbare Transaktion ist, wurde ein entsprechender Schnittstellendienst für die Erfassung der Waren durch den Menschen spezifiziert (*MCK_Lieferschein_Eingabe*).

Ebene 4: Nachdem die Außensicht die Nachrichten- und Ereignisbeziehungen zwischen den betrieblichen Aufgaben festlegt, wird in der Innensicht das Lösungsverfahren der einzelnen Vorgänge als eigenständiges Ablaufdiagramm modelliert. Start- (Eingang der Bestellanforderung) und Endereignis (Auslösen der Annahmeanweisung) des Vorgangs *sende Bestellung (Bestellung>)* sowie die gleichnamige Aktivität ergeben sich direkt aus den Beziehungen des Vorgangs. Darüber hinaus wurde der Workflow um weitere Aktivitäten ergänzt, die auf die Operatoren des Entitäten-Dienstes *Beschaffungsverwaltung* zurückgreifen.

Ebene 5: Ausgangspunkt der softwaretechnischen Spezifikation bildet die Detaillierung des KOS und IOS. Erst die strenge Typisierung der Attribute und Parameter ermöglicht die Generierung des XML-Schemas und der zugehörigen Web-Services. Das fachliche VOS aus Außensicht wird als initiales Modell übernommen und anschließend derart überarbeitet, dass es sich als Ableitungsbasis für die Choreographie der beteiligten Vorgänge eignet. Da es sich im Wesentlichen um Typisierungen von Variablen sowie die Zuweisung von Operatoren handelt, sind diese Änderungen in der Abbildung nicht sichtbar. Nach der Überarbeitung enthält es ausschließlich Nachrichtenflüsse, die mit Hilfe von Web-Services ausgetauscht werden. Da die Bestellung an den Lieferanten per EDI übertragen wird, ist ein Schnittstellendienst für die Kommunikation mit einem externen AwS notwendig (*CCK_Sende_Bestellung*).

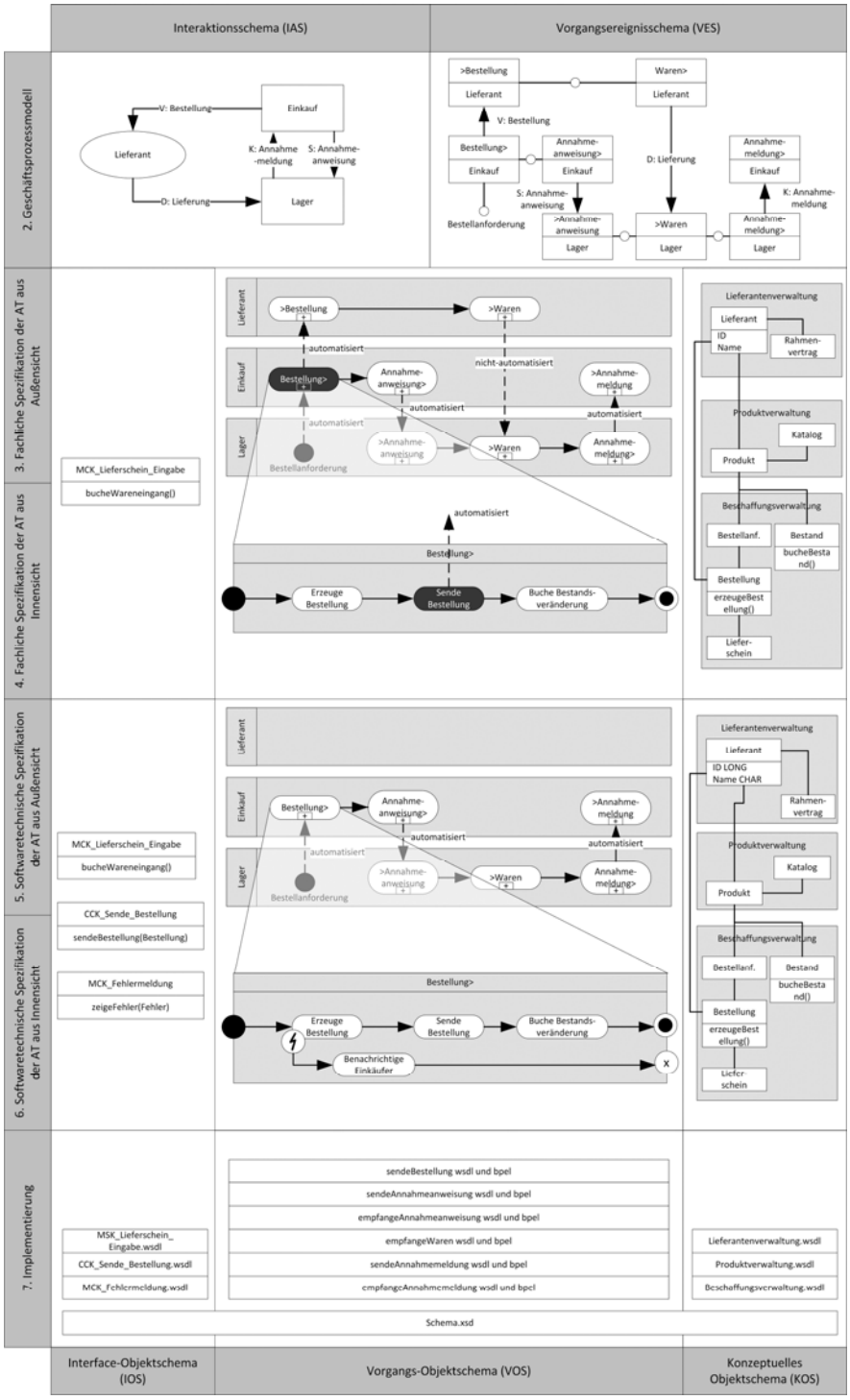

Abbildung 4: Geschäftsprozess *Warenbereitstellung*

Ebene 6: Das initiale Modell dieser Ebene basiert auf der fachlichen Spezifikation der Innensicht, deren Nachrichten- und Ereignisbeziehungen um die softwaretechnischen Details der Ebene 5 erweitert wurden. Darüber hinaus beinhaltet das Diagramm einen alternativen Kontrollfluss für die Ausnahmebehandlung beim Erstellen einer Bestellung. Um einen derartigen Fehler behandeln zu können, wurde ein neuer Schnittstellendienst für die Kommunikation mit dem Einkäufer modelliert (*MCK_Fehlermeldung*).

Ebene 7: Die Abbildung zeigt eine Übersicht der aus den Ebenen 5 und 6 generierten Artefakte der SOA. Auf Basis der Schnittstellendefinitionen kann mit dem Entwurf und der Implementierung der Komponenten fortgefahren werden.

5. Zusammenfassung und verwandte Arbeiten

Die Konstruktion von serviceorientierten Architekturen beginnt mit der Analyse und Modellierung der Geschäftsprozesse eines betrieblichen Systems. Die Überführung der Geschäftsprozessmodelle in technische Workflows und zugehörige Basisdienste ist jedoch auf Grund unterschiedlicher Zielsetzungen der Modelle nicht einfach. Durch Zerlegung des Modellsystems in mehrere Abstraktionsebenen kann die semantische Lücke zwischen Aufgaben- und Aufgabenträgerebene in kleinen Schritten überwunden werden. In dieser Arbeit wurde deshalb ein modellgetriebener Ansatz vorgestellt, der den Entwurf einer SOA in sieben Teilmodellsysteme gliedert und die Übergänge zwischen den Ebenen durch Beziehungsmetamodelle definiert. Die Beziehungsmetamodelle bestehen aus eindeutigen Ableitungsregeln [4, S. 57ff], die eine automatisierte Transformation zwischen den Ebenen erlauben. Die Regeln gewährleisten auch die Konsistenz zwischen den Modellebenen, sodass der Modellierer sich auf die Überarbeitung und Detaillierung der abgeleiteten Modelle konzentrieren kann. Die automatisierbare Transformation sowie weitere Automatisierungspotentiale wurden mittels eines vertikalen Prototyps untersucht. Dieser stützt sich auf MDA-Standards für die Definition der Metamodelle und die Modell-zu-Modell-Transformationen[3]. Der Prototyp wurde für das in dieser Arbeit vorgestellte Fallbeispiel eingesetzt, sodass eine vollautomatisierte Transformation der Modelle und die Generierung der Artefakte möglich waren.

Einen dieser Arbeit sehr verwandten Ansatz haben Thomas et al. [16] vorgeschlagen, deren Architekturmodell die Gestaltungs-, die Konfigurations- und die Ausführungsebene unterscheidet: Ausgehend von der Systemabgrenzung und Geschäftsprozessmodellierung (Gestaltungsebene) erfolgt die fachliche und softwaretechnische Konfiguration auf den Ebenen 3 bis 6 (Konfigurationsebene) und schließlich die Spezifikation der ausführbaren BPEL-Workflows auf Ebene 7 (Ausführungsebene). Einen wesentlich umfangreicheren Entwicklungsprozess hat IBM aufbauend auf dem Rational Unified Process definiert [17]. Obwohl sich diese Methode über alle Phasen der Softwareentwicklung erstreckt, liegen ihre Schwerpunkte in der Entwurfs- und Implementierungsphase. Durch die strukturierte Vorgehensweise über mehrere Ebenen und Sichten hinweg, systematisiert der in dieser Arbeit vorgestellte Ansatz diese frühen Phasen und kann als Ausgangsbasis für einen derartigen Entwicklungsprozess verwendet werden. Ein weiterer Ansatz für den Entwurf von serviceorientierten Architekturen basiert auf der von Scheer konzipierten Architektur integrierter Informationssysteme (ARIS) [13]. Aus den mit der ARIS-Methode analysierten Geschäftsprozessen werden die Anforderungen an die technische Infrastruktur abgeleitet und anschließend in 10 Schritten in eine SOA umgesetzt [9]. Die von IDS Scheer entwickelte Methode konzentriert sich wie dieser Artikel auf die frühen Phasen des Entwicklungsprozesses und basiert ebenfalls auf einem geschäftsprozessorientierten Ansatz. Im Gegensatz zu dem in dieser Arbeit und dem von Thomas et al. vorgeschlagenen Architekturmodell erfolgt die Konfiguration der SOA aber direkt in den EPKs. Eine klare Trennung zwischen Geschäftsprozessebene und Workflow-Ebene bzw. zwischen fachlichen und technischen Anforderungen ist dabei nicht gewährleistet.

