Modellgetriebene Geschäftsprozessautomatisierung

STEPHAN ROSER

TR 2005-9

(basierend auf der Diplomarbeit von Stephan Roser)

Universität Augsburg

Institut für Informatik

April 2005

Institut für Informatik
Universität Augsburg
D-86135 Augsburg, Germany
www.informatik.uni-augsburg.de/vs

Kurzfassung

Die Überwachung, Steuerung und Optimierung von Geschäftsprozessen sind heutzutage kritische Erfolgsfaktoren eines Unternehmens. Darüber hinaus erfordern neue Anforderungen an ein Unternehmen, die beispielsweise durch die Anpassung an neue Kunden oder Geschäftspartner entstehen, oft eine schnelle Änderung von bestehenden komplexen Geschäftsprozessen. Ziel ist es daher, schnell ausführbare Prozesse bereitzustellen und schnell auf Prozessänderungen reagieren zu können. In dieser Arbeit wird als möglicher Lösungsansatz eine modellbasierte Entwicklung bzw. Architektur eingesetzt. In einer modellbasierten Entwicklung werden jedoch für die Abbildung der Geschäftsprozesse eines Unternehmens und der Modellierung eines Informations- und Kommunikationssystems oft unterschiedliche und nicht hinreichend integrierte Modellierungsansätze verwendet. Dies führt zu einem ‚Methodenbruch' bei der Modellierung, so dass die Systemfunktionalität eines Informations- und Kommunikationssystems die Prozesse eines Unternehmens nicht korrekt oder unzureichend abbildet.

Diese Arbeit beschreibt, wie prozessorientierte Modelle eines Unternehmens auf business-level in möglichst äquivalente Modelle von Informations- und Kommunikationssystemen umgewandelt werden können. Dazu wird zunächst erläutert, wie mit dem Business Process Definition Metamodel, einer Erweiterung der Unified Modeling Language zur Geschäftsprozessbeschreibung, Unternehmensabläufe aus einer prozessorientierten Sicht beschrieben werden können. Als Hauptbeitrag dieser Arbeit wird anschließend eine Abbildung zwischen der Architektur integrierter Informationssysteme und dem Business Process Definition Metamodel entwickelt und mit 120 Abbildungsregeln formell spezifiziert. Schließlich illustriert eine Fallstudie die Anwendung der entwickelten Abbildung anhand eines Praxisbeispiels. Die in dieser Arbeit mit der Abbildung entwickelten Modelle stellen eine durchgängige und konsistente Prozessmodellierung von Fachabteilungen hin zu einem Informations- und Kommunikationssystem sicher und können eine Basis für eine maschinenlesbare Beschreibung von automatisierbaren Geschäftsprozessen bilden.

Inhaltsverzeichnis

1 Einführung

1.1 Motivation

Ende der 80er und Anfang der 90er Jahre sahen sich viele Unternehmen aufgrund der Öffnung der globalen Märkte und dem Wegfall von juristischen und traditionellen Handelsbeschränkungen mit einem steigenden Wettbewerbsdruck konfrontiert [Dav01]. Höhere Anforderungen bezüglich Lieferzeit, Produktionskosten und Qualität ihrer Produkte verlangten nach effizienteren, effektiveren und flexibleren Organisationsformen. Als Konsequenz wurden die Prozesse der Unternehmen als die zu optimierenden unternehmenskritischen Faktoren identifiziert. Viele Unternehmen, insbesondere solche mit funktionalen oder divisionalen Organisationsstrukturen, mussten die Art, wie sie die Leistungserstellung betrachteten, wie sie arbeiteten und organisiert waren, ändern. Die vertikale Ausrichtung der Organisationsstruktur behinderte vielfach den horizontalen Fluss der Prozesse, die zur Produktion, deren Planung und Steuerung notwendig sind. Als Lösung wird eine horizontal ausgerichtete Prozessorganisation gesehen. Diese rückt die Prozesse eines Unternehmens in den Mittelpunkt der Betrachtung und weist eine starke Marktorientierung auf. Dazu kommt eine unternehmensübergreifende Betrachtung, die vor allem Lieferanten- und Kundenbeziehungen umfasst, und eine informationelle Vernetzung, durch die neuartige unternehmensübergreifende Strukturen ermöglicht und unterstützt werden [Sei02].

Betrachtet man Informations- und Kommunikationssysteme (IuK-Systeme), die zur Überwachung, Steuerung und Optimierung der Unternehmensprozesse dienen sollen, so stellt man fest, dass deren Entwicklung und Gestaltung oft nicht aus einer prozessorientierten Sicht erfolgte und zum Teil immer noch nicht erfolgt. Dies liegt vor allem daran, dass viele Modellierungsmethoden, die zum Systementwurf verwendet werden, über keine ausgefeilte Prozessdarstellung oder Diagramme zur Beschreibung von organisatorischen Aspekten und Leistungsflüssen verfügen [Sch98b]. So werden für die Abbildung der Geschäftsprozesse eines Unternehmens und der Modellierung des IuK-Systems oft unterschiedliche und nicht hinreichend integrierte Modellierungsansätze verwendet. Dieser ‚Methodenbruch' führt bei vielen IuK-Systemen dazu, dass deren Systemfunktionalität die Prozesse des Unternehmens nicht korrekt, unzureichend oder überhaupt nicht abbilden kann.

Geschäftsprozesse stellen, wie eingangs schon erwähnt, einen kritischen Erfolgsfaktor für Unternehmen dar. Außerdem sind Geschäftsprozesse ständigen Änderungen und Anpassungen unterworfen. Deshalb ist es wünschenswert, die Prozesse eines Unter-

nehmens flexibel und korrekt in einem IuK-System abzubilden. Nur so kann eine effiziente und effektive Überwachung, Steuerung und Optimierung der Geschäftsprozesse mit Hilfe von IuK-Systemen erreicht werden. Sind anfangs zur Steuerung der Prozesse noch manuelle Interaktionen nötig, so können diese schrittweise in den IuK-Systemen immer weiter automatisiert werden.

Für die Entwicklung von IuK-Systemen lassen sich aufgrund des eingangs skizzierten Trends mehrere kritische Faktoren identifizieren [Fra03]. Die Qualität von IuK-Systemen muss erhöht werden, insbesondere im Hinblick auf eine korrekte Abbildung von Geschäftsprozessen. Ebenso muss die Lebenszeit der IuK-Systeme verlängert werden, indem sie flexibel an neue Geschäftsprozesse und auch Systemplattformen angepasst werden können. Nicht zuletzt müssen aufgrund eines steigenden Wettbewerbsdrucks die Entwicklungszeiten und -kosten gesenkt werden.

Zwar werden IuK-Systeme auch schon zum heutigen Stand der Technik mit Hilfe von Modellen entwickelt, aber diese Modelle sind oft nicht ausreichend plattformunabhängig [Fra03]. Bei einem Wechsel der Plattform oder der Integration von anderen Plattformen führt dies zu einer aufwändigen Überarbeitung der Modelle und des IuK-Systems. Ähnliche Probleme treten bei Änderungen von Geschäftprozessen eines Unternehmens auf, was oft eine aufwändige Überarbeitung oder einen wiederholten Neuentwurf von IuK-Systemen zur Folge haben kann. Verfügt man jedoch über ein plattformunabhängiges Modell, so kann dieses wieder verwendet werden und muss bei einem Plattformwechsel ‚nur' noch an die neue Plattform angepasst oder bei einer Änderung der Geschäftsprozesse dementsprechend überarbeitet werden. Somit wird zum einen sichergestellt, dass bewährte und erprobte Abbildungen von Prozessen erhalten bleiben, was zu einer höheren Qualität von IuK-Systemen aber auch zu einer längeren Lebenszeit einer schon entwickelten Lösung führt. Zum anderen verringern sich durch die erhöhte Wiederverwendung die Entwicklungszeiten und Entwicklungskosten von IuK-Systemen.

Allein die Erstellung von plattformunabhängigen Modellen ist aber noch nicht genug. Plattformunabhängige Modelle abstrahieren zwar von speziellen, durch unterschiedliche Plattformen bereitgestellte Dienstimplementierungen, nutzen aber deren Dienste, um die Geschäftsprozesse des zu entwickelnden IuK-Systems umsetzen zu können. Folglich ist die Beschreibung aus der Sicht eines plattformunabhängigen Modells auf ein IuK-System nicht gleich einer Beschreibung einer informationstechnologisch unabhängigen Sicht von Fachabteilungen und Prozessverantwortlichen eines Unternehmens auf deren Geschäftsprozesse. Das zu entwickelnde IuK-System wird durch Modelle auf unterschiedliche Abstraktionsebenen beschrieben, die zwar die Prozesse desselben Unternehmens beschreiben, aber zwangsläufig dennoch nicht gleich sind. Analog verhält es sich mit Beschreibungen aus der Sicht eines plattformspezifischen Modells und der Sicht eines plattformunabhängigen Modells auf ein IuK-System. In diesem Zusammenhang erscheint es bei der Entwicklung eines IuK-Systems nur wichtig, dass die entsprechenden Modelle konsistent ineinander übergeführt werden. Wünschenswert ist es also, dass aus informationstechnologisch unabhängigen Geschäftsprozessmodellen plattformunabhängige Modelle abgeleitet werden können, die IT-Aspekte beinhalten. Aus diesen plattformunabhängigen Modellen sollten dann plattformabhängige Modelle abgeleitet werden können, die zugleich als Implementierungsvorlage dienen. Verschiedenen Rollen, die durch unterschiedliche Personen des Unternehmens wahrgenommen werden (Fachbereiche, Prozessverantwortliche, Systemdesigner, etc.), wird es dadurch ermöglicht, die Prozesse aus ihrer Sicht zu beschreiben, ohne dass die verschiedenen Modelle zueinander inkonsistent werden. Dies verlangt nach einer integrierten Vorgehensweise, die unterschiedliche Modelltypen, deren Verknüpfungen und die Ableitung von Modellen aus anderen beschreibt.

Durch ein solches Vorgehen wird die Qualität des IuK-Systems erhöht, indem vor allem die Geschäftsprozesse korrekt in Bezug auf die Modellierung durch Prozessverantwortliche und Fachabteilungen umgesetzt werden. Eine automatische Generierung eines Modells aus Modellen einer höheren Abstraktionsebene verringert zugleich die Entwicklungszeit und den Entwicklungsaufwand eines IuK-Systems.

1.2 Aufgabenstellung und Zielsetzung

Die Diplomarbeit findet in Zusammenarbeit mit der CT IC 6 der Siemens AG statt. Die Siemens AG lässt die unternehmensinternen Prozesse von ihren Gruppen und Regionen nach vorgegebenen Referenzprozessen standardisiert erfassen. Zur Modellierung der Prozesse wird die Architektur integrierter Informationssysteme (ARIS) eingesetzt. Andererseits sollen zur Beschreibung von IuK-Systemen im Wesentlichen UML 2.0 Modelle einschließlich ihrer Erweiterungen durch UML-Profile wie das Business Process Definition Metamodel (BPDM) verwendet werden.

Ziel dieser Arbeit ist es, im Rahmen der Model Driven Architecure (MDA) eine Abbildung (Mapping) zwischen den in ARIS erstellten Prozessmodellen und BPDM-Modellen für IuK-Systeme zu entwerfen. Die Abbildung soll möglichst in Form von Abbildungsregeln von ARIS nach BPDM spezifiziert werden. Sind in den ARIS-Modellen Geschäftsprozesse aus berechnungsunabhängiger Sicht beschrieben, so sollen die zu entwerfenden BPDM-Modelle IT-Aspekte beinhalten und lediglich plattformunabhängig sein. Grundlage der Abbildung sind die im Siemens Referenzprozesshaus zur Modellierung von Geschäftprozessen eingesetzten ARIS-Bestandteile. Dem Entwurf der BPDM-Modelle soll eine prozessorientierte Vorgehensweise zugrunde liegen.

1.3 Aufbau und Inhalt der Arbeit

Diese Arbeit behandelt die Entwicklung und Spezifikation einer Abbildung von in ARIS-Modellen beschriebenen Geschäftsprozessen nach BPDM. Der Aufbau der Arbeit gliedert sich dabei wie folgt:

In Kapitel 2 werden grundlegende Technologien, auf denen die ARIS-BPDM Abbildung basiert, vorgestellt. Dazu gehört die Model Driven Architecture (MDA), die Architektur integrierter Informationssysteme (ARIS), die Unified Modeling Language (UML), das Business Process Definition Metamodel (BPDM) und die serviceorientierte Architektur (SOA).

Das darauf folgende Kapitel 3 widmet sich den Entwurfsprinzipien, auf denen die ARIS-BPDM Abbildung basiert. Dort wird die Modellierung von Geschäftprozessen mit dem BPDM besprochen, um anschließend den methodischen Ansatz zur Entwicklung der ARIS-BPDM Abbildung zu diskutieren. Nachdem die in der Abbildung verwendeten Konzepte von ARIS und BPDM vorgestellt wurden, wird noch ein Überblick über die in Prozessmodellierung mit der ARIS-BPDM Abbildung inklusive der verwendeten Diagrammtypen gegeben.

Schließlich wird in Kapitel 4 die ARIS-BPDM Abbildung besprochen und spezifiziert. Neben dem Entwurf von 120 Transformationsregeln, deren formelle Spezifikation sich größtenteils im Anhang dieser Arbeit befindet, wird eine möglichst äquivalente Umwand-

lung von ARIS-Modellen in BPDM-Modelle unter anderem anhand des ARIS Metamodells erörtert.

Anschließend wird die Anwendung der in Kapitel 4 definierten Abbildung in Kapitel 5 anhand eines Praxisbeispiels besprochen. Dies umfasst neben der Vorstellung des UseCases die Erläuterung der wichtigsten Transformationsregeln anhand von ausgewählten Diagrammen des Praxisbeispiels.

In Kapitel 6 wird zum Schluss eine Zusammenfassung der Ergebnisse dieser Arbeit gegeben und die ARIS-BPDM Abbildung in den Kontext einer modelgetriebenen Entwicklung von IuK-Systemen eingeordnet. Zusätzlich werden der Nutzen und die Vorteile der in dieser Arbeit entwickelnden Abbildung besprochen und den in der Motivation erläuterten kritischen Erfolgsfaktoren für die Entwicklung von IuK-Systemen zugeordnet. Schließlich wird noch ein Ausblick gegeben, der Erweiterungsmöglichkeiten der Abbildung und Arbeit, die über die Abbildung selbst hinausgeht, vorstellt.

2 Grundlagen

Dieses Kapitel umfasst die Vorstellung der grundlegenden Technologien, auf denen die ARIS-BPDM Abbildung basiert. Zunächst wird der Ansatz der Softwareentwicklung, die Model Driven Architecture (MDA), in den die Modelle der Abbildung einzuordnen sind, in Abschnitt 2.1 eingeführt. Anschließend folgt in Kapitel 2.2 eine Erläuterung der Architektur integrierter Informationssysteme, die neben einer Notation auch Methoden zur Geschäftsprozessmodellierung bereitstellt. Abschnitt 2.3 und 2.4 bestehen aus der Beschreibung der Unified Modeling Language (UML) und des Business Process Definition Metamodells (BPDM). Schließlich wird mit der serviceorientierten Architektur in Abschnitt 2.5 noch ein Ansatz besprochen, der es ermöglicht, Prozesse als (Software-) Komponenten zu betrachten.

2.1 Model Driven Architecture

Die Model Driven Architecture (MDA) ist ein von der Object Management Group[1] (OMG) eingeführter Ansatz zur Softwareentwicklung. Grundgedanke (mit dem Ziel effizienter langlebige und qualitativ hochwertigere IuK-Systeme zu produzieren) der MDA ist es, Modelle eines Systems in das Zentrum aller Phasen der Systementwicklung zu stellen. Modelle dienen nicht mehr nur der abstrakten Beschreibung eines Systems, sondern werden auch zur automatisierten Gewinnung weiterer Modelle desselben Systems oder zur Erzeugung der Komponenten (automatische Codegenerierung) eines Systems herangezogen. Durch dieses Vorgehen wird es möglich, Modelle als Spezifikationen für unterschiedliche Problemlösungen wieder verwenden zu können. Softwarekomponenten können automatisch aus technologiebezogenen Modellen gewonnen und mit bereits existierenden Komponenten auf der Basis der im Modell enthaltenen und in Modelldatenbanken gespeicherten Informationen integriert werden. Ändert sich im Laufe der Zeit die Implementierungstechnologie, so werden neue Softwarekomponenten erzeugt. Die zugrunde liegenden die Problemlösung beschreibenden Modelle bleiben gleich. Die MDA sieht als Metadefinitionssprache zur Spezifikation von Modelltypen die von der OMG definierte Meta Object Facility (MOF) vor [BHK04]. Dadurch wird es möglich, die Software-

[1] Die OMG (http://www.omg.org) ist eine non-profit Vereinigung zur Erarbeitung von Spezifikationen für interoperable Unternehmensanwendungen. Zu den Spezifikationen der OMG gehören unter anderem CORBA, UML und MDA.

entwicklung von der Anforderungsanalyse bis hin zur Inbetriebnahme durchgängig mit generischen Entwicklungswerkzeugen zu unterstützen.

Im Rahmen des MDA Development Lifecycles werden Modelle auf unterschiedlichen Abstraktionsebenen entwickelt, die den Kern der MDA bilden [KWB03].

- **Computation Independent Model (CIM):** Dieses Modell bildet die höchste Abstraktionsebene der MDA. Es werden die betriebswirtschaftlichen Abläufe, die Umgebung des Systems sowie die Anforderungen an das System unabhängig von IT-Technologie modelliert. Von Details wie der Struktur und der internen Arbeitsweise eines Systems wird abstrahiert.
- **Platform Independent Model (PIM):** Auch dieses Modell wird auf einer hohen Abstraktionsebene definiert. Es werden Aspekte wie das Design und die Architektur des Systems als auch die Anforderungen an das System unabhängig von jeglicher Implementierungstechnologie beschrieben. Das System wird so modelliert, dass es das Unternehmen am besten unterstützt. Ein PIM kann als langfristiger Wissensspeicher eines Unternehmens angesehen werden, da es für die im CIM erfassten betriebwirtschaftlichen Abläufe und das Domänenwissen plattformunabhängige Problemlösungen bereitstellt (nach [BHK04]).
- **Platform Specific Model (PSM):** Ein PIM wird in ein oder mehrere PSMs transformiert. Für jede spezifische Plattform wird ein PSM in Abhängigkeit der dort verfügbaren Technologie generiert. Da heutzutage viele Systeme auf unterschiedlichen Plattformtechnologien eingesetzt werden, existieren gewöhnlicherweise zu einem PIM mehrere PSMs.
- **Code:** Der letzte Schritt in der Softwareentwicklung ist die Transformation der PSMs zu Code. Dieser Transformationsschritt ist meist eine einfache und direkte Abbildung eines PSMs auf Quellcode, da PSMs an die jeweiligen Implementierungstechnologien gut angepasst sind.

Auch traditionelle Vorgehensweisen sehen den Einsatz von Modellen auf unterschiedlichen Abstraktionsebenen bei der Entwicklung eines Softwaresystems vor. Während die Transformation der Modelle dabei meist manuell erledigt werden muss, soll diese bei der MDA durch Entwicklungswerkzeuge automatisch erledigt werden. Dadurch können Systementwickler auf einem höheren Abstraktionsniveau arbeiten und komplexere Systeme mit niedrigerem Aufwand entwerfen. Nach [KWB03] können bei der Softwareentwicklung durch die MDA die Produktivität, die Portabilität, die Interoperabilität, die Wartung und die Dokumentation verbessert werden.

- **Produktivität:** In traditionellen Ansätzen werden die Modelle unterschiedlicher Abstraktionsebenen manuell auseinander abgeleitet. Ändern sich die Anforderungen an ein System, muss diese Aufgabe stets von neuem erledigt werden. Durch eine automatische Modelltransformation können sich Softwareentwickler auf eine exaktere Beschreibung von Modellen einer höheren Abstraktionsebene konzentrieren, als sich bei jeder Änderung und Anpassung des Systems um plattformspezifische Details kümmern zu müssen.
- **Portabilität:** Um Wettbewerbsvorteile erzielen zu können, sind Unternehmen oft gezwungen, in kurzen Zeitabständen neue Technologien einzuführen. Investitionen in Systeme von Vorgängertechnologien verlieren dabei an Wert oder werden sogar ganz wertlos. Die MDA erhöht das Abstraktionsniveau bei der Beschreibung von Softwaremodellen und ermöglicht durch die automatische Generierung von PSMs aus einem

PIM die Portierung eines einmal auf PIM-Ebene modellierten Systems auf unterschiedlichste Technologieplattformen.

- **Interoperabilität:** Bei der Entwicklung von Software wird meist die Technologie eingesetzt, mit der das zu entwickelnde IuK-System am besten umgesetzt werden kann. Um es auf PSM-Ebene beschriebenen IuK-Systemen zu ermöglichen, Informationen auszutauschen müssen Kommunikationsbrücken manuell entwickelt werden. Da bei der Entwicklung von Systemen mit der MDA in den Modelltransformationen festgehalten wird, aus welchen Konzepten auf PIM-Ebene Konzepte auf PSM-Ebene abgeleitet werden, kann diese Aufgabe von Entwicklungswerkzeugen übernommen werden.
- **Wartung und Dokumentation:** Für die Pflege und Wartung eines IuK-Systems ist eine gute und aktuelle Dokumentation oft unerlässlich oder stellt zumindest eine wichtige Voraussetzung dar. Durch die automatische Generierung von Modellen in der MDA können Abhängigkeiten zwischen diesen verfolgt und Inkonsistenzen vermieden werden. Bei Änderungen des Codes ist es beispielsweise möglich, dass das PSM automatisch angepasst wird. Darüber hinaus gehören high-level Modelle, wie ein PIM, zur Beschreibung des IuK-Systems auf einer hohen Abstraktionsebene.

2.2 Architektur integrierter Informationssysteme

Die Architektur integrierter Informationssysteme (ARIS) stellt ein integriertes Konzept zur Architektur, Methodenauswahl und Werkzeugunterstützung bei der Entwicklung von IuK-Systemen zur Verfügung. ARIS wurde in den neunziger Jahren von Scheer an der Universität Saarbrücken theoretisch entwickelt und von der von ihm gegründeten IDS Scheer AG als Softwareprodukt *‚ARIS-Toolset'* 1993 auf den Markt gebracht [Sei02]. Im Zusammenhang mit ARIS werden vier Anwendungsaspekte unterschieden (nach [Sch98a]):

- Das **ARIS-Konzept** ist ein Rahmenwerk zur Beschreibung von Unternehmen und betriebswirtschaftlichen Anwendungssystemen. Die Komplexität der Geschäftprozessbeschreibung wird durch die Strukturierung in Beschreibungssichten und ein Phasenmodell (Unterteilung der Beschreibungssichten in Ebenen) reduziert. Das Konzept wird als ARIS-Haus (vgl. Abbildung 1) dargestellt.
- Das ARIS-Konzept stellt **Modellierungsmethoden** bereit, die in die Sichten und Ebenen des ARIS-Hauses eingeordnet werden. Zu diesem Zweck wird ihre Metastruktur beschrieben und zu einem detaillierten ARIS-Informationsmodell zusammengestellt.
- Das ARIS-Konzept ist Basis des von der IDS Scheer AG **entwickelten ARIS-Toolsets**. Dieses Softwarewerkzeug unterstützt den Modellierer bei der Erstellung und Verwaltung von Modellen.
- ARIS stellt mit dem **House of Business Engineering** (HOBE) einen Ansatz zum ganzheitlichen computergestützten Geschäftsprozessmanagement bereit. Dies umfasst die Gestaltung von Geschäftsprozessen, deren Planung und Steuerung bis zur Umsetzung durch Workflow-Systeme.

Abbildung 1: ARIS-Haus

Kernbestandteil des ARIS-Konzepts ist die Zusammenfassung von Klassen mit ihren Beziehungen zu Beschreibungssichten. Diese dienen zur Strukturierung und zur Vereinfachung von Geschäftsprozessmodellen. Auch werden durch die Sichteneinteilung Redundanzen, die bei einer Mehrfachverwendung von Objekten innerhalb eines Prozessmodells entstehen, vermieden. Für die einzelnen Beschreibungssichten können sichtenspezifische Modellierungsmethoden verwendet werden, die sich dort besonders bewährt haben [Sch98b]. Die Zerlegung von ARIS-Modellen erfolgt in die im ARIS-Haus (Abbildung 1) dargestellte Funktions-, Organisation-, Daten-, Leistungs- und Steuerungssicht.

- **Funktionssicht:** In der Funktionssicht werden die Aufgaben und Tätigkeiten als Funktionen bzw. Prozesse beschrieben, die zur Umsetzung der Unternehmensziele nötig sind.
- **Organisationssicht:** In der Organisationssicht wird die Aufbauorganisation eines Unternehmens beschrieben. Die Aufbauorganisation befasst sich mit der Strukturierung von Aufgaben, Aufgabenträgern und deren Beziehungen.
- **Datensicht:** Die Datensicht beschreibt die logischen Datenstrukturen eines Anwendungsfalles. Sie enthält Datenobjekte, die von Funktionen manipuliert werden können.
- **Leistungssicht:** Leistungen als Ergebnis von Prozessen werden in der Leistungssicht modelliert. Da Leistungen eng mit der Beschreibung von Prozessen gekoppelt sind, werden diese auch oft in der Steuerungssicht mit den Prozessen modelliert.
- **Steuerungssicht:** Aufgabe der Steuerungssicht ist es, die zunächst getrennt behandelten ARIS-Sichten wieder zu verbinden. Dabei werden sowohl strukturelle Beziehungen als auch das Verhalten des Systems beschrieben. In der Steuerungssicht werden Prozessabläufe modelliert.

Zusätzlich zu der Zerlegung von Geschäftsprozessmodellen in Beschreibungssichten definiert ARIS ein Phasenmodell. Dazu werden die im ARIS-Haus beschriebenen Sichten in die drei Beschreibungsebenen Fachkonzept, DV-Konzept und Implementierung unterteilt. Die Beschreibungsebenen ermöglichen es, die Modelle der einzelnen Sichten mit einem Top-Down Ansatz von einer betriebswirtschaftlichen Fachbeschreibung in Konstrukte der Informations- und Kommunikationstechnik zu transformieren.

2.3 Unified Modeling Language

Die Unified Modeling Language (UML) ist eine Notation und Modellierungssprache zur Beschreibung, Spezifikation, Dokumentation und graphischen Darstellung von Softwareartefakten. Sie entstand in den 90er Jahren im Wesentlichen aus den Ansätzen für objektorientierte Analyse- und Designtechniken von Booch, Rumbaugh und Jacobson. Mit der Unterstützung eines firmenübergreifenden Konsortiums (u.a. Rational, IBM, Microsoft und HP) wurde 1997 die Version 1.0 fertig gestellt. Seit der Annahme von UML durch die OMG 1998 wird die Entwicklung von UML durch eine Arbeitsgruppe der OMG geleitet.

Obwohl die UML Versionen 1.2 - 1.5 mehrmals überarbeitet und erweitert wurden, weisen sie einige Schwächen auf, die zum Teil in den ersten Versionen von UML begründet liegen. Dort sind z.B. die einzelnen Diagrammtypen durch die Metabeschreibung nicht ausreichend getrennt. Durch sehr viele Beziehungen zwischen den einzelnen UML-Modelltypen sind Teile von UML in vielen Fällen nicht allein stehend einsetzbar. Außerdem fehlen einige, von Entwicklern insbesondere zur Modellierung des Systemverhaltens, verwendete Diagrammtypen. Die UML Version 2.0 stellt eine völlig neu überarbeitete Version der Unified Modeling Language dar, deren Veröffentlichung Ende 2004 erwartet wird. Dabei wurden die Spezifikation, das Metamodell und der Abdeckungsumfang von UML 2.0 weitgehend neu definiert [WOe04]. Das Metamodell von UML ist mit Hilfe der MOF[2] beschrieben, wobei es oberstes Prinzip war, mit möglichst wenigen Modellelementen auszukommen und die Abhängigkeiten zwischen unterschiedlichen Modelltypen auf ein Minimum zu reduzieren. Zusätzlich dazu kann der Standardwortschatz der UML durch so genannte UML-Profile erweitert werden, ohne dass das UML Metamodell geändert werden muss. Schließlich wurden in UML 2.0 noch zusätzliche Konzepte und Diagrammtypen aufgenommen, wie z.B. Verteilungs-, Timing- oder Interaktionsübersichtsdiagramme. Die UML deckt nun ein breites Spektrum von Anwendungsgebieten ab und eignet sich zur Modellierung von technischen Systemen ebenso wie zur Modellierung von Informations- und Kommunikationssystemen.

Im Rahmen der Softwareentwicklung mit der MDA bietet sich UML 2.0 als Notation und Modellierungssprache an, da sie einerseits für die Modelle der unterschiedlichen Abstraktionsebenen der MDA jeweils passende Diagrammtypen bereitstellt und andererseits selbst durch die Metabeschreibungssprache MOF ausreichend beschrieben ist. Auf CIM-Ebene können z.B. UseCase-, Aktivitäts- und Interaktionsdiagramme für Business-Modelle und Klassendiagramme für Informationsmodelle verwendet werden. Auf PIM und PSM-Ebene können je nach Problemstellung alle UML 2.0 Diagrammtypen zum Einsatz kommen. Stets wird jedoch die Granularität der Modelle der darüber liegenden Ebene erhöht.

2.4 Business Process Definition Metamodel

Das Business Process Definition Metamodel (BPDM) wurde unter der Federführung von IBM entworfen, um ein abstraktes Modell zur Definition von Geschäftsprozessen bereitzustellen (vgl. [IAF+04]). BPDM ist als UML 2.0 Profil spezifiziert, um es generischen

[2] Die Meta Object Facility (MOF) ist eine abstrakte Sprache zur Beschreibung der Metamodelle von Modellierungssprachen wie z.B. der UML oder der MOF selbst. Die MOF wird von der OMG gepflegt und verwaltet (siehe auch http://www.omg.org /cwm).

UML Werkzeugen im Rahmen des MDA Ansatzes zu ermöglichen, bestehende Modelle zu importieren als auch neu zu erstellen. Das BPDM stellt eine Untermenge von UML 2.0 dar, wobei die Standardkonstrukte von UML 2.0 um weitere Stereotypen in einem UML 2.0 Profil erweitert wurden. Beim Entwurf des BPDM stand vor allem die Aktivität der Geschäftsmodellierung im Vordergrund, in deren Kontext die Modellierung der Geschäftsprozesse stattfindet [IAF+04]. Dabei ist das BPDM in die Teilbereiche *Information*, *Organization*, *Resources*, *Process* und *Automation* aufgeteilt.

In der MDA entspricht das BPDM der Abstraktionsebene eines PIMs. Das BPDM stellt Konzepte zur Geschäftprozessmodellierung bereit, die zum Teil schon in mit UML 2.0, EDOC[3] oder BPMN[4] modellierten berechnungsunabhängigen Modellen (CIMs) genutzt werden [IAF+04].. So zeigt z.B. ein Vergleich in [Whi04] von BPMN und BPDM (d.h. Aktivitätsdiagrammen, die von BPDM zur Modellierung des Kontrollflusses von Workflows eingesetzt werden) die Ähnlichkeit der beiden Ansätze anhand von 21 ausgewählten Patterns[5] zur Modellierung von Geschäftsprozessen. Fast alle Geschäftsprozesspatterns können durch die beiden Notationen ausreichend gut mit modelliert werden. Insgesamt kommt BDMN im Vergleich zu Aktivitätsdiagrammen mit weniger Kernelementen aus und bietet Variationen dieser Element an, um die Komplexität der Geschäftsprozessmodellierung zu handhaben. Obwohl die Zielgruppe von BPDM auch Nutzer aus dem Geschäftsumfeld sind, die sich mehr mit der Erfassung der betriebswirtschaftlichen Zusammenhänge zwischen den Prozessen als mit den Details einer Laufzeitimplementierung befassen, sind die Einflüsse, durch die Ursprünge der UML im Softwareentwurf und dem sehr technisch orientierten Geschäftprozessmodellierungsansatz EDOC, auf die Aktivitätsdiagramme von UML 2.0 nicht verkennbar. Im Vergleich zu BPMN, das explizit für Betriebswirtschaftler entwickelt wurde, ist BPDM mit den UML 2.0 Aktivitätsdiagrammen eher technisch orientiert. Mappings von Notation wie BPMN nach BPDM wurden entwickelt und sind in Entwicklung (siehe Abbildung 2). Darüber hinaus werden auch Abbildungen von BPDM zu laufzeitumgebungsspezifischen Modellen für wie J2EE[6] oder BPEL4WS[7] bereitgestellt. In [IAF+04] finden sich z.B. Ansätze eines Mappings von BPDM zu einem UML 1.5 Profil für die Beschreibung automatisierbarer Geschäftsprozesse als PSMs. Aus einem solchen PSM kann direkt ausführbarer BPEL4WS Code generiert werden (siehe [IAF+04]).

---

[3] Das UML Profile for Enterprise Distributed Object Computing (EDOC) bietet eine standardisierte Notation um Geschäftprozesse mit der UML zu modellieren. Da EDOC eine eher technische Notation darstellt, ist der Nutzerkreis aus dem betriebswirtschaftlichen Umfeld gering. (siehe http//www.omg.org)

[4] Die Business Process Modeling Notation (BPMN) wurde Ende 2002 von Business Process Management Inititative (http://www.bpmi.org) herausgegeben. Sie stellt ein graphische Notation zur Modellierung von Geschäftprozessen in Business Process Diagrammen (BPD) zur Verfügung, die sowohl für Geschäftleute als auch für Entwickler leicht verständlich ist.

[5] Van der Aalst, Hofstede, Kiepuszewski und Barros identifizieren in ihrer Arbeit 21 Patterns zur Modellierung des Verhaltens von Geschäftsprozessen.
(siehe auch http://tmitwww.tm.tue.nl/research/patterns/patterns.htm)

[6] Die Java 2 Platform, Enterprise Edition (J2EE) (http://java.sun.com/j2ee) definiert einen Standard zur Entwicklung von komponentenbasierten mehrschichtigen Geschäftsanwendungen dar.

[7] Die Business Process Execution Language for Web Services (BPEL4WS) definiert eine Notation zur Beschreibung von Geschäftsprozessen nach dem Paradigma der servicesorientierten Architektur. Die aktuelle Spezifikation von BPEL4WS ([Ibm03]) wurde im Mai 2003 herausgegeben und liegt in der Version 1.1 vor.

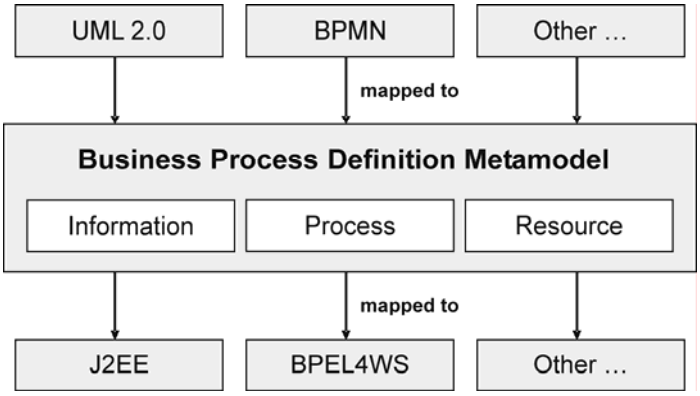


Abbildung 2: BPDM zur Notation von PIMs

2.5 Serviceorientierte Architektur

Wie andere Software Architekturen beschreibt die serviceorientierte Architektur (SOA) die Struktur von Komponenten, die nach außen sichtbaren Schnittstellen und die Beziehungen zwischen Komponenten eines Softwaresystems. In der SOA werden Komponenten als Dienste angesehen, die über ein Netzwerk dynamisch gesucht und genutzt werden können. Dienste, die meist ein Stück Geschäftslogik implementieren und selbst aus eine Menge von Dienstkomponenten bestehen können, kommunizieren miteinander und nutzen die Funktionalität von anderen Diensten. Als ein wesentliches Merkmal trennt die SOA zwischen dem ‚Wie' der Implementierung und dem ‚Was' der Schnittstellen von Diensten. Ein Dienstnutzer sieht einen Dienst als einfachen Zugriffsendpunkt über den er eine bestimmte Funktionalität nutzen kann.

Die serviceorientierten Architektur basiert auf den Interaktionen der Teilnehmer, die drei Rollen (siehe Abbildung 3) einnehmen können: Service Requestor, Service Provider und Service Registry. Als Interaktionen werden im Allgemeinen die Veröffentlichung (publish), das Auffinden (find) und das Binden (bind) eines Dienstes unterschieden. Der Service Provider veröffentlicht eine Beschreibung seines Dienstes bei einer Service Registry (publish). Dort kann der Service Requestor nach geeigneten Diensten, möglicherweise unter Angabe bestimmter Auswahlkriterien, suchen und diese finden (find). Er erhält ein Beschreibung des Dienstes, mit deren Hilfe er den Dienst lokalisieren, kontaktieren und aufrufen kann (bind).

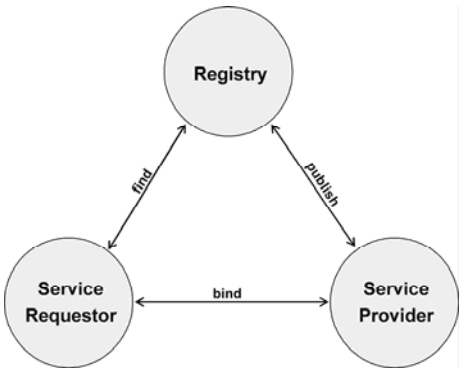


Abbildung 3: Serviceorientierte Architektur

Ein großer Vorteil des SOA besteht darin, dass durch die SOA die Kommunikation zwischen den Fachabteilungen und der IT verbessert werden kann [Dat04]. Wirtschafts-

wissenschaftler, die in Geschäftsprozessen und Geschäftabläufen denken und diese modellieren, müssen ihre Konzepte nicht einer großenteils technisch motivierten Architektur formulieren bzw. ihre Konzepte von IT-Architekten in eine solche übersetzen lassen. Die SOA schafft eine gemeinsame architektonische Grundlage, in der sowohl Wirtschaftswissenschaftler als auch Informatiker modellieren und ihre Modelle diskutieren können.

Darüber hinaus zeichnet sich die SOA durch folgende technische Merkmale aus:

- Dienstkomponenten sind **lose gekoppelt**.
- Dienste können **grobkörnig** beschrieben werden.
- Dienste sind modular und selbstbeschreibend.
- Dienstkomponenten sind **frei kombinierbar**.
- Dienste sind über ein Netzwerk erreichbar.
- Dienste unterstützen die Suche und das dynamische Binden.
- Dienstkomponenten sind **ortstransparent**.
- Die **Interoperabilität** wird erhöht.

3 Entwurfsprinzipien der ARIS-BPDM Abbildung

In diesem Kapitel werden die Entwurfsprinzipien vorgestellt, auf denen die ARIS-BPDM Abbildung basiert. Generell stehen in den Modellen der Abbildung die Geschäftsprozesse von Unternehmen im Mittelpunkt. In Abschnitt 3.1 werden die in der Abbildung verwendeten grundlegenden Konzepte zur Modellierung von Geschäftsprozessen beschreiben. Anschließend wird in Abschnitt 3.2 der methodische Ansatz zur Entwicklung der ARIS-BPDM Abbildung besprochen. Kapitel 3.3 umfasst die Beschreibung der in der Abbildung verwendeten Konzepte von ARIS und BPDM. In Abschnitt 3.4 werden die in den Modellen der Abbildung verwendeten Diagrammtypen besprochen. Schließlich gibt Abschnitt 3.5 mit einem Beispiel einen Überblick über die Prozessmodellierung mit der ARIS-BPDM Abbildung.

3.1 Grundlagen zur Geschäftsprozessmodellierung

Prozesse, die die spezifischen Abläufe eines Unternehmens regeln, werden Geschäftprozesse genannt. Geschäftsprozesse beschreiben die Aktivitäten, die ein Unternehmen zur Erreichung seiner Ziele durchführt und mit denen ein wirtschaftlicher Mehrwert gewonnen wird. Man kann also davon sprechen, dass durch Geschäftprozesse Wertschöpfungsketten implementiert werden. Dabei kommt es zu sowohl unternehmensinternen als auch unternehmensübergreifenden Interaktionen mit Geschäftspartnern. Wie oben bei der Vorstellung von für diese Arbeit wichtigen Technologien motiviert, bietet es sich an für die Modellierung von Geschäftprozessen das Paradigma der serviceorientierten Architektur (SOA) anzuwenden. In Modellen, denen die SOA zugrunde liegt, kann die grundlegende Zusammenarbeit von Geschäftsprozessen mit Partnerprozessen und die Komposition von Geschäftsprozessen leicht beschreiben werden (siehe [FGJ04]). In einer SOA stellen Geschäftsprozesskomponenten die von ihnen implementierten Dienste und Funktionalitäten als Service anderen Geschäftsprozesskomponenten zur Verfügung und treten selbst als Nachfrager auf. Schließlich kann durch die IuK-Systeme von Unternehmen ein Netzwerk von gegenseitig in Beziehung stehenden Geschäftsprozessen aufgebaut werden. Ein und derselbe Geschäftsprozess kann dabei in verschiedenen Geschäftsszenarien verwandt und im Zusammenspiel mit unterschiedlichsten Geschäftspartnern eingesetzt werden.

Im Rahmen der Zusammenarbeit von Geschäftprozesskomponenten wird oft zwischen einer *Orchestration* und einer *Choreography* unterschieden. [Pel03] definiert *Orchestration*

und *Choreography* für die Zusammenarbeit zwischen Geschäftprozesskomponenten folgendermaßen:

- **Orchestration:** Orchestration bezieht sich auf einen ausführbaren Geschäftsprozess, der sowohl mit internen als auch externen Diensten interagiert. Orchestration beschreibt die Interaktionen von Diensten inklusive der Geschäftslogik und die Ausführungsreihenfolge der Interaktionen. In einer Orchestration wird der Gesamtprozess stets aus der Perspektive eines Geschäftspartners kontrolliert.
- **Choreography:** In einer Choreography werden Prozesse kollaborativ beschreiben, wobei jeder an dem Prozess beteiligte Geschäftspartner seine Rolle beschreibt, die er in den Interaktionen einnimmt. Die Choreography legt die Reihenfolge des Nachrichtenaustausches zwischen mehreren Partnern fest. Oft wird Choreography mit dem öffentlichen Nachrichtenaustausch, der zwischen mehreren Diensten stattfindet, in Verbindung gebracht.

Eine Orchestration unterscheidet sich also von einer Choreography, indem sie einen Prozessfluss zwischen Servicekomponenten beschreibt, der von einem einzigen Teilnehmer kontrolliert wird (wie in Abbildung 4 ist der kontrollierende Teilnehmer oft der Gesamtprozess oder auch der globale Prozess). Choreography beschreibt kollaborativ die Reihenfolge des öffentliche Nachrichtenaustausches zwischen mehreren Partnern, wobei keiner der Teilnehmer die Konversation besitzt, indem er den gesamten Prozessfluss kontrolliert (siehe Abbildung 4). Ist in der weiteren Arbeit von Prozess Orchestration oder Prozess Choreography die Rede, so sind diese im eben vorgestellten Sinne zu verstehen. Prozesse stellen dabei eigenständige Dienstkomponenten dar, die in Prozessflüsse eingebunden sind oder diese steuern.

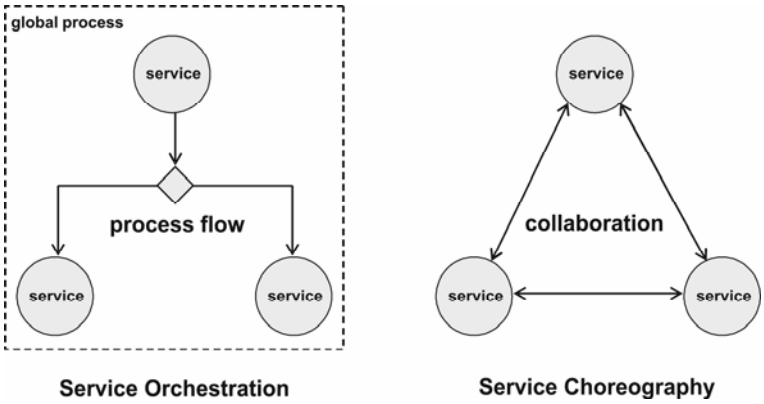


Abbildung 4: Orchestration und Choreography

[FGJ04] unterscheidet bei der Modellierung von Geschäftprozessen und deren Zusammenspiel zwischen einer *internen* und einer *externen Prozesssicht*. Je nach Sicht wird ein Prozess als *ausführbarer*, *abstrakter* oder *kollaborativer* Prozess beschrieben (vgl. Abbildung 5).

- **Ausführbarer Prozess:** In der *internen Sicht* wird das ‚Wie' eines Prozesses in einem dem Modellierer bekannten Detaillierungsgrad beschrieben. Der tatsächliche Prozessablauf ist eine teilweise geordnete Menge von auszuführenden Aufgaben. [Ibm03] bezeichnet solche Prozesse als *ausführbare Prozesse*. Da der durch den ausführbaren Prozess beschriebene Prozess den Ablauf seines internen Verhaltens koordiniert, d.h. beispielsweise den Aufruf von Subprozessen, liegt in einem solchen Fall im Allgemeinen eine Prozess Orchestration vor.

- **Abstrakter Prozess:** Die *externe Sicht* beschreibt das ‚Was' eines Prozesses. Dabei stellt ein Prozess seine Rollen, die er in der Zusammenarbeit mit anderen Prozessen einnimmt, als Schnittstelle zur Verfügung. Diese Schnittstellen werden als *abstrakte Prozesse* bezeichnet. Ein abstrakter Prozess beschreibt zusätzlich seine nach außen sichtbaren Interaktionen die er aufgrund seiner Rolle bei einer Kommunikation durchführt. Er macht aber keine Angaben über die eigentliche Zusammenarbeit mit anderen Prozessen oder seine eigene Realisierung.
- **Kollaborativer Prozess:** Im Falle einer Prozess Choreography wird das Zusammenspiel von abstrakten Prozessen in *kollaborativen Prozessen* beschrieben. Kollaborative Prozesse nutzten abstrakte Prozesse, um den für einen externen Beobachter sichtbaren Nachrichtenaustausch zwischen Prozessen und dessen Reihenfolge zu modellieren. Es wird die Zusammenarbeit zwischen den externen Rollen der beteiligten (Geschäfts-) Prozesse als Interaktionspattern aus der Sicht eines externen Beobachters beschrieben. Kollaborative Prozesse werden nach [Ibm03] auch als *Business Protokolle* bezeichnet, da sie den zwischen beteiligten Teilnehmern stattfindenden und nach außen hin sichtbaren Nachrichtenaustausch wie bei Kommunikationsprotokollen als Interaktionspattern vorgeben.

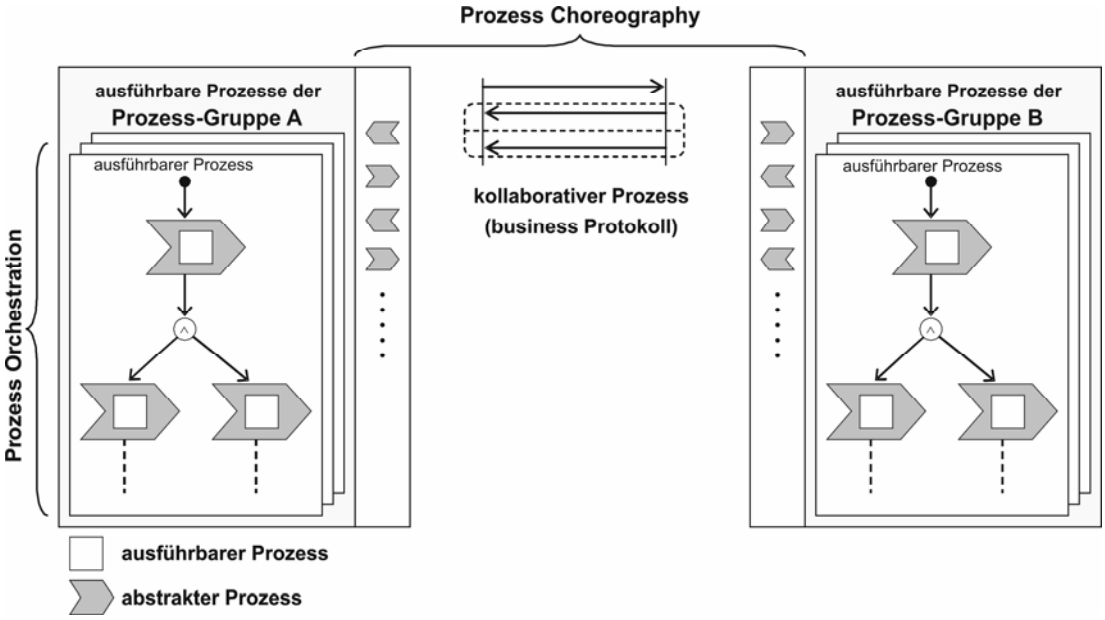


Abbildung 5: Prozessmodellierung im BPDM

Kollaborative Prozesse bzw. Business Protokolle können auf unterschiedliche Art und Weise modelliert werden. Die beiden vielversprechendsten Alternativen werden im Folgenden erläutert. Diese unterscheiden sich in der Architektur, wie der in den kollaborativen Prozessen definierte Nachrichtenaustausch und Kommunikationsfluss gesteuert wird.

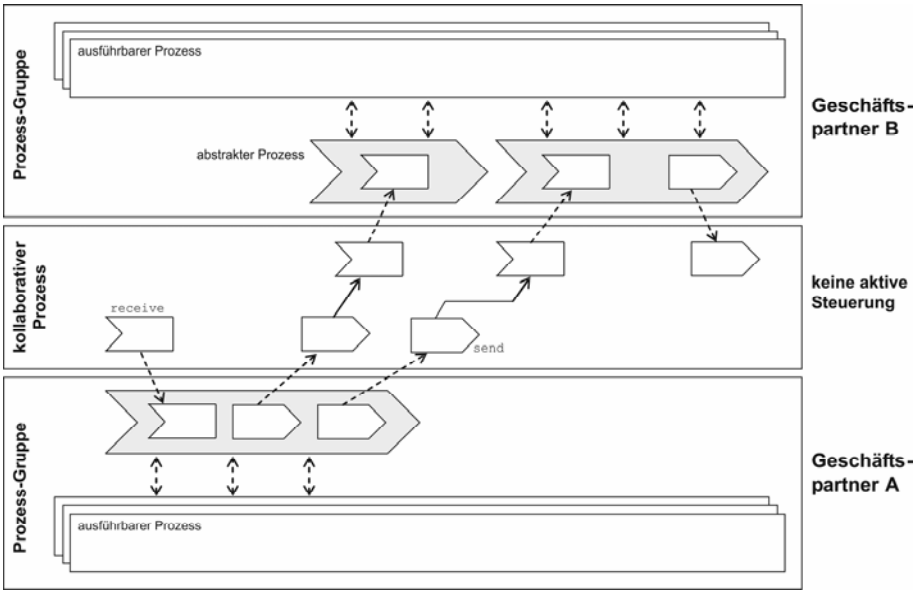


Abbildung 6: Kollaborativer Prozess ohne aktive Steuerung

- In der ersten Alternative stellt ein kollaborativer Prozess nur eine logische Sicht auf unternehmens- bzw. prozessgruppenübergreifende Prozesse dar. Der kollaborative Prozess legt die Reihenfolge des Nachrichtenaustausches zwischen den Kommunikationspartnern fest. Dies geschieht, indem er Verknüpfungen zwischen den abstrakten Prozessen von Nachrichtensender und -empfänger definiert (siehe Abbildung 6). Die Beschreibung des kollaborativen Prozesses besitzt jedoch keine Kontrollelemente (wie z.B. Fallunterscheidungen), die den Ablauf des prozessgruppenübergreifenden Prozesses aktiv steuern können. Solche Entscheidungen werden in den ausführbaren Prozessen, d.h. genauer in den die abstrakten Prozesse implementierenden ausführbaren Prozessen, getroffen. Dies ist insofern problematisch, da die Steuerung des Kontrollflusses des kollaborativen Prozesses auf mehrere prozessgruppeninterne ausführbare Prozesse verteilt wird. Eine Änderung des kollaborativen Protokolls zieht somit im Allgemeinen eine Änderung von mehreren ausführbaren Prozessen oder den Entwurf von Wrapperprozessen nach sich.

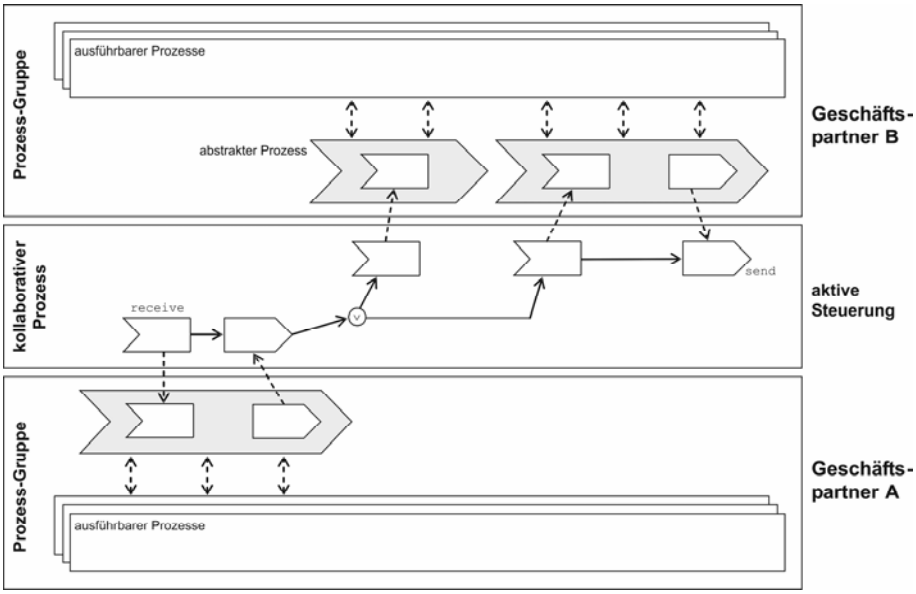


Abbildung 7: Kollaborativer Prozess mit aktiver Steuerung

- In der zweiten Alternative stellt ein kollaborativer Prozess auch eine Sicht auf unternehmens- bzw. prozessgruppenübergreifende Prozesse dar, die aber nicht ausschließlich logischen Charakter hat. Im Gegensatz zur vorigen Alternative handelt der kollaborative Prozess als Intermediär bzw. Broker, der den Kontrollfluss zwischen den Kommunikationspartnern aktiv steuert. Dabei wird durch Kontrollelemente (wie z.B. das ODER in Abbildung 7) entschieden, an welche abstrakten Prozesse eine Nachricht weiterzuleiten ist. Diese Entscheidung könnte z.B. aufgrund des Zustands der in der Nachricht enthaltenen Daten getroffen werden. Eine Änderung des Business Protokolls würde lediglich eine Änderung des kollaborativen Prozesses nach sich ziehen. Die internen ausführbaren Prozesse der Kommunikationspartner könnten unverändert bleiben.

Die in der Diplomarbeit zu entwickelnde ARIS-BPDM Abbildung legt sich nicht für eine der beiden Ansätze fest. Wie wir später sehen werden, werden in ARIS nur ausführbare und abstrakte Prozesse und keine kollaborativen Prozesse modelliert. Trotzdem soll die ARIS-BPDM Abbildung so entwickelt werden, dass die Modelle des BPDMs um kollaborative Prozesse, gleich welchen Ansatz sie verfolgen, erweiterbar bleiben.

3.2 Methodischer Ansatz

Angesichts der eingangs erwähnten Tatsache, dass Experten wie Wirtschaftswissenschaftler und Informatiker ohne das Vorhandensein von disziplinübergreifenden Standards mit der Geschäftprozessmodellierung angefangen haben, ist es nahe liegend, dass unterschiedliche Ansätze zur Geschäftsprozessmodellierung entstanden sind. Bei der Geschäftsprozessmodellierung kann im Rahmen der MDA zusätzlich zwischen einem top-down und einem bottom-up Ansatz unterschieden werden.

- In dem top-down Ansatz werden Geschäftsprozesse zunächst berechnungsunabhängig beschrieben. Unternehmensinterne als auch unternehmensübergreifende Geschäftprozesse werden von Wirtschaftswissenschaftlern modelliert. Um der CIM-Ebene der MDA zuordenbare Geschäftsprozessmodelle zu entwerfen, setzen viele Unternehmen die Modellierungssprache ARIS ein. ARIS verfügt insbesondere um so genannte erweiterte ereignisgesteuerte Prozessketten (eEPKs), mit denen die Prozessabläufe und deren Zusammenhänge mit der statischen Unternehmensstruktur dargestellt werden können.
- Auf der anderen Seite haben Informatiker in den meisten Fällen einen bottom-up Ansatz zur Geschäftsprozessmodellierung gewählt. Ihr Hauptaugenmerk galt nicht der gesamtheitlichen Modellierung der Geschäftsprozesse eines Unternehmens. Stattdessen entwickelnden sie plattformspezifische Modelle (z.B. zur Darstellung von BPEL4WS, BPML[8], ebXML[9], etc.), die eine automatische Prozessausführung erlau-

[8] Die BPML (Business Process Model Language) ist eine Sprache zur Beschreibung von automatisierten Geschäftsprozessen. Die BPML Spezifikation wird von der Business Process Management Initiative (http://www.bpmi.org) gepflegt.

[9] ebXML (Electronic Business using eXtensible Markup Language) ist eine Spezifikation die es Unternehmen unabhängig von ihrer Größe und ihres Standortes ermöglichen soll, Geschäfte über das Internet durchzuführen. Zur Choreography und Durchführung von kollaborativen Geschäftsprossen stellt ebXML ein spezielles Business Process Spezifikation Schema (ebXML BPSS) bereit. (siehe http://www.ebxml.org)

ben. Neuerdings entstehen Beschreibungssprachen wie BPDM für plattformunabhängige Geschäftsprozessmodelle. Diese Modelle können auf plattformspezifische Modelle abgebildet werden. Angesichts der Tatsache, dass sich heutzutage beim Entwurf von IuK-Systemen zur computerbasierten Unterstützung und Steuerung von Unternehmen objektorientierte Systeme durchgesetzt haben, ist es offensichtlich für deren Modellierung UML zu verwenden.

In einer solchen Umgebung ist es für die Entwickler von IuK-Systemen wichtig sicherzustellen, dass prozessorientierte ARIS-Modelle der Wirtschaftswissenschaftler mit objektorientierten BPDM-Modellen von Informatikern konsistent sind. Ein solcher Wechsel der Modellierungsmethode stellt bei dem Entwurf eines IuK-Systems einen kritischen Punkt dar, da an dieser Stelle verursachte Fehler in den Prozessen und anderen Aspekten des Unternehmens oft erst im produktiven Einsatz des IuK-Systems gefunden werden. Auch ein vermehrtes konventionelles Testen des Systems gegenüber den Anforderungen (d.h. den UML-Modellen auf CIM-Ebene) schafft hier kaum Abhilfe, da die objektorientierten Modelle der unterschiedlichen Entwurfsebenen konsistent sein können, obwohl sie Fehler in Beschreibung des Unternehmens enthalten. Es würde sozusagen ein Test gegenüber einer falschen Spezifikation erstellt, wodurch Fehler erst im produktiven Einsatz bemerkt würden. Durch falsch abgebildete Geschäftsprozesse eines Unternehmens wird der Wert eines IuK-Systems zur Prozesssteuerung, -optimierung oder automatischen Prozessausführung im Allgemeinen erheblich gemindert.

In dieser Arbeit wird eine Abbildung von ARIS nach BPDM vorgestellt. Die Abbildung ermöglicht es, BPDM-Modelle auf PIM-Ebene aus ARIS-Modellen auf CIM-Ebene abzuleiten. Die Prozesse auf PIM-Ebene sollen so beschrieben werden, dass sie z.B. in Modelle von Prozessausführungssprachen wie BPEL4WS auf PSM-Ebene transformiert werden können. Dabei muss die entwickelte Abbildung im Wesentlichen zwei Aufgaben erfüllen:

- Die Umwandlung der ARIS-Modelle in möglichst äquivalente objektorientierte UML-Modelle (siehe a1) in Abbildung 8). Dies umfasst neben dem Mapping der relevanten statischen Strukturen vor allem die dynamischen Aspekte von IuK-Systemen. Der Fokus liegt dabei auf der Transformation von prozessorientierten eEPKs, die für die Repräsentation von dynamischen Aspekten in ARIS verwendet werden, in UML-Modelle.
- Eine möglichst automatische Generierung von BPDM-Modellen auf PIM-Ebene aus den schon vorhandenen Modellen auf CIM-Ebene (siehe a2) in Abbildung 8). Die BPDM-Modelle sollen so generiert werden, dass die Anforderungen an das System (wie z.B. vorgegebene Prozessabläufe oder Unternehmensstrukturen) vorgegeben sind, der Systemdesigner aber das Design und die Architektur des Systems anpassen kann.

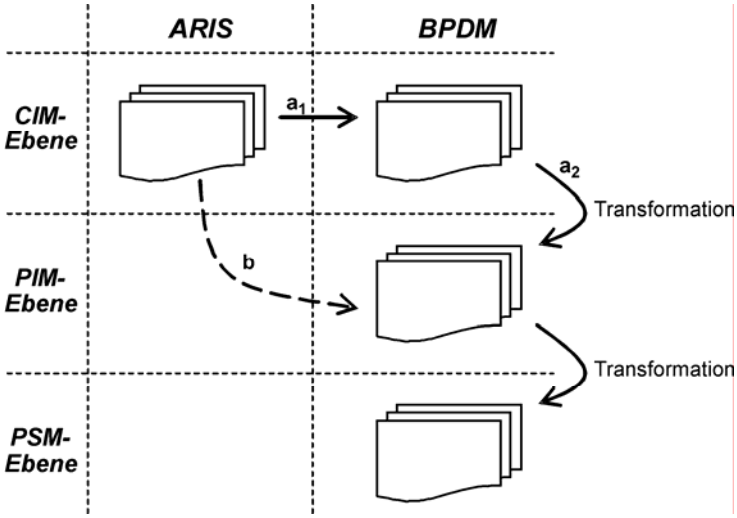


Abbildung 8: Abbildungsschritte

Auch hier ergeben sich wieder zwei Alternativen für die Entwicklung einer ARIS-BPDM Abbildung die beide Vor- und Nachteile aufweisen. In der Alternative a) (siehe Abbildung 8) werden die beiden vorhin identifizierten Aufgabenschwerpunkte in jeweils separaten Schritten durchgeführt. In Alternative b) werden die Transformationen von ARIS nach UML und die Erweiterung der UML-Modelle auf PIM-Ebene in einem einzigen Schritt vorgenommen (vgl. b) in Abbildung 8).

- Für Variante b) spricht, dass die Abbildung von ARIS auf CIM-Ebene nach BPDM auf PIM-Ebene in einem einzigen Schritt durchgeführt wird. Es wird die Erstellung und somit auch die Existenz eines zweiten Modells auf CIM-Ebene vermieden. Der schon in ARIS modellierte Sachverhalt würde nur ein zweites Mal redundant in UML beschrieben. Obwohl eine hinreichend gute Abbildung das Mapping und die Transformation der ARIS-Modelle in einem einzigen Schritt vornehmen könnte, ist ein solches Vorgehen in mehrerlei Hinsicht problematisch. So führt eine parallele und verschachtelte Ausführung des Mappings und der Transformation von ARIS-Modellen zu einer großen, monolithischen und schwer nachvollziehbaren Abbildung. Darüber hinaus würden mit ARIS und BPDM zwei nicht ausreichend integrierte Modellierungssprachen zur Entwicklung eines IuK-Systems eingesetzt. Die Integration von verschiedenen Entwicklungsmodellen ist einer der zentralen Punkte der MDA.
- Variante a) weist diese Nachteile nicht auf. Die Abbildung wird in die oben beschriebenen beiden Aufgaben unterteilt. Dadurch wird die Abbildung für einen Nutzer transparenter und leichter zu verstehen. Es ist besser nachvollziehbar, welche Modellelemente ihren Ursprung in ARIS-Diagrammen haben und welche Modellelemente bei der Transformation zur PIM-Ebene erzeugt wurden. Darüber hinaus können zur Systementwicklung voll integrierte und aufeinander abgestimmte Modelle verwendet werden. Obwohl nun mehrere Modelle auf CIM-Ebene existieren, stellt dies keinen Nachteil der Variante a) dar. Wirtschaftswissenschaftler modellieren in ARIS Geschäftsprozesse aus ihrer Sichtweise. Die UML-Modelle auf CIM-Ebene beschreiben Geschäftprozesse aus einer technischeren Sicht vom Standpunkt eines Systementwicklers. Darüber hinaus können BPDM-Modelle auf CIM-Ebene um Aspekte wie unternehmensübergreifende Geschäftsprozesse erweitert werden, was in ARIS normalerweise nicht vorgesehen ist.

Für die in dieser Arbeit entwickelte Abbildung wird als Vorgehensweise Alternative a) gewählt. Als Zusammenfassung sollen nochmals die Zusammenhänge und Unterschiede

der unterschiedlichen Modelltypen hervorgehoben werden. ARIS-Modelle werden gewöhnlich zur Beschreibung von Unternehmen und deren Geschäftsprozesse auf CIM-Ebene von einem betriebswirtschaftlichen Standpunkt aus eingesetzt. BPDM-Modelle werden zur Modellierung von Unternehmen und Geschäftsprozessen von einem objektorientierten technisch motivierten Standpunkt aus eingesetzt. Trotzdem werden UML- und BPDM-Modelle aufgrund der Bereitstellung von geeigneten Konzepten zur (Geschäfts-) Prozessmodellierung auf CIM-Ebene zur Geschäftsmodellierung eingesetzt. Auf PIM-Ebene werden mit BPDM-Modellen Geschäftsprozesse sowie die für das zu entwickelnde IuK-System relevanten Aspekte des Unternehmens detaillierter beschrieben. Schließlich werden auf PSM-Ebene Modelle für spezielle Plattformen, wie z.B. ein UML 1.5 Profil zur Modellierung von automatisierten Geschäftsprozessen für eine BPEL4WS-Engine, verwendet.

3.3 Kernelemente

Die in der ARIS-BPDM Abbildung verwendeten Technologien und Spezifikationen sind großenteils noch in der Entwicklung oder werden gerade überarbeitet. Da zum jetzigen Zeitpunkt nicht bekannt ist, wie sich diese von den in der ARIS-BPDM Abbildung verwendeten Spezifikationen unterschieden werden, werden die dieser Arbeit zugrunde liegenden Dokumente und Spezifikationen einschließlich ihrer Version im Folgenden festgelegt. Weitere Informationen zu den Dokumenten oder den Herausgebern finden sich im Literaturverzeichnis.

Dokumente zum Siemens Referenz Prozesshaus

- ARIS Konventionen zur Prozessmodellierung Version 2.2,
  Version 2.2 vom 24.10.2003
- Modellierungs-Handbuch Version 1.0,
  Version 1.0 vom 11.03.2004
- Levelkonzept zum Siemens Referenz Prozess Haus Version 1.0,
  Version 1.0 vom November 2003

Dokumente zum Business Process Definition Metamodel

- Business Process Definition Metamodel v1.0.2,
  Version 1.0.2 vom 02.01.2004
- Business Process Definition Metamodel – Concepts and Overview,
  herausgegeben am 08.04.2004

In diesem Abschnitt werden nun noch die in der ARIS-BPDM Abbildung behandelten Konzepte von ARIS und BPDM vorgestellt. Dabei wird der bei Siemens zur Modellierung von Geschäftsprozessen eingesetzte Teil des in [Sch98a] beschriebenen ARIS-Konzepts besprochen. Die Vorstellung des BPDMs basiert auf den eben erwähnten Dokumenten.

3.3.1 Siemensspezifisches ARIS

Sowohl das ARIS-Konzept als auch das ARIS-Toolset beschreiben und unterstützen eine Vielzahl von Modellierungselementen und Diagrammtypen. Beim Einsatz von ARIS in der Praxis beschränkt man sich bei der Modellierung bestimmter Sachverhalte auf eine

Teilmenge der angebotenen Modellierungselemente und Diagrammtypen. Das ARIS-Toolset bzw. die Vorschriften für die zu verwendenden Diagramme werden dem jeweiligen Einsatzzweck angepasst. In diesem Abschnitt werden die von Siemens zur Modellierung von Geschäftprozessen vorgeschriebenen und verwendeten Modellelemente und Diagramme besprochen. Zusätzlich wird mit dem Referenz Prozesshaus ein Modellierungsrahmen zur Erfassung und Modellierung der Siemens internen Prozesse vorgestellt.

3.3.1.1 Diagrammtypen

Folgende Diagrammtypen müssen bei Siemens zur Modellierung von Geschäftsprozessen eingesetzt werden [Sie03].

Wertschöpfungskettendiagramm

Wertschöpfungskettendiagramme (WKDs) werden als Einstiegs- und Überblicksmodelle zur Prozessdarstellung auf einer hohen Abstraktionsebene verwendet (siehe Abbildung 9). Es werden Funktionen bzw. Prozessschritte beschrieben, die direkt an der Wertschöpfung des Unternehmens beteiligt sind. Funktionen können in Form einer Funktionsfolge miteinander verbunden werden und bilden damit eine Wertschöpfungskette. Zusätzlich kann ein Wertschöpfungskettendiagramm eine hierarchische Struktur ähnlich einem Funktionsbaum aufweisen, wodurch eine prozessorientierte Über- und Unterordnung dargestellt wird (nach [Sei02]). In Wertschöpfungsdiagrammen des Siemens RPHs finden sich folgende Modellelemente und Beziehungstypen:

- **Funktion:** Mit dem Modellelement Funktion werden die wertschöpfenden Prozesse in einem WKD modelliert.
- **„ist prozessorientiert übergeordnet“:** Mit diesem Beziehungstyp kann eine Prozesshierarchie beschrieben werden.
- **„ist Vorgänger von“:** Dieser Beziehungstyp modelliert die zeitlich-logischen Prozessfolgen in einem WKD.

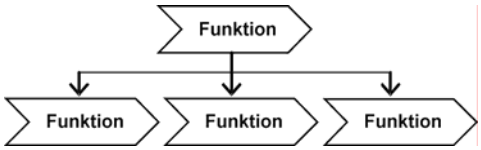


Abbildung 9: Wertschöpfungskettendiagramm

Ereignisgesteuerte Prozesskette

Ereignisgesteuerte Prozessketten (EPKs) beschreiben als Modelle der Steuerungssicht den ablaufbezogenen Zusammenhang von Funktionen (siehe Abbildung 10). Durch das Hintereinanderschalten von Ereignissen und Funktionen entsteht eine zusammenhängende Kette, die den logischen Ablauf eines Prozesses wiedergibt. Dieser Ablauf stellt dar, wie ein betrieblicher Vorgang durch einen Prozess gesteuert wird (nach [Sei02]). EPKs bestehen im Wesentlichen aus folgenden Modellelementen [Ren03]:

- **Ereignis:** Ein Ereignis stellt den Eintritt eines betriebswirtschaftlich relevanten Zustands, der den weiteren Ablauf von Prozessen steuert und beeinflusst, dar. Ereignisse lösen Funktionen aus und sind Ergebnisse von Funktionen. Ein Ereignis ist stets auf einen Zeitpunkt bezogen.

- **Funktion:** Eine Funktion repräsentiert einen Prozess oder eine fachliche Aufgabe zur Unterstützung eines oder mehrerer Unternehmensziele.
- **Schnittstelle:** (Prozess-) Schnittstellen stellen die Durchgängigkeit der Prozessketten untereinander sicher. Sie können zur Kennzeichnung von vor- und nachgelagerten Prozessketten aber auch einer Übergabe des Kontrollflusses an einen externen Prozess dienen. In einer EPK werden sie als Modellelemente wie Funktionen behandelt.
- **Regel UND, ODER und XOR:** Regeln stellen Verknüpfungsoperatoren dar, mit denen die logischen Verbindungen von Ereignissen und Funktionen in Prozessketten festgelegt werden.

Als Beziehungstypen werden in EPKs Kanten zwischen Ereignissen und Funktionen und umgekehrt verwendet. Durch die Zwischenschaltung von Regeln können alternative und parallele Prozessabläufe modelliert werden. Eine ausführliche Auflistung der erlaubten Kombinationsmöglichkeiten von Regeln, Ereignissen und Funktionen findet sich in [Sei02] S. 70ff.

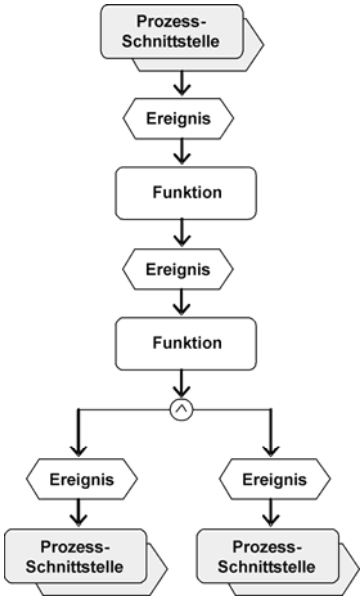


Abbildung 10: Ereignisgesteuerte Prozesskette

Funktionszuordnungsdiagramm

Mit Funktionszuordnungsdiagrammen (FZDs) kann die enge Verbindung zwischen der Funktions- und Datensicht in ARIS modelliert werden (siehe Abbildung 11). Dazu wird die Transformation von Inputdaten zu Outputdaten durch eine Funktion dargestellt. Darüber hinaus werden in FZDs auch die wichtigsten Objekttypen von erweiterten ereignisgesteuerten Prozessketten verwendet [Sei02]. So kann z.B. durch den Objekttyp Personentyp die Verbindung zur Organisationssicht modelliert werden. Folgende (für die Arbeit relevante) Modellelemente und Beziehungstypen sind durch das Siemens RPH definiert:

- **Fachbegriff:** Ein Fachbegriff stellt eine, zur Beschreibung der in einem Unternehmen betrachteten Informationsobjekte verwendete, Begrifflichkeit dar. In FZDs werden Input- und Outputdaten durch Fachbegriffe modelliert.
- **Personentyp:** Ein Personentyp entspricht einer Rolle. Er stellt eine Typisierung einzelner Personen dar, die gleiche Eigenschaften wie Rechte, Verantwortlichkeiten oder Fähigkeiten aufweisen.

- **„ist Input für“:** Beschreibt die Beziehung zwischen einer Funktion und einem Fachbegriff. Der Fachbegriff repräsentiert einen Inputparameter der Funktion.
- **„hat Output“:** Beschreibt die Beziehung zwischen einer Funktion und einem Fachbegriff. Der Fachbegriff repräsentiert einen Outputparameter der Funktion.
- **„führt aus“:** Beschreibt eine Beziehung zwischen einer Funktion und einem Personentyp. Der Personentyp hat Führungsverantwortung bei der Ausführung der Aktivitäten einer Funktion.
- **„wirkt mit bei“:** Beschreibt eine Beziehung zwischen einer Funktion und einem Personentyp. Der Personentyp nimmt aktiv an der Ausführung der Funktion teil.

Abbildung 11: Funktionszuordnungsdiagramm

3.3.1.2 Referenzprozesshaus

Die in diesem Abschnitt enthaltenen Siemens internen Informationen wurden gelöscht.

3.3.2 BPDM

Das Business Process Definition Metamodel ist ein UML 2.0 Profil, mit dem Geschäftsprozesse inklusive der relevanten Informationen, Organisationen, Personen, Business Objekten und Ressourcen abgebildet werden können. Neben den im BPDM definierten Konstrukten identifiziert das BPDM eine notwendige Teilmenge von Elementen des UML 2.0 Metamodells, die zur Beschreibung von Geschäftsprozessen notwendig sind. Das BPDM schließt aber auch keine Konzepte des UML 2.0 Metamodells explizit aus, so dass die volle Ausdrucksmächtigkeit von UML 2.0 dem Nutzer erhalten bleibt. In Abbildung 12 findet sich ein konzeptuelles Modell des BPDMs, in dem ein Überblick über die vom BPDM abgedeckten Bereiche gegeben wird. (nach [IAF+04])

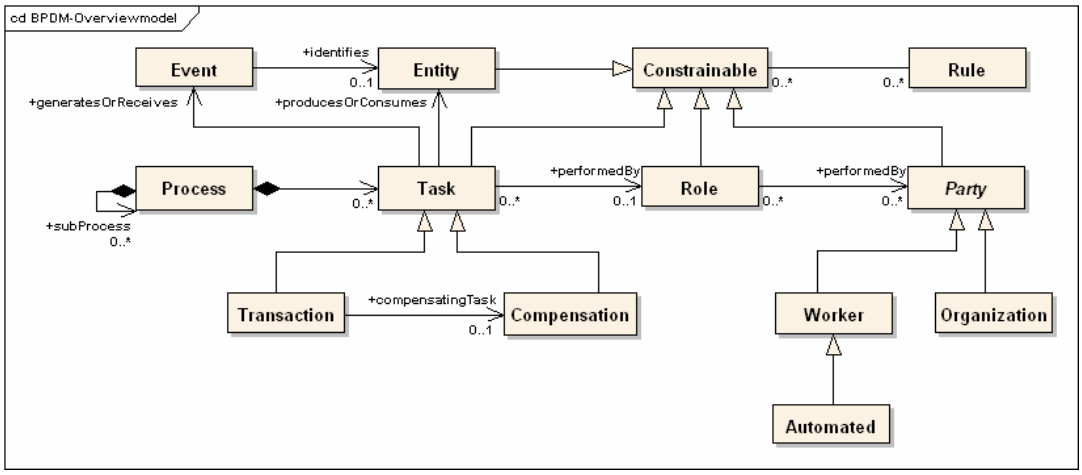


Abbildung 12: BPDM-Überblick

Ein *Prozess* ist ein Container für Zustand und Verhalten. Er stellt nach außen hin sichtbare Schnittstellen und eine Menge von Operationen zur Verfügung. Das Verhalten eines Prozesses kann wiederum aus *Subprozessen* und *Tasks* bestehen. Subprozesse ermög-

lichen die Aufteilung und Wiederverwendung des modellierten Verhaltens. Tasks repräsentieren atomare Schritte in einen Prozess. Ein atomarer Schritt kann eine Berechnung, das Lesen von Zustandsdaten, des Senden eines Events oder der Aufruf eines Subprozesses sein. Darüber hinaus können Tasks eine transaktionelle Semantik besitzen. Mit Hilfe von Compensation Aktivitäten können Transaktionen im Falle eines Fehlers rückgängig gemacht werden.

In der UML 2.0 werden Datenflüsse innerhalb und zwischen Prozessen durch Objektflüsse zwischen Aktivitäten und Aktionen dargestellt. In BPDM werden Informationsobjekte in *Entities* gekapselt, die Business Entities und Business Objects repräsentieren. Business Entities haben Identität, Verhalten und einen eigenen Zustand und können aus betriebswirtschaftlicher Sicht als eine feste Begrifflichkeit verstanden werden. Für die tatsächliche Parameterübergabe zwischen Prozessen sieht BPDM jedoch so genannte *Business Documents* vor (diese sind in dem konzeptuellen BPDM-Modell nicht ersichtlich). Business Documents haben im Gegensatz zu Business Entities keine Identität, kein Verhalten und keinen eigenen Zustand und sind stets einer Business Entity zugeordnet. Ziel dieses Konzeptes ist, dass als Übergabeparameter nicht eine Business Entity selbst dienen muss, sondern Business Documents als Kopien übergeben werden. Dabei enthält ein Business Document stets nur die aus Sicht des Prozesses benötigten oder durch den Zustand der Business Entity vorhandenen Daten.

Das Konzept einer *Rolle* stellt in BPDM den Bedarf eines Tasks nach einer Ressource dar, durch den er erfüllt wird.[10] Eine Organisation (vgl. *Organization*) kann für die Durchführung eines Tasks verantwortlich sein, indem sie die zu ihr gehörigen, zur Durchführung der Aufgabe benötigten Ressourcen zur Verfügung stellt. Aufgaben werden von Menschen (vgl. *Worker*) oder wenn möglich und sinnvoll von automatisierten Systemen (vgl. *Automated*) ausgeführt. Sowohl ein *Worker* als auch eine *Organization* erben von der Klasse *Party*. Es ist zu erwarten, dass das Organisationsmodell des BPDM in naher Zukunft durch eine neue Version ersetzt wird (siehe [IAF+04] Seite 24).

Schließlich sieht das BPDM noch das Konzept von Business Rules (vgl. Klasse *Rule*) vor. Dadurch sollen IuK-Systeme durch die Parametrisierung der Businesslogik flexibel anpassbar gemacht werden. Bei Änderungen im Umfeld des Systems oder der Unternehmensrichtlinien müssen das System oder die Prozesse nicht explizit geändert, sondern können durch einen Wechsel der Business Rules angepasst werden. (Anmerkung: Business Rules sind nicht zu verwechseln mit Business Protokollen.)

Zur Umsetzung der eben vorgestellten Konzepte sieht das BPDM sowohl schon im UML 2.0 Metamodell vorhandene als auch in einem UML 2.0 Profil eigens definierte Konstrukte vor. Eine grafische Darstellung des vom BPDM definierten UML 2.0 Profils findet sich in Abbildung 13.

[10] Im Folgenden wird noch ein zweites, sich von diesem Rollenkonzept unterscheidendes, Konzept einer *‚partnerRole'* eingeführt.

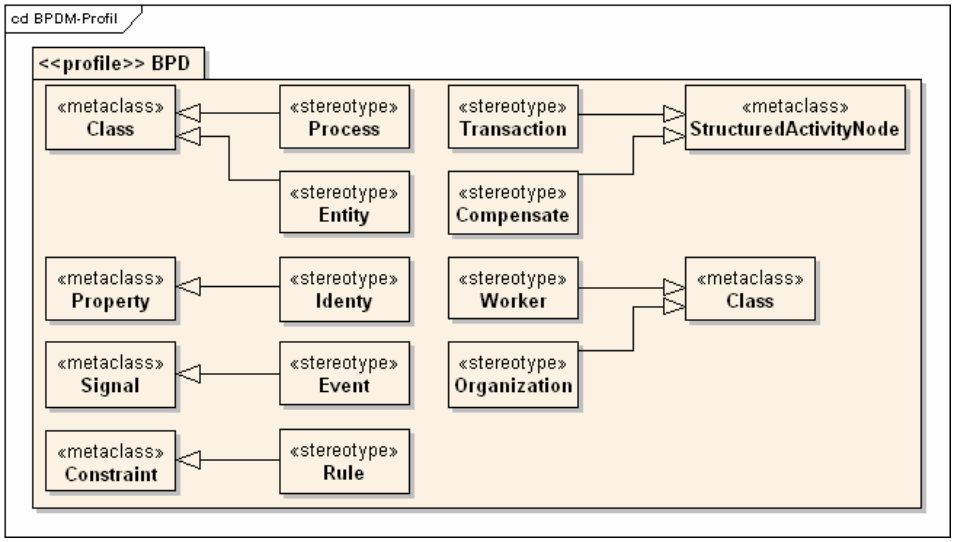


Abbildung 13: BPDM als UML 2.0 Profil

Wie eingangs erläutert, stellt das BPDM Beschreibungsmöglichkeiten für Geschäftprozesse auf einer Abstraktionsebene zur Verfügung, die unabhängig von jeglicher Implementierungstechnologie ist. Dabei werden Konzepte zur Modellierung von ausführbaren Prozessen verwendet. Jedoch werden keine Angaben über die Art der Prozessausführung selbst getroffen. So kann es sich bei einem ausführbaren Prozess um einen automatisierten oder aber um einen manuell durchgeführten Prozess handeln. In der Spezifikation des BPDMs [IAF+04] finden sich keine Konzepte zur Modellierung von abstrakten Prozessen. Die Prozesse werden dort stets aus einer internen Sicht beschrieben. Erst das Whitepaper zum Business Process Definition Metamodell [FGJ04] unterscheidet zwischen einer internen und einer externen Sicht auf Prozesse. In der internen Sicht werden die ausführbaren Prozesse beschrieben, während in der externen Sicht abstrakte und kollaborative Prozesse modelliert werden. Im Rahmen der Abbildung werden die für die ARIS-BPDM Abbildung notwendigen Konzepte zur Beschreibung von abstrakten Prozessen detailliert beschrieben.

Während die Spezifikation des BPDMs [IAF+04] eine Rolle als einen Bedarf nach einer Ressource definiert, sieht das Whitepaper [FGJ04] zum BPDM zwei unterschiedliche Typen von Rollen vor. Wie in [IAF+04] kann eine Rolle eine funktionelle Fähigkeit darstellen, die Aufgaben zu ihrer Erfüllung benötigen. Solche Rollen werden von Ressourcen, d.h. Organisationen, Personen oder Maschinen zur Verfügung gestellt. Anderseits können Rollen in einer komponentenbasierten Sichtweise die Abhängigkeit zwischen einem Prozess und seinen Partnerprozessen ausdrücken. So genannte *‚partnerRoles'* spezifizieren Rollen die von Prozesskomponenten für einen erfolgreichen Kommunikations- und Prozessfluss erfüllt werden müssen. Um diese beiden Ansätze nicht zu verwechseln wird in der weiteren Arbeit des Öfteren explizit hervorgehoben um welches Rollenkonzept es sich handelt.

3.4 Designentscheidungen der Abbildung

Während für die ARIS-BPDM Abbildung die zu verwendenden ARIS Diagramme und Modellelemente durch das Siemens RPH vorgegeben sind, stehen bei der Modellierung mit UML 2.0 und BPDM eine Vielzahl von Diagrammtypen und Modellelementen zur Auswahl. Dieser Abschnitt stellt für die Modellierung relevante Diagrammtypen vor und be-

spricht deren Vor- und Nachteile bezüglich einer Verwendung für die ARIS-BPDM Abbildung. Unter Berücksichtigung der in der MDA definierten Abstraktionsebenen, der Nutzer der unterschiedlichen Modell- und Diagrammtypen und der zu modellierenden Sachverhalte werden die in der Abbildung verwendeten Diagrammtypen identifiziert.

Vor der Betrachtung der einzelnen in der Abbildung vorkommenden Modelle soll noch die Beschreibung von statischen und dynamischen Aspekten mit Hilfe von Diagrammen der UML vorgestellt werden. Zur Modellierung von statischen Strukturen bieten sich Klassendiagramme an, die es erlauben, den inneren Aufbau von Unternehmen und Systemen zu beschreiben. Klassendiagramme können durch z.B. Paket- und Kompositionsstrukturdiagramme ergänzt werden, die Klassen logisch gliedern und Architekturzusammenhänge beschreiben. Zur Modellierung von dynamischen Aspekten sieht die UML eine Reihe von Verhaltensdiagrammen vor. Verhaltensdiagramme unterteilen sich im Wesentlichen in Aktivitätsdiagramme, Interaktionsdiagramme und Zustandsautomaten, die je nach Einsatzzweck unterschiedlich gut geeignet sind, Abläufe zu modellieren.

- **Zustandsautomaten:** Zustandsautomaten ermöglichen die präzise Abbildung eines Zustandsmodells und bieten die Möglichkeit, das Verhalten beliebiger *Classifier* zu modellieren. Dabei wird geklärt, wie sich ein System in einem bestimmten Zustand bei gewissen Ereignissen verhält. Die in der UML verwendeten Zustandsautomaten bilden eine Erweiterung von endlichen Automaten. Deren Möglichkeiten reichen nicht aus, um zum Beispiel komplexe Systeme überschaubar und vollständig auf einer detaillierten Ebene zu beschreiben (siehe [JRH+04]). Da Unternehmen und deren Geschäftsprozesse jedoch oft sehr komplexe Systeme bilden, werden Zustandsautomaten zur Verhaltensmodellierung von Geschäftsprozessen im ARIS-BPDM nicht verwendet. Auch scheinen sie für die Zielgruppen von CIMs nur bedingt geeignet, die die Unternehmensabläufe und Kommunikation innerhalb eines Unternehmens abbilden und optimieren wollen.
- **Aktivitätsdiagramme:** In Aktivitätsdiagrammen wird der Ablauf von Prozessen oder von Algorithmen beschrieben. Dabei steht eine vom System zu bewältigende Aufgabe im Vordergrund, die in Einzelschritte zerlegt wird. Aktivitätsdiagramme lassen sich zur Beschreibung von Prozessen unterschiedlichster Granularität, wie z.B. Systemprozessen oder Geschäftsprozessen, einsetzten. Zur Modellierung, wie das System sein Verhalten realisiert, können dabei sowohl Kontroll- als auch Datenflüsse in einem Aktivitätsdiagramm abgebildet werden. Aktivitätsdiagramme stellen somit eine sehr gute Möglichkeit dar, die internen Abläufe und die Realisierung von Geschäftsprozessen zu modellieren.
- **Interaktionsdiagramme:** Interaktionsdiagramme dienen zur Modellierung der zwischen Kommunikationspartnern ausgetauschten Nachrichten und Daten. Durch die Vorgabe von Interaktionsreihenfolgen, zeitlichen und logischen Ablaufbedingungen wird ein Verhalten zwischen den Kommunikationspartner beschrieben. In diesem Abschnitt wird das Sequenzdiagramm, als das am weitesten verbreitete Interaktionsdiagramm der UML mit dem größten Einsatzbereich, betrachtet. Mit Sequenzdiagrammen kann die Kommunikation in einem System modelliert werden. Der Fokus liegt auf dem Informationsaustausch zwischen beliebigen Kommunikationspartnern innerhalb eines Systems oder zwischen Systemen generell. Sequenzdiagramme beschreiben dabei jeweils ein mögliches Kommunikationsszenario zwischen Kommunikationspartnern. Insbesondere bei komplexeren Abläufen können alle durch ein System realisierten Nachrichtenfolgen nicht mehr durch Sequenzdiagramme oder allgemein Interaktionsdiagramme wiedergegeben werden. Daher sind Sequenzdiagramme weniger

zur Modellierung der internen Abläufe von Geschäftsprozessen geeignet. Die Kommunikation zwischen Geschäftsprozessen, d.h. vor allem über deren Prozessschnittstellen, lassen sich gut durch Sequenzdiagramme beschreiben und visualisieren.

- **Interaktionsübersichtsdiagramme:** Interaktionsdiagramme zeigen das Zusammenspiel verschiedener Interaktionen, indem es Abfolgen von Interaktionen und Interaktionsreferenzen mittels einer Variante des Aktivitätsdiagramms darstellt [JRH+04]. Bei der Beschreibung größerer Systeme in denen viele Komponenten miteinander Kommunizieren entsteht fast unweigerlich ein unüberschaubarer Satz von Sequenz- bzw. Interaktionsdiagrammen. Ein Interaktionsübersichtsdiagramm bietet nun z.B. die Möglichkeit, die Abhängigkeiten und Reihenfolgen zwischen Sequenzdiagramme zu beschreiben. Für die Beschreibung von Geschäftsprozessen bieten sich Interaktionsübersichtsdiagramme an, da mit ihnen die Zusammenhänge zwischen den Interaktionen eines Prozesses mit seinen Kommunikationspartnern und seinem internen Ablauf beschrieben können.

ARIS auf CIM-Ebene

Das Siemens RPH gibt zur Modellierung auf CIM-Ebene drei Diagrammtypen fest vor: Wertschöpfungskettendiagramme, Funktionszuordnungsdiagramme und ereignisgesteuerte Prozessketten. Die ARIS-BPDM Abbildung beschränkt sich auf diese Diagrammtypen und baut gleichzeitig die Abbildungsregeln darauf auf.

BPDM auf CIM-Ebene

Auf CIM-Ebene werden Geschäftsmodelle entworfen, die Aspekte von Unternehmen oder Geschäftsabläufen beschreiben, aber nicht notwendigerweise Angaben über zu verwendende IuK-Systeme machen. Daher werden diese Modelle auch berechnungsunabhängige Modelle genannt. Wann immer Geschäftabläufe durch ein IuK-System unterstützt werden sollen, so müssen zu diesem Zweck spezielle Modelle der Software auf PIM-Ebene erstellt werden. CIM-Modelle können auch als Anforderungsdokumente gesehen werden, die ein logisches Modell eines Systems ohne Berücksichtigung technischer Aspekte beschreiben (siehe [Fra03] & [KWB03]). Hauptzielgruppe der Modelle auf CIM-Ebene sind Business Analysten und Prozess Spezialisten, die für den Entwurf und die Optimierung der Geschäftsmodelle verantwortlich sind. Aber auch für die Geschäftsführung und den Anwendungsentwickler sollen diese Modelle lesbar und verständlich sein.

Aus statischer Sicht werden in der ARIS-BPDM Abbildung bei der Modellierung mit dem BDPM auf CIM-Ebene Klassendiagramme eingesetzt. Mit ihnen können wie oben beschrieben alle wesentlichen Strukturen abgebildet werden. Zur Geschäftsprozessmodellierung aus dynamischer Sicht bieten sich prinzipiell Aktivitätsdiagramme und Sequenzdiagramme an. Im BPDM werden, wie in der serviceorientierten Architektur, Geschäftprozesse als Komponenten umgesetzt [FGJ04]. Daher bietet es sich an zwischen der Beschreibung der internen Realisierung einer Prozesskomponente, und der Kommunikation dieser mit Partnerprozessen zu unterscheiden. Ausführbare Prozesse werden in der ARIS-BPDM Abbildung mit Hilfe von Aktivitätsdiagrammen beschrieben. Die Kommunikation mit Partnerprozessen wird in Sequenzdiagrammen beschrieben. Für Modellierung abstrakter Prozesse ist jedoch auch die Reihenfolgen der Nachrichten relevant. Daher wird ein abstrakter Prozess in einem Interaktionsübersichtsdiagramm das auf die zum Prozess gehörigen Sequenzdiagramme und den ausführbaren Prozess verweist modelliert.

BPDM auf PIM-Ebene

In einem PIM wird ein IuK-System plattformunabhängig beschrieben. Dabei steht das zu entwickelnde System mit seinen technischen Aspekten, wie die Architektur, die Zerlegung in Komponenten oder die Definition von Schnittstellen, im Mittelpunkt der Betrachtung. Ein PIM stellt eine Sicht auf das in einem CIM beschriebenen Unternehmen dar. Es werden die Teile eines Unternehmens unter Berücksichtigung informationstechnischer Gesichtspunkte beschrieben, die durch ein IuK-System unterstützt werden sollen. Obwohl die Zielgruppe von Modellen auf PIM-Ebene hauptsächlich Nutzer aus einem technischen Umfeld, wie Anwendungsentwickler oder Systemarchitekten, umfasst, werden diese Modelle auch von Prozessspezialisten oder Anforderungsanalysten genutzt.

Wie auf CIM-Ebene nutzt die ARIS-BPDM Abbildung Klassendiagramme zur Modellierung der statischen Aspekte des zu entwickelnden Systems. Es werden die Klassendiagramme der CIM-Ebene überarbeitet und um weitere Konzepte erweitert. Wie auch bei allen anderen Diagrammen wird die Granularität der Beschreibung auf niedrigeren Abstraktionsebenen im Vergleich zu höheren stets erhöht. Auch die Diagramme zur Beschreibung des Verhaltens eines Systems werden von dem CIM übernommen. Neben einer weiteren Verfeinerung aller Diagrammtypen werden vor allem die Aktivitätsdiagramme den Datenfluss bei Prozessabläufen erweitert. Aktivitätsdiagramme stellen gleichzeitig die Grundlage dar um ausführbare Prozesse, die automatisiert werden können, in ein Modell für eine Plattform zur automatischen Prozessausführung, wie z.B. BPEL4WS, zu übertragen

3.5 Überblick über die Prozessmodellierung in der Abbildung

Wie eingangs schon erwähnt, nimmt bei der Modellierung von Prozessen die Sichtweise auf einen Prozess eine entscheidende Rolle ein. Je nachdem aus welcher Sicht ein oder mehrere Prozesse beschrieben werden, handelt es sich in den modellierten Diagrammen um ausführbare, abstrakte oder kollaborative Prozesse. In diesem Abschnitt wird vorgestellt, in welcher Form sich die unterschiedlichen Prozesstypen in den in der ARIS-BPDM Abbildung verwendeten Diagrammen wieder finden. Gleichzeitig wird ein Überblick über die dynamischen Aspekte der ARIS-BPDM Abbildung geboten, der auch als Grundlage für die im nächsten Kapitel spezifizierten Transformationsregeln der ARIS-BPDM Abbildung dient.

ARIS auf CIM-Ebene

Prozessabläufe werden im ARIS des Siemens RPHs mit Hilfe von ereignisgesteuerten Prozessketten modelliert. In einer EPK können sowohl der interne Ablauf eines Prozesses als auch dessen externe Schnittstellen modelliert werden. Abbildung 14 zeigt eine EPK eines Prozesses dessen Prozessschnittstellen grau hinterlegt sind und in die Beschreibungen der abstrakten Prozesse eingehen. Ein abstrakter Prozess spezifiziert die Interaktionen eines Prozesses mit Partnerprozessen und deren Reihenfolge. Die Interaktionen werden in ARIS explizit durch Prozessschnittstellen modelliert, während die Interaktionsreihenfolge implizit dem ausführbaren Prozess entnommen werden muss. Der ausführbare Prozess, in Abbildung 14 der nicht grau hinterlegte Diagrammteil, bildet den internen Ablauf des modellierten Prozesses ab. Kollaborative Prozesse sind im Siemens RPH zur Modellierung prinzipiell nicht vorgesehen.

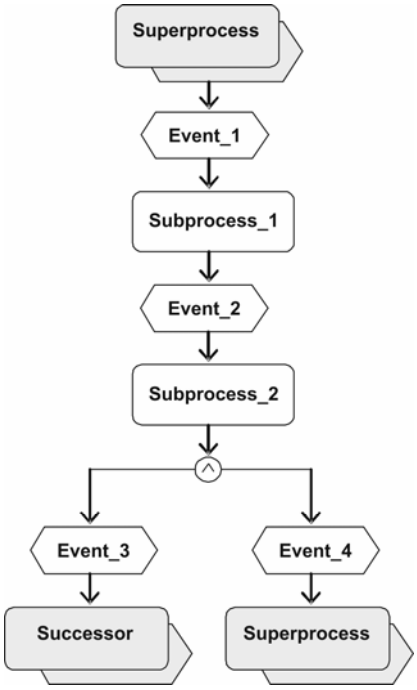


Abbildung 14: ARIS – ereignisgesteuerte Prozesskette

Im Siemens RPH kann sowohl der Prozessfluss von Prozess Choreography Beziehungen als auch von Prozess Orchestration Beziehungen modelliert werden. Abbildung 15 zeigt eine schematische Darstellung zur Modellierung einer Prozess Orchestration. Der Prozessablauf einer Funktion, die in der EPK eines Prozesses als Subprozess modelliert ist, wird durch eine eigene EPK beschrieben. Dabei wird der Vorgängerprozess und Nachfolgerprozess des Subprozesses als Prozessschnittstelle in die EPK des Subprozess übernommen. Auch das Vorgänger- und Nachfolgerereignis des Subprozesses müssen in dessen EPK modelliert werden. Sind diese Voraussetzungen nicht erfüllt, kann davon ausgegangen werden, dass es sich bei der Beziehung zwischen Prozessen um eine Prozess Choreography handelt.

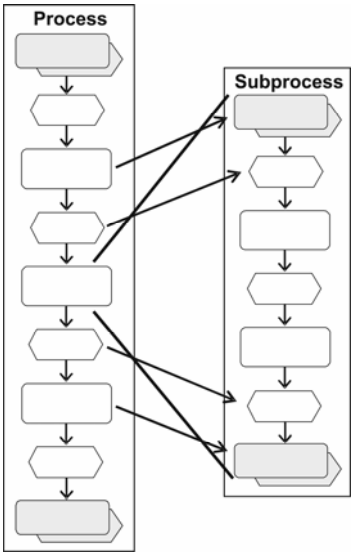


Abbildung 15: ARIS – Prozess Orchestration

BPDM auf CIM-Ebene

Auf CIM-Ebene werden zur Modellierung von Prozessabläufen Aktivitätsdiagramme eingesetzt. Sequenzdiagramme beschreiben die Kommunikation mit Prozesspartnern, während Interaktionsübersichtsdiagramme die Zusammenhängen zwischen Sequenz- und Aktivitätsdiagrammen modellieren. In der ARIS-BDPM Abbildung werden die ausführbaren und abstrakten Prozesse modelliert, die schon in ARIS Diagrammen vorgeben werden. Abbildung 16 zeigt einen ausführbaren Prozess der durch ein Aktivitätsdiagramm modelliert ist.

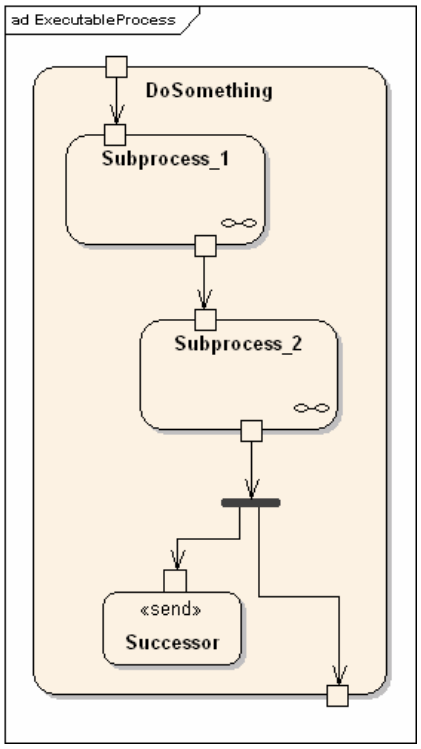


Abbildung 16: BPDM – ausführbarer Prozess des CIMs

Abstrakte Prozesse werden in Interaktionsübersichtsdiagrammen beschreiben (siehe Abbildung 17). Darüber hinaus bestünde die Möglichkeit durch Kombination der Sequenzdiagramme, die die Kommunikation zwischen Prozesspartnern beschreiben, ein CIM um kollaborative Prozesse, unter Berücksichtigung der abstrakten Prozesse, zu erweitern. Ob ein Prozess mit Partnerprozessen eine Prozess Orchestration oder eine Prozess Choreography eingeht, lässt sich aus dem den ausführbaren Prozess modellierenden Aktivitätsdiagramm eines Prozesse ablesen. Choreography Beziehungen eines Prozesses werden durch Aktionen mit den Stereotypen <<*receive*>> und <<*send*>> abgebildet. Die Nutzung eines Prozesses durch übergeordnete Prozesse über Input- und Outputparameter oder den Aufruf von Subprozessen in einem Aktivitätsdiagramm deutet auf eine Prozess Orchestration hin.

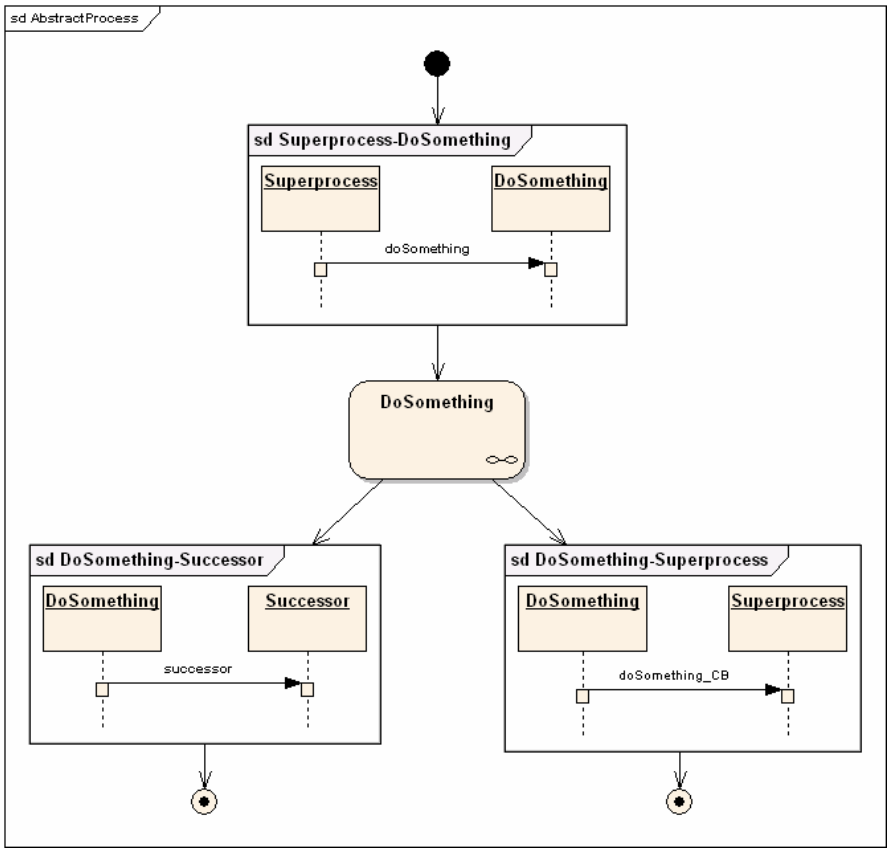


Abbildung 17: BPDM – abstrakter Prozess des CIMs

BPDM auf PIM-Ebene

Auf PIM-Ebene steht die Beschreibung des internen Verhaltens von Prozessen noch mehr als auf CIM-Ebene im Vordergrund. Aktivitätsdiagramme werden daher zusätzlich um den Datenfluss zwischen Aktivitäten und Aktionen ergänzt. In Abbildung 18 sind die Datenobjekte durch eine Pin-Notation an den Aktionen modelliert (beispielsweise *Object1*, *Object2* und *Object3*). Zusätzlich bieten Datencontainer, so genannte <<*datastores*>>, die Möglichkeit Daten ab- und zwischenzuspeichern. An der Modellierung von ausführbaren, abstrakten und kollaborativen Prozessen ändert sich im Vergleich zur CIM-Ebene nichts. Auch die Prozess Orchestration und die Prozess Choreography wird im PIM aus dem CIM übernommen.

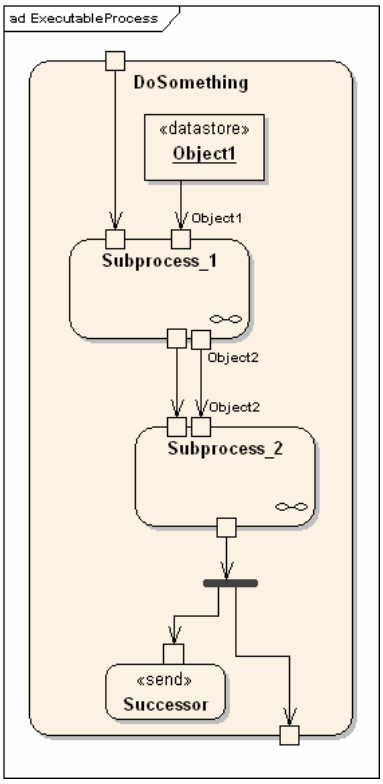


Abbildung 18: BPDM – ausführbarer Prozess des PIMs

In der ARIS-BPDM Abbildung werden ausführbare Prozesse im Allgemeinen behandelt, d.h. es spielt keine Rolle durch wen ein Prozess ausgeführt wird. Dabei kann es sich z.B. um eine Person handeln, die den Prozess manuell durchführt, oder um einen Softwareagenten, der einen Prozess automatisch ausführen kann. Automatisierbare Prozesse, d.h. Prozess die z.B. durch einen Softwareagenten automatisch durchgeführt werden können, stellen eine Spezialisierung von ausführbaren Prozessen dar. Obwohl es ein langfristiges Ziel ist, möglichst viele Prozesse zu automatisieren, bringt auch eine informationstechnische Unterstützung von lediglich ausführbaren Prozessen Fortschritte. So können Prozessabläufe von IuK-Systemen gesteuert, überwacht oder neu angestossen werden.

4 Spezifikation der ARIS-BPDM Abbildung

In diesem Kapitel wird die ARIS-BPDM Abbildung im Detail vorgestellt und spezifiziert. Wie in Abschnitt 3.2 schon erwähnt, ist die ARIS-BDPM Abbildung in zwei Transformationsschritte unterteilt, die auch die Gliederungsstruktur dieses Kapitels bestimmen. Abschnitt 4.1 spezifiziert eine möglichst äquivalente Umwandlung von ARIS-Modellen in UML- bzw. BPDM-Modelle (siehe a1) in Abbildung 19). Dabei werden auch schon Konzepte des BPDMs verwendet. Die Generierung von BPDM-Modellen auf PIM-Ebene aus den schon auf CIM-Ebene vorhandenen Modellen wird in Abschnitt 4.2 behandelt (siehe a2) in Abbildung 19).

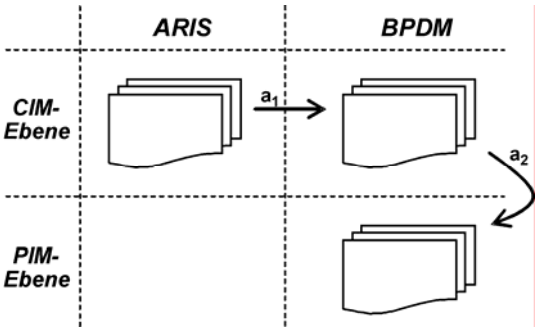


Abbildung 19: Gliederungsstruktur der Abbildungsspezifikation

Die Gliederung der Abschnitte 4.1 und 4.2 orientiert sich an der Unterteilung des ARIS-Hauses in Funktions-, Organisations-, Daten-, Leistungs- und Steuerungssicht. Durch die Unterteilung der Modelle in Beschreibungssichten wird eine Komplexitätsreduktion von Systembeschreibungen erzielt, die auch die Spezifikation der ARIS-BPDM Abbildung übersichtlicher gestalten lässt.

Beim Entwurf der ARIS-BPDM Abbildung wurde darauf Wert gelegt, nur Modellelemente zu verwenden, die durch die UML 2.0 oder das BPDM definiert sind. Bei der Identifizierung von zu den ARIS-Elementen möglichst äquivalenten Konzepten, wurden BPDM spezifische Konzepte alternativen UML-Konzepten vorgezogen. Dies ist auch ein Grund, dass BPDM-Konzepte schon auf CIM-Ebene zum Einsatz kommen.

Die in der ARIS-BPDM Abbildung betrachteten ARIS-Diagrammtypen sind Wertschöpfungskettendiagramme, ereignisgesteuerte Prozessketten und Funktionszuordnungsdiagramme. Obwohl alle drei Diagrammtypen der Steuerungssicht zuzuordnen sind, enthalten sie für die Abbildung relevante statische Aspekte. Statische Aspekte der UML- bzw. BPDM-Modelle werden in der ARIS-BPDM Abbildung in Klassendiagrammen model-

liert. Für die dynamischen Aspekte stehen Aktivitäts-, Sequenz- und Interaktionsübersichtsdiagramme zur Verfügung.

4.1 Mapping auf CIM-Ebene

In diesem Abschnitt wird der erste Teil der ARIS-BPDM Abbildung, d.h. das Mapping von ARIS-Modellen in UML- bzw. BPDM-Modelle, vorgestellt. Nach einem Überblick über die Mappingregeln folgt eine Betrachtung der einzelnen Beschreibungssichten anhand von Ausschnitten des ARIS-Metamodells. Anschließend werden die von Siemens bei der Modellierung mit dem ARIS-Toolset verwendeten Symbole den Konstrukten des Metamodells zugeordnet. Abschließend werden für jede Beschreibungssicht die Mappingregeln eingeführt und beschrieben.

Das Mapping von ARIS nach UML bzw. BPDM auf CIM-Ebene wird mit Hilfe von Regeln beschrieben. Aus dem Namen einer Regel wie z.B. *Regel_CIM_CD_Prozess* kann zum Teil abgeleitet werden, wann und zu welchem Zweck sie eingesetzt werden kann.

- Der Bestandteil **Regel**_ des Regelnamens kennzeichnet die Regel als Regel.
- Der Bestandteil _**CIM**_ des Regelnamens ordnet die Regel der Abbildung von ARIS nach BPDM auf CIM-Ebene zu.
- Der Bestandteil _**CD**_ des Regelnamens gibt Aufschluss darüber, in welchen Diagrammtyp das generierte BPDM dargestellt wird.
- Der letzte Bestandteil des Regelnamens ist der eigentliche Name der Regel. Er gibt normalerweise einen Hinweis auf den Inhalt oder Zweck der Abbildungsregel.

Die Transformation von ARIS Konzepten nach BPDM aus statischer Sicht wird mit Abbildungsregeln in tabellarischer Form formell beschrieben (vgl. Tabelle 1 bis Tabelle 3). In der ersten Zeile findet sich der Name der Abbildungsregel. Die zweite Spalte beschreibt den Regeltyp, während ab der dritten Spalte die Tabelle spaltenweise zu lesen ist. Ist der Regeltyp eine Objektabbildung (siehe Tabelle 1), so werden in der linke Spalte die ARIS-Konzepte inklusive des im ARIS-Toolset verwendeten (Symboltyps) beschrieben, die auf ein BPDM-Konzept der rechten Spalte abgebildet werden.

Regelname	Beispielregel
Regeltyp	Objektabbildung
ARIS-Konzept (Symboltyp)	BPDM-Konzept
ARISElement (Symbol)	BPDMElement

Tabelle 1: Regelbeispiel Objektabbildung

Im Falle einer Beziehungsabbildung (siehe Tabelle 2) werden Beziehungstyp, Quellobjekt und Zielobjekt der Beziehung von ARIS nach BPDM gemappt.

Regelname	Beispielregel	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„depends on“*	Assoziation

Quellobjekt	ARISElement1	BPDMElement1
Zielobjekt	ARISElement2	BPDMElement2

Tabelle 2: Regelbeispiel Beziehungsabbildung

Liegt eine Attributabbildung vor (siehe Tabelle 3), werden Attribute eines ARIS Konzepts nach BPDM abgebildet.

Regelname	Beispielregel
Regeltyp	Attributabbildung
Attributtypname	BDPM
ARIS Attribut 1	BPDM Attribut 1
ARIS Attribut 2	BPDM Attribut 2

Tabelle 3: Regelbeispiel Attributabbildung

Die Regeln zur Erzeugung der Verhaltensdiagramme lassen sich nicht übersichtlich in Tabellenform darstellen. Deshalb sind diese formell in einer regelbasierten *‚wenn-dann'* Form in Anhang A detailliert beschrieben.

4.1.1 Regelübersicht

Die Abbildung von ARIS nach BPDM auf CIM-Ebene umfasst 71 Transformationsregeln. In den Abbildungen 20 und 21 findet sich jeweils eine Übersicht über die Struktur der Abbildungsregeln. Die Pfeile gruppieren die in den rechteckigen Kästen dargestellten Regeln und bilden eine Ausführungsreihenfolge der Regeln ab. Ist beispielsweise ein Pfeil von der Regel [Regel_CIM_CD_Prozess] zur Regel [Regel_CIM_CD_Prozessanordnung] vorhanden, so wird die Regel zur Prozessanordnung nach der Regel [Regel_CIM_CD_Prozess] ausgeführt. Da die Abbildungen lediglich eine Übersicht über die Gruppierung der Regeln liefern soll, kann nicht davon ausgegangen werden, dass Regeln, die nicht durch Pfeile verbunden sind, stets in einer beliebigen Reihenfolge angewandt werden können.

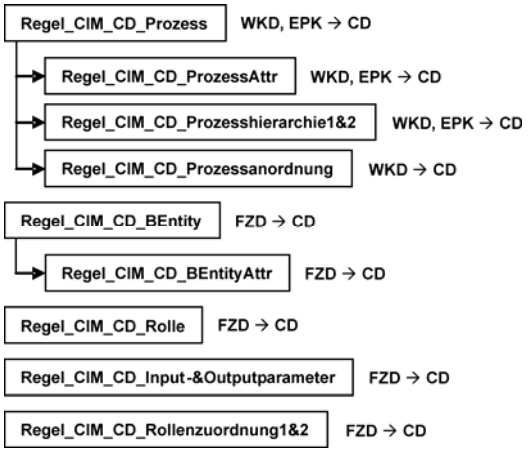


Abbildung 20: Mapping auf CIM-Ebene – Regelübersicht1

Schließlich enthalten die Regelübersichten noch Informationen zwischen welchen Diagrammen die in einer Regel spezifizierte Abbildung stattfindet. Dies ist hinter dem Regelnamen in der Form *, . → . '* angegeben. Im ersten Operand werden die abzubildenden

Diagramme notiert. Der zweite Operand spezifiziert die Zieldiagramme der Abbildung. Die Regel [Regel_CIM_CD_Prozess] bildet beispielsweise Elemente aus Wertschöpfungskettendiagrammen (WKD) und ereignisgesteuerten Prozessketten (EPK) auf Elemente von Klassendiagrammen (CD) ab.

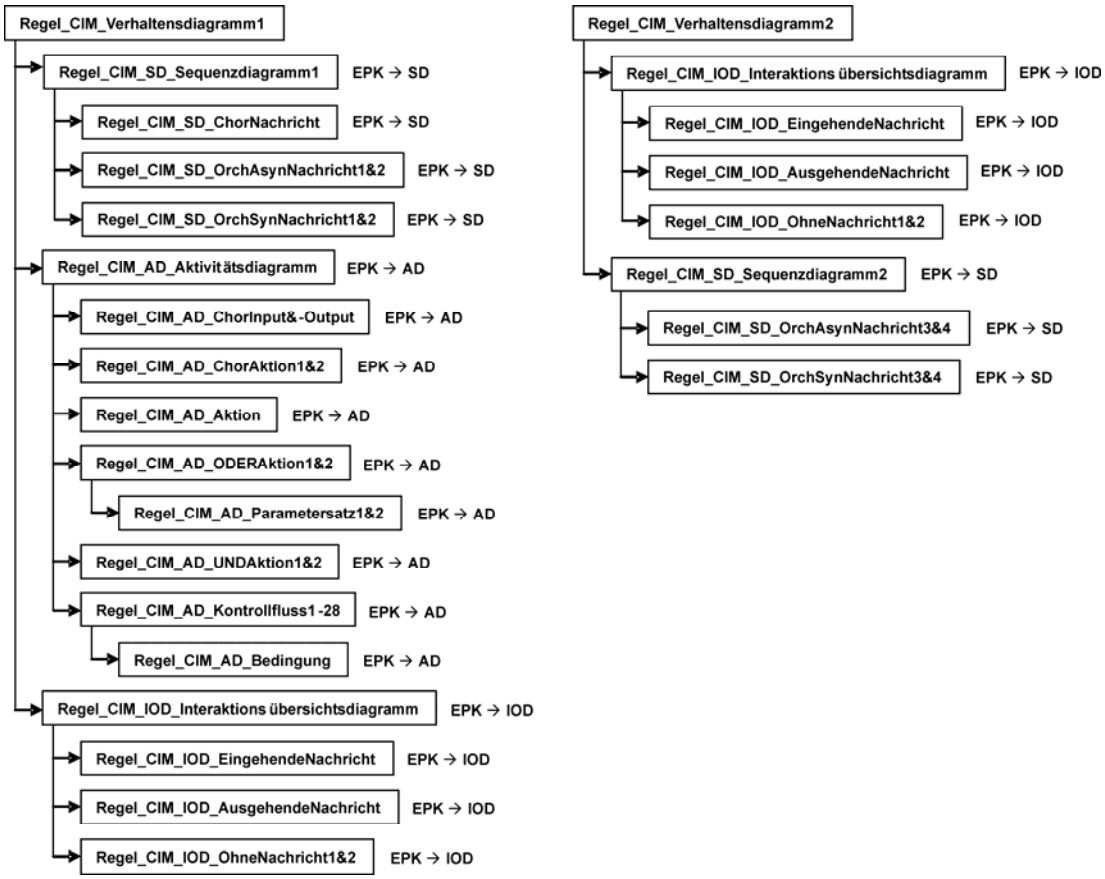


Abbildung 21: Mapping auf CIM-Ebene – Regelübersicht2

4.1.2 Funktionssicht

In der ARIS Funktionssicht werden die Aufgaben und Tätigkeiten beschrieben, die zur Umsetzung der Unternehmensziele nötig sind. Sie enthält Beschreibungen der Funktionen und der zwischen den Funktionen bestehenden Beziehungen, wie Nachfolgefunktion, Subfunktion, etc.. In einem Metamodell gibt ARIS (siehe [Sch98a]) die bei der Beschreibung der Funktionssicht eines IuK-Systems zu verwendenden Objekttypen und Assoziationstypen vor.

ARIS Metamodell

Im ARIS Metamodell (siehe [Sch98a]) wird jede Funktion einmal durch die Klasse Allgemeine Funktion AFUNKTION als Objekttyp erfasst. So werden Funktionen[11] wie z.B. Auftragsannahme oder Verfügbarkeitsprüfung nur einmal als Ausprägung der Klasse AFUNKTION definiert. Klassen wie Auftragsannahme oder Verfügbarkeitsprüfung können nun in unterschiedlichen Prozesszusammenhängen wiederverwendet werden. Auf den

[11] Hinweis: Funktionen entsprechen in diesem Zusammenhang Prozessen.

Allgemeinen Funktionen ist eine verrichtungsorientierte Gliederungsstruktur[12] AFKT-STRUKTUR definiert, mit der Gruppierungen von Funktionen mit gleichen oder ähnlichen Transformationsvorschriften erstellt werden können. Geschäftprozesse sind in dem ARIS Metamodell durch die Klasse GESCHÄFTSPROZESS modelliert. Mit Hilfe der Assoziationsklasse GESCHÄFTPROZESS-STRUKTUR können Geschäftprozesse in Unter- oder Teilprozesse zu hierarchischen Strukturen zusammengefasst werden. Dabei können Teilprozesse auch in mehrere übergeordnete Prozesse eingehen. Den Geschäftprozessen werden die zu ihnen gehörenden Allgemeinen Funktionen über die Assoziationsklasse FUNKTION zugeordnet. Nachdem Funktionen im Zusammenhang mit Geschäftprozessen beschrieben wurden, kann auf den Funktionen eine prozessorientierte Funktionsstruktur FUNKTIONSSTRUKTUR definiert werden. Die statische Geschäftprozessstruktur und die prozessorientierte Funktionsstruktur sind oft sehr ähnlich. Schließlich bietet das ARIS Metamodell mit der Assoziationsklasse ANORDNUNG die Möglichkeit, die Anordnung der Funktionen durch Vorgänger- und Nachfolgerbeziehungen zu modellieren.

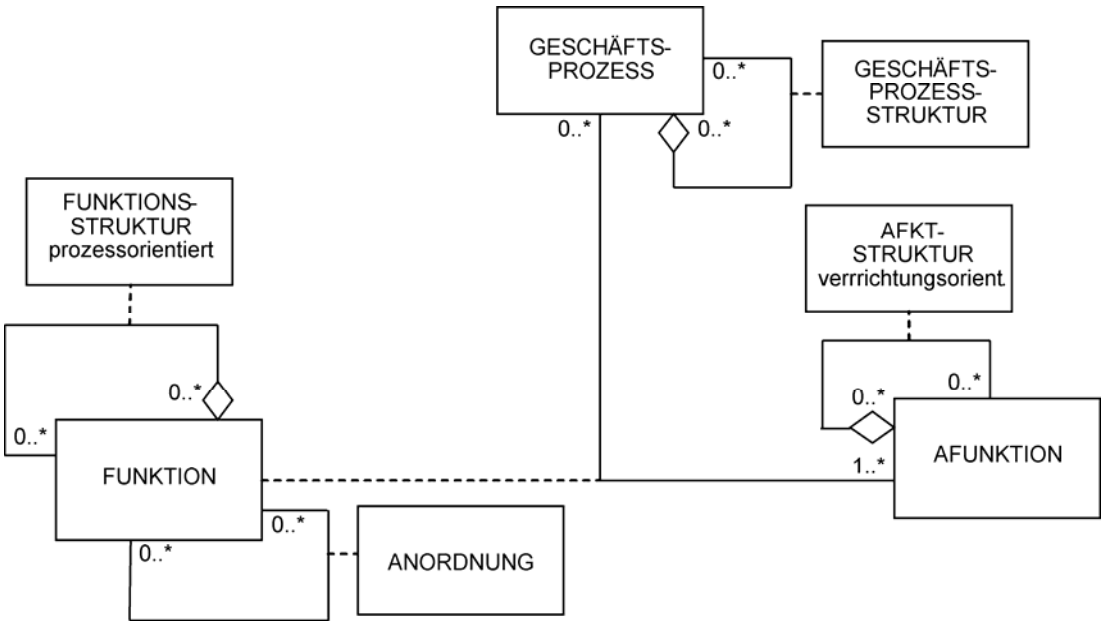


Abbildung 22: Ausschnitt des ARIS Metamodells der Funktionssicht [Sch98a]

Siemens ARIS

Aus Funktionssicht sieht das Siemens RPH lediglich den Objekttyp Funktion vor. Funktionen stellen dabei (Geschäfts-)Prozesse dar, auf denen eine hierarchische prozessorientierte Struktur und eine Ordnungsstruktur definiert sind. Es werden keine Allgemeinen Funktionen definiert, die den entsprechenden Geschäftsprozessen zugeordnet werden. Ansätze einer verrichtungsorientierten Funktionsstruktur finden sich lediglich auf Level 2 des Siemens RPH. Dort werden voneinander abweichende Prozessabläufe als Prozesskategorien, Prozessmodelle und Prozessvarianten hierarchisch definiert. Jedoch geschieht dies nicht kernprozess- oder gar prozessgruppenübergreifend.

[12] In einer verrichtungsorientierte Gliederungsstruktur werden Funktionen mit gleichen oder ähnlichen Transformationsvorschriften zusammengefasst. Ist das Gliederungskriterium nicht die Verrichtung sondern z.B. der Geschäftsprozess werden die an einem Prozess beteiligten Funktionen gruppiert.

Das Siemens RHP sieht zur Modellierung von Funktionen mit Hilfe des *ARIS-Toolsets* drei unterschiedliche Symboltypen vor: Funktion, Prozess und Wertschöpfungskette.

- Eine **Wertschöpfungskette** ist eine fachliche Aufgabe bzw. Tätigkeit an einem Objekt zur Unterstützung eines oder mehrerer Unternehmensziele.
- Ein **Prozess** repräsentiert Funktionen eines Szenarioprozesses, die durch hinterlegte Prozessmodelle näher beschrieben werden können.
- Eine **Funktion** ist eine fachliche Aufgabe bzw. Tätigkeit an einem Objekt zur Unterstützung eines oder mehrerer Unternehmensziele. Auf Level 3 bis n sowie den untersten Implementierungsleveln werden Funktionen verwendet, um Teilprozesse und Aktivitäten darzustellen.

Im Siemens RPH sind sowohl eine Anordnungsstruktur mit Vor- und Nachfolgerprozessen als auch eine Prozesshierarchie vorgesehen. Die Anordnungsstruktur wird im Siemens RPH in WKDs mit dem Beziehungstyp „*ist Vorgänger von*“ modelliert. Eine Prozesshierarchie wird sowohl in WKDs als auch mit Hilfe von EPKs festgelegt. WKDs können mit dem Beziehungstyp „*ist prozessorientiert übergeordnet*“ ähnlich einem Funktionsbaum hierarchisiert werden. Des Weiteren besteht eine prozessorientierte Über- bzw. Unterordnung, wenn einer Funktion eine EPK zugeordnet ist. Der Ablauf der Funktion wird in der zugeordneten EPK detaillierter beschrieben. Oft wird dabei von der Hinterlegung einer Funktion mit einer EPK gesprochen. Die in der EPK zur Beschreibung des Prozessablaufs verwendeten Funktionen werden der hinterlegten Funktion untergeordnet.

ARIS-BPDM Mapping

Regel CIM-1: (Regel_CIM_CD_Prozess)

Im BPDM wird der ARIS Objekttyp *Funktion* als Klasse mit dem Stereotyp *«process»* abgebildet. Nach dem BPDM [IAF+04] repräsentiert eine Klasse dieses Stereotyps die externe Sicht auf einen Prozess. Dabei dient sie auch als Container für den internen Zustand und das Verhalten des Prozesses. In Tabelle 4 findet sich die Mappingregel *Regel_CIM_CD_Prozess* für den ARIS Objekttyp *Funktion*.

Regelname	Regel_CIM_CD_Prozess
Regeltyp	Objektabbildung
ARIS-Konzept (Symboltyp)	BPDM-Konzept
Funktion (Funktion) Funktion (Prozess) Funktion (Wertschöpfungskette)	Klasse mit dem Stereotyp *«process»*

Tabelle 4: Regel zum Objekttyp Funktion

Regel CIM-2: (Regel_CIM_CD_ProzessAttr)

Tabelle 5 gibt Aufschluss über die bei der Modellierung von Funktionen verwendeten Attribute. Diese Attribute werden im BPDM in den *«process»* Klassen übernommen. Spalte eins gibt den Attributtypnamen des Attributs im Siemens RPH an. In der zweiten Spalte wird angegeben wie das Attribut in BPDM umgesetzt wird.

Regelname	Regel_CIM_CD_ProzessAttr
Regeltyp	Attributabbildung

Attributtypname	BDPM
Identifizierer	Name der *«process»* Klasse
Name	Attribut Name
Beschreibung/Definition	Attribut Beschreibung
Sortierreihenfolge	Attribut Sortierreihenfolge

Tabelle 5: Regel zu den Attributen der Klasse «process»

Regel CIM-3 bis 4: (Regel_CIM_CD_Prozesshierarchie1&2)

Die hierarchisch prozessorientierten Funktionsstrukturbeziehungen des Siemens RPHs werden durch eingebettete Klassen modelliert, die sich auch im konzeptionellen Übersichtsmodell von BPDM wieder finden. Nach [JRH+04] S.42 entspricht die grafische Darstellungsform eingebetteter Klassen einer gerichteten Kante von der umgebenden zur enthaltenen Klasse. Neben der Richtung der Assoziation vom übergeordneten zum untergeordneten Prozess wird der untergeordnete Prozess zusätzlich mit der Rolle *‚subProcess'* gekennzeichnet. In der hierarchischen Funktionsstruktur sind untergeordnete Prozesse Teil des übergeordneten Prozesses. Es wird aber nicht festgelegt, auf welche Weise und inwieweit die Subprozesse in den Prozessablauf des übergeordneten Prozesses eingebunden werden. In Tabelle 6 und Tabelle 7 finden sich die Regeln zur Prozesshierarchie zur Umwandlung von ARIS Prozesshierarchien nach BPDM.

Regelname	Regel_CIM_CD_Prozesshierarchie1	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„ist prozessorientiert übergeordnet“*	Assoziation (gerichtet von der Quelle zum Ziel)
Quellobjekt	Funktion (WKD)	*«process»* Klasse
Zielobjekt	Funktion (WKD)	*«process»* Klasse (Rolle *‚subProcess'*)

Tabelle 6: Regel zur Prozesshierarchie1

Regelname	Regel_CIM_CD_Prozesshierarchie2	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	einer Funktion ist eine EPK hinterlegt	Assoziation (gerichtet von der Quelle zum Ziel)
Quellobjekt	Funktion	*«process»* Klasse
Zielobjekt	Funktion (EPK)	*«process»* Klasse (Rolle *‚subProcess'*)

Tabelle 7: Regel zur Prozesshierarchie2

Regel CIM-5: (Regel_CIM_CD_Prozessanordnung)

Im Siemens RPH wird neben einer Prozesshierarchie eine Anordnungsbeziehung mit Vor- und Nachfolgerprozessen definiert. Dazu gibt es im BPDM keine direkte Entsprechung. Jedoch kann diese Beziehung leicht in UML durch eine Assoziation zwischen zwei Prozessen unter Angabe der Rollen *‚predecessor'* und *‚successor'* umgesetzt werden (siehe Tabelle 8).

Regelname	Regel_CIM_CD_Prozessanordnung	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„ist Vorgänger von"*	Assoziation
Quellobjekt	Funktion (WKD)	*«process»* Klasse (Rolle *‚predecessor'*)
Zielobjekt	Funktion (WKD)	*«process»* Klasse (Rolle *‚successor'*)

Tabelle 8: Regel zur Prozessanordnung

4.1.3 Organisationssicht

In der ARIS Organisationssicht wird die Aufbauorganisation eines Unternehmens beschrieben. Diese umfasst sowohl die Organisationseinheiten mit den zwischen ihnen bestehenden Kommunikations- und Weisungsbeziehungen als auch ein Rollenkonzept, das das Anforderungsprofil einer Organisationseinheit definiert.

ARIS Metamodell

Organisationseinheiten sind in dem ARIS Metamodell mit der Klasse ORGANISATIONSEINHEIT modelliert. Strukturelle Beziehungen zwischen über- und untergeordneten Organisationseinheiten werden durch die Klasse ORGANISATIONSSTRUKTUR erfasst. Eine STELLE ist die kleinste Einheit einer Organisationsstruktur. Da sie auch eine Organisationseinheit ist, erbt sie von der Oberklasse ORGANISATIONSEINHEIT. Schließlich wird durch das ARIS Metamodell noch ein Rollenkonzept definiert. Die Klasse ROLLE repräsentiert einen bestimmten Mitarbeitertyp mit einer definierten Qualifikation und Kompetenz. Einer Rolle werden über die Assoziation STELLENBESETZUNG geeignete Stellen zugeordnet, die die eigentlichen Tätigkeiten ausführen.

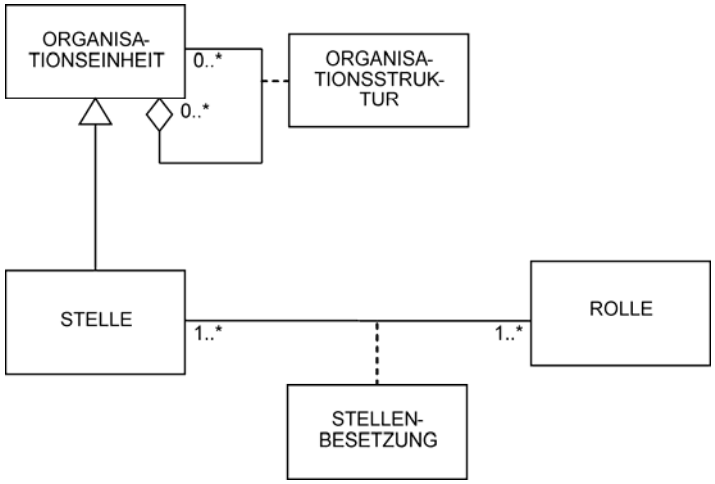


Abbildung 23: Ausschnitt des ARIS Metamodells der Organisationssicht [Sch98a]

Siemens ARIS

Zur Modellierung der ARIS Organisationssicht schreibt das Siemens RPH den Objekttyp *Personentyp* vor. Ein *Personentyp* entspricht einer Rolle im ARIS Metamodell. Darüber hinaus könnten mit dem ARIS-Toolset Stellen des ARIS Metamodells mit dem der Objekttyp *Person* modelliert werden. Eine *Organisationseinheit* deckt sich mit einer ARIS Organisationseinheit, während es für den Objekttyp *Organisationseinheitstyp* kein Konzept im ARIS Metamodell gibt, das diesem direkt entspricht.

Das Siemens RPH sieht folgenden Symboltyp zwingend vor:

- Ein **Personentyp** (Objekttyp *Personentyp*) stellt die Typisierung einzelner (externer und interner) Personen dar, die gleiche Eigenschaften aufweisen. Diese Eigenschaften können sich z.B. auf gleichartige Rechte und Verantwortlichkeiten beziehen. Der *Personentyp* entspricht einer Rolle.

In Funktionszuordnungsdiagrammen müssen den Funktionen, d.h. den Prozessen, verpflichtend Personentypen zugeordnet werden. Die Angabe von Organisationseinheiten ist optional (siehe [Ren03] S. 20). In WKDs und EPKs sind keine Konzepte zur Modellierung der Organisationssicht vorgesehen. Beziehungen auf Organisationsebene würden im Siemens RPH in Organigrammen modelliert. Da Organigramme weder verpflichtend vorgeschrieben sind noch genau spezifiziert wurden, werden Beziehungen auf Organisationssicht nicht näher betrachtet.

ARIS-BPDM Mapping

Regel CIM-6: (Regel_CIM_CD_Rolle)

In UML wird der ARIS Objekttyp *Personentyp* (bzw. im ARIS Metamodell das Konzept *Rolle*) als Klasse mit dem Stereotyp *«role»* abgebildet. Das Konzept einer Rolle stellt den Bedarf einer Aufgabe nach einer Ressource dar, durch die sie erfüllt wird [IAF+04][13]. Diese Ressourcen können z.B. wie in ARIS nach Qualifikationen, Rechten oder Verantwortlichkeiten unterschieden werden. Aber auch andere Unterscheidungsmerkmale sind möglich. In Tabelle 9 findet sich die Mappingregel für den ARIS Objekttyp *Personentyp*.

[13] ARIS definiert kein zum Konzept der *‚partnerRole'* äquivalentes Konstrukt. Daher werden *‚partnerRoles'* im ARIS-BPDM Mapping nicht verwendet.

Regelname	Regel_CIM_CD_Rolle
Regeltyp	Objektabbildung
ARIS-Konzept (Symboltyp)	BPDM-Konzept
Personentyp (Personentyp)	Klasse mit dem Stereotyp *«role»*

Tabelle 9: Regel zum Objekttyp Rolle

Im Siemens RPH sind keine verpflichtenden Attribute für Personentypen definiert. Deshalb wird im Mapping der Name des Personentyps als Klassenname für eine Rolle in UML übernommen. Attribute eines Personentyps könnten leicht als Attribute in UML Klassendiagramm übernommen werden.

Ein Mapping von Beziehungen innerhalb der Organisationssicht ist nicht erforderlich, da in den relevanten Diagrammen (WKDs, FZDs und EPKs) solche Beziehungen nicht modelliert werden.

4.1.4 Datensicht

Die ARIS Datensicht enthält Beschreibungen von Datenobjekten, die von Funktionen manipuliert werden (siehe [Sch98a]).

ARIS Metamodell

Zur Modellierung von Datenobjekten sieht das ARIS Metamodell die Klasse MAKRO-DATENOBJEKT vor. Durch ein Datenobjekt werden in ARIS unterschiedlichste Konzepte, wie Ereignisse und Nachrichten aber auch Funktionsinput und Funktionsoutput repräsentiert. Das ARIS Metamodell erlaubt es, eine Beziehungsstruktur (MDO-STRUKTUR) zwischen den Datenobjekten abzubilden. Dabei besteht ein übergeordnetes Datenobjekt aus keinem oder mehreren untergeordneten Datenobjekten. In ARIS werden Makrodatenobjekte schließlich in kleinere Einheiten, den INFORMATIONSOBJEKTen zerlegt, mit denen die detaillierte Struktur eines fachlichen Anwendungsbereichs abgebildet werden kann. Informationsobjekte sind den Makrodatenobjekten über die MDO-ZUORDNUNG zugeordnet.

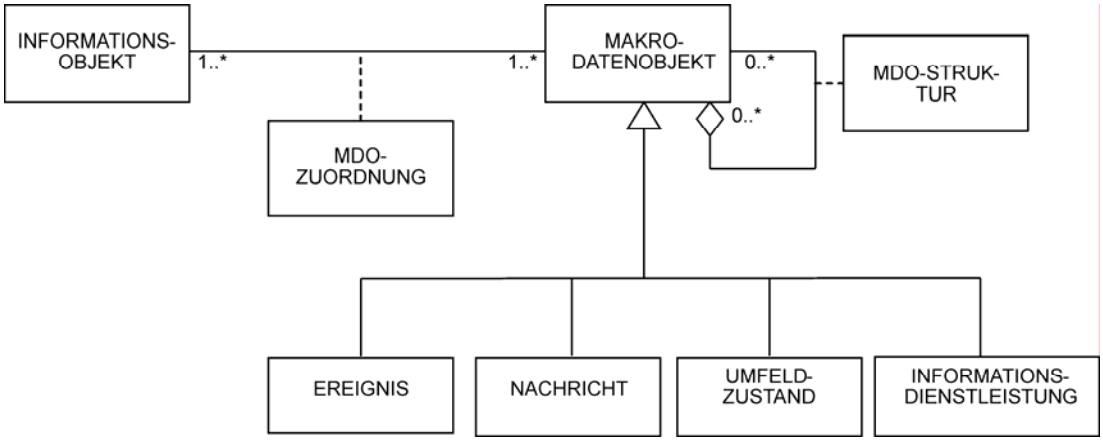


Abbildung 24: Ausschnitt des ARIS Metamodells der Datensicht [Sch98a]

Siemens ARIS

Zur Modellierung der ARIS Datensicht verwendet das Siemens RPH den ARIS Objekttyp *Fachbegriff*. Im ARIS Metamodell entspricht dies weitestgehend einem Informationsobjekt. Das Siemens RPH sieht zur Modellierung des Objekttyps *Fachbegriff* den Symboltyp *Fachbegriff* vor, der im Konventionenhandbuch [Ren03] wie folgt definiert ist:

- **Fachbegriffe** stellen die in einem Unternehmen zur Beschreibung der betrachteten Informationsobjekte existierenden Begrifflichkeiten dar. Inputs und Outputs werden im Funktionszuordnungsdiagramm durch Kantenbeziehungen dargestellt.

Fachbegriffe werden im Siemens RPH in FZDs als Input- und Outputparameter von Funktionen verwandt. Eine Modellierung der Beziehungsstruktur zwischen Fachbegriffen wird durch das Siemens RPH noch nicht unterstützt. Wie schon die Organigramme der Organisationssicht, ist auch das Fachbegriffsmodell der Datensicht noch nicht ausreichend spezifiziert.

ARIS-BPDM Mapping

Regel CIM-7: (Regel_CIM_CD_BEntity)

Im BPDM sind Informationsobjekte als so genannte Business Entities modelliert, die mit dem Stereotyp *«entity»* gekennzeichnet werden. Diese haben Identität, Verhalten und einen eigenen Zustand und können aus betriebswirtschaftlicher Sicht als eine feste Begrifflichkeit verstanden werden. In Tabelle 10 findet sich die Mappingregel für den ARIS Objekttyp Fachbegriff.

Regelname	Regel_CIM_CD_Entity
Regeltyp	Objektabbildung
ARIS-Konzept (Symboltyp)	BPDM-Konzept
Fachbegriff (Fachbegriff)	Klasse mit dem Stereotyp *«entity»*

Tabelle 10: Regel zum Objekttyp Business_Entity

Regel CIM-8: (Regel_CIM_CD_BEntityAttr)

Auch für die Modellierung von Fachbegriffen gibt das Siemens RPH eine Anzahl von zu modellierenden Attributtypen vor. Tabelle 11 gibt Aufschluss darüber, wie das Mapping diese auf eine *«entity»* im BPDM abbildet.

Regelname	Regel_CIM_CD_BEntityAttr
Regeltyp	Attributabbildung
Attributtypname	BDPM
Name	Name der *«entity»* Klasse
Beschreibung/Definition	Attribut Beschreibung

Tabelle 11: Regel zu den Attributen der Klasse «entity»

Da die Beziehungen zwischen Fachbegriffen im Siemens RPH noch nicht ausreichend spezifiziert sind, werden diese in der Abbildung auch nicht weiter betrachtet.

4.1.5 Leistungssicht

Die Modellierung der Leistungssicht ist im Siemens RPH noch nicht spezifiziert und auch nicht verpflichtend vorgeschrieben. Da aus diesen Gründen die Leistungssicht für die ARIS-BPDM Abbildung nicht weiter relevant ist, wird sie auch nicht weiter behandelt.

4.1.6 Steuerungssicht

Aufgabe der Steuerungssicht ist es, die zunächst getrennt behandelten ARIS Sichten (Funktions-, Organisations-, Daten- und Leistungssicht) wieder zu verbinden. Dabei werden sowohl strukturelle Beziehungen als auch das Verhalten des Systems beschrieben. ARIS bietet Modelle zur Beschreibung der paarweisen Beziehungen zwischen den Sichten und Gesamtmodelle, in die alle ARIS Sichten integriert werden (nach [Sch98a]).

ARIS Metamodell

Die Konzepte der Steuerungssicht sollen nun anhand eines groben Metamodells (siehe Abbildung 25) für ein Gesamtmodell vorgestellt werden. Da der Fokus dieser Arbeit auf den Prozessen von Unternehmen liegt, wird ein Gesamtmodell, das eine prozessorientierte Sichtweise abbildet, verwendet. Alternativ würden auch Gesamtmodelle für beispielsweise eine Objektsicht in ARIS zur Verfügung stehen.

In einem prozessorientierten Gesamtmodell für Vorgangskettendiagramme (VKDs) oder auch EPKs steht die FUNKTION im Mittelpunkt. Ihr werden die Konzepte der anderen ARIS Sichten zugeordnet. Aus der Organisationssicht werden der FUNKTION ORGANISATIONSEINHEITEN zugeordnet. In der OE-FKT-ZUORDNUNG werden die Assoziationen zwischen Funktionen und Organisationseinheiten näher spezifiziert (z.B. der Typ der Assoziation). Aus Datensicht werden INFORMATIONSOBJEKTE FUNKTIONEN zugeordnet. In der IO-FKT-ZUORDNUNG kann beispielsweise festgelegt werden, ob es sich bei einem Informationsobjekt um einen Input- oder Outputparameter einer Funktion handelt. Schließlich stehen auch noch EREIGNISSE mit FUNKTIONEN über die EREIG-FKT-ZUORDNUNG in Beziehung. Ereignisse lösen Funktionen aus und werden von Funktionen erzeugt.

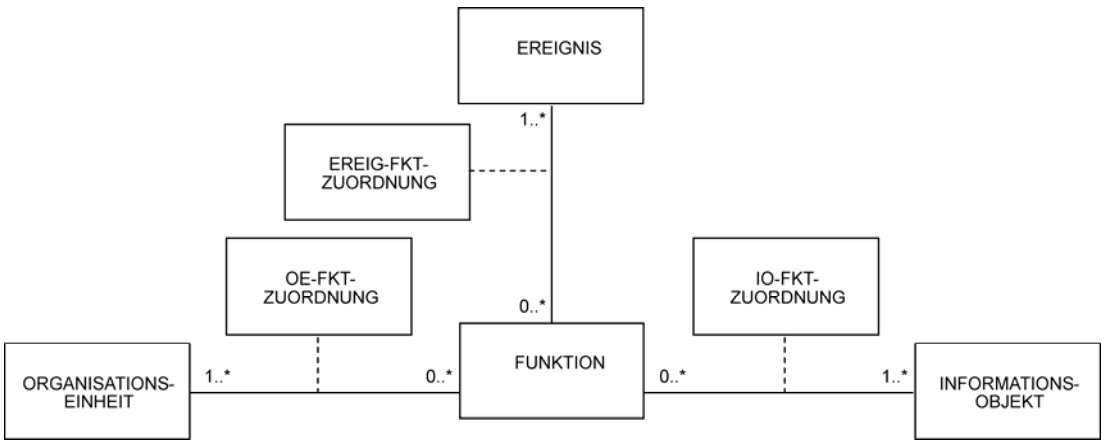


Abbildung 25: Ausschnitt des ARIS Metamodells für VKDs und EPKs [Sch98a]

Siemens ARIS

Im Siemens RPH sind Wertschöpfungskettendiagramme, ereignisgesteuerte Prozessketten und Funktionszuordnungsdiagramme zur Modellierung der Steuerungssicht verpflichtend vorgesehen. Da die in den WKDs verwendeten Objekt- und Beziehungstypen in der Abbildung schon in der Funktionssicht berücksichtigt wurden (vgl. Abschnitt 4.1.2), entfällt eine detaillierte Betrachtung der WKDs aus Steuerungssicht.

Mit Hilfe von Funktionszuordnungsdiagrammen werden die Beziehungen zwischen den ARIS Sichten modelliert. Die zugehörigen Objekttypen wurden in dieser Abbildung schon in den jeweiligen ARIS Sichten berücksichtigt. Beziehungstypen, die sichtenübergreifende Zuordnungen modellieren, werden nun in der Steuerungssicht näher betrachtet. Diese sind im Einzelnen:

- **„ist Input für“:** Beschreibt die Beziehung zwischen einer Funktion und einem Fachbegriff. Der Fachbegriff repräsentiert einen Inputparameter der Funktion.
- **„hat Output“:** Beschreibt die Beziehung zwischen einer Funktion und einem Fachbegriff. Der Fachbegriff repräsentiert einen Outputparameter der Funktion.
- **„führt aus“:** Beschreibt eine Beziehung zwischen einer Funktion und einem Personentyp. Der Personentyp hat Führungsverantwortung bei der Ausführung der Aktivitäten einer Funktion.
- **„wirkt mit bei“:** Beschreibt eine Beziehung zwischen einer Funktion und einem Personentyp. Der Personentyp nimmt aktiv an der Ausführung der Funktion teil.

In EPKs wird der interne Ablauf von Prozessen modelliert. Neben den schon vorgestellten Funktionen sieht das Siemens RPH weitere Symbol- und Objekttypen zur Modellierung von EPKs vor:

- Eine **Prozessschnittstelle** (Objekttyp *Funktion*) dient der Kennzeichnung der vorgelagerten und nachgelagerten Prozesskette auf dem gleichen Level. Sie stellt die Durchgängigkeit der Prozessketten untereinander sicher.
- Eine **externe Schnittstelle** (Objekttyp *Funktion*) (auch Prozessverweis genannt) dient der Kennzeichnung eines Levelsprungs im Prozess oder einer Übergabe an einen externen Prozess.
- Ein **Ereignis** (Objekttyp *Ereignis*) stellt den Eintritt eines betriebswirtschaftlich relevanten Zustands, der den weiteren Ablauf von Prozessen steuert und beeinflusst, dar. Ereignisse lösen Funktion aus und sind Ergebnisse von Funktionen. Ein Ereignis ist stets auf einen Zeitpunkt bezogen.
- Eine **UND**, **ODER** oder **exklusives ODER** Regel (Objekttyp *Regel*) stellt einen Verknüpfungsoperator dar, mit dem die logische Verbindung von Ereignissen und Funktionen in EPKs festgelegt wird.

Je nach Einsatz der Symboltypen *Prozessschnittstelle* und *externe Schnittstelle* kann im Siemens RPH die Zusammenarbeit der Prozesse im Sinne einer Prozess Orchestration oder einer Prozess Choreography modelliert werden. Mit den beiden Schnittstellentypen wird der abstrakte Prozess eines Prozesses modelliert.

- Der in **einer** EPK modellierte Ablauf eines Prozesses stellt grundsätzlich eine Prozess Orchestration dar. Die EPK beschreibt den detaillierten Prozessablauf, d.h. die interne Sicht auf einen Prozess. Dieser Prozess steuert als globaler Prozess den Gesamtab-

lauf seiner untergeordneten Subprozesse. Um die Funktionalität der Subprozesse nutzen zu können, müssen ihm nach dem black-box Prinzip nur deren abstrakte Prozesse bekannt sein. Die Subprozesse werden wiederum detaillierter durch separate EPKs beschrieben, in denen sich jeweils ein Prozessverweis auf den vorgelagerten und den nachgelagerten Subprozess findet.

- In allen anderen Fällen liegt eine Prozess Choreography vor. Dies kann z.B. dann der Fall sein, wenn ein Prozess über keinen übergeordneten Prozess verfügt, in dessen EPK er eingebunden ist. Der Prozess leitet dann den Kontrollfluss an einen oder mehrere andere Prozesse weiter, auf die er mit Hilfe von externen Schnittstellen oder Prozessschnittstellen verweist. Als Sonderfall können auch eine Prozess Choreography und eine Prozess Orchestration kombiniert werden. Dies ist nötig, wenn ein Prozess zwar als Subprozess in eine Prozess Orchestration eingebunden ist, aber zusätzlich auf Prozesse verweist, mit denen er in einem Choreographyverhältnis steht.

Beziehungstypen für EPKs werden im Siemens RPH nicht explizit definiert. Wie bei EPKs allgemein üblich, werden Ereignisse und Funktionen durch Kanten im Sinne eines Kontrollflusses miteinander verbunden. Zur Abbildung von komplexeren Kontrollflüssen werden Regeln als logische Verknüpfungen eingesetzt. Dabei wird zwischen parallelen Abläufen (*UND*-Regel), alternativen Abläufen (*exklusives ODER*-Regel) und parallelen und alternativen Abläufen (*ODER*-Regel) unterschieden. Erlaubte Kombinationsmöglichkeiten von Ereignissen, Funktionen und Regeln sind in [Sei 02] S.74ff beschrieben.

ARIS-BPDM Mapping

Kontrollfluss[14]

Wie schon vorgestellt, wird der im Rahmen des Siemens RPHs modellierte Kontrollfluss in BPDM auf CIM-Ebene durch Interaktionsübersichtsdiagramme, die durch Sequenz- und Aktivitätsdiagramme verfeinert werden, modelliert. Sequenzdiagramme beschreiben dabei die Interaktion eines Prozesses mit einem Partnerprozess über eine Schnittstelle. Aktivitätsdiagramme bilden den internen Prozessablauf, d.h. den ausführbaren Prozess, eines Prozesses ab. Interaktionsübersichtsdiagramme repräsentieren abstrakte Prozesse, die das externe Verhalten von Prozessen beschreiben. In ihnen werden die Sequenzdiagramme mit dem Aktivitätsdiagramm eines Prozesses in Zusammenhang gebracht. Interaktionsübersichts-, Sequenz- und Aktivitätsdiagramme werden stets dem zugehörigen Prozess zugeordnet, d.h. dem Prozess, dem die EPK im ARIS-Modell hinterlegt ist.

Bei der Beschreibung von Prozessen mit Verhaltensdiagrammen muss unterschieden werden, ob ein Prozess durch eine EPK hinterlegt ist oder nicht.

Regel CIM-9: (Regel_CIM_Verhaltensdiagramm1)

- Ist ein Prozess mit einer EPK hinterlegt, so wird aus der EPK ein Aktivitätsdiagramm generiert, in dem sein ausführbarer Prozess beschrieben ist. Zusätzlich dazu werden Sequenzdiagramme zur Modellierung der Interaktionen mit Partnerprozess und ein Interaktionübersichtsdiagramm zur Modellierung des abstrakten Prozesses erzeugt. (formell siehe Anhang A)

[14] Zugunsten einer kürzeren und übersichtlicheren Beschreibung werden für das Mapping des Kontrollflusses Mappingregeln nicht in tabellarischer sondern in textueller Form angegeben.

Regel CIM-10: (Regel_CIM_Verhaltensdiagramm2)

- Ist ein Prozess durch keine EPK hinterlegt und tritt der Prozess als Funktion in einer oder mehreren EPKs von übergeordneten Prozessen auf, wird er auch durch Sequenz- und Interaktionsdiagramme beschrieben. Da die Realisierung seines internen Verhaltens nicht bekannt ist, wird für diesen Prozess kein Aktivitätsdiagramm generiert. (formell siehe Anhang A)

Sequenzdiagramme

In Sequenzdiagrammen werden die zwischen Prozessen ausgetauschten Nachrichten modelliert. Wie eingangs schon angedeutet, muss dabei unterschieden werden, ob eine Prozessschnittstelle in einer EPK zur Modellierung eines Prozess Choreographyverhältnisses oder einer Prozess Orchestration verwandt wird.

Ist ein Prozess mit einer EPK hinterlegt, so ist bei der Generierung der zugehörigen Sequenzdiagramme folgendermaßen vorzugehen:

Regel CIM-11: (Regel_CIM_SD_Sequenzdiagramm1)

- Jede Prozessschnittstelle der EPK wird in einem separaten Sequenzdiagramm umgesetzt. Jedes dieser Sequenzdiagramme erhält eine Lebenslinie für den durch die EPK beschriebenen Prozess und eine Lebenslinie für den durch die Prozessschnittstelle referenzierten Prozess. Der Name eines Sequenzdiagramms setzt sich folgendermaßen zusammen: *‚<AufrufenderProzess>_<AufgerufenerProzess>'*. (formell siehe Anhang A)

Regel CIM-12: (Regel_CIM_SD_ChorNachricht)

- Liegt eine Prozess Choreography vor, wird die Nachricht zwischen dem Prozess, der durch die EPK beschrieben wird, und dem Prozess, der durch die Prozessschnittstelle referenziert wird, ausgetauscht. Beide Prozesse werden im Sequenzdiagramm durch Lebenslinien, die deren Namen erhalten, modelliert. Verweist die Prozessschnittstelle auf einen Vorgängerprozess, so ist die Richtung der ausgetauschten Nachricht von dem Prozess, auf den die Prozessschnittstelle verweist, zu dem durch die EPK beschriebenen Prozess. Verweist die Prozessschnittstelle auf einen Nachfolgeprozess, wird die Nachricht von dem durch die EPK beschriebenen Prozess zum Nachfolgeprozess verschickt. Die zwischen zwei Prozessen modellierte Nachricht erhält den Namen des Prozesses, an den sie geschickt wird. (formell siehe Anhang A)

- Im Falle einer Prozess Orchestration wird eine Nachricht nicht mit einem vor- oder nachgelagerten Prozess ausgetauscht, sondern mit dem übergeordneten Prozess, der den durch die EPK beschriebenen Prozess in seinen Prozessablauf einbindet. Verglichen zu den Sequenzdiagrammen einer Prozess Choreography wird lediglich der Name des Kommunikationspartners durch den Namen des übergeordneten Prozesses ersetzt.[15] Darüber hinaus wird in Sequenzdiagrammen des CIMs zwischen synchronem und asynchronem Nachrichtenaustausch unterschieden.

Regel CIM-13 bis 14: (Regel_CIM_SD_OrchAsynNachricht1&2)

- Im Falle eines asynchronen Nachrichtenaustausches erhält die zu dem Prozess, der durch die EPK beschrieben wird, gesandte Nachricht den Namen des Prozes-

[15] Da Sequenzdiagramme zur Modellierung von abstrakten Prozessen verwendet werden, werden die Nachrichten, die ein Prozess mit untergeordneten Prozessen austauscht nicht, modelliert. Sie sind auch vom Standpunkt eines externen Beobachters nicht sichtbar bzw. nicht von Interesse.

ses. Die Antwortnachricht wird auch mit dem Namen dieses Prozesses einschließlich dem Zusatz *‚_CB'* versehen. (formell siehe Anhang A)

Regel CIM-15 bis 16: (Regel_CIM_SD_OrchSynNachricht1&2)

- Bei synchroner Kommunikation erhalten die bei dem durch die EPK beschriebenen Prozess eingehende Nachrichten, wie bei asynchroner Kommunikation, den Namen des Prozesses. Lediglich die Antwortnachricht erhält keinen Namen, da sie nur die Rückgabe des Ergebnisparameters und nicht eine eigenständige Nachricht modelliert. (formell siehe Anhang A)

Nicht alle Prozesse, die externe Schnittstellen zur Verfügung stellen, sind mit EPKs hinterlegt. Zu diesen Prozessen werden auch Sequenzdiagramme generiert.

Regel CIM-17: (Regel_CIM_SD_Sequenzdiagramm2)

- Ein Prozess ist nicht durch eine EPK hinterlegt, sondern ist in einer oder mehreren EPKs eines übergeordneten Prozesses nur als Funktion modelliert. Dieser Prozess steht mit seinem übergeordneten Prozess in einem Orchestrationverhältnis. Es werden zwei Sequenzdiagramme erzeugt (eines zur Modellierung der eingehenden Nachricht mit dem Namen *‚<ÜbergeordneterProzess>-<UntergeordneterProzess>'* und eines zur Modellierung der Antwortnachricht mit dem Namen *‚<UntergeordneterProzess>-<ÜbergeordneterProzess>'*), die jeweils eine Lebenslinie für den zum Sequenzdiagramm gehörigen Prozess und eine Lebenslinie für den übergeordneten Prozess enthalten. (formell siehe Anhang A)

Regel CIM-18 bis 19: (Regel_CIM_SD_OrchAsynNachricht3&4)

- Liegt eine asynchrone Kommunikation vor, werden die Regeln [Regel_CIM_SD_OrchAsynNachricht1+2] auf die beiden Sequenzdiagramme angewendet, mit der Änderung, dass der in den Regeln durch eine EPK hinterlegte Prozess nun durch keine EPK hinterlegt ist. (formell siehe Anhang A)

Regel CIM-20 bis 21: (Regel_CIM_SD_OrchSynNachricht3&4)

- Liegt eine synchrone Kommunikation vor, werden die Regeln [Regel_CIM_SD_OrchSynNachricht1+2] auf die beiden Sequenzdiagramme angewendet, mit der Änderung, dass der in den Regeln durch eine EPK hinterlegte Prozess nun durch keine EPK hinterlegt ist. (formell siehe Anhang A)

Aktivitätsdiagramme

Aktivitätsdiagramme werden zur Modellierung des internen Prozessablaufs genutzt. Sie repräsentieren ausführbare Prozesse und sind mit den zugehörigen abstrakten Prozessen verknüpft. Auf CIM-Ebene wird in Aktivitätsdiagrammen nur der Kontrollfluss und kein Datenfluss modelliert. Folgende Modellelemente werden bei der Modellierung von Aktivitätsdiagrammen eingesetzt (nach [JRH+04]):

- **Aktivität:** Aktivitäten werden nach außen durch ein Rechteck mit abgerundeten Ecken abgrenzt. Aktivitäten können geschachtelt werden, indem in einer Aktivität enthaltene Aktionen wiederum Aktivitäten aufrufen. Während mögliche Parameter mit der Pin-Notation dargestellt werden, sieht UML 2.0 keine Notation zur Unterscheidung von synchronen und asynchronen Aktivitätsaufrufen vor (diese Information muss z.B. als Kommentar modelliert werden).
- ***Aktion*:** Eine Aktion wird durch ein Rechteck mit abgerundeten Ecken dargestellt. Eine Aktion beschreibt einen Einzelschritt, der zur Realisierung des durch die Aktivität beschriebenen Verhaltens beiträgt. Dabei steht eine Aktion für den Aufruf eines Ver-

haltens (einer weiteren Aktivität) oder für die Bearbeitung von Daten, die nicht weiter innerhalb einer weiteren Aktivität zerlegt werden. Zur Modellierung von abstrakten Prozessen werden zusätzlich die *Stereotypen «receive»* und *«send»* für Aktionen eingeführt.

- **Objektknoten:** Ein Objektknoten innerhalb einer Aktivität repräsentiert Ausprägungen eines bestimmten Typs. Objektknoten bilden das logische Gerüst, um Daten und Werte innerhalb einer Aktion während eines Ablaufs zu transportieren. Zur Modellierung von Objektknoten als Eingabe- und Ausgabeparameter einer Aktion kann die *Pin-Notation* verwendet werden. Bei dieser Schreibweise ist der Objektknoten an der Aktion unmittelbar angeheftet. Die Richtung der Kanten an dem Objektknoten gibt an, ob es sich um einen Eingangs- oder Ausgangspin handelt. Der Name eines Pins entspricht dem Objektknotentyp.
- **Kanten:** Kanten sind die Übergänge zwischen Aktionen in einem Aktivitätsdiagramm. Kanten sind stets gerichtet und können sowohl mit *Bedingungen* in eckigen Klammern oder Namen versehen werden. Der *Kontrollfluss*, und der *Objektfluss* werden mit Kanten modelliert. In der ARIS-BPDM Abbildung wird zur Modellierung sowohl des Kontroll- als auch des Objektflusses die Pin-Notation verwendet. Die Pins des Kontrollflusses erhalten keinen Namen, da sie keinen Objekttyp repräsentieren.
- **Kontrollelemente:** Die Aufgabe von Kontrollelementen ist es, den Ablauf der Aktivität zu steuern. Mit Verzweigungs-, Verbindungs-, Synchronisations- und Parallelisierungsknoten können alternative und parallele Abläufe modelliert werden. Wie bei Sequenzdiagrammen, gibt es auch in Aktivitätsdiagrammen kein Kontrollelement, das ein *ODER* repräsentiert. Es steht lediglich ein *exklusives ODER* zur Verfügung. In der ARIS-BPDM Abbildung werden, wie in [FGJ04] eingeführt, Kontrollelemente durch spezielle Aktionen umgesetzt (für eine genauere Beschreibung siehe [FGJ04] S.11f). Diese Aktionen können als Kontrollschritte angesehen werden, die Zeit und Ressourcen benötigen.
- **Parametersatz:** Ein- und Ausgabeparameter von Aktionen und Aktivitäten lassen sich zu Parametersätzen gruppieren. Ein Parametersatz legt fest, dass an allen Pins eines Satzes (Daten-)Tokens anliegen müssen, damit eine Aktion oder eine Aktivität ausgeführt wird. Das heißt, zwischen den Pins eines Parametersatzes besteht ein implizites UND. Zwischen den Parametersätzen selbst besteht eine Entweder-oder-Beziehung. Die (Daten-)Tokens eines Parametersatzes werden dann gemeinsam in die Aktion überführt.
- **Aktivitätsbereich:** Mit einem Aktivitätsbereich lässt sich eine Aktivität in Bereiche mit gemeinsamen Eigenschaften unterteilen. Im Fall der ARIS-BPDM Abbildung werden mit Aktivitätsbereichen Rollen modelliert. Dabei handelt es sich um die in der Organisationssicht als Ressourcen modellierte Rollen und um keine *‚partnerRoles'*.

Nachdem die für die ARIS-BPDM Abbildung wichtigen Konzepte von Aktivitätsdiagrammen vorgestellt wurden, können nun auch die Transformationsschritte von ereignisgesteuerten Prozessketten zu Aktivitätsdiagrammen angegeben werden.

Regel CIM-22: (Regel_CIM_AD_Aktivitätsdiagramm)

- Zu jedem Prozess, dessen Ablauf in ARIS durch eine EPK beschrieben wird, wird im BPDM ein Aktivitätsdiagramm mit dem Namen *‚<Prozessname>-ExecutableProcess'* generiert. Ein Aktivitätsdiagramm enthält dazu eine Aktivität mit dem Namen des Prozesses, die die interne Realisierung des Verhaltens des beschriebenen Prozesses

genauer spezifiziert. Darüber hinaus wird das Aktivitätsdiagramm dem Prozess, dessen Verhalten es beschreibt, zugeordnet. (formell siehe Anhang A)

- Transformation des Kontrollflusses

Regel CIM-23 bis 24: (Regel_CIM_AD_ChorAktion1&2)

- Die Prozessschnittstellen einer EPK, die ein Prozess Choreographyverhältnis repräsentieren, werden in Aktivitätsdiagrammen als Aktionen mit den Stereotypen *«receive»* und *«send»* umgesetzt. Sie erhalten den Namen des Vor- oder Nachfolgerprozesses, auf den sie verweisen. (formell siehe Anhang A)

Regel CIM-25 bis 26: (Regel_CIM_AD_OrchInput&-Output)

- Ist durch Prozessschnittstellen einer EPK ein Prozess Orchestrationverhältnis modelliert, so wird dies nicht durch Aktionen umgesetzt. Stattdessen erhält die Aktivität des durch die EPK beschriebenen Prozesses einen Eingangs- und einen Ausgangsparameter zur Modellierung des Kontrollflusses, durch den der Ablauf der Aktivität angestoßen bzw. die Beendigung des Ablaufs der Aktivität mitgeteilt wird. (formell siehe Anhang A)

Regel CIM-27: (Regel_CIM_AD_Aktion)

- Funktionen einer EPK werden in einem Aktivitätsdiagramm als Aktionen modelliert. Sie sind dem im Aktivitätsdiagramm beschriebenen Prozess untergeordnet, der sie zur Realisierung seines internen Verhaltens benötigt und deren Zusammenspiel koordiniert. Eine Aktion erhält den Namen der in der EPK modellierten Funktion. Dabei kann eine Aktion als Aufruf des Verhaltens eines Subprozesses dienen oder als elementare Funktion Berechnungsschritte beschreiben, die nicht weiter zerlegt werden. Da in UML keine eigene Notation zur Modellierung von asynchronen und synchronen Aktivitätsaufrufen vorhanden ist, wird für den Aufruf von Subprozessen diese zusätzliche Information in Kommentaren hinzugefügt. (formell siehe Anhang A)

Regel CIM-28 bis 55: (Regel_CIM_AD_Kontrollfluss1-28)

- Der Kontrollfluss wird mit *Kontrollflusskanten* zwischen den Pins der Aktionen modelliert. Diese Pins erhalten keinen Namen, da sie nur den Austausch von virtuellen Kontrolltokens und keine Datentokens repräsentieren. (formell siehe Anhang A)

Regel CIM-56 bis 57: (Regel_CIM_AD_Parametersatz1&2)

- Pins können durch Parametersätze zu Gruppen zusammengefasst werden, um alternative Input- oder Outputparametersätze einer Aktion abzubilden. Wird kein Parametersatz modelliert, so entspricht dies einem einzigen Parametersatz, der alle Input- bzw. Outputpins umfasst. (formell siehe Anhang A)

- Zur Transformation der ARIS Regeln *exklusives ODER* und *UND* können in Aktivitätsdiagrammen Kontrollelemente eingesetzt werden. Das BPDM sieht darüber hinaus die alternative Verwendung von speziellen Kontrollaktionen vor, die in der ARIS-BPDM Abbildung eingesetzt werden. Da die UML und das BPDM kein Kontrollelement ODER vorsehen, kann die ARIS Regel ODER nicht übersetzt werden.[16] Zur

[16] Eine Möglichkeit wäre, die Semantik des ODER Kontrollelements der UML anzupassen und die Wahl beliebig vieler (statt einer) Alternativen zuzulassen. Welche Auswirkungen ein solcher Inter-

Kennzeichnung alternativer Gruppierungen von Pins als Input oder Output einer Aktion sind Parametersätze einzusetzen.

Regel CIM-58 bis 59: (Regel_CIM_AD_ODERAktion1&2)

- Eine *ODER*-Regel einer EPK kann in Aktivitätsdiagrammen durch zwei unterschiedliche Kontrollaktionen umgesetzt werden. Liegt eine Verzweigung vor, so wird diese durch einen Inputpin und mehrere durch Parametersätze getrennte Outputpins modelliert. In einer Verbindung werden alternative Prozessabläufe durch mehrere durch Parametersätze getrennte Inputpins und einen Outputpin zusammengefasst. (formell siehe Anhang A)

Regel CIM-60 bis 61: (Regel_CIM_AD_UNDAktion1&2)

- Auch *UND*-Regeln werden durch zwei unterschiedliche Kontrollaktionen umgesetzt. Im Falle einer Parallelisierung erhält die Kontrollaktion einen Inputpin und mehrere in einem Parametersatz zusammengefasste Outputpins. Analog dazu hat eine Synchronisation mehrere nicht durch Parametersätze getrennte Inputpins und einen Outputpin. (formell siehe Anhang A)

Regel CIM-62: (Regel_CIM_AD_Bedingung)

- Ereignisse in ARIS EPKs werden BPDM Aktivitätsdiagrammen als Bedingungen auf den Kontrollflusskanten in eckigen Klammern modelliert. (formell siehe Anhang A)
- Mit Hilfe von Aktivitätsbereichen werden die den Subprozessen zugeordneten Rollen in Aktivitätsdiagrammen modelliert. Dazu werden die Aktionen, die Subprozessaufrufe darstellen, den zugehörigen Aktivitätsbereichen zugeordnet. Da die Aussagekraft von Aktivitätsdiagrammen durch Aktivitätsbereiche oft nicht verbessert wird und die semantische Information schon im statischen Teil eines Modells vorhanden ist, bleibt es dem Nutzer überlassen, Aktivitätsbereiche einzusetzen. Die Ursache liegt darin, dass durch die zusätzlich eingeführte Information Aktivitätsdiagramme unübersichtlicher strukturiert und schwerer verständlich werden.

Interaktionsübersichtsdiagramme

Interaktionsdiagramme werden im BPDM zur Modellierung von abstrakten Prozessen eingesetzt. Der Kontrollfluss wird in Interaktionsübersichtsdiagrammen analog zu Aktivitätsdiagrammen modelliert. Als Aktionen enthält ein Interaktionsübersichtsdiagramm Interaktionsdiagramme oder Referenzen auf Interaktionsdiagramme. In der ARIS-BPDM Abbildung besteht ein Interaktionsübersichtsdiagramm für einen Prozess aus den zum Prozess gehörigen Sequenzdiagrammen und einer Aktion, die einen Aufruf des im Aktivitätsdiagramm modellierten ausführbaren Prozesses darstellt. Ein Interaktionsübersichtsdiagramm wird wie folgt erstellt:

Regel CIM-63: (Regel_CIM_IOD_Interaktionsübersichtsdiagramm)

- Alle zu dem beschriebenen Prozess gehörigen Sequenzdiagramme werden im Interaktionsübersichtsdiagramm aufgenommen. Der ausführbare Prozess wird im Interaktionsübersichtsdiagramm als Aktion repräsentiert. Der Name des Interaktionsübersichtsdiagramms setzt sich aus dem Namen des beschriebenen Prozesses und dem Zusatz *‚-AbstractProcess'* zusammen. (formell siehe Anhang A)
- Modellierung des Kontrollflusses

---

determinismus (insbesondere bezüglich der in einem Aktivitätsdiagramm kreisenden Token) auf eine spätere automatische Ausführung von Prozessen hat, müsste vorher noch untersucht werden.

Regel CIM-64; (Regel_CIM_IOD_EingehendeNachricht)

- Ist in einem Sequenzdiagramm der Versand einer Nachricht von einem Vorgängerprozess zum beschriebenen Prozess modelliert, so wird das Interaktionsübersichtsdiagramm um eine Kante von dem Sequenzdiagramm zu der Aktion ergänzt. Das Sequenzdiagramm erhält zusätzlich noch eine Kante als Input, die von einem Startknoten ausgeht. (formell siehe Anhang A)

Regel CIM-65: (Regel_CIM_IOD_AusgehendeNachricht)

- Ist in einem Sequenzdiagramm der Versand einer Nachricht zu einem Vorgängerprozess von dem beschriebenen Prozess modelliert, so wird das Interaktionsübersichtsdiagramm um eine Kante zu dem Sequenzdiagramm von der Aktion ergänzt. Das Sequenzdiagramm erhält zusätzlich noch eine Kante als Output, die in einem Endknoten mündet. (formell siehe Anhang A)

Regel CIM-66 bis 67: (Regel_CIM_IOD_OhneNachricht1&2)

- Verfügt eine Aktion nach der Anbindung der Sequenzdiagramme an die Aktion über keine Inputkante, so wird diese inklusive einem Startknoten ergänzt. Analog wird bei einem Fehlen einer Outputkante eine solche inklusive eines Endknotens dem Interaktionsübersichtsdiagramm hinzugefügt. (formell siehe Anhang A)

Beziehungen zwischen Funktions- und Datensicht

Regel CIM-68 bis 69: (Regel_CIM_CD_Input-&Outputparameter)

Informationsobjekte können sowohl Inputparameter als auch Outputparameter von Funktionen sein. Die Modellierung von Input- und Outputparametern ist im Siemens RPH verpflichtend vorgeschrieben. Im BPDM werden diese Beziehungen als Assoziationen zwischen den Prozessen und den Business Entities modelliert. In Tabelle 12 und Tabelle 13 finden sich die Mappingregeln zur Transformation der Beziehungen zwischen Funktions- und Datensicht.

Regelname	Regel_CIM_CD_Inputparameter	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„ist Input für“*	Assoziation (Name *‚isInput‘*)
Quellobjekt	Fachbegriff (FZD)	*«entity»* Klasse
Zielobjekt	Funktion (FZD)	*«process»* Klasse

Tabelle 12: Regel zu Inputparametern

Regelname	Regel_CIM_CD_Outputparameter	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„hat Output“*	Assoziation (Name *‚hasOutput‘*)
Quellobjekt	Funktion (FZD)	*«entity»* Klasse
Zielobjekt	Fachbegriff (FZD)	*«process»* Klasse

Tabelle 13: Regel zu Outputparametern

Beziehungen zwischen Funktions- und Organisationssicht

Regel CIM-70 bis 71: (Regel_CIM_CD_Rollenzuordnung1&2)

Im Siemens RPH werden in den FZDs den Funktionen Personentypen zugeordnet. Ein Personentyp repräsentiert eine Rolle, die eine Funktion ausführen kann. Im BPDM werden diese Beziehungen durch Assoziationen zwischen Prozessen und Rollen abgebildet. Die passenden Mappingregeln finden sich in Tabelle 14 und Tabelle 15.

Regelname	Regel_CIM_CD_Rollenzuordnung1	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„wirkt mit bei”*	Assoziation (Name *‚contributesTo‘*)
Quellobjekt	Funktion (WKD)	*«role»* Klasse
Zielobjekt	Personentyp (WKD)	*«process»* Klasse

Tabelle 14: Regel zur Rollenzuordnung 1

Regelname	Regel_CIM_CD_Rollenzuordnung2	
Regeltyp	Beziehungsabbildung	
	ARIS	BPDM
Beziehungstyp	*„führt aus“*	Assoziation (Name *‚executes‘*)
Quellobjekt	Funktion (WKD)	*«role»* Klasse
Zielobjekt	Personentyp (WKD)	*«process»* Klasse

Tabelle 15: Regel zur Rollenzuordnung 2

Obwohl im BPDM Rollen Tasks zugeordnet werden und nicht Prozessen, stellt das Mapping keinen Widerspruch zum Übersichtsmodell des BPDMs dar. Tasks sind die auszuführenden Aufgaben eines Prozesses. Dies ist oft der Aufruf eines Subprozesses, der durch eine bestimmte Rolle ausgeführt werden kann. In einem Aktivitätsdiagramm wird z.B. ein Subprozess durch einen Task modelliert. Da ein Task als (black-box) Repräsentation eines Prozesses gesehen werden kann, ist es unproblematisch, Rollen dem Prozess selbst und nicht einem Task zuzuordnen.

Beziehungen zwischen Daten- und Organisationssicht

Beziehungen zwischen der Daten- und der Organisationssicht sind im Siemens RPH noch nicht spezifiziert und werden deshalb in der Abbildung nicht weiter behandelt.

4.2 Transformation zur PIM-Ebene

In diesem Abschnitt wird der zweite Teil der ARIS-BPDM Abbildung, d.h. die Generierung von BPDM-Modellen auf PIM-Ebene aus den schon auf CIM-Ebene vorhandenen Modellen, behandelt. Nach einer Regelübersicht ist, wie schon im ersten Teil der Spezifikation der Abbildung, auch dieser Abschnitt in die ARIS Beschreibungssichten unterteilt. Jede Beschreibungssicht verfügt über einen Überblick, in dem die auf PIM-Ebene neu hinzugefügten Konzepte und die von der CIM-Ebene übernommenen Konzepte vorgestellt werden. Anschließend werden die zur Modellierung verwendeten Konzepte erläutert, wobei unterschieden wird, ob ein Konzept im BPDM beschrieben ist oder durch die Abbildung zusätzlich aufgenommen wurde. Abschließend wird die Transformation von CIM zu PIM Modellen in textueller Form im Detail besprochen wobei sich die formellen Transformationsregeln jeweils im Anhang B finden.

Regel PIM-1: (Regel_PIM_Identität)

Die Identitätsregel kann in jeder Sicht angewandt werden. Durch Sie werden Diagramme bzw. Konzepte die auf CIM-Ebene in BPDM beschrieben wurden, in das PIM übernommen. (formell siehe Anhang B)

4.2.1 Regelübersicht

In Abbildung 26 findet sich eine Übersicht der Regeln die zur Transformation des CIMs zu einem PIM verwendet werden. Die Semantik der in der Regelübersicht verwendeten Notation entspricht der von Kapitel 4.1.1. Da bei der Transformation von CIMs zu PIMs die Diagramme lediglich erweitert werden und keine Abbildung zwischen unterschiedlichen Diagrammtypen stattfindet, wird auf eine Angabe der Quell- und Zieldiagramme der Transformation verzichtet.

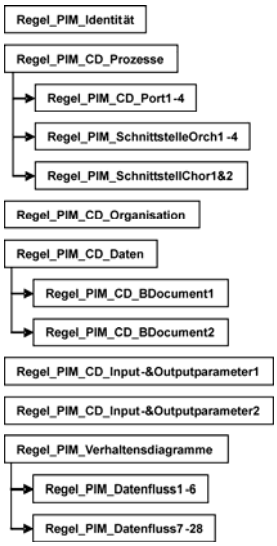


Abbildung 26: Transformation zur PIM-Ebene - Regelübersicht

4.2.2 Funktionssicht

Überblick

Das CIM Modell der Funktionssicht besteht aus Klassendiagrammen. In diesem werden die Prozesse als Klassen mit dem Stereotyp *«process»* modelliert. Darüber hinaus bilden Assoziationsbeziehungen Prozesshierarchien mit über- und untergeordneten Prozessen und Prozessanordnungen mit Vorgänger- und Nachfolgerprozessen ab. Auf PIM-Ebene werden diese Klassendiagramme ohne Änderung zur Modellierung der Funktionssicht übernommen. Zusätzlich werden Prozessschnittstellen generiert, mit Hilfe derer die Funktionalität von Prozessen genutzt werden kann.

Verwendete BPDM Konzepte

- **Process Stereotype:** Ein *Prozess* ist ein Container für Zustand und Verhalten. Eine Klasse mit dem Stereotyp *«process»* inklusive zugehöriger Ports und Schnittstellen repräsentiert die externe Sicht auf einen Prozess. In einem Prozess können Subprozesse enthalten sein, die dazu beitragen, das Verhalten des Prozesses zu realisieren. Das interne Verhalten eines Prozesses wird mit Hilfe des *Process Flow Concepts* modelliert.
- **Sub-process Concept:** Subprozesse dienen zur Modellierung von Prozesshierarchien von über- und untergeordneten Prozessen. Das BPDM sieht zur grafischen Darstellung von Subprozessen eingebettete Klassen vor. Nach [JRH+04] S. 42 entsprechen in UML 2.0 eingebettete Klassen einer gerichteten Assoziation von der umgebenden Klasse zur enthaltenen Klasse. In der Abbildung werden zur Darstellung von Prozesshierarchien gerichtete Assoziationen verwendet.
- **Ports und Schnittstellen:** In der Spezifikation des BPDMs werden Ports und Schnittstellen vor allem in Beispielen immer wieder erwähnt, es wird aber nie exakt spezifiziert, wie diese einzusetzen sind. In der ARIS-BPDM Abbildung werden Ports daher als Zugriffsendpunkte umgesetzt, über die ein Prozess den Dienst eines anderen Prozesses nutzen kann. Ein Port kann auf mehrere Schnittstellen verweisen, die Operationen zur Nutzung eines Dienstes definieren. Je nachdem, ob ein Prozess einen Dienst nutzen will oder zur Verfügung stellt, bindet er die passende Schnittstelle als *required interface* oder als *provided interface* an den zum Dienst gehörigen Port.

Zusätzliche Konzepte zu BPDM

- **Prozessanordnung:** Das BPDM sieht kein Konzept zur Modellierung von Prozessanordnungen vor- und nachgelagerter Prozesse aus statischer Sicht vor. Da nicht ausgeschlossen werden kann, dass Wertschöpfungsketten von Prozessen auch für ein zu implementierendes IuK-System relevant sind, werden Prozessanordnungen auch auf PIM-Ebene modelliert. Hierzu werden die auf CIM-Ebene definierten Assoziationen (siehe Tabelle 8) zur Modellierung von Prozessanordnungen auf PIM-Ebene übernommen.[17]

[17] Zur Beschreibung von Prozessanordnungen bzw. des Prozessablaufs aus dynamischer Sicht werden in ARIS EPKs und in BPDM Aktivitätsdiagramme verwendet. Da in WKDs kein konkreter Prozessablauf beschrieben wird, kann der in ihnen beschriebene Sachverhalt nur unbefriedigend in Aktivitätsdiagrammen abgebildet werden.

Transformation des CIM zum PIM

Regel PIM-2: (Regel_PIM_CD_Prozesse)

In der Funktionssicht werden die Klassendiagramme der CIM-Ebene auf PIM-Ebene übernommen. Zusätzlich werden die *«process»* Klassen um Ports und Schnittstellen erweitert. Eine Erweiterung der *«process»* Klassen um zusätzliche Attribute ist möglich, jedoch zum momentanen Stand nicht vorgesehen. (formell siehe Anhang B)

Modellierung von Ports und Schnittstellen

Prozesse bieten anderen Prozessen Dienste an und nutzen Dienste von anderen Prozessen. In Klassendiagrammen werden mit *Ports* Zugriffsendpunkte spezifiziert, über die Interaktionen zwischen einem Prozess und seiner Umgebung, d.h. im Fall der ARIS-BPDM Abbildung mit den anderen Prozessen, stattfinden.[18]

Ein Port kann Dienste zusammenfassen, die ein Prozess seiner Umgebung zur Verfügung stellt oder von der Umgebung benötigt. Dienste werden als Schnittstellen, d.h. Klassen mit dem Stereotyp *«interface»*, modelliert. Bei Schnittstellen wird zwischen *provided interfaces*, die Schnittstelle wird vom Prozess zur Verfügung gestellt, und *required interfaces*, die Schnittstelle wird vom Prozess benötigt, unterschieden.[19]

Regel PIM-3 bis 6: (Regel_PIM_CD_Port1-4)

- Ein Prozess erhält für jeden Prozess[20], mit dem seine Verhaltensspezifikation Interaktionen vorsieht, einen separaten Port. Der Name des Port setzt sich aus dem Namen des Prozesses selbst und dem Namen des Partnerprozesses zusammen: *‚<Prozessname>_<Partnerprozessname>'*. (formell siehe Anhang B)
- Finden Interaktionen zwischen Prozessen statt, so werden zur Modellierung der Interaktionen Schnittstellen eingesetzt. Die Umsetzung der Schnittstelle unterscheidet sich je nachdem, ob Prozesse in einem Orchestration- oder Choreographyverhältnis stehen.

Regel PIM-7 bis 10: (Regel_PIM_CD_SchnittstelleOrch1-4)

- Stehen zwei Prozesse in einem Orchestrationverhältnis, so wird eine Schnittstelle generiert, über die der untergeordnete Prozess seine Dienste zur Verfügung stellen kann. Diese Schnittstelle erhält folgenden Namen: ‚*<UntergeordneterProzessname>_ORCH'*. Diese Schnittstelle wird dem übergeordneten Prozess als *required interface* und dem untergeordneten Prozess als *provided interface* zugeordnet. Findet zwischen den Prozessen eine asynchrone Kommunikation statt, so wird zusätzlich eine Callback-Schnittstelle mit dem Namen ‚*<UntergeordneterProzessname>_CB'* generiert. Diese wird dem übergeordneten Prozess als *provided interface* und dem untergeordneten Prozess als *required interface* zugeordnet. (formell siehe Anhang B)

[18] Prozesse nehmen bei der Kommunikation mit anderen Prozessen Rollen ein, die im BPDM als *‚partnerRoles'* bezeichnet werden. Da ARIS solche Rollen nicht vorsieht, werden sie im ARIS-BPDM Mapping auch nicht modelliert, können aber nachträglich ergänzt werden. Dann bietet es sich an, einen *Port* eines Prozesses den *‚partnerRoles'* zuzuordnen, die er erfüllen kann.

[19] Soll ein Port eine spezielle *‚partnerRole'* erfüllen, so können ihm nachträglich leicht weitere Schnittstellen zugeordnet werden.

[20] Dies umfasst auch untergeordnete Prozesse im Falle einer Prozess Orchestration.

Regel PIM-11 bis 12: (Regel_PIM_CD_SchnittstelleChor1&2)

- Liegt ein Choreographyverhältnis zwischen Prozessen vor, so wird dieses durch Referenzen auf die jeweiligen Partnerprozesse modelliert. Besitzt ein Prozess eine Referenz auf einen Vorgänger- oder einen Nachfolgerprozess, so wird diese, falls nicht schon durch einen anderen Prozess geschehen, in einer Schnittstelle umgesetzt und dem Prozess als *provided* bzw. *required interface* zugewiesen. Der Name der Schnittstelle setzt sich folgendermaßen zusammen: *‚<NameNachfolgerprozess>_<NameVorgängerprozess>_CHOR'.* (formell siehe Anhang B)
- Zwei Schnittstellen sind zueinander kompatibel, d.h. das *provided interface* kann den Bedarf des *required interfaces* erfüllen, wenn sie vom gleichen Typ sind oder das *provided interface* eine Subklasse der *required interfaces* ist. Für Choreographyverhältnisse zwischen Prozessen muss diese Definition der Kompatibilität von Schnittstellen erweitert werden. Zwei Schnittstellen sind demnach auch kompatibel, wenn der Nachfolgerprozess, der das *provided interfaces* zur Verfügung stellt, ein Subprozess des im *required interface* definierten Nachfolgerprozesses ist oder wenn der Vorgängerprozess, der das *required interface* bereitstellt, ein Subprozess des im *provided interface* spezifizierten Vorgängerprozesses ist.

4.2.3 Organisationssicht

Überblick

Aus Organisationssicht sind die Modelle auf CIM-Ebene und auf PIM-Ebene identisch. Es kommen dabei jeweils Klassendiagramme zum Einsatz, die bei der Transformation von der CIM zur PIM-Ebene lediglich übernommen und nicht modifiziert oder erweitert werden.

Verwendete BPDM Konzepte

- **Role Concept:** Im Rahmen des BPDM Übersichtsmodells (Abbildung 12) wird eine Rolle als der Bedarf einer Aufgabe nach einer Ressource, durch die sie erfüllt wird, beschrieben [IAF+04]. Trotz dieser Definition eines festen Konzeptes sieht die Spezifikation des BPDM lediglich Aktivitätsbereiche in Aktivitätsdiagrammen zur Modellierung von Rollen vor. Ein Konzept zur Repräsentation von Rollen aus statischer Sicht, wie z.B. als stereotypisierte Klassen in Klassendiagrammen, ist nicht vorgesehen. Deshalb wird in der Abbildung auf CIM-Ebene der Stereotyp *«role»* für Klassen eingeführt, um das in ARIS vorhandene Konzept einer Rolle in BPDM statisch modellieren zu können. Es ist nahe liegend, diesen Stereotypen auch auf PIM-Ebene weiter zu verwenden.
- **PartnerRole Concept:** Dieses zweite Konzept einer Rolle wird im BPDM Whitepaper vorgestellt. *‚partnerRoles'* repräsentieren die Rollen, die Prozesse bei Interaktionen mit Partnerprozessen einnehmen. Da ARIS kein äquivalentes Konzept vorsieht, macht die ARIS-BPDM Abbildung keinen Gebrauch von *‚partnerRoles'*. Sollten diese dennoch nachträglich benötigt und den BPDM-Modellen hinzugefügt werden, so könnten Prozesse über geeignete, von ihnen zur Verfügung gestellten, *Ports* den *‚partnerRoles'* zugeordnet werden.

Transformation des CIM zum PIM

Regel PIM-13: (Regel_PIM_CD_Organisation)

Alle Klassendiagramme inklusive deren Modellelemente, die die Organisationssicht auf CIM-Ebene beschreiben, werden ohne Änderung oder Erweiterung auf PIM-Ebene übernommen. (formell siehe Anhang B)

4.2.4 Datensicht

Überblick

Zur Modellierung der Datensicht werden sowohl auf CIM-Ebene als auch auf PIM-Ebene Klassendiagramme eingesetzt. Im Rahmen der Transformation der Klassendiagramme von CIMs zu PIMs können alle Modellelemente ohne Änderung übernommen werden. Darüber hinaus werden Modelle der Datensicht um zusätzliche Konzepte, den Business Documents, erweitert.

Verwendete BPDM Konzepte

- **Entity Stereotype:** Eine *Entity* ist ein Container für eine Menge von Geschäftsdaten. *Entities* haben Identität, Verhalten und einen eigenen Zustand und werden aus betriebswirtschaftlicher Sicht oft als eine feste Begrifflichkeit verstanden.
- **Document Stereotype:** Ein *Document* stellt eine Sicht auf eine Entity dar. Documents werden zur Parameterübergabe zwischen Prozessen verwendet. Im Gegensatz zu *Entities* haben *Documents* keine Identität, kein Verhalten und keinen eigenen Zustand. Für verschiedene Zustände einer *Entity* ist es sinnvoll, jeweils separate *Documents* zu erzeugen. Diese unterscheiden sich z.B. in Attributen, die sie für einen bestimmten Zustand der *Entity* zur Verfügung stellen.

Transformation des CIM zum PIM

Regel PIM-14: (Regel_PIM_CD_Daten)

Die Klassendiagramme zur Modellierung der Datensicht auf CIM-Ebene werden auf PIM-Ebene ohne Änderung übernommen. Die Klassendiagramme werden um das zusätzliche Konzept *Business Document* erweitert. (formell siehe Anhang B)

Modellierung von Business Documents

Regel PIM-15: (Regel_PIM_CD_BDocument1)

- Die Erzeugung eines neuen Business Documents wird nötig, wenn auf CIM-Ebene eine Business Entity einem Prozess als Input- oder Outputparameter zugeordnet ist. Für einen Prozess wird stets maximal ein Business Document für jede Business Entity generiert, auch wenn eine Business Entity einem Prozess mehrfach zugeordnet ist.
- Business Documents werden als Datentypen mit dem Stereotyp *«document»* modelliert. Die Attribute von Business Documents einer Business Entity können sich unterscheiden und sind normalerweise nicht aus dem CIM abzuleiten. Daher erhält ein Business Document als Standard in der ARIS-BPDM Abbildung alle Attribute der zugehörigen Business Entity.

- Der Name eines Business Documents setzt sich aus dem Namen der zugehörigen Business Entity und dem zugehörigen Prozess folgendermaßen zusammen: *‚<Name Entity>_<Name Prozess>'*. (formell siehe Anhang B)

Regel PIM-16: (Regel_PIM_CD_BDocument2)

- Jedes Business Document ist einer Business Entity zugeordnet. Diese Verbindung wird im BPDM durch eine gerichtete Abhängigkeitsbeziehung (engl. Dependency) vom Business Document zur Business Entity repräsentiert. (formell siehe Anhang B)

4.2.5 Leistungssicht

Da schon bei der Abbildung auf CIM-Ebene keine relevanten Konzepte der Leistungssicht definiert waren, wird sie auch bei Transformation des CIMs zu einem PIM nicht weiter betrachtet.

4.2.6 Steuerungssicht

Überblick

Aus Steuerungssicht ergeben sich bei der Modellierung des PIMs im Vergleich zum CIM Änderungen. Der in den Aktivitätsdiagrammen modellierte Kontrollfluss wird auf PIM-Ebene um einen Datenfluss ergänzt. Auch die Beziehung zwischen Funktions- und Datensicht müssen überarbeitet werden. Lediglich die Beziehungen zwischen Funktions- und Organisationssicht können ohne Änderung übernommen werden.

Verwendete BPDM Konzepte

- **Process Flow Concept:** Das interne Verhalten eines Prozesses wird mit Hilfe einer Aktivität in einem Aktivitätsdiagramm modelliert. Die Aktivität wird dem Verhalten der *«process»* Klasse zugeordnet.
- **Atomic Task Concept:** Tasks sind die elementaren auszuführenden Schritte eines Prozesses. Sie werden in Aktivitätsdiagrammen als Aktionen modelliert. Dabei kann es sich um eine Berechnung, ein Update, dem Lesen von Zustandsdaten, dem Senden eines Ereignisses oder dem Aufruf eines Subprozesses handeln.
- **Data Object Concept:** Datenobjekte werden als Input- und Outputparameter zwischen Aktivitäten und Aktionen ausgetauscht. Sie werden in Aktivitätsdiagrammen als Objektknoten modelliert. Es handelt sich bei Datenobjekten stets um Business Documents und niemals um Business Entities.
- **Sequence flow concept:** Der Kontrollfluss regelt die Ausführungsreihenfolge von Aktivitäten und Aktionen bei der Beschreibung des Verhaltens eines Prozesses.
- **Data flow concept:** Mit dem Datenfluss wird die Austauschreihenfolge von Datenobjekten zwischen Aktivitäten und Aktionen beschrieben.
- **(Initial) receive und send:** Diese Konzepte dienen zur Modellierung von abstrakten Prozessen in Aktivitätsdiagrammen und des Kontrollflusses. Da in der Spezifikation des BPDM [IAF+04] nur ausführbare Prozesse behandelt werden, sind diese Konzepte dort noch nicht definiert. Erst in [FGJ04], das sich mit abstrakten und kollaborativen Prozessen auseinandersetzt, werden *(initial) receive* und *send* eingeführt und informell besprochen. In der ARIS-BPDM Abbildung werden diese als Aktionen mit den Stereotypen *«receive»* und *«send»* definiert.

Transformation des CIM zum PIM

Regel PIM-17: (Regel_PIM_Verhaltensdiagramme)

Bei der Transformation des CIMs zu einem PIM müssen aus Steuerungssicht die Diagramminhalte, bis auf die Beziehungen zwischen Funktions- und Organisationssicht, überarbeitet oder erweitert werden. Im Folgenden wird, wie schon beim Mapping von ARIS nach BPDM auf CIM-Ebene, die Steuerungssicht in mehrere Bereiche unterteilt, für die die Transformationsschritte im Einzelnen vorgestellt werden. (formell siehe Anhang B)

Kontroll- und Datenfluss

Aktivitätsdiagramme wurden schon, einschließlich ihrer Elemente zur Datenflussmodellierung, auf CIM-Ebene vorgestellt. Deshalb wird auf eine wiederholte Vorstellung verzichtet und gleich die Erweiterung der Aktivitätsdiagramme um den Datenfluss beschrieben.

Daten werden in Klassendiagrammen den Prozessen als Input- und Outputparameter statisch zugeordnet. Zusätzlich dazu ist es für das Verständnis von Prozessen und deren Abläufe hilfreich, den Datenfluss zwischen den Prozessen explizit abzubilden. Dies geschieht auf PIM-Ebene in Aktivitätsdiagrammen. Dort werden der Kontroll- und der Datenfluss parallel modelliert. Zur Umsetzung des Datenflusses in Aktivitätsdiagrammen sind folgende Punkte zu beachten:

Regel PIM-18 bis 19: (Regel_PIM_AD_Datenfluss1&2)

- Die den Prozessen als Input- und Outputparameter zugeordneten Business Documents (die auch schon auf CIM-Ebene als *«entities»* den Prozessen zugeordnet waren) werden in Aktivitätsdiagrammen als Pins der zugehörigen Aktionen modelliert. Die Pins erhalten dabei den Namen der Business Documents. (formell siehe Anhang B)

Regel PIM-20 bis 23: (Regel_PIM_AD_Datenfluss3-6)

- Aktionen mit den Stereotypen *«receive»* und *«send»* erhalten als Output- und Inputpins das zu dem Prozess, in dem sie modelliert werden, gehörige Business Document.[21] Das Business Document muss auch hier schon auf CIM-Ebene als Business Entity dem Prozess zugeordnet gewesen sein. Liegt eine Orchestration vor, werden die Business Documents der Aktivität des beschriebenen Prozesses als Eingangs- bzw. Ausgangsparameter zugeordnet.

Regel PIM-24 bis 26: (Regel_PIM_AD_Datenfluss7-9)

- Der Daten- bzw. Objektfluss wird durch Kanten zwischen den Pins der Aktionen, Objektknoten und den Eingabe- und Ausgabeparametern einer Aktivität modelliert. Dabei müssen die Objekttypen der Pins an den Enden einer Kante zueinander kompatibel sein. (formell siehe Anhang B)

Regel PIM-27 bis 29: (Regel_PIM_AD_Datenfluss10-12)

- Weist eine Aktion einen Ausgangsparameter auf, den die Nachfolgeaktion nicht als Inputparameter hat, so wird das Datentoken in einem Objektknoten mit dem Stereotyp *«datastore»* gespeichert. Die Kante zwischen dem Ausgangspin und dem Objektknoten wird mit der Bezeichnung *‚{copy}'* versehen. Fehlt einer Aktion ein Ausgangsparameter, den die Nachfolgeaktion als Eingangsparameter besitzt, so kopiert sich die

[21] Dies sind nicht die Business Documents der Prozesse, due die Aktionen repräsentieren.

Nachfolgeaktion das benötigte Datentoken aus einem Datenspeicher. Auch hier wird die Kante zwischen dem Datenspeicher und der Nachfolgeaktion mit der Bezeichnung *‚{copy}'* versehen. (formell siehe Anhang B)

Regel PIM-30 bis 45: (Regel_PIM_AD_Datenfluss13-28)

- Kontrollaktionen und elementare Aktionen sind aus statischer Sicht durch Klassendiagramme keine Input- und Outputparameter zugeordnet. In Aktivitätsdiagrammen ist es jedoch notwendig, Kontrollaktionen in den Objektfluss mit einzubeziehen. Daher werden ihnen Pins als Input- und Outputparameter zugewiesen. Führt z.B. eine Kante von einer Aktion zu einer Kontrollaktion, so erhält die Kontrollaktion prinzipiell alle Pins, die Outputparameter der Aktion sind, als Inputparameterpins. Die zueinandergehörigen Pins werden mit Kanten verbunden. Analog erhält eine Kontrollaktion Pins als Outputparameter, die eine nachfolgende Aktion als Inputparameter hat. Ausnahmen können sich hier bei dem Einsatz von Objektknoten mit dem Stereotyp *«datastore»* ergeben (für Details siehe formale Regeln im Anhang). (formell siehe Anhang B)

Beziehungen zwischen Funktions- und Datensicht

Wurden auf CIM-Ebene Prozessen noch Business Entities zugeordnet, so werden auf PIM-Ebene Input- und Outputparameter von Prozessen durch Business Documents repräsentiert. Die Transformation der Beziehungen zwischen Funktions- und Datensicht von CIM zur PIM-Ebene erfolgt folgendermaßen.

Regel PIM-46 bis 47: (Regel_PIM_CD_Input-&Outputparameter1)

- Die auf CIM-Ebene modellierten Assoziationen zwischen Prozessen und Business Entities werden im Klassendiagramm auf PIM-Ebene nicht übernommen. Stattdessen wird für jede dieser Assoziationen eine neue Assoziation zwischen dem Prozess und dem zugehörigen, in der Datensicht neu generierten, Business Document erstellt. Der Name der Assoziation *‚isInput'* oder *‚hasOutput'* wird dabei von der Assoziation auf CIM-Ebene übernommen. (formell siehe Anhang B)

Regel PIM-48 bis 49: (Regel_PIM_CD_Input-&Outputparameter2)

- Zusätzlich erhält jeder Prozess die Input- und Outputparameter (Business Documents) seiner im Sinne einer Prozess Orchestration untergeordneten Prozesse als Input- und Outputparameter zugewiesen. (formell siehe Anhang B)

Beziehungen zwischen Funktions- und Organisationssicht

Die Beziehungen zwischen Funktions- und Organisationssicht ändern sich bei der Transformation eines CIMs zu einem PIM nicht.

Beziehungen zwischen Daten- und Organisationssicht

Im CIM waren keine Beziehungen zwischen der Daten- und der Organisationssicht vorhanden und es werden im PIM auch keine neuen Beziehungen eingeführt.

5 Fallstudie

Wurde die ARIS-BPDM Abbildung im vorhergehenden Abschnitt spezifiziert, so wird sie in einer Fallstudie an einem Praxisbeispiel angewendet. Die Fallstudie soll die Funktionsweise der in Kapitel 4 besprochenen Abbildung darstellen und deren Praxistauglichkeit exemplarisch aufzeigen. Dazu werden nach der Vorstellung eines UseCases aus der Automobilindustrie in Abschnitt 5.1 die Abbildungsschritte mit ihren wichtigsten Transformationsregeln an Diagrammausschnitten in Kapitel 5.2 und 5.3 besprochen.

5.1 Vorstellung des Fallstudienbeispiels

Die Fallstudie zum Einsatz der ARIS-BPDM Abbildung wird anhand eines Beispiels aus der Automobilindustrie durchgeführt. Dort ist es unüblich, dass Automobilhersteller ihre Zulieferer für Produktionsreihen oft und kurzfristig wechseln. Zuliefererverträge haben meist eine mittel- bis langfristige Vertragslaufzeit und dementsprechend hohe Auftragsvolumina. Aus diesem Grund kommt dem Prozess der Zuliefererauswahl besondere Bedeutung zu. Es ist wichtig, diesen Prozess exakt zu modellieren und durch IuK-Systeme zu unterstützen. IuK-Systeme führen den Prozess zwar nicht automatisch durch, stellen aber z.B. ein Monitoring zur Verfügung oder koordinieren die Angebotseinholung bei den Zulieferern. Durch ein IuK-System kann darüber hinaus die Überprüfung der Bedingungen bezüglich der Zusammensetzung der Zulieferer oder die Kombinierbarkeit der Zuliefererteile erleichtert werden.

Das Fallstudienbeispiel umfasst die Angebotseinholung bei der Auswahl von Automobilteilezulieferern durch einen Automobilhersteller. Obwohl die Prozesse des Beispiels im Vergleich zur Realität stark verkürzt sind, spiegeln sie doch wesentliche Vorgänge der Angebotseinholung bei den Zulieferern und die Auswahl von Zulieferern wieder. Das Fallstudienbeispiel ist so gestaltet, dass die Funktionsweise der ARIS-BPDM Abbildung in ihm gezeigt und jeder Zeit um weitere Prozesse erweitert werden kann.

An dem unternehmensübergreifenden Prozessbeispiel sind drei Rollen beteiligt, die von Unternehmen oder deren Abteilungen eingenommen werden können.

- **OEM (Original Equipment Manufacturer):** Der OEM ist der Automobilhersteller, der einen neuen Automobiltyp produzieren möchte.
- **PO (Purchasing Organisation):** Die PO kann ein eigenständiges Unternehmen oder eine Abteilung des OEMs sein, die die Angebotseinholung bei den Zulieferern durch-

führt und die endgültige Auswahl der Zulieferer trifft, um die Zulieferverträge abschließen zu können.

- **SU (Supplier):** Der SU ist ein Automobilteilezulieferer, der mit dem OEM bzw. der PO einen Zuliefervertrag abschließen möchte.

In Abbildung 27 findet sich eine Übersicht der von den drei Rollen unterstützen Prozesse[22]. Die Rollen OEM, PO und SU werden mit Aktivitätsbereichen beschrieben. Befindet sich ein Prozess im Aktivitätsbereich, z.B. des SUs, so wird der Prozess vom SU durchgeführt und implementiert.

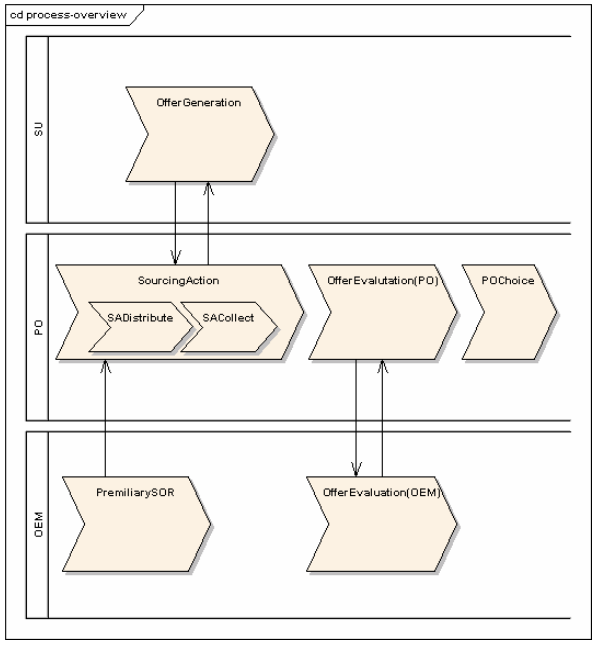


Abbildung 27: Fallbeispiel - Prozessübersicht

Der unternehmensübergreifende Prozessfluss beginnt mit dem Prozess *Preliminary-SOR*, der vom OEM durchgeführt wird. Nachdem die Konstruktionsphase eines neuen Automobiltyps abgeschlossen ist, werden in *PreliminarySOR* die Anforderungen an eine Produktion ermittelt. Resultat des *PreliminarySOR* ist ein so genanntes ‚Statement of Requirements' (SOR). Der OEM stellt das SOR der PO zur Verfügung, die den Prozess *SourcingAction* durchführt. *SourcingAction* besteht wiederum aus zwei Subprozessen, *SADistribute* und *SACollect*. In *SADistribute* wird aus dem SOR ein ActionPlan generiert, der Informationen zu den zu beschaffenden Zulieferteilen enthält (wie z.B. Objektbeschreibungen, Typ, Menge, Maximalkosten oder die Zeit, in der Teile in der Supply Chain bereitgestellt werden müssen). Anschließend werden aus dem ActionPlan OfferRequests erzeugt und an geeignete SUs versandt. Ein SU prüft im Prozess *OfferGeneration*, ob und zu welchem Preis er bestimmte Teile zuliefern kann. Er erzeugt ein Offer, das er an die PO zurückschickt. Die PO sammelt im Prozess *SACollect* die eingehenden Offers ein und speichert diese in einem einheitlichen Format in einer Datenbank ab. Sind alle Angebote eingegangen, beginnt die Evaluierung der Angebote. Diese wird vom OEM und der PO gleichzeitig durchgeführt. Der OEM prüft im Prozess *OfferEvaluation(OEM)* die eingegangenen Offers aus technischen Gesichtspunkten, d.h. beispielsweise, ob die versprochene

[22] Diese Übersicht ist nicht Bestandteil der eigentlichen Prozessbeschreibung. Sie dient zum besseren Verständnis des Fallbeispiels und wird beim Mapping von ARIS nach BPDM auch nicht beachtet.

Qualität bestimmter Teile für die Produktion ausreichend ist. Die PO untersucht im Prozess *OfferEvaluation(PO)* die Offers aus einer ökonomischen Sicht, d.h. wie zum Beispiel mögliche Konstellationen der Zulieferer aussehen oder woher die Zulieferteile beschafft werden müssen. Anschließend vergleicht die PO ihre Ergebnisse mit denen des OEMs und startet bei befriedigenden Resultaten den Prozess *POChoice*. In *POChoice* werden die SUs ausgewählt, mit denen Zulieferverträge abschlossen werden sollen.

5.2 Mapping von ARIS nach BPDM auf CIM-Ebene

In diesem Abschnitt wird das ARIS-BPDM Mapping auf CIM-Ebene vorgestellt. Dabei wird von den drei zur Modellierung in ARIS verwendeten Diagrammen ausgegangen und besprochen, wie diese Diagramme nach BPDM abgebildet werden.

5.2.1 Wertschöpfungskettendiagramme in ARIS

In Abbildung 28 ist ein Wertschöpfungskettendiagramm des OEMs zu sehen. Der Prozess *OEM* hat zwei Subprozesse, *PreliminarySOR* und *OfferEvaluation(OEM)*, die ihm prozessorientiert untergeordnet sind. Auf das Wertschöpfungsdiagramm lassen sich nun zwei Regeln des ARIS-BPDM Mappings anwenden.

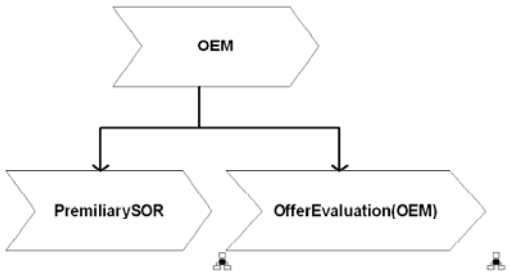


Abbildung 28: Fallbeispiel – ARIS-WKD

Mit der Regel **[Regel_CIM_CD_Prozess]** werden aus den Prozessen des WKDs Klassen mit dem Stereotyp *«process»* im Klassendiagramm des BPDM-Modells (siehe Abbildung 29). Die prozessorientierte Hierarchie wird durch die Regel **[Regel_CIM_CD_Prozesshierarchie1]** abgebildet. Die gerichtete Kante vom Prozess *OEM* zu dem Prozess *PreliminarySOR* wird beispielsweise durch eine gerichtete Assoziation von der *«process»* Klasse OEM zu der *«process»* Klasse PreliminarySOR abgebildet. Dem untergeordneten Prozess (in diesem Fall die *«process»* Klasse PreliminarySOR) wird die Rolle *‚subProcess'* zugeordnet.

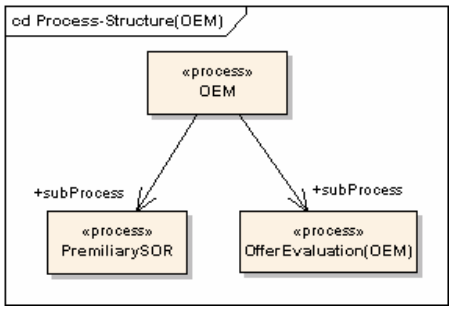


Abbildung 29: Fallbeispiel – CIM Prozessstruktur1

5.2.2 Ereignisgesteuerte Prozessketten in ARIS

Ereignisgesteuerte Prozessketten werden auf unterschiedliche Diagrammtypen eines BPDM-Modells abgebildet. Dieser Abschnitt beschreibt die Abbildungsregeln anhand des Fallstudienbeispiels nach Diagrammtypen gruppiert.

Als ARIS EPK wird die EPK zu dem Prozess *OfferEvaluation(OEM)* besprochen (siehe Abbildung 30). Der Prozessablauf kann durch die in den Prozessschnittstellen modellierten Prozesse *SourcingAction* und *OfferEvaluation(PO)* angestossen werden. Geschieht dies durch den Prozess *SourcingAction*, so muss das Ereignis *‚all offer requests returned'* gelten. Anschließend wird der Subprozess *CheckOffer(OEM)* ausgeführt. Ist dieser beendet, gilt das Ereignis *‚OEM check complete'* und über der Prozess *OfferEvaluation(PO)* wird eine Prozessschnittstelle aufgerufen.

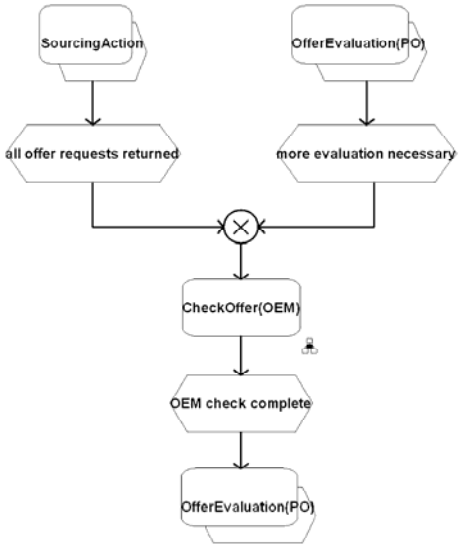


Abbildung 30: Fallbeispiel – ARIS-EPK

Klassendiagramm

Zur Abbildung von in der EPK modellierten Konzepten auf Klassendiagramme des BPDM-Modells können die Regeln **[Regel_CIM_CD_Prozess]** und **[Regel_CIM_CD_Prozesshierarchie2]** des ARIS-BPDM Mappings verwendet werden. Die in der EPK modellierte Funktion *CheckOffer(OEM)* wird nach Regel **[Regel_CIM_CD_Prozess]** durch die *«process»* Klasse CheckOffer(OEM) abgebildet (siehe Abbildung 31). Da *CheckOffer(OEM)* ein Subprozess des in der EPK modellierten Prozesses *OfferEvaluation(OEM)* ist, kann Regel **[Regel_CIM_CD_Prozesshierarchie2]** angewendet werden. Diese bildet die Prozesshierarchie durch eine gerichtete Kante von der *«process»* Klasse OfferEvaluation(OEM) zur *«process»* Klasse CheckOffer(OEM) ab.

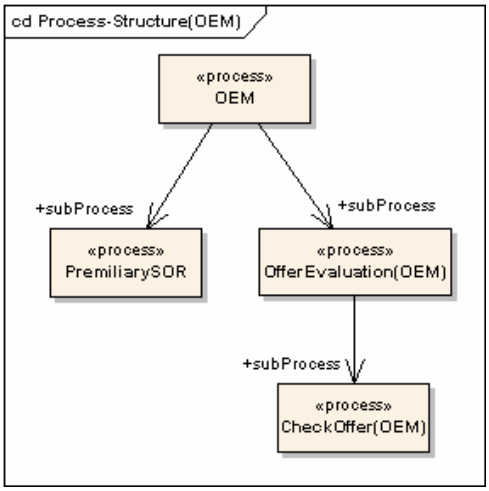


Abbildung 31: Fallbeispiel – CIM Prozessstruktur2

Sequenzdiagramm

Die Prozessschnittstellen einer EPK werden in Sequenzdiagrammen abgebildet. Die Erzeugung eines Sequenzdiagramms ist in der Regel **[Regel_CIM_SD_Sequenzdiagramm1]** beschrieben. Aus der Prozessschnittstelle *SourcingAction* wird ein Sequenzdiagramm mit dem Namen *‚SourcingAction_OfferEvaluation(OEM)'* generiert (siehe Abbildung 32). Die Prozesse *SourcingAction* und *OfferEvaluation(OEM)* werden im Sequenzdiagramm als Lebenslinien übernommen. Für die Modellierung der zwischen den beiden Prozessen ausgetauschten Nachricht muss nun unterschieden werden, ob zwischen den Prozessen ein Orchestration- oder Choreographyverhältnis vorliegt. Da ein Choreographyverhältnis vorliegt, kann Regel **[Regel_CIM_SD_ChorNachricht]** angewandt werden. Das Sequenzdiagramm wird zwischen den Prozessen *SourcingAction* und *OfferEvaluation* um eine asynchrone Nachricht mit dem Namen *‚offerEvaluation(OEM)'* ergänzt.

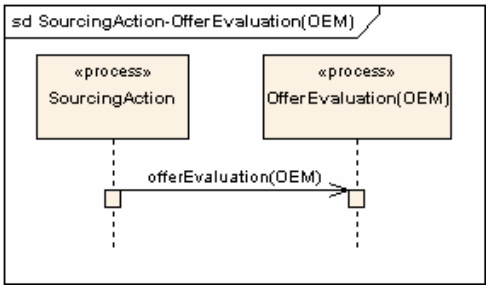


Abbildung 32: Fallbeispiel – CIM Sequenzdiagramm1

Da der Prozess CheckOffer(OEM) nicht durch eine EPK hinterlegt ist, kann die Regel **[Regel_CIM_SD_Sequenzdiagramm2]** angewandt werden. Es werden zwei Sequenzdiagramme erzeugt (siehe Abbildung 33), die jeweils den Prozess *CheckOffer(OEM)* und den übergeordneten Prozess *OfferEvaluation(OEM)* als Lebenslinie enthalten. Da die beiden Prozesse in einem Orchestrationverhältnis stehen und da nach Annahme eine asynchrone Kommunikation vorliegt, werden nun die Regeln **[Regel_CIM_SD_OrchAsynNachricht3]** und **[Regel_CIM_SD_OrchAsynNachricht4]** zur Modellierung der Nachrichten in den Sequenzdiagrammen angewandt.

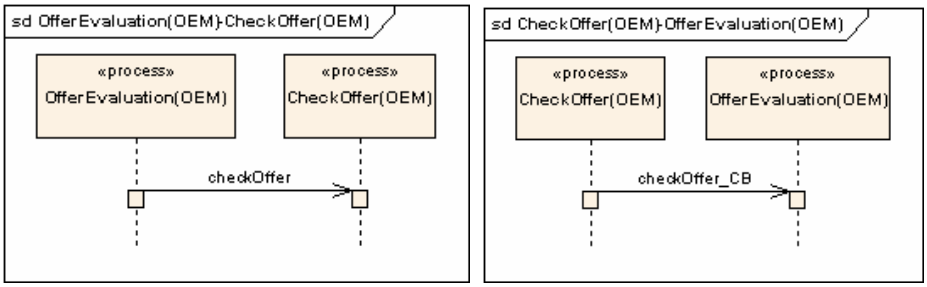


Abbildung 33: Fallbeispiel – CIM Sequenzdiagramm2+3

Aktivitätsdiagramm

Eine EPK beschreibt die interne Realisierung eines Prozesses. Der Prozessablauf wird in BPDM in Aktivitätsdiagrammen modelliert, das nach Regel **[Regel_CIM_AD_Aktivitätsdiagramm]** mit dem Namen *‚OfferEvaluation(OEM)-ExecutableProcess'* erzeugt wird (siehe Abbildung 34). Wendet man die Regel **[Regel_CIM_AD_ChorAktion1]** auf die Prozessschnittstellen an, so werden im Aktivitätsdiagramm zwei Aktionen mit dem Stereotyp *«receive»* generiert. Analog wird die Aktion OfferEvaluation(PO) mit dem Stereotyp *«send»* hinzugefügt (Regel **[Regel_CIM_AD_ChorAktion2]**). Der Subprozess CheckOffer(OEM) wird nach Regel **[Regel_CIM_AD_Aktion]** als Aktion, die einen Subprozess repräsentiert, umgesetzt. Schließlich wird noch das Kontrollelement nach BDPM transformiert (Regel **[Regel_CIM_AD_ODERAktion2]**).

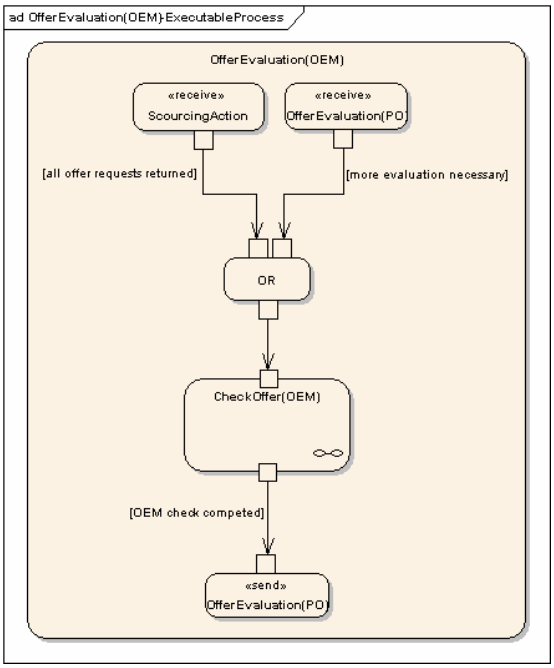


Abbildung 34: Fallbeispiel – CIM Aktivitätsdiagramm

Abschließend können die Regeln zur Abbildung des Kontrollflusses der EPK angewendet werden. Im vorliegenden EPK treffen die Regeln **[Regel_CIM_AD_Kontrollfluss4]**, **[Regel_CIM_AD_Kontrollfluss15]** und **[Regel_CIM_AD_Kontrollfluss25]** zu und werden zur Modellierung des Kontrollflusses im Aktivitätsdiagramm eingesetzt.

Interaktionsübersichtsdiagramm

Ein Interaktionsübersichtsdiagramm wird zur Modellierung eines abstrakten Prozesses eingesetzt. Regel **[Regel_CIM_IOD_Interaktionsübersichtsdiagramm]** schreibt im Falle

des Prozesses *OfferEvaluation(OEM)* die Erstellung eines Interaktionsübersichtsdiagramms mit dem Namen *‚OfferEvaluation(OEM)-AbstractProcess'* vor (siehe Abbildung 35). Der Kontrollfluss wird zwischen den Interaktionsreferenzen und der Aktion, die den ausführbaren Prozess repräsentiert, modelliert. Dabei kommen Regel **[Regel_CIM_IOD_EingehendeNachricht]** und Regel **[Regel_CIM_IOD_AusgehendeNachricht]** zu Einsatz.

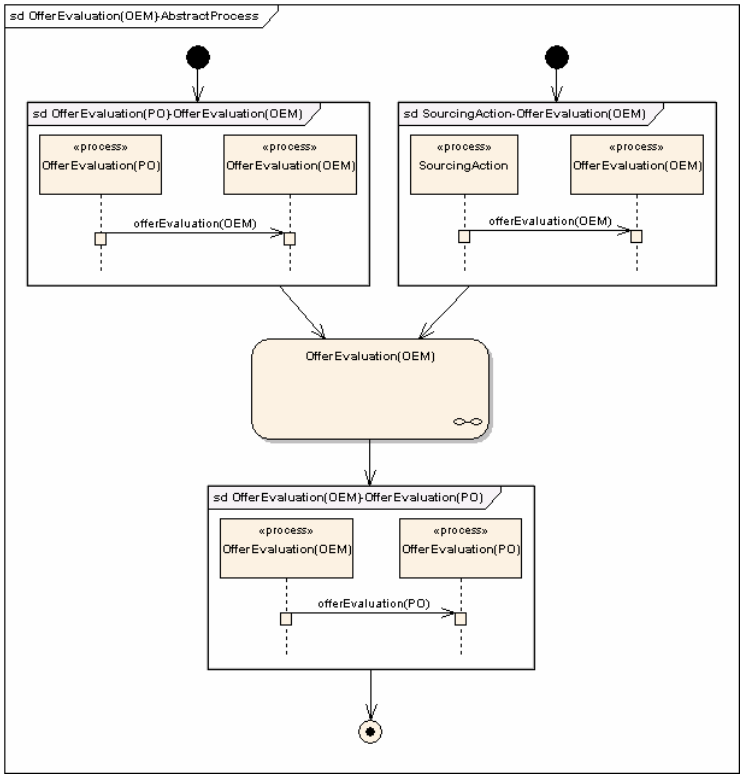


Abbildung 35: Fallbeispiel – CIM Interaktionsübersichtsdiagramm

5.2.3 Funktionszuordnungsdiagramme in ARIS

Funktionszuordnungsdiagramme dienen der Zuordnung von Funktionen bzw. Prozessen zu Konzepten anderer ARIS Beschreibungssichten. In Abbildung 36 ist ein FZD zum Prozess *OfferEvaluation(OEM)* zu sehen. Dem Prozess werden Fachbegriffe aus der Datensicht und Personentypen aus der Organisationssicht zugeordnet.

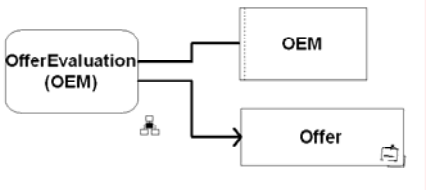


Abbildung 36: Fallbeispiel – ARIS-FZD

Der Fachbegriff *Offer* wird mit Hilfe der Regel **[Regel_CIM_CD_BEntity]** als Klasse mit dem Stereotyp *«entity»* nach BPDM transformiert. Auf den Personentyp OEM muss, zu einer Abbildung auf eine Klasse des Stereotyps *«role»*, die Regel **[Regel_CIM_CD_Rolle]** angewendet werden. Schließlich wird das Klassendiagramm in Abbildung 37 noch um Beziehungstypen ergänzt. Da der Fachbegriff *Offer* ein Outputparameter der Funktion *OfferEvaluation(OEM)* ist, kann die Regel **[Regel_CIM_CD_Outputparameter]** zur Abbil-

dung dieser Beziehung eingesetzt werden. Schließlich bildet Regel **[Regel_CIM_CD_Rollenzuordnung2]** noch die Beziehung *„führt aus“* nach BPDM als Assoziation mit der Bezeichnung *‚executes’* ab.

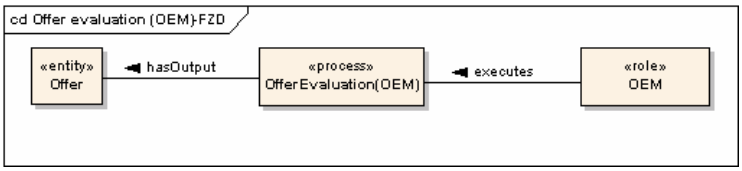


Abbildung 37: Fallbeispiel – CIM BPDM-Funktionszuordnung

5.3 Transformation des BPDM-CIMs zu einen BPDM-PIM

In diesem Abschnitt wird die Transformation des BPDM-Modells auf CIM-Ebene in ein PIM besprochen. Der Abschnitt ist in die Transformation der statischen und der dynamischen Modellbestandteile unterteilt. Es wird dabei erläutert, wie die Diagramme des PIMs aus Diagrammen des CIMs abgeleitet werden.

5.3.1 Transformation der statischen Modellbestandteile

In den Klassendiagrammen des CIMs werden Prozesse als Klassen mit dem Stereotyp *«process»* einschließlich Prozesshierarchien und -anordnungen modelliert. Diese werden in das PIM übernommen und um Ports und Schnittstellen erweitert.

Für die Erweiterung eines Prozesses um Ports und Schnittstellen müssen auch die Aktivitätsdiagramme betrachtet werden, in denen der interne Ablauf eines Prozesses oder ein Prozess als Aktion modelliert ist. Im Falle des Prozesses *OfferGeneration* weist das Aktivitätsdiagramm aus Abbildung 38 zwei Aktionen mit den Stereotypen *«receive»* oder *«send»* auf.

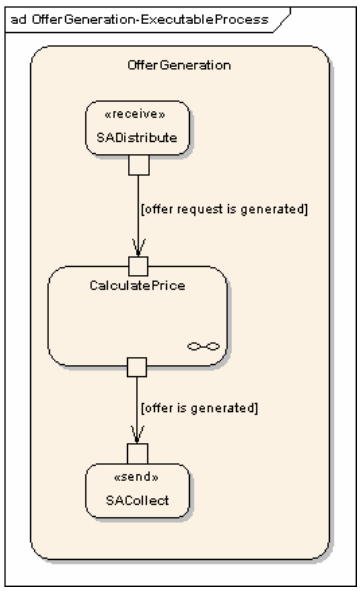


Abbildung 38: Fallbeispiel – CIM Aktivitätsdiagramm1

Der Prozess *OfferGeneration* kommuniziert mit den Prozessen *SADistribute*, *SACollect* und *CalculatePrice*. Für jeden dieser Prozesse erhält er aufgrund der Regeln

[Regel_PIM_CD_Port1&2&3] für jeden Partnerprozess einen eigenen Port (wie in Abbildung 39 z.B. für den Partnerprozess *SACollect* den Port *OfferGeneration_SACollect*). Nach Regel **[Regel_PIM_CD_SchnittstelleChor1]** wird der Port *OfferGeneration_SACollect* um das *required Interface SACollect_OfferGeneration_CHOR* erweitert. Regel **[Regel_PIM_SchnittstelleChor2]** fügt dem Port *OfferGeneration_SADistribute* das *provided Interface OfferGeneration_SADistribute_CHOR* hinzu. Da nach Annahme eine asynchrone Kommunikation vorliegt, werden die Ports *OfferGeneration_CalculatePrice* und *CalculatePrice_OfferGeneration* der Prozesse *OfferGeneration* und *CalculatePrice* um Schnittstellen mit den Regeln **[Regel_PIM_CD_SchnittstelleOrch1&2]** erweitert. So werden die beiden Schnittstellen *CalculatePrice_ORCH* und *CalculatePrice_CB* erzeugt (siehe Abbildung 40). In Abbildung 39 werden diese Schnittstellen den entsprechenden Ports der Prozesse *OfferGeneration* und *CalculatePrice* als *provided* bzw. *required Interfaces* zugeordnet.

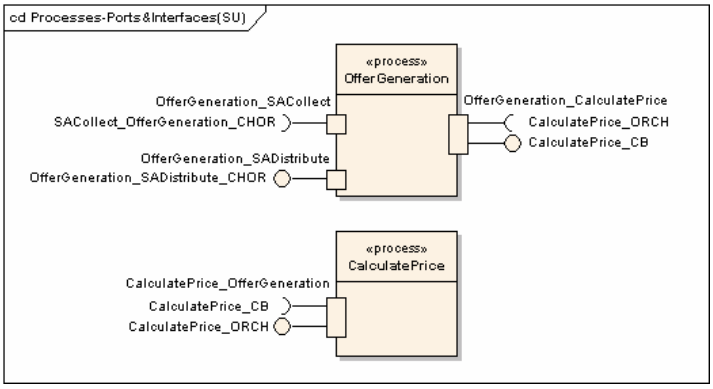


Abbildung 39: Fallbeispiel – PIM Ports & Prozessschnittstellen

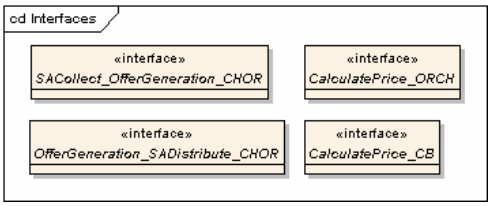


Abbildung 40: Fallbeispiel - Schnittstellen

Während die Regel **[Regel_PIM_CD_Organisation]** keine Erweiterung der Klassendiagramme um Aspekte der Organisationssicht vorsieht, wird die Datensicht um das Konzept der Business Documents erweitert. Abbildung 41 und Abbildung 42 zeigen die Business Entities, die als Input- bzw. Outputparameter der Prozesse *OfferGeneration* und *CalculatePrice* auf CIM-Ebene modelliert wurden.

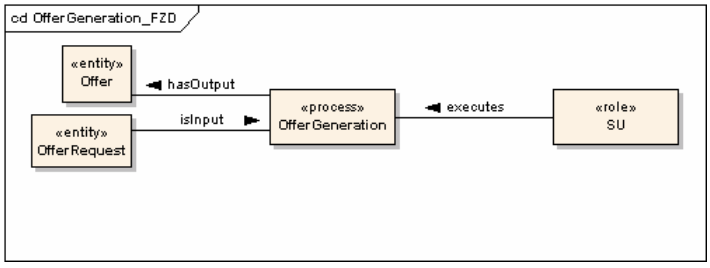


Abbildung 41: Fallbeispiel – CIM Funktionszuordnung1

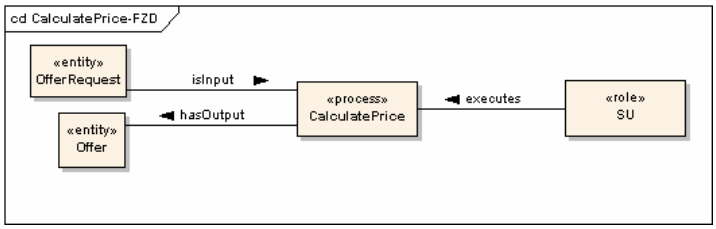


Abbildung 42: Fallbeispiel – CIM Funktionszuordnung2

Nach Regel **[Regel_PIM_CD_BDocument1]** werden nun zu den Business Entities der Prozesse spezifische Business Documents erzeugt. Abbildung 43 zeigt die *«document»* Klassen einschließlich der Abhängigkeitsbeziehungen zu den zugehörigen *«entity»* Klassen (nach Regel **[Regel_PIM_CD_BDocument2]**).

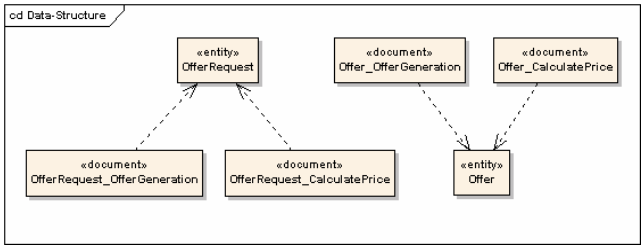


Abbildung 43: Fallbeispiel - Datenstruktur

Mit den Regeln [**Regel_PIM_CD_Inputparameter1]** und **[Regel_PIM_CD_Outputparameter1]** werden diese neu erzeugten *«document»* Klassen den beiden Prozessen als Input- und Outputparameter zugewiesen. Zusätzlich erhält der Prozess *OfferGeneration* die Inputparameter seines Subprozesses *CalculatePrice* als Outputparameter und umgekehrt (nach Regel [**Regel_PIM_CD_Inputparameter2]** und **[Regel_PIM_CD_Outputparameter2]**) (siehe Abbildung 44 und Abbildung 45).

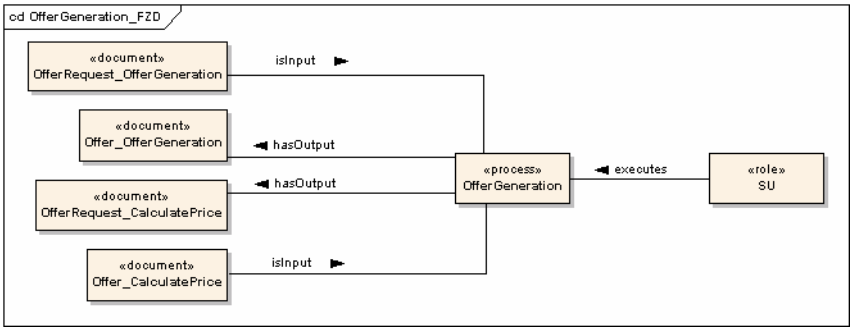


Abbildung 44: Fallbeispiel – PIM Funktionszuordnung1

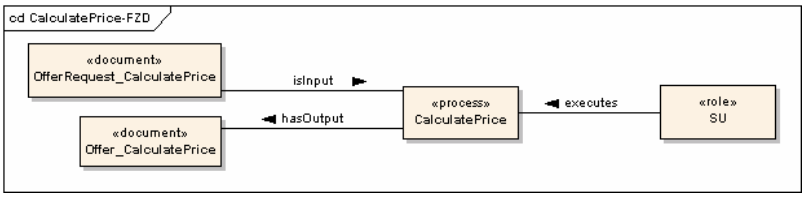


Abbildung 45: Fallbeispiel – PIM Funktionszuordnung2

5.3.2 Transformation der dynamischen Modellbestandteile

Bei der Generierung eines PIMs aus einem CIM werden nach Regel **[Regel_PIM-Verhaltensdiagramm]** alle Diagramme zur Verhaltensmodellierung (d.h. Sequenz-, Aktivi-

täts- und Interaktionsübersichtsdiagramme) ohne Änderungen übernommen. Lediglich die in den Aktivitätsdiagrammen modellierten Prozessabläufe werden um einen Datenfluss erweitert.

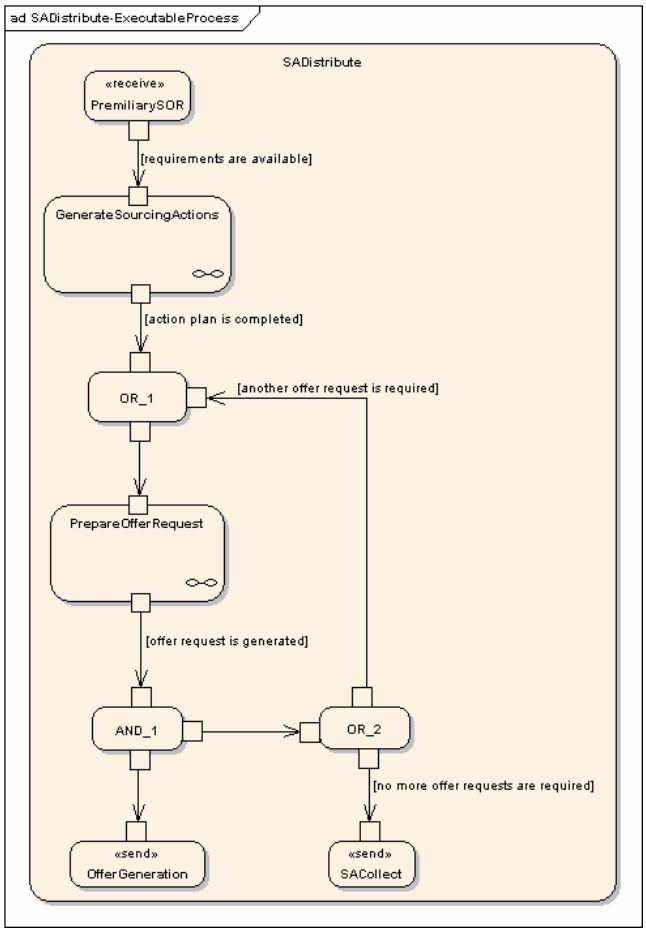


Abbildung 46: Fallbeispiel – CIM Aktivitätsdiagramm2

Zur Modellierung des Datenflusses wird im Fallsbeispiel der Prozess *SADistribute* betrachtet (das Aktivitätsdiagramm aus Abbildung 46 wird um den Datenfluss in Abbildung 50 erweitert). Zusätzlich muss auch noch die Zuordnung von Business Entities als Input- und Outputparameter zu den beschriebenen Prozessen bekannt sein (vgl. Klassendiagrammen in Abbildung 47 bis Abbildung 49).

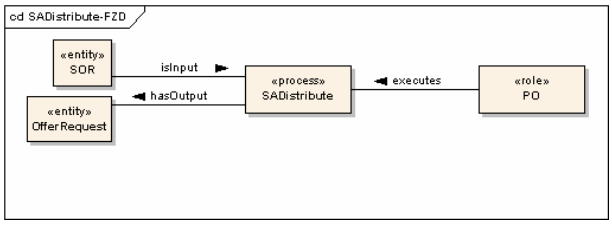


Abbildung 47: Fallbeispiel – CIM Funktionszuordnung3

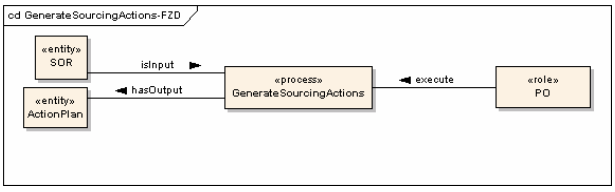


Abbildung 48: Fallbeispiel – CIM Funktionszuordnung4

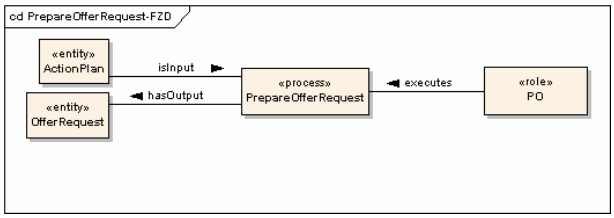


Abbildung 49: Fallbeispiel – CIM Funktionszuordnung5

Nun kann mit der Regel **[Regel_PIM_AD_Datenfluss1]** ein Inputpin *‚SOR_GenerateSourcingActions'* dem Subprozess *GenerateSourcingActions* hinzugefügt werden. Der Outputpin *‚ActionPlan_GenerateSourcingActions'* wird durch die Regel **[Regel_PIM_AD_Datenfluss2]** generiert. Analog dazu wird der Prozess *PrepareOfferRequest* um den Inputpin *‚ActionPlan_PrepareOfferRequest'* und den Outputpin *‚OfferRequest_PrepareOfferRequest'* erweitert. Um die Aktionen mit den Stereotypen *«receive»* und *«send»* mit Outputpins und Inputpins zu versehen, müssen nun die Regeln **[Regel_PIM_AD_Datenfluss3&4]** angewendet werden. (vgl. Abbildung 50)

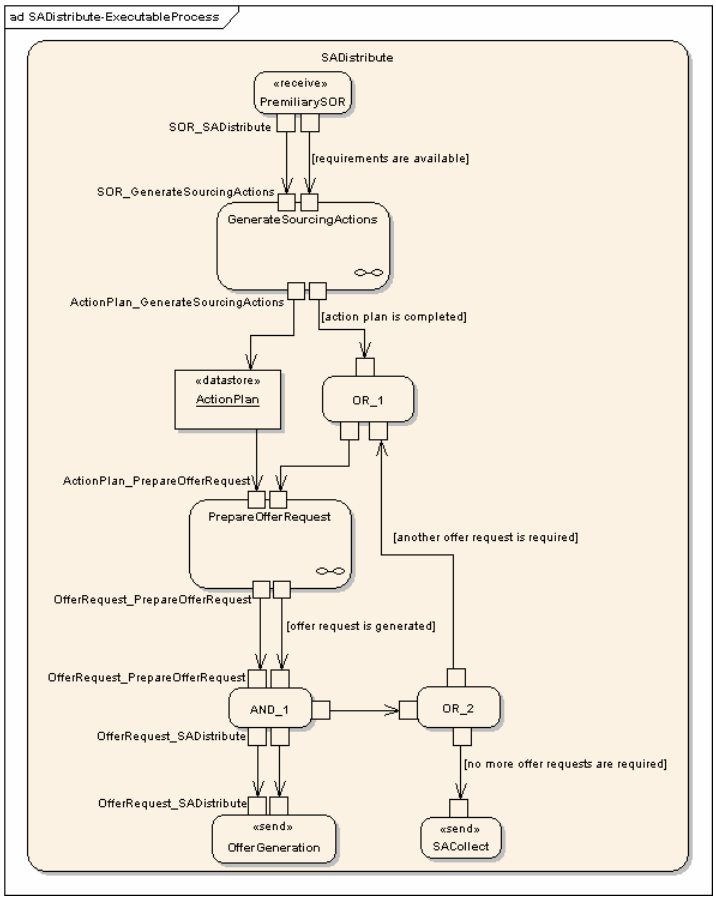


Abbildung 50: Fallbeispiel – PIM Aktivitätsdiagramm1

Zur Modellierung der Datenflusskanten stehen, wie auf CIM-Ebene bei der Modellierung der Kontrollflusskanten, eine Reihe von Transformationsregeln zur Auswahl. Mit der Regel **[Regel_PIM_AD_Datenfluss7]** kann eine Datenflusskante zwischen dem Outputpin *‚SOR_SADistribute'* der Aktion *PreliminarySOR* mit dem Stereotyp *«receive»* und dem Inputpin *‚SOR_GenerateScourcingActions'* der Aktion *GenerateSourcingActions* modelliert werden. Die Datenflusskante vom Outputpin *‚ActionPlan_GenerateScourcingActions'* der Aktion *GenerateSourcingActions* zum Objektknoten *ActionPlan* wird durch Anwendung der Regel **[Regel_PIM_AD_Datenfluss17]** hinzugefügt. Zur Modellierung der vom

Objektknoten *ActionPlan* ausgehenden Datenflusskante wird die Regel **[Regel_PIM_AD_Datenfluss18]** eingesetzt. Schließlich werden noch die Datenflusskanten, die in die Kontrollaktion *AND_1* ein- und ausgehen, analog zu den Regeln **[Regel_PIM_AD_Datenfluss21]** und **[Regel_PIM_AD_Datenfluss22]** dem Aktivitätsdiagramm hinzugefügt.

Bis jetzt wurden in der Fallstudie Abbildungsregeln lediglich auf Prozesse angewendet, die zueinander in einem Choreographyverhältnis stehen. Die Abbildung von Prozessen die in einem Orchestrationverhältnis stehen erfolgt jedoch analog. Wie in Abbildung 48 zu sehen ist, verfügt der Prozess *GenerateSourcingActions* über einen Inputparameter vom Typ *SOR* und einen Outputparameter vom Typ *ActionPlan*. Nach Regel **[Regel_PIM_AD_Datenfluss5&6]** werden diese Input- und Outputparameter der Aktivität in Abbildung 51 als Eingangs- bzw. Ausgangsparameter zugeordnet. Anschließend wird noch der Datenfluss im ausführbaren Prozess von *GenerateSourcingActions* mit den Regeln **[Regel_PIM_AD_Datenfluss8]** und **[Regel_PIM_AD_Datenfluss9]** modelliert.

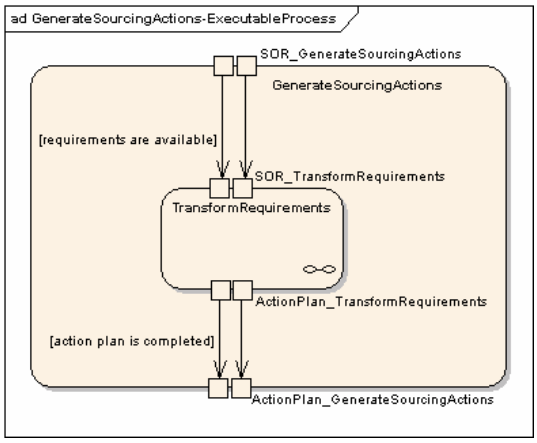


Abbildung 51: Fallbeispiel – PIM Aktivitätsdiagramm2

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde eine Abbildung zwischen der Architektur integrierter Informationssysteme und dem Business Process Definition Metamodel entworfen. Um sich auf eine für die Prozessbeschreibung mit ARIS wichtige Menge von ARIS-Konzepten festlegen zu können, wurde sowohl das Siemens Referenzprozesshaus als auch das ARIS Metamodel betrachtet. Des Weiteren wurde evaluiert, wie Prozesse mit dem BPDM möglichst ‚ideal' modelliert werden. Durch die Abbildung der identifizierten ARIS-Konzepte auf eine ‚idealtypische' Darstellung von Prozessen mit dem BPDM, konnte eine Abbildung von ARIS-Modellen auf erweiterbare und möglichst generische BPDM-Modelle erreicht werden. So könnten die in der Abbildung entworfenen BPDM-Modelle ohne große Änderungen um kollaborative Prozesse erweitert werden. Die Abbildung selbst wurde in zwei Abbildungsschritte unterteilt, die mit insgesamt 120 Abbildungsregeln formell beschrieben sind. Der erste Abbildungsschritt umfasst ein Mapping von ARIS-Modellen zu BPDM-Modellen auf CIM-Ebene der MDA, in dem ARIS-Modelle in möglichst semantisch äquivalente BPDM-Modelle umgeformt werden. Im zweiten Abbildungsschritt wurde das BPDM-Modell auf CIM-Ebene um Konzepte wie Ports, Schnittstellen oder Datenfluss erweitert und zu einem PIM transformiert. Abschließend wird die Anwendung der ARIS-BPDM Abbildung an einem praxisorientierten Fallbeispiels illustriert.

Die in dieser Arbeit entwickelte ARIS-BPDM Abbildung stellt einen Baustein der Automatisierung von Geschäftsprozessen im Rahmen des MDA-Ansatzes dar. Die Abbildung stellt Regeln zur Transformation von berechnungsunabhängigen Modellen zu plattformunabhängigen Modellen zur Verfügung. Zur Unterstützung einer modellgetriebenen Geschäftprozessautomatisierung müssen nun noch die Abbildung zu plattformspezifischen Modellen und ausführbarem Code bereitgestellt werden. Die Abbildung der PIMs zu PSMs könnte beispielsweise durch eine Transformation des BPDM-Modells in ein plattformspezifisches Modell, dem als Notation ein UML 1.5 Profil zur Modellierung automatisierbarer Prozesse zugrunde liegt, realisiert werden. Aus einer solchen plattformspezifischen Beschreibung könnte leicht mit der in [IAF+04] beschriebenen Abbildung ausführbarer BPEL4WS-Code erzeugt werden. Darüber hinaus sind natürlich auch Abbildungen eines PIM zu PSMs anderer Technologien wie Agentenplattformen oder peer-to-peer Netzwerken denkbar.

Betrachtet man den eben skizzierten Zusammenhang einer Geschäftprozessautomatisierung im Rahmen der MDA, so können die in dieser Arbeit durch die Abbildung erzeugten Modelle eine Basis für eine maschinenlesbare Beschreibung von automatisierbaren Geschäftsprozessen bilden. Des Weiteren lassen sich die in der Motivation beschriebe-

nen kritischen Erfolgsfaktoren durch die Ergebnisse dieser Arbeit verbessern. So kann die Entwicklungszeit von IuK-Systemen verkürzt und deren Lebenszeit verlängert werden, indem auf Änderungen und Anpassungen von Geschäftsprozessen durch automatische Modellgenerierung schneller und flexibler reagiert wird. Eine einmal definierte und dann automatisch durchgeführte Modelltransformation stellt darüber hinaus eine durchgängige und konsistente Prozessmodellierung von den Fachabteilungen hin zu einem IuK-System sicher, indem manuelle Abbildungsfehler und Abbildungsabweichungen vermieden werden. Die Qualität von IuK-Systemen wird erhöht. Schließlich müssen sich Systementwickler bei dem Design und der Architektur von IuK-Systemen nicht mehr mit der Komplexität von Geschäftprozessen auseinandersetzen, da deren Struktur und Abläufe automatisch aus business-level Modellen generiert werden. Die Entwicklungszeiten und -kosten von IuK-Systemen können gesenkt werden.

Wurde in dieser Arbeit eine Abbildung zwischen ARIS und BPDM definiert und deren Einsatz an einer Fallstudie gezeigt, so ist die Abbildung noch nicht in einer Implementierung umgesetzt. Eine Implementierung der Abbildung könnte durch den Einsatz einer Transformationssprache wie XSLT[23] realisiert werden. Dabei würde zunächst ein in ARIS modelliertes Modell in eine XML-Repräsentation[24] dieses Modells exportiert. Anschließend kann die eigentliche Abbildung mit Hilfe von z.B. XSLT durchgeführt werden. Diese hat eine XML-Repräsentation des generierten BPDM-Modells zum Ergebnis. Die Struktur der XML-Repräsentation des BPDM-Modells ist durch ein XML-Schema[25] festgelegt, das mit der XML Metadata Interchange (XMI)[26] aus dem UML Metamodel generiert wurde. Diese einheitliche Repräsentation von UML in XML-Dokumenten ermöglicht es beliebigen Modellierungswerkzeugen, die den Import von UML unterstützen, durch die Abbildung generierte Modelle zu importieren und darzustellen.

Schließlich bieten sich auch für die Spezifikation der Abbildung selbst Erweiterungsmöglichkeiten. Zum einen könnte in die Abbildung auch das Mapping von unternehmensübergreifenden Prozessen integriert werden. Sind zu der Beschreibung von unternehmensübergreifenden Prozessen in BPDM kollaborative Prozesse vorgesehen, so müsste man sich vorher bei der Prozessmodellierung mit ARIS erst auf eine Beschreibung von unternehmensübergreifenden Geschäftsprozessen einigen. Hier wäre es z.B. auch denkbar, die ARIS-Modelle um eine nicht im ARIS Metamodell vorgesehene Notation zu erweitern. Zum anderen konzentriert sich die in dieser Arbeit entworfene Abbildung auf die Prozesse eines Unternehmens, d.h. vor allem auf die Funktions- und Steuerungssicht von ARIS. Daher wäre es sicherlich wünschenswert, die Abbildung um Aspekte zur Beschreibung der Organisations- und Datenstrukturen von Unternehmen zu erweitern.

---

[23] Die eXtensible Stylesheet Language Transformations (XSLT) ist eine XML-Sprache zur Übersetzung von beliebigen XML Dokumenten. Häufigste Anwendung dürfte die Übersetzung von XML Dokumenten in andere XML Formate sein.

[24] Die eXtensible Markup Language (XML) ist eine Auszeichnungsprache für Dokumente die strukturierte Infromationen enthalten. Sie wird vom World Wide Web Consortium gepflegt. (siehe http://www.w3.org/xml)

[25] Die W3C XML Schema Definition Language (XML-Schema) ist eine XML Sprache zur Beschreibung und Einschränkung des Inhalts von XML Dokumenten. Ein XML Dokument ist *wohlgeformt*, wenn es lediglich die XML Syntax einhält. Genügt ein XML Dokument noch einer vorgegebenen XML-Schema Definition, so ist es *gültig*.

[26] Die XML Metadata Interchange (XMI) definiert Prinzipien zur Abbildung von Modellen, deren Metamodelle der MOF genügen, auf XML Strukturen. Die Spezifikation 2.0 beschreibt die Generierung von XML-Schemas aus Metamodellen für Modellierungssprachen. (siehe http://www.omg.org)

Literaturverzeichnis

BHK04 BORN, Marc, Eckhardt HOLZ und Olaf KATH: *Softwareentwicklung in UML 2*, Addison-Wesley, 2004.

Dav01 DAVIS, Rob: Business process modelling with ARIS, Springer, 2001.

Dat04 DATZ, Todd: *What You Kneed to Know About Service-Oriented Architecture*, CIO Magazine (http://www.cio.com), 15th January 2004.

FGJ04 FRANK, Joachim, Tracy GARDNER und Simon JOHNSTON: *Business Process Definition Metamodel – Concepts and Overview*, IBM, 2004.

Fra03 FRANKEL, David S.: Model Driven Architecture. Appling MDA to Enterprise Computing, OMG Press, 2003.

IAF+04 Iyengar, Amsden, Frank, Gardner, Johnston, White, Rivett, Cobb, Goland, Frank, Miller, Fischer, Casanave, Cummins, Malhorta und Koethe: *Business Process Definition Metamodel v1.0.2*, IBM, Adaptive, BEA, Borland, BPMI, DAT, EDS, Unisys, 88 Solutions, 2004.

Ibm03 IBM: *Business Process Execution Language for Web Services Version 1.1*, http://www.ibm.com/developerworks/library/ws-bpel/, 2003.

Ids03 IDS SCHEER: ARIS Methode, ARIS 6 Collaborative Suite, IDS Scheer, 2003.

JRH+04 JECKLE, Mario, Chris RUPP, Jürgen HAHN, Barbara ZENGLER und Stefan QUEINS: *UML 2 glasklar*, Hanser, 2004.

KWB03 KLEPPE, Anneke, Jos WARMER und Wim BAST: MDA explained. The Model Driven Architecture: Practice and promise, Addison-Wesley, 2003.

Pel03 PELTZ, Chris: *Web Service Orchestration and Choreography*, Web Services Journal Volume3 Issue 7, 2003.

Ren03 RENGER, Susanne: ARIS Konventionen zur Prozessmodellierung Version 2.2, Siemens, 2003.

Ren04 RENGER, Susanne: Modellierungs-Handbuch Version 1.0, Siemens, 2004.

Sch98a SCHEER, August-Wilhelm: ARIS – Modellierungsmethoden, Metamodelle, Anwendungen, Springer, 1998.

Sch98b SCHEER, August-Wilhelm: ARIS – vom Geschäftsprozess zum Anwendungssystem, Springer, 1998.

Sei02 SEIDLMEIER, Heinrich: *Prozessmodellierung mit ARIS*, Vieweg, 2002.

Sie03 SIEMENS: Levelkonzept zum Siemens Referenz Prozess Haus Version 1.0, Siemens, 2003.

WOe04 WEILKIENS, Tim und OESTEREICH, Bernd: *UML 2 Zertifizierung*, dpunkt.verlag, 2004.

Whi04 WHITE, Stephen: Process Modeling Notations and Workflow Patterns, IBM, 2004.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ARIS	Architektur integrierter Informationssysteme
BPDM	Business Process Definition Metamodel
BPEL4WS	Business Process Execution Language for Web Services
BPML	Business Process Modeling Language
BPMN	Business Process Modeling Notation
CIM	Computation independent Model
ebXML	Electronic Business using eXtensible Markup Language
EDOC	Enterprise Distributed Object Computing
EPK	Ereignisgesteuerte Prozesskette
FZD	Funktionszuordnungsdiagramm
HOBE	House of Business Engineering
J2EE	Java 2 Platform Enterprise Edition
MDA	Model driven Architecture
MOF	Meta Object Facility
OMG	Object Management Group
PIM	Platform independent Model
PSM	Platform specific Model
RPH	Referenz Prozesshaus
SOA	Service oriented Architecture
UML	Unified Modeling Language
VKD	Vorgangskettendiagramm
WKD	Wertschöpfungskettendiagramm
XMI	XML Metadata Interchange
XML	EXtensible Markup Language
XSLT	EXtensible Stylesheet Language Transformations

Anhang A

In diesem Anhang finden sich die Transformationsregeln zur Abbildung von ARIS-Modellen in BPDM-Modelle auf CIM-Ebene. Die Regeln haben dieselben Bezeichnungen und Nummerierung wie in Kapitel 4 dieser Arbeit. Die Regeln sind in einer *‚wenn-dann'*-Form formuliert: Gilt die unter *‚wenn'* formulierte Bedingung, so werden die unter *‚dann'* angegebenen Aktionen ausgeführt.

Transformationsregeln Verhaltensdiagramme

Regel CIM-9:

[Regel_CIM_Verhaltensdiagramm1]
wenn Prozess ist durch eine EPK hinterlegt
dann wende Regel **[Regel_CIM_SD_Sequenzdiagramm1]** an
und wende Regel **[Regel_CIM_AD_Aktivitätsdiagramm]** an
und wende Regel **[Regel_CIM_IOD_Interaktionsübersichtsdiagramm]** an

Regel CIM-10:

[Regel_CIM_Verhaltensdiagramm2]
wenn Prozess ist nicht durch eine EPK hinterlegt
und Prozess ist in einer oder mehreren EPKs von übergeordneten Prozessen als Funktion modelliert
dann wende Regel **[Regel_CIM_SD_Sequenzdiagramm2]** an
und wende Regel **[Regel_CIM_IOD_Interaktionsübersichtsdiagramm]** an

Transformationsregeln Sequenzdiagramme

Regel CIM-11:

[Regel_CIM_SD_Sequenzdiagramm1]
wenn Prozess ist durch eine EPK hinterlegt
dann generiere für jede in der EPK enthaltene Prozessschnittstelle ein Sequenzdiagramm (Name: *‚<AufrufenderProzess>>-<<AufgerufenerProzess>'*)
und füge jedem dieser Sequenzdiagramme eine Lebenslinie für den durch die EPK beschriebenen Prozess hinzu (Name: *‚<NameBeschriebenerProzess>'*)
und füge jedem dieser Sequenzdiagramme eine Lebenslinie für den durch die Prozessschnittstelle referenzierten Prozess hinzu (Name: *‚<NameReferenzierterProzess>'*)
und wende Regel **[Regel_CIM_SD_ChorNachricht]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchAsynNachricht1]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchAsynNachricht2]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchSynNachricht1]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchSynNachricht2]** auf alle Sequenzdiagramme an

Regel CIM-12:

[Regel_CIM_SD_ChorNachricht]

wenn die durch das Sequenzdiagramm beschriebene Prozessschnittstelle modelliert ein Choreographyverhältnis
dann erzeuge im Sequenzdiagramm eine asynchrone Nachricht vom sendenden zum empfangenden Prozess
(Name: *‚<NameEmpfangenderProzess>'*)

Regel CIM-13:

[Regel_CIM_SD_OrchAsynNachricht1]
wenn die durch das Sequenzdiagramm beschriebene Prozessschnittstelle modelliert ein Orchestrationverhältnis
und die Prozessschnittstelle repräsentiert einen Aufruf des untergeordneten, durch die EPK beschriebenen, Prozesses
und es liegt eine asynchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine asynchrone Nachricht vom übergeordneten, durch die Prozessschnittstelle referenzierten, Prozess zum untergeordneten, aufgerufenen Prozess
(Name: *‚<NameUntergeordneterProzess>'*)

Regel CIM-14:

[Regel_CIM_SD_OrchAsynNachricht2]
wenn die durch das Sequenzdiagramm beschriebene Prozessschnittstelle modelliert ein Orchestrationverhältnis
und die Prozessschnittstelle repräsentiert die Callback-Nachricht des untergeordneten zum übergeordneten Prozess
und es liegt eine asynchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine asynchrone Nachricht vom untergeordneten Prozess zum übergeordneten, durch die Prozessschnittstelle referenzierten Prozess
(Name: *‚<NameUntergeordneterProzess>_CB'*)

Regel CIM-15:

[Regel_CIM_SD_OrchSynNachricht1]
wenn die durch das Sequenzdiagramm beschriebene Prozessschnittstelle modelliert ein Orchestrationverhältnis
und die Prozessschnittstelle repräsentiert einen Aufruf des untergeordneten, durch die EPK beschriebenen, Prozesses
und es liegt eine synchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine synchrone Nachricht vom übergeordneten, durch die Prozessschnittstelle referenzierten, Prozess zum untergeordneten, aufgerufenen Prozess
(Name: *‚<NameUntergeordneterProzess>'*)

Regel CIM-16:

[Regel_CIM_SD_OrchSynNachricht2]
wenn die durch das Sequenzdiagramm beschriebene Prozessschnittstelle modelliert ein Orchestrationverhältnis
und die Prozessschnittstelle repräsentiert die Ergebnisparameterübergabe vom untergeordneten zum übergeordneten Prozess
und es liegt eine synchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine synchrone Nachricht vom untergeordneten Prozess zum übergeordneten, durch die Prozessschnittstelle referenzierten, Prozess
(Name: *ERHÄLT KEINE BEZEICHNUNG*)

Regel CIM-17:

[Regel_CIM_SD_Sequenzdiagramm2]
wenn Prozess ist durch keine EPK hinterlegt
und Prozess ist in einer oder mehreren EPKs von übergeordneten Prozessen als Funktion modelliert
dann generiere ein Sequenzdiagramm zur Modellierung der bei dem, durch die Funktion beschriebenen, Prozess eingehenden Nachricht
(Name: *‚<ÜbergeordneterProzess>-<UntergeordneterProzess>'*)
und generiere ein Sequenzdiagramm zur Modellierung der, durch den untergeordneten Prozess verschickten, Antwortnachricht
(Name: *‚<UntergeordneterProzess>-<ÜbergeordneterProzess>'*)

und füge jedem dieser Sequenzdiagramme eine Lebenslinie für den untergeordneten, durch die Funktion beschriebenen, Prozess hinzu (Name: *‚<NameUntergeordneterProzess>'*)
und füge jedem dieser Sequenzdiagramme eine Lebenslinie für den übergeordneten Prozess hinzu (Name: *‚<NameÜbergeordneterProzess>'*)
und wende Regel **[Regel_CIM_SD_OrchAsynNachricht3]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchAsynNachricht4]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchSynNachricht3]** auf alle Sequenzdiagramme an
und wende Regel **[Regel_CIM_SD_OrchSynNachricht4]** auf alle Sequenzdiagramme an

Regel CIM-18:

[Regel_CIM_SD_OrchAsynNachricht3]
wenn das Sequenzdiagramm beschreibt den Aufruf eines untergeordneten Prozesses
und es liegt eine asynchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine asynchrone Nachricht vom übergeordneten Prozess zum untergeordneten, aufgerufenen Prozess (Name: *‚<NameUntergeordneterProzess>'*)

Regel CIM-19:

[Regel_CIM_SD_OrchAsynNachricht4]
wenn das Sequenzdiagramm beschreibt die vom untergeordneten Prozess verschickte Antwortnachricht
und es liegt eine asynchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine asynchrone Nachricht vom untergeordneten Prozess zum übergeordneten Prozess (Name: *‚<NameUntergeordneterProzess>_CB'*)

Regel CIM-20:

[Regel_CIM_SD_OrchSynNachricht3]
wenn das Sequenzdiagramm beschreibt einen Aufruf eines untergeordneten Prozesses
und es liegt eine synchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine synchrone Nachricht vom übergeordneten Prozess zum untergeordneten, aufgerufenen Prozess (Name: *‚<NameUntergeordneterProzess>'*)

Regel CIM-21:

[Regel_CIM_SD_OrchSynNachricht4]
wenn das Sequenzdiagramm beschreibt die Ergebnisparameterübergabe vom untergeordneten zum übergeordneten Prozess
und es liegt eine synchrone Kommunikation vor
dann erzeuge im Sequenzdiagramm eine synchrone Nachricht vom untergeordneten Prozess zum übergeordneten Prozess (Name: *ERHÄLT KEINE BEZEICHNUNG*)

Transformationsregeln Aktivitätsdiagramme

Regel CIM-22:

[Regel_CIM_AD_Aktivitätsdiagramm]
wenn TRUE
dann erzeuge Aktivitätsdiagramm (Name: *‚<NameBeschriebenerProzess>-ExecutableProcess'*)
und füge dem Aktivitätsdiagramm eine Aktivität mit dem Namen des durch das Aktivitätsdiagramm beschriebenen Prozesses hinzu (alle weitern Element werden innerhalb dieser Aktivität modelliert)

und wende Regel **[Regel_CIM_AD_OrchInput]** auf jede Prozessschnittstelle der EPK an
und wende Regel **[Regel_CIM_AD_OrchOutput]** auf jede Prozessschnittstelle der EPK an
und wende Regel **[Regel_CIM_AD_ChorAktion1]** auf jede Prozessschnittstelle der EPK an
und wende Regel **[Regel_CIM_AD_ChorAktion2]** auf jede Prozessschnittstelle der EPK an
und wende Regel **[Regel_CIM_AD_Aktion]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_ODERAktion1]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_ODERAktion2]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_UNDAktion1]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_UNDAktion2]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss1]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss2]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss3]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss4]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss5]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss6]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss7]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss8]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss9]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss10]** auf jede Prozessschnitt-stelle der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss11]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss12]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss13]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss14]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss15]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss16]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss17]** auf jede Funktion der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss18]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss19]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss20]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss21]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss22]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss23]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss24]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss25]** auf jedes Kontrollelement der EPK an

und wende Regel **[Regel_CIM_AD_Kontrollfluss26]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss27]** auf jedes Kontrollelement der EPK an
und wende Regel **[Regel_CIM_AD_Kontrollfluss28]** auf jedes Kontrollelement der EPK an

Regel CIM-23:

[Regel_CIM_AD_ChorAktion1]
wenn durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Prozessschnittstelle repräsentiert einen Aufruf des in dem Aktivitätsdiagramm beschriebenen Prozess
dann füge Aktion mit dem Stereotypen *«receive»* dem Aktivitätsdiagramm hinzu
(Name: *‚<NameAufrufenderProzess>'*)
und füge der Aktion einen Outputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG***)**

Regel CIM-24:

[Regel_CIM_AD_ChorAktion2]
wenn durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Prozessschnittstelle repräsentiert den Aufruf eines Prozesses, der dem in dem Aktivitätsdiagramm beschriebenen Prozess nachfolgt
dann füge Aktion mit dem Stereotypen *«send»* dem Aktivitätsdiagramm hinzu
(Name: *‚<NameAufgerufenerProzess>'*)
und füge der Aktion einen Inputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG***)**

Regel CIM-25:

[Regel_CIM_AD_OrchInput]
wenn durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Prozessschnittstelle repräsentiert einen Aufruf des in dem Aktivitätsdiