Model-Driven Integration Engineering in der E-Government-Domäne Meldewesen

Stefan Kühne, Maik Thränert

Institut für Informatik, Universität Leipzig
04103 Leipzig
{kuehne,thraenert}@informatik.uni-leipzig.de

Werner Rotzoll, Jan Lehmann

DVZ Datenverarbeitungszentrum Mecklenburg-Vorpommern GmbH
19059 Schwerin
{rotzoll,lehmann}@dvz-mv.de

Abstract

Im Rahmen des E-Government integrieren IT-Dienstleister der öffentlichen Verwaltung bestehende Anwendungen von Bund, Ländern und Kommunen in komplexen Infrastrukturen. Durch domänenspezifische Modellierungssprachen lässt sich die Komplexität der Integrationsvorhaben reduzieren. Am Beispiel der E-Government-Domäne Meldewesen wird in diesem Beitrag der Ansatz Model-Driven Integration Engineering (MDIE) vorgestellt. Er beschreibt u. a. die Entwicklung kompakter, intuitiver Modellierungssprachen für Integrationsvorhaben in einer fachlichen Domäne und fokussiert dabei die direkte Weiterverwendung der modellierten Sachverhalte über Modelloperationen.

1 Integrationsproblematik im E-Government

Derzeit werden in allen Bundesländern große Anstrengungen in der E-Government-Domäne Meldewesen unternommen, eine Infrastruktur aufzubauen und erste Dienste, wie *einfache Melderegisterauskunft* [Osci06, S. 352] oder *Rückmeldung* [Osci06, S. 133] anzubieten. Allgemein betrachtet, entwickeln IT-Dienstleister der öffentlichen Verwaltung E-Government-Dienste, indem sie bestehende Anwendungen von Bund, Ländern und Kommunen in komplexen Infrastrukturen vernetzen und integrieren. Dabei sind sie mit typischen Herausforderungen von In-

tegrationsprojekten konfrontiert, die sich aus der Vielfältigkeit und Heterogenität bestehender Systeme ergeben. Zusätzliche Schwierigkeiten ergeben sich aus Änderungen z. B. hinsichtlich Kundenanforderungen, Gesetzesvorgaben oder verfügbaren Implementierungstechnologien.
Die Komplexität der Integrationsvorhaben zeigt sich bereits in aus fachlicher Sicht vergleichsweise einfachen Prozessen, wie der *einfachen Melderegisterauskunft.* Deren technische Umsetzung basiert auf verschiedenen Infrastruktur- und Basiskomponenten einer E-Government-Plattform, wie Portal, Virtuelle Poststelle oder Verzeichnisdiensten. Die Orchestrierung dieser Komponenten erfolgt unter Nutzung von Workflows in einem Business Process Management System.
Für eine effizientere Umsetzung derartiger Integrationsprojekte im E-Government wird in diesem Beitrag der Ansatz *Model-Driven Integration Engineering (MDIE)* vorgeschlagen, welcher die Anwendung modellgetriebener Techniken im Anwendungsbereich des Integration Engineering (IE) [Raut93, FäTW05, SDTK05] fokussiert. Der MDIE-Ansatz wird am Beispiel der E-Government-Domäne Meldewesen vorgestellt.[1] Es wird gezeigt, wie Konzepte dieses Bereichs identifiziert und analysiert, das gewonnene Domänenwissen in eine kompakte, spezifische Modellierungssprache überführt und Modelle der Sprache über Transformationen weiterverarbeitet werden können. Die Anwendung des MDIE-Ansatzes ermöglicht die (Integrations-) projektübergreifende Wiederverwendung von Domänenwissen sowie Komplexitätsreduktionen, womit Verbesserungen des Entwicklungsprozesses, insbesondere hinsichtlich Flexibilität, Entwicklungsgeschwindigkeit, Entwicklungskosten, Qualität sowie Transparenz für beteiligte Personengruppen (z. B. Auftraggeber), adressiert werden.

2 Der Ansatz des Model-Driven Integration Engineering

2.1 Modellgetriebene Entwicklungsansätze

Modellgetriebene Entwicklungsansätze wie das Model-Driven Engineering (MDE) [Bézi05, FaNg05, Schm06], Model-Driven Software Development (MDSD) [StVö06] oder Generative Software Development (GSD) [Czar04, CzEi00] betonen die zentrale Rolle formaler Modelle im Lebenszyklus eines Softwaresystems bzw. einer Softwaresystemfamilie. Ein Modell dient hierbei als stellvertretende Beschreibung eines zugrunde liegenden Systems (Original), in wel-

[1] Beispielhaft wurden hierfür die Ergebnisse des Projektvorhabens Meldewesen für das Bundesland Mecklenburg-Vorpommern [LeRo06] in die Untersuchungen einbezogen.

chem ausschließlich zweckdienliche Informationen repräsentiert sind. Da ein Modell selbst als System aufzufassen ist, kann vom technischen Informationssystem in mehreren Stufen hinsichtlich verschiedener Sichten und Abstraktionsebenen abstrahiert werden.

Ein zentraler Bestandteil modellgetriebener Ansätze ist die Definition von Modelloperationen, wie Modelltransformationen, Modellvalidierungen oder Modelldifferenzen, auf Basis formaler Modelle zur *Automatisierung* des Entwicklungsprozesses. Modelltransformationen ermöglichen beispielsweise die Anreicherung von Modellen mit zusätzlichen Informationen (z. B. technischen), die informationserhaltende Formüberführung von Modellen oder die Extraktion und Aggregation bestimmter Eigenschaften zur Bewertung von Modellen. Modellvalidierungen überprüfen, ob modellierte Sachverhalte gültig hinsichtlich definierter Anforderungen sind. Die Modelloperationen rechtfertigen somit die zentrale Rolle formaler Modelle im Entwicklungsprozess, da sie direkt zu signifikanten Mehrwerten, wie effizientere Entwicklung oder bessere Qualität, beitragen können.

In den modellgetriebenen Ansätze MDE, MDSD und GSD werden die Konzepte technologieunabhängig definiert. Für den praktischen Einsatz existieren zahlreiche Konkretisierungen, wie beispielsweise die Model Driven Architecture [ObjeoJ], das Model-integrated Computing [KSLB03], das architecture-centric Model-Driven Software Development [StVö06, S. 21 ff.] oder Microsofts Software Factories [GrSh04]. Dabei werden die Konzepte für verschiedene Anwendungsbereiche mit Methoden, Werkzeugen und Standards hinterlegt.

Erfahrungen zeigen, dass die Ansätze in sehr unterschiedlichen Anwendungsbereichen, wie bei der Entwicklung verteilter eingebetteter Systeme oder komplexer Enterprise-Anwendungen, erfolgreich eingesetzt werden und zu Verbesserungen im Entwicklungsprozess führen können [Schm06]. Ein noch wenig betrachteter Anwendungsbereich modellgetriebener Ansätze ist das Integration Engineering.

2.2 Integration Engineering

Beim Integration Engineering (IE) handelt es sich um eine Engineering-Disziplin, *„die Prinzipien, Modelle, Methoden und Werkzeuge entwickelt, bewertet und zur Verfügung stellt, sodass diese in einem ingenieurmäßigen Prozess der Integrierung eingesetzt werden können.“* [ThKü06, S. 14] Vereinfacht formuliert, liefert das IE *„das »Handwerkszeug« mit zugehörigen Einsatzanweisungen, damit Integrierung durchgeführt werden kann.“* [Thrä05, S. 21]

Dabei betrachtet das IE Informationssysteme (IS) als relevante Integrationsobjekte, sodass diese zu Integrierten Informationssystemen (IIS) umgestaltet werden. Da es sich bei einem Informati-

onssystem um ein „sozio-technisches System“ [Birk05] handelt, spielen neben den technischen auch die organisatorischen Aspekte einer Integration eine wichtige Rolle, obgleich sich die Ausführungen in diesem Beitrag primär den technischen Integrationsproblemen zuwenden.

Grundsätzlich sind „sowohl die Neuentwicklung von integrierten Informationssystemen (‚Integration ex ante’) als auch die Zusammenführung bereits existierender Informationssysteme (‚Integration ex post’)“ [KuRa96, S. 170] Aufgabenbereiche des IE. Zur ex-post-Integration gehören z. B. die Anpassung und Integrierung eines Informationssystems in eine bestehende IT-Infrastruktur, die Integrierung von IS oder deren Komponenten untereinander sowie die Anbindung bestehender IS an eine zentrale Plattform. Zur ex-ante-Integration gehört auch die Desintegration eines IS bzw. IIS mit dem Ziel eines Refactoring, sodass eine bessere Integrierbarkeit dieses IS bzw. IIS erreicht wird. Ebenso zur ex-ante-Integration gehört die Harmonisierung der Anwendungsarchitektur, wie z. B. die Einführung einer einheitlichen technischen Kommunikationsbasis unter Nutzung von z. B. Web Services oder auch die Harmonisierung von Anwendungsmodellen durch Schaffung einer einheitlichen Metamodellinfrastruktur.

Die Anwendungsbereiche des IE sind dabei sehr vielseitig. Dazu gehören beispielsweise der Maschinenbau [WeSo05], E-Government [LeRK06] oder E-Commerce [HäKü06; FFSD06]. Das Integration Engineering versucht dabei ständig, durch neue Methoden, Werkzeuge und Vorgehensweisen eine Effizienzsteigerung zu erzielen. In diesem Sinne wurde auch der im Folgenden dargestellte Ansatz des Model-Driven Integration Engineering entwickelt.

2.3 Model-Driven Integration Engineering

Die Anwendung des Paradigmas modellgetriebener Entwicklung im Anwendungsbereich des IE führt zur Disziplin des *Model-Driven Integration Engineering (MDIE)* [ThKü06]. Dabei werden modellgetriebene Ansätze zur Lösung von IE-Problemen ausgewählt und angepasst. Konkreter formuliert liegt der Fokus des MDIE in

- der Auswahl, Anpassung und ggf. Neuentwicklung
- von Prinzipien, Modellen, Methoden, Werkzeugen und Vorgehensweisen modellgetriebener Ansätze
- zur formalen domänenspezifischen Modellierung und Anwendung von Modelloperationen (wie Modelltransformationen oder Modellvalidierungen)

- für die IE-Aufgabenbereiche Beschreibung der Integrationsform als Ausgangsbasis und Ziel einer Integrierung sowie Überführung von Integrationsformen.

Eine Zielsetzung des MDIE ist die kompakte, problemangepasste und intuitive Modellierung als Basis für ein besseres Verständnis sowie leichtere Handhabbarkeit im Vergleich zum zugrunde liegenden System. Modellierungssprachen mit weit gefasstem Anwendungsbereich (Unified Languages) unterliegen der Problematik, dass die fokussierten Aspekte des Systems nur mit generischen Sprachkonstrukten repräsentiert werden können. Dies führt tendenziell zu feingranularen Beschreibungen. Problemangepasste Modellierungssprachen (Specific Languages) adressieren dagegen einen meist fachlich beschränkten Anwendungsbereich (Domäne). Die spezifischen Gegebenheiten und Charakteristika der Domänen werden berücksichtigt, indem für die Konzepte der Domäne spezifische Sprachkonstrukte zur Verfügung gestellt werden. Dieses Vorgehen ermöglicht eine direkte Repräsentation der Sachverhalte und damit kompakte Modellierung, was zu reduziertem Aufwand bei der Erstellung von Modellen und zu einer verbesserten Kommunikation zwischen den beteiligten Personengruppen führt.
Ein weiteres Ziel des MDIE ist die Definition von Modelloperationen auf Basis formaler domänenspezifischer MDIE-Modellierungen. Entsprechende Transformationen ermöglichen bspw. die Anreicherung von Modellen mit technischen Informationen oder die Extraktion bestimmter Modellaspekte beispielsweise zur Bewertung der Integrationsform. Mithilfe von Validierungen können Modelle bspw. hinsichtlich Konsistenz, Vollständigkeit oder Anforderungskonformität überprüft werden. Die im Rahmen modellgetriebener Ansätze definierten Modelloperationen führen zur Automatisierung von manuellen Arbeitsschritten.

2.4 Anwendung des MDIE im E-Government

Die Potenziale modellgetriebener Entwicklungsansätze übertragen sich auf das MDIE. Insbesondere verspricht dessen Anwendung Verbesserungen hinsichtlich Qualität, Portabilität, Flexibilität und Transparenz der Integrationslösung. Dem gegenüber stehen zusätzliche Aufwände, die sich aus Erstellung und Pflege zusätzlicher MDIE-Artefakte wie domänenspezifische Sprachen, Transformationen und Validierungen ergeben. Inwiefern sich bezüglich des Entwicklungsprozesses langfristige Kostensenkungen und Steigerungen der Entwicklungsgeschwindigkeit ergeben, hängt demnach von den Gegebenheiten des Einsatzgebietes (der Domäne) ab. Hierbei spielen Faktoren wie die Stabilität von Konzepten oder die Anzahl und Umfang der Projekte eine wesentliche Rolle.

Wie in Abschnitt 1 geschildert, stellt die Umsetzung behördenübergreifender E-Government-Prozesse auf Basis existierender Fachverfahren eine aktuelle Herausforderung für IT-Dienstleister der öffentlichen Verwaltung dar. Die effiziente Anwendbarkeit des MDIE-Ansatzes in der Domäne E-Government wird durch folgende Charakteristika begünstigt:

- Der Domäne liegt eine gewachsene und gefestigte Terminologie zugrunde.
- Durch Vorgaben des Gesetzgebers sowie einer Reihe von Standards und Richtlinien von E-Government-Initiativen und Arbeitsgruppen bestehen stabile Konzepte, hinsichtlich des Gegenstandsbereichs der Integrierung und der Integrationsmittel.
- Anzahl und Umfang potenzieller Integrationsprojekte sind beträchtlich. Die einzelnen Integrationsprojekte stehen in einem zeitlichen und logischen Zusammenhang. Demzufolge bestehen Bedarfe hinsichtlich Wiederverwendung von Entwicklungsartefakten, Erfahrungen, usw.

Im Folgenden soll der MDIE-Ansatz anhand der E-Government-Domäne Meldewesen demonstriert werden. Zunächst wird in Abschnitt 3 die Domäne eingegrenzt, die am Entwicklungsprozess beteiligten Personengruppen identifiziert, deren spezifischen Informationsbedarfe in eine Modellierungsarchitektur eingeordnet sowie der für den MDIE-Ansatz relevante Bereich abgegrenzt. In Abschnitt 4 werden die für die ausgewählte Modellierungsebene relevanten Konzepte identifiziert, analysiert und anschließend in eine domänenspezifische Modellierungssprache überführt. Abschnitt 5 zeigt, wie Modelle der entwickelten Sprache auf eine Ausführungsumgebung abgebildet werden können.

3 Die E-Government-Domäne Meldewesen

3.1 Der Nachrichtenstandard OSCI-XMeld

Das Online Services Computer Interface (OSCI) ist ein zweischichtiger Protokollstandard zum sicheren Austausch von Nachrichten über das Internet mit besonderer Eignung für das E-Government [Bund06]. Die sichere und vertrauliche Übertragung digital signierter Dokumente ist in der *Transportschicht* (OSCI-Transport) beschrieben. Hier wird beispielsweise die OSCI-Nachrichtenstruktur definiert, welche sich in die drei Sicherheitsebenen Administrationsebene, Auftragsebene und Inhaltsdaten untergliedert [Osci01]. Die Strukturierung und semantische

Standardisierung von Nachrichteninhalten erfolgt in OSCI in der *Anwendungsschicht.* Hierzu werden in verschiedenen Projekten, wie XMeld, XBau und XJustiz, entsprechende Standards definiert. Für den Bereich Meldewesen ist der Standard XMeld [Osci06] relevant.

Das Meldewesen versorgt eine Vielzahl staatlicher Stellen mit verwaltungsinternen Daten der Einwohner. Rechtliche Grundlage für eine Umsetzung im Sinne des E-Government ist das novellierte Melderechtsrahmengesetz (MRRG), dessen länderspezifische Umsetzung in Landesmeldegesetzen (LMG) und die erste Bundesmeldedatenübermittlungsverordnung (1. BMeldDÜV). Im Standard XMeld werden Syntax und Semantik von Nachrichten definiert, welche zwischen Meldeämtern und Kommunikationspartnern, wie Bürger, gewerblicher Kunde oder auswärtiges Meldeamt unter bestimmten, festgelegten Umständen ausgetauscht werden. Für die Einordnung der Nachrichten in einen Prozesskontext werden verschiedene Szenarien wie *Anmeldung, Rückmeldung* oder *einfache Melderegisterauskunft* definiert.

3.2 Stakeholderanalyse für eine domänenspezifische Modellierung

Für eine domänenspezifische Modellierung nach MDIE sind zunächst relevante Personengruppen und deren Informationsbedarfe zu identifizieren. Zur Umsetzung von E-Government-Diensten gehören folgende Gruppen:

Auftraggeber (Land, Kommune, Meldebehörde): Der Auftraggeber hat einen Informationsbedarf hinsichtlich der Umsetzung von gesetzlichen Grundlagen (Bundesgesetze, Landesgesetze und Richtlinien von Arbeitsgruppen), z. B. in welcher Form behördenübergreifende Prozesse ablaufen. Im Fall des Meldewesens, welches auf Landesebene umgesetzt wird, sind die fachlichen Anforderungen an Komponenten der Meldebehörden relevant.

Entwickler und Systemintegrator: Relevant sind hier funktionale und technische Aspekte (z. B. IT-Prozess, Prozessschnittstellen und fachliche Anforderungen an Komponenten), welche für die Entwicklung neuer bzw. die Integration vorhandener Komponenten erforderlich sind.

Systembetreuer: Der Systembetreuer verwaltet die Systeme im laufenden Betrieb. Er muss Fehler im Wirkbetrieb Prozessen und Komponenten zuordnen können. Der Informationsbedarf ist demnach ähnlich dem der Gruppe Entwickler und Systemintegrator, jedoch aus einer anderen Perspektive.

Service-Manager: Der Service-Manager steht zwischen den Gruppen Auftraggeber, Entwickler/Systemintegrator und Systembetreuer. Er ist für die fachgerechte Umsetzung von Diensten verantwortlich. Demzufolge sind organisatorische Aspekte wie Service-Level-Agreements,

Konfiguration/Change-Management, aber auch funktionale Aspekte wie Prozessablauf und Prozessschnittstellen relevant.

Weitere Gruppen: Weitere Gruppen sind u. a. Marketing, Datenschützer, User Help Desk, Medien, Bürger, Nutzer der Anwendung, Diensteanbieter und Zertifizierungslabor.

Die unterschiedlichen Zielstellungen dieser Personengruppen erfordern eine Modellierung auf verschiedenen Abstraktionsebenen (vgl. Abb. 1). Das Spektrum reicht dabei von fachlich-abstrakten Modellierungsebenen, welche für Auftraggeber und Service-Management relevant sind, bis zu technisch-konkreten Modellierungsebenen für die Gruppen Entwicklung und Systemintegration. Dabei ist die dynamische Prozesssicht, in der statische Sichten, wie Organisations-, Ressourcen- oder Datensicht, miteinander verknüpft sind, für alle beteiligte Personengruppen von hoher Priorität. Auf dieser Basis können bei der Entwicklung von E-Government-Diensten bspw. die Anforderungen an das Integrationssystem oder der systemübergreifende Entwurf spezifiziert werden. Auch im Betrieb ist die Prozesssicht relevant, z. B. bei der Zuordnung von Fehlern zu Systemkomponenten. Neben den gruppenbezogenen Informationsbedarfen ist die Prozesssicht ebenso für die gruppenübergreifende Abstimmung relevant.

In Abb. 1 ist eine mögliche Modellierungsarchitektur dargestellt. Diese setzt auf einer vorhandenen E-Government-Plattform, bestehend aus verschiedenen Infrastruktur- und Basiskomponenten, auf. Die technische Integration der Komponenten wird auf Integrationsebene beschrieben. Hier können z. B. auf Basis von Web-Service-Technologien die Komponenten systemübergreifend orchestriert werden. Die darüber liegenden Modellierungsebenen, wie die technische und fachliche Prozessmodellierungsebene, beschreiben Prozesse mit abnehmend technischem Detaillierungsgrad bzw. zunehmend fachlichem Abstraktionsgrad.

Die Modellierungsebene Technische Prozessmodellierung ist primär für die Personengruppe Systemintegration relevant und liegt daher im Fokus des MDIE-Ansatzes. Sie dient jedoch auch

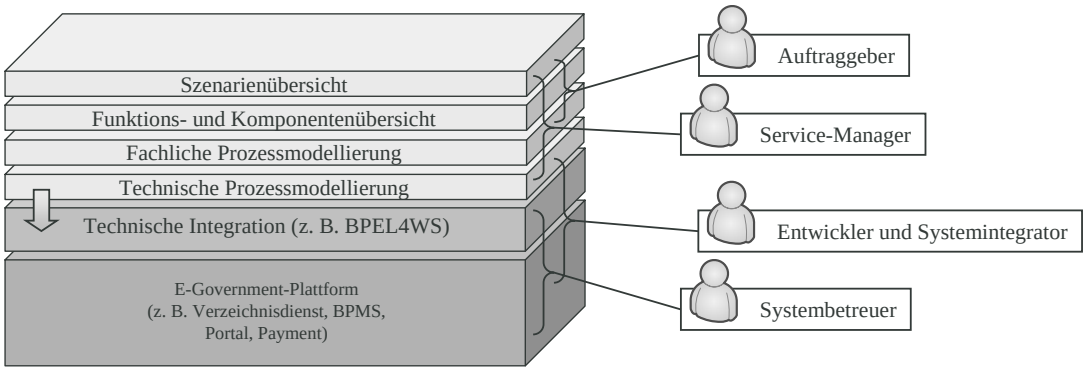

Abb. 1: Modellierungsebenen

zur Abstimmung mit den Gruppen Entwicklung und Service-Management und bildet die Schnittstelle zwischen fachlicher Modellierung und technischer Umsetzung. Sie abstrahiert von technischen Details zugrunde liegender Integrationssprachen und enthält Aspekte der fachlichen Domäne (E-Government, Meldewesen). Zielstellung des MDIE-Ansatzes ist die automatisierte Abbildung von Modellen der technischen Prozessmodellierungsebene auf die technische Integrationsebene (vgl. Pfeil in Abb. 1). Dies wird realisiert über Modelltransformationen, welche die fachlichen Modellierungskonstrukte mit den abstrahierten technischen Aspekten anreichern.

4 Entwicklung einer domänenspezifischen Sprache für die technische Prozessmodellierung im Bereich Meldewesen

4.1 Identifizierung und Analyse relevanter Konzepte

Nach Einschränkung der Domäne und Identifikation relevanter Personengruppen besteht der nächste Schritt in der Identifikation und Analyse relevanter Domänenkonzepte. Für den auf die technische Prozessmodellierungsebene eingeschränkten Bereich wurden hierfür Spezifikationsdokumente aus der Domäne E-Government bzw. der Subdomäne Meldewesen (z. B. SAGA-Architektur [Kbst05], XMeld-Spezifikation [Osci06]) sowie Implementierungsartefakte einer exemplarischen Umsetzung des Szenarios *einfache Melderegisterauskunft* (wie Orchestrierungen und Teilorchestrierungen von am Prozess beteiligten Komponeten auf Basis der Business Process Execution Language for Web Services (BPEL4WS), Schnittstellenbeschreibungen von Diensten auf Basis der Web Service Description Language (WSDL) und Nachrichtenformate auf Basis von XML-Schema-Beschreibungen) ausgewertet. Die Teilergebnisse der Konzeptanalyse wurden sodann in einem iterativen/inkrementellen Prozess mit Domänenexperten aus den Gruppen Systemintegrator, Entwickler und Service-Manager diskutiert und abgestimmt.

Als allgemeine Basiskonzepte für die technische Prozessmodellierung wurden für Prozessbeschreibungssprachen typische Konstrukte wie *Prozess, Aktivität, Nachricht, Attribut, Ereignis, Prozessstart, Prozessende, System* und *Systemschnittstelle* identifiziert. Zwischen diesen allgemeinen Konzepten bestehen vielfältige Beziehungen. So besteht ein Prozess z. B. aus Aktivitäten, eine Nachricht ist Input für eine Aktivität oder ein Prozess terminiert mit dem Prozessende. Diese Konzepte sind elementar und treten mehrfach in verschiedenen Ausprägungen auf.

Auf diesen Basiskonzepten bauen fachdomänenspezifische Konzepte auf, wie *XMeldNachricht, VerzeichnisdienstNachricht, XMeldAttribut, Fehlernachricht, Protokollierung, Fehlerbehandlung, Nachrichtenkonvertierung, Verzeichnisdienst, Intermediär.* Diese Konzepte haben noch stark technischen Charakter, erweitern die zugrunde liegenden Basiskonzepte jedoch mit domänenspezifischer Semantik. So ist beispielsweise das Konzept *XMeldNachricht* eine Einschränkung des Konzepts *Nachricht* auf die im XMeld-Standard definierten Nachrichtentypen (dargestellt als Vererbungsbeziehungen in Abb. 2).

Neben den Basiskonzepten und domänenspezifischen Konzepten gibt es weiterhin zusammengesetzte Konzepte, welche sich aus Komposition bestehender Konzepte zusammensetzen. Diese repräsentieren Muster, die in zugrunde liegenden Implementierungsartefakten gehäuft und in unterschiedlichen Ausprägungen auftreten. Beispiele hierfür sind *Daten empfangen, Daten senden, E-Mail versenden, Kommunikation mit Verzeichnisdienst, Kommunikation mit Web Service* oder *XMeldNachricht bearbeiten.* Das Konzept *Kommunikation mit Verzeichnisdienst* bspw. ist abgeleitet vom Konzept Aktivität. Es umfasst neben dem Aufruf einer Verzeichnisdienstoperation durch Senden und Empfangen von Nachrichten an einen Verzeichnisdienst optional auch eine Fehlerbehandlung, in der die Operation nach vorgegebenen Zeitintervallen n-mal wiederholt wird. Anschließend kann eine Fehlerprotokollierung erfolgen.

In Abb. 3 sind zwei mögliche Ausprägungen des Konzepts *Kommunikation mit Verzeichnisdienst* (auf technischer Integrationsebene) dargestellt. Die Menge der möglichen Ausprägungen eines Konzepts wird durch dessen Variabilität bestimmt. Eine Analyse der Variabilitäten der Konzepte einer Domäne ist Grundlage für die spätere Entwicklung der domänenspezifischen Sprache, da durch sie die Beziehungen der Konzepte, z. B. bestimmte Wechselwirkungen, expliziert werden. Die Erfassung der Variabilitäten kann mithilfe der in [AnCz04] beschriebenen

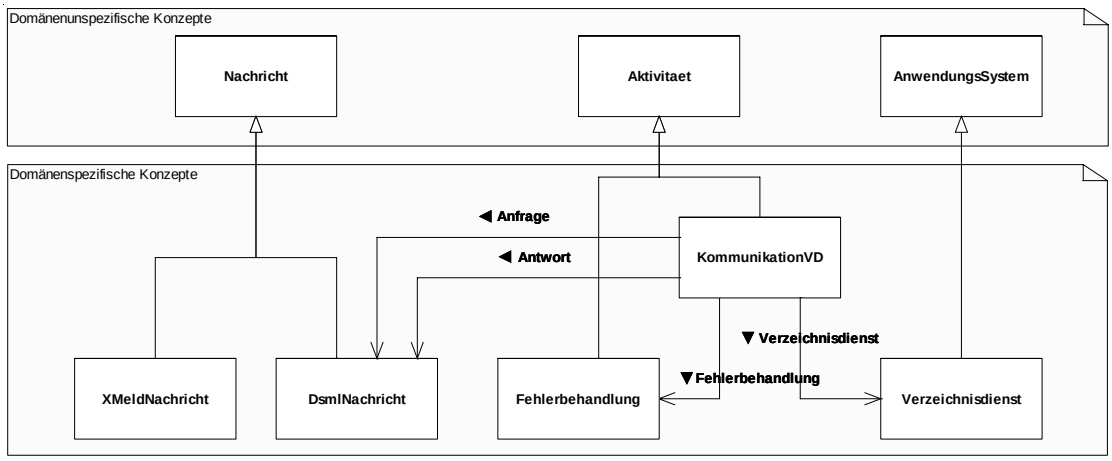

Abb. 2: Konzepte der E-Government-Domäne Meldewesen (Ausschnitt)

Merkmalanalyse durchgeführt werden. Hierfür wird für die Konzepte einer Domäne ein Merkmalmodell erstellt, welches für jedes Konzept ein Merkmaldiagramm enthält. In Abb. 3 (Mitte) ist das zugehörige Merkmaldiagramm für das Konzept *Kommunikation mit Verzeichnisdienst* angegeben. Es enthält das Merkmal Nachrichtenkommunikation, welche aus den Konzepten *Nachrichten senden* und *Nachrichten empfangen* besteht. Abhängig vom Prozesskontext wird dabei eine der aufgeführten Operationen ausgeführt. Ist die optionale Fehlerbehandlung in einer Konfiguration ausgeprägt, so sind hierfür die Anzahl der Anfrageversuche sowie die Wartezeit zwischen den Wiederholungen zu spezifizieren.

Die zusammengesetzten Konzepte sind speziell. Durch die Komposition entstehen Abhängigkeitsbeziehungen von Basiskonzepten, was zur Folge hat, dass zusammengesetzte Konzepte weniger stabil hinsichtlich Änderungen in der Domäne sind. Des Weiteren entsteht Redundanz im Konzeptraum, was zu verringerter Übersichtlichkeit führt. Dennoch wird im MDIE-Ansatz

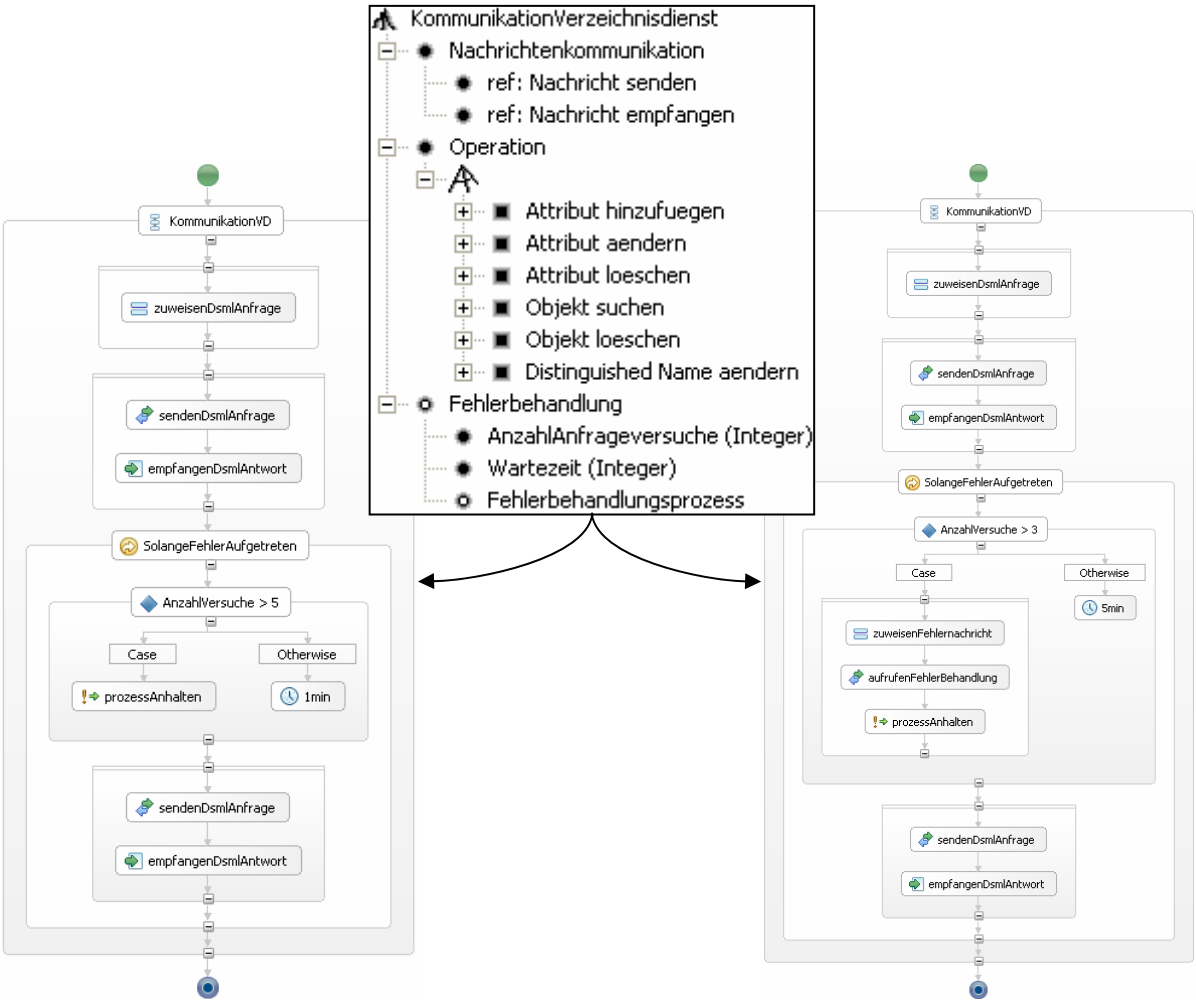

Abb. 3: Merkmaldiagramm des Konzepts *Kommunikation mit Verzeichnisdienst* (erstellt mit dem Eclipse-Plugin fmp [AnCz04]) und zwei verschiedene Ausprägungen (links und rechts)

die Einführung von zusammengesetzten Konzepten bei häufig auftretenden Kombinationen als sinnvoll erachtet. Sie sind die Grundlage für eine kompakte und damit übersichtliche bzw. für Fachexperten intuitive Modellierung.

4.2 Abbildung der Konzepte auf eine domänenspezifische Sprache

Nach der Exploration des Konzeptraums besteht der nächste Schritt in der Definition der domänenspezifische Sprache, welche im Folgenden als $EGov_{TP}$ bezeichnet wird, sowie deren Implementierung in Form eines Editors.

Domänenspezifische Sprachen können in unterschiedlicher Form abgebildet werden. Czarnecki klassifiziert bspw. die Ansätze hinsichtlich Flexibilität und unterscheidet zwischen dem Entscheidungsbaum-basierten Konfigurationsansatz, dem Merkmal-basierten Konfigurationsansatz und der Graph-basierten Modellierungssprache [Czar04].

Für die betrachtete Subdomäne technische Prozessmodellierung im Meldewesen ist die Flexibilität des letztgenannten Ansatzes erforderlich. Für die Definition einer graphischen domänenspezifischen Modellierungssprache existieren leicht- und schwergewichtige Ansätze. Die Grundlage in *leichtgewichtigen Ansätzen* bildet eine universelle Modellierungssprache, welche durch entsprechende Mechanismen erweitert bzw. eingeschränkt werden kann. Durch Vorgabe von Annotationsmöglichkeiten können domänenspezifischen Konzepten in die Sprache eingebracht werden. Ein Beispiel für dieses Vorgehen ist der UML-Profil-Mechanismus, welcher die Definition von Stereotypen und Constraints für Modellierungselemente erlaubt.

Bei *schwergewichtigen Ansätzen* werden grundlegend neue Modellierungssprachen mit Hilfe von Metasprachen definiert. Dieses Vorgehen ist aufwendiger, da hierbei Grundkonzepte nachmodelliert werden müssen, die beim leichtgewichtigen Ansatz bereits durch die Basissprache gegeben sind. Schwergewichtige Ansätze sind jedoch flexibler, da Modellierungselemente über Konstrukte der Metamodellierungssprache (z. B. Vererbungsbeziehungen, Aggregationsbeziehungen) präziser und vielschichtiger definiert werden können. Ein weiterer Vorteil schwergewichtiger Ansätze ist, dass bei der Definition der konkreten Syntax der Modellierungssprache größerer Freiraum besteht. So können beispielsweise für Fachexperten intuitive graphische Symbole definiert werden. Zu den schwergewichtigen Ansätzen gehören bspw. Meta Object Facility 2.0 (MOF 2.0) [Obje06], das Eclipse Modeling Framework (EMF) bzw. das darauf aufbauende Eclipse Graphical Modeling Framework (GMF), MetaGME [BGKS06], MetaEdit+ von MetaCase oder Microsoft DSL Tools. Alle Metamodellierungswerkzeuge ermöglichen in ähnlicher Weise das Editieren domänenspezifischen Sprachen, das Erzeugen zugehöriger do-

mänenspezifischer Modellierungswerkzeuge, sowie deren Anwendung bzw. Validierung [At-Kü05].[2]

Im Fall der Sprache $EGov_{TP}$ fiel die Entscheidung auf den schwergewichtigen Ansatz, da er die direkte Abbildung der Domänenkonzepte sowie deren vielfältigen Beziehungen auf Sprachkonstrukte ermöglicht. Die Implementierung der $EGov_{TP}$ erfolgte auf Basis der Eclipse-Werkzeugplattform (insbes. Eclipse EMF und GMF). Die dem EMF-Framework zugrunde liegende Metasprache Ecore [BSME03] ermöglicht die direkte Repräsentation der identifizierten Domänenkonzepte (vgl. Abschnitt 4.1) in einem entsprechenden Metamodell, daher kann Abb. 2 als Ausschnitt aus dem Metamodell gesehen werden. Basierend auf dem Ecore-Metamodell können verschiedene graphische Editoren definiert werden. Hierzu wird für jedes Metamodellelement spezifiziert, in welcher (graphischen) Form entsprechende Instanzen darzustellen sind. Aus diesen Editor-Spezifikationen können mit GMF als Eclipse-Plugin lauffähige Editoren (vgl. Abb. 4) generiert werden. Das in Abb. 4 dargestellte Prozessdiagramm zeigt einen Ausschnitt aus der Beispielmodellierung *einfache Melderegisterauskunft*. Das Konzept *Kommunikation mit Verzeichnisdienst* tritt dabei in den Instanzen *MeldebehoerdeAngeschlossen* und *HoleSchnittstellenInformationen* auf. Ein Vergleich der Instanzen mit Ausprägungen desselben Konzepts auf technischer Integrationsebene (vgl. Abb. 3) zeigt, dass sich auf technischer Prozessmodellierungsebene eine deutlich kompaktere und damit übersichtlichere Modellierung erreichen lässt.

4.3 Weiterverwendung von domänenspezifischen Modellen durch Modelltransformationen

Die technische Prozessmodellierung anhand der domänenspezifischen Sprache erlaubt eine kompakte und für Fachexperten verständliche Modellierung der umzusetzenden Integrationsprozesse. Prozessmodelle bilden anschließend die Grundlage für automatisierte Modelloperationen. Die grundlegendste Operation ist die Modelltransformation, da sie die Wiederverwendung der im Modell repräsentierten Informationen in anderen Kontexten ermöglicht. In [Viss01] werden Transformationen hinsichtlich des Abstraktionsgrad der Quell- und Zielsprache in vertikale und horizontale Transformationen untergliedert. Für den betrachteten Anwendungsfall ist die Abbildung der technischen Prozessmodellierung auf Sprachen der Integrationsebene (z. B. BPEL4WS, WSDL) relevant. Es handelt sich dabei um vertikale Modelltransformationen (Refinement) einer High-Level-Modellierungssprache auf Low-Level-Sprachen.

[2] Atkinson und Kühne verweisen auf prinzipielle Probleme, z. B. Inkompatibilitäten zwischen Modellen und veränderter domänenspezifischer Modellierungssprache. Dennoch ist dieser Ansatz derzeit stark verbreitet.

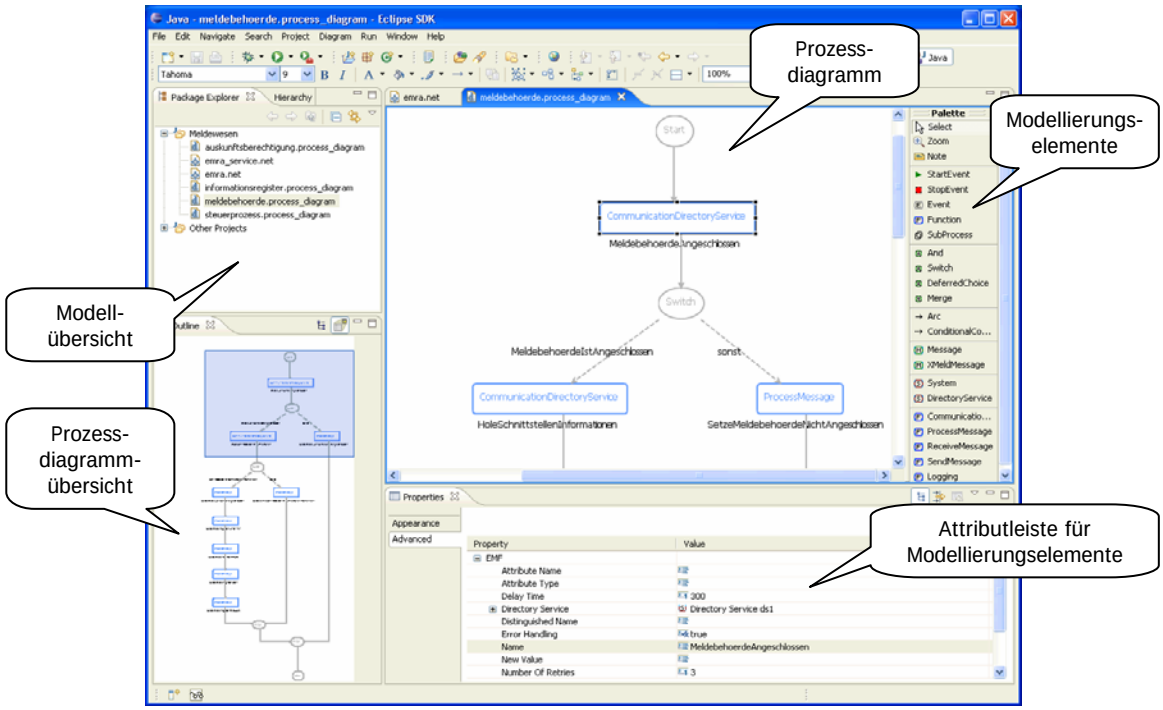

Abb. 4: Graphisches Modellierungswerkzeug für die technische Prozessmodellierungssprache $EGov_{TP}$

Die Ergebnisse der Domänenanalyse sind die Grundlage für die Definition von Transformationsregeln, welche die Abbildungsvorschriften von $EGov_{TP}$-Modellen auf Sprachen der Integrationsebene definieren. Im MDIE-Ansatz fließen die Konzepte der Domänenanalyse direkt in die domänenspezifische Modellierungssprache ein. Die Merkmaldiagramme explizieren die den Konzepten zugrunde liegenden Variabilitäten. Den Ausprägungen dieser Merkmaldiagramme (Konfigurationen) werden konkrete Ausprägungen wie in Abb. 3 zugeordnet (vgl. hierzu [CzAn05]). Die Beziehungen der technischen Prozessmodellierungen (z. B. Kontrollflusskanten) geben an, wie diese Konzeptausprägungen in den Sprachen der Integrationsebene zu verknüpfen sind. In [CzHe06, MeGo05] werden verfügbare Transformationstechniken und -werkzeuge für eine konkrete Umsetzung dieser Transformationsstrategie klassifiziert.

5 Zusammenfassung

In diesem Artikel wurde der MDIE-Ansatz vorgestellt, welcher eine Anwendung modellgetriebener Techniken auf den Anwendungsbereich des Integration Engineering darstellt. Der Schwerpunkt lag dabei auf der Integration domänenspezifischer Konzepte in Prozessmodellierungssprachen zur Reduktion der Modellierungskomplexität. Die Anwendung des MDIE-

Ansatzes in der E-Government-Domäne Meldewesen zeigte exemplarisch, dass die angewendeten Abstraktionsmechanismen die Komplexität der zugrunde liegenden Integrationssprachen reduzieren können. Es wurde eine Transformationsstrategie skizziert, welche die direkte Weiterverwendung repräsentierter Sachverhalte ermöglicht. Andere Modelltransformationen wie die Extraktion bestimmter Modellaspekte (Systeme, Schnittstellen, Nachrichtenformate) zur Analyse bzw. andere Modelloperationen wie Modellvalidierung oder Modelldifferenzen sind gleichfalls denkbar.

Für eine umfassende Bewertung des MDIE-Ansatzes sind jedoch weiterführende Arbeiten erforderlich. So sind für quantitative Kosten-/Nutzenbewertungen empirische Studien durchzuführen. Des Weiteren sind Vorgehensaspekte der MDIE-Artefakte, z. B. der Weiterentwicklung domänenspezifischer Sprachen im betrachteten Bereich sowie der Abstimmung zwischen verschiedenen domänenspezifischen Sprachen (DSL-Management) zu bestimmen.

Literaturverzeichnis

[AnCz04] Antkiewicz, M.; Czarnecki, K.: FeaturePlugin: Feature Modeling Plug-In for Eclipse. In: OOPSLA'04 Eclipse Technology eXchange (ETX) Workshop, http://www.swen.uwaterloo.ca/~kczarnec/etx04.pdf, 2004.

[AtKü05] Atkinson, C.; Kühne, T.: Concepts for Comparing Modeling Tool Architectures. In: ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems, MoDELS / UML 2005, http://www.mm.informatik.tu-darmstadt.de/staff/kuehne/publications/papers/concepts.pdf, 2005.

[Bézi05] Bézivin, J.: On the Unification Power of Models. In: Software and System Modeling (SoSym) 4 (2005) 2, S. 171–188.

[BGKS06] Balasubramanian, K.; Gokhale, A.; Karsai, G.; Sztipanovits, J.; Neema, S.: Developing Applications Using Model-Driven Design Environments. In: IEEE Computer, Februar 2006, 2006, S. 33–40.

[Birk05] Birker, K.: Das neue Lexikon der BWL. Cornelsen, 2005.

[BSME03] Budinsky, F.; Steinberg, D.; Merks, E.; Ellersick, R.; Grose, T. J.: eclipse Modeling Framework. Addison-Wesley (the eclipse series), 2003.

[Bund06] Bundesamt für Sicherheit in der Informationstechnik (BSI), Projektgruppe EGovernment (Hrsg.): Das E-Government-Glossar: Pragmatische Definitionen, Begriffserläuterungen und Abkürzungsverzeichnis. http://www.bsi.de/fachthem/egov/download/6_EGloss.pdf, 2006-01.04, Abruf am 2006-11-15.

[CzAn05] Czarnecki, K.; Antkiewicz, M.: Mapping Features to Models: A Template Approach Based on Superimposed Variants. In: Generative Programming and Component Engineering (GPCE 2005), 2005, S. 422–437.

[Czar04] Czarnecki, K.: Overview of Generative Software Development. In: Banâtre, J.-P.: Unconventional Programming Paradigms (UPP) 2004. Springer, LNCS 3566, 2004, S. 313–328.

[CzEi00] Czarnecki, K.; Eisenecker, U. W.: Generative Programming: Methods, Tools, and Applications. Addison-Wesley, 2000.

[CzHe06] Czarnecki, K.; Helsen, S.: Feature-based survey of model transformation approaches. In: IBM Systems Journal 45 (2006) 3, S. 621–645.

[FaNg05] Favre, J.-M.; NGuyen, T.: Towards a Megamodel to Model Software Evolution Through Transformations. In: Electronic Notes in Theoretical Computer Science 127 (2005) 3, S. 59–74.

[FäTW05] Fähnrich, K.-P.; Thränert, M.; Wetzel, P. (Hrsg.): Umsetzung von kooperativen Geschäftsprozessen auf eine internetbasierte IT-Struktur: Arbeiten aus dem Forschungsvorhaben Integration Engineering. Leipziger Beiträge zur Informatik. Band III, Eigenverlag Leipziger Informatik-Verbund (LIV), 2005.

[FFSD06] Fötsch, D.; Feja, S.; Sauer, S.; David, A.: Problemstellungen agiler Schnittstellen am Beispiel des Commerce Management Systems von Truition/AGETO. In: [FKSW06], S. 49–56.

[FKSW06] Fähnrich, K.-P.; Kühne, S.; Speck, A.; Wagner, J.: Integration betrieblicher Informationssysteme, Problemanalysen und Lösungsansätze des Model-Driven Integration Engineering. Eigenverlag Leipziger Informatik-Verbund (LIV), Leipzig, 2006.

[GrSh04] Greenfield, J.; Short, K.: Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools. Wiley Publishing, 2004.

[HäKü06] Hänsgen, P.; Kühne, S.: Modellgetriebene Softwareentwicklung zur Lösung von Integrations- und Migrationsproblemen am Beispiel des E-Commerce-Systems Intershop Enfinity. In: [FKSW06], S. 31–41.

[Kbst05] KBSt: SAGA Version 2.1: Standards und Architekturen für E-Government-Anwendungen Koordinierungs- und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung (KBSt), http://www.kbst.bund.de, 2005, Abruf am 2006-11-15.

[KSLB03] Karsai, G.; Sztipanovits, J.; Ledeczi, A.; Bapty, T.: Model-integrated development of embedded software. In: Proceedings of the IEEE. 91 (2003) 1, S. 145–164.

[KuRa96] Kurbel, K.; Rautenstrauch, C.: Integration Engineering: Konkurrenz oder Komplement zum Information Engineering? – Methodische Ansätze zur Integration von Informationssystemen. In: Heilmann, H. (Hrsg.): Information Engineering: Wirtschaftsinformatik im Schnittpunkt von Wirtschafts-, Sozial- und Ingenieurwissenschaften. Oldenbourg, 1996, S. 167–191.

[LeRK06] Lehmann, J.; Rotzoll, W.; Kühne, S.: Analyse der E-Government-Domäne Meldewesen am Beispiel des Dienstes „einfache Melderegisterauskunft“. In: [FKSW06], S. 20–30.

[LeRo06] Lehmann, J.; Rotzoll, W.: E-Government-Dienste im Meldewesen Mecklenburg-Vorpommern. Interne Entwicklerdokumentationen, 2006.

[MeGo05] Mens, T.; Gorp, P. v.: A Taxonomy of Model Transformation. In: International Workshop on Graph and Model Transformation (GraMoT) 2005. 2005, S. 7–23.

[Obje06] Object Management Group: Meta Object Facility (MOF) Core Specification: Version 2.0, Object Management Group, http://www.omg.org/cgi-bin/doc?formal/2006-01-01, 2006-01-01, Abruf am 2006-11-15.

[ObjeoJ] Object Management Group: OMG Model Driven Architecture. http://www.omg.org/mda, Abruf am 2006-11-15.

[Osci01] OSCI Leitstelle (Hrsg.): OSCI: Die informelle Beschreibung – Eine Ergänzung zur OSCI Spezifikation. Technischer Bericht, http://www.osci.de/materialien/summary.pdf, 2001, Abruf am 2006-11-15.

[Osci06] Projektgruppe OSCI-XMeld (Hrsg.): OSCI–XMeld 1.3.0. Spezifikation. http://www.osci.de/xmeld130/xmeld-130.zip, 2006, Abruf am 2006-11-15.

[Raut93] Rautenstrauch, C.: Integration Engineering: Konzeption, Entwicklung und Einsatz integrierter Softwaresysteme. Addison-Wesley, 1993.

[Schm06] Schmidt, D. C.: Model-Driven Engineering. In: IEEE Computer 39 (2006) 2, S. 25–31.

[SDTK05] Specht, T.; Drawehn, J.; Thränert, M.; Kühne, S.: Modeling Cooperative Business Processes and Transformation to a Service Oriented Architecture. In: 7th International IEEE Conference on E-Commerce Technology, 2005, S. 249–256.

[StVö06] Stahl, T.; Völter, M.: Model-Driven Software Development: Technology, Engineering, Management. Wiley Publishing, 2006.

[ThKü06] Thränert, M.; Kühne, S.: Model-Driven Integration Engineering und seine Anwendung im Projekt OrViA. In: [FKSW06], S. 13–19.

[Thrä05] Thränert, M.: Integration – Eine Begriffsbestimmung. In: [FäTW05], S. 11–22.

[Viss01] Visser, E.: A Survey of Rewriting Strategies in Program Transformation Systems. In: Electronic Notes in Theoretical Computer Science 57 (2001) 2.

[WeSo05] Wetzel, P.; Soutchilin, A.: Integration Engineering im Maschinen- und Anlagenbau. In: [FäTW05], S. 23–33.