Der hier vorgestellte Ansatz integriert verschiedene Modellierungsansätze und Standards in einem Architekturmodell für die modellgetriebene Entwicklung einer SOA. Das Architekturmodell ist offen gestaltet, so dass sich in zusätzlichen Sichten weitere Aspekte einer SOA erfassen lassen. Mit Hilfe von Patterns könnte darüber hinaus heuristisches Entwurfswissen integriert werden. Denn letztlich können Modelle, Methoden und Vorgehensweisen nur einen Beitrag zur durchgängigen

[3] Der Prototyp wurde mit Hilfe des Eclipse Modeling Frameworks (EMF) und der Atlas Trasformation Language (ATL) implementiert. Siehe [4, S. 74ff] für eine Vorstellung des Prototypen.

Entwicklung einer SOA leisten und den Modellierer bei der Komplexitätsbewältigung unterstützen. Die Qualität des zu konstruierenden Systems hängt immer noch in entscheidendem Maße von der Erfahrung und dem Wissen der beteiligten Entwickler ab.

6. Literaturangaben

[1] AMBERG, M., Konzeption eines Software-Architekturmodells für die objektorientierte Entwicklung betrieblicher Anwendungssysteme, Dissertation an der Otto-Friedrich-Universität Bamberg, 1993.

[2] ARSANJANI, A., Serviceoriented modeling and architecture - How to identify, specify, and realize services for your SOA. In: IBM Developerworks, http://www.ibm.com/developerworks/library/ws-soa-design1 (Letzter Aufruf: 2008-11-14), 2004.

[3] ARSANJANI, A., ZHANG, L.-J., ELLIS, M., ALLAM, A. und CHANNABASAVAIAH, K., Design an SOA solution using a reference architecture In: IBM Developerworks, http://www-128.ibm.com/developerworks/library/ar-archtemp/index.html (Letzter Aufruf: 2008-11-14), 2007.

[4] BERNHARD, R., Modellgetriebene Entwicklung von service-orientierten Architekturen auf Basis der SOM-Methodik, Diplomarbeit an der Otto-Friedrich-Universität Bamberg, 2008.

[5] ERL, T., Serviceoriented architecture : concepts, technology, and design, Prentice Hall, München 2005.

[6] FERSTL, O. K. und SINZ, E. J., Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). In: WIRTSCHAFTSINFORMATIK 32(6), S. 566-581, 1990.

[7] FERSTL, O. K. und SINZ, E. J., Grundlagen der Wirtschaftsinformatik, 6. Auflage, Oldenbourg, München 2008.

[8] FRANCE, R. und RUMPE, B., Model-driven Development of Complex Software: A Research Roadmap. In: FOSE '07: 2007 Future of Software Engineering, S. 37 – 54, IEEE Computer Society, Washington, DC, USA, 2007.

[9] KLÜCKMANN, J., In 10 Schritten zur Business-Driven SOA, http://www.ids-scheer.com/set/82/ARIS_Expert_Paper_-_10_Steps_to_SOA_Klueckmann_2007-03_en.pdf (Letzter Aufruf: 2008-11-14), 2007.

[10] KRAFZIG, D., BANKE, K. und SLAMA, D., Enterprise SOA : serviceoriented architecture best practices, Prentice Hall, NJ 2005.

[11] KURBEL, K., STRUNZ, H., Handbuch der Wirtschaftsinformatik, Poeschel, Stuttgart, 1990.

[12] MALISCHEWSKI, C., Generierung von Spezifikationen betrieblicher Anwendungssysteme auf der Basis von Geschäftsprozeßmodellen, Shaker, Aachen 1997.

[13] SCHEER, A.-W., Architektur integrierter Informationssysteme - Grundlage der Unternehmensmodellierung, Springer, Berlin 1992.

[14] SIEDERSLEBEN, J., SOA revisited: Komponentenorientierung bei Systemlandschaften. In: WIRTSCHAFTSINFORMATIK Sonderheft 2007, S. 110-117, 2007.

[15] SINZ, E. J., SOA und die bewährten methodischen Grundlagen der Entwicklung betrieblicher IT-Systeme. In: WIRTSCHAFTSINFORMATIK 50(1), S. 70-72, 2008.

[16] THOMAS, O., LEYKING, K., DREIFUS, F., FELLMANN, M., Serviceorientierte Architekturen: Gestaltung, Konfiguration und Ausführung von Geschäftsprozessen. In: Loos, P. (Hrsg.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, Nr. 189, Saarbrücken : Universität des Saarlandes, 2007.

[17] WAHLI, U., ACKERMAN, L., DI BARI, A. und HODGKINSON, G., Building SOA Solutions Using the Rational SDP, IBM Redbooks Publication, IBM International Technical Support Organization 2007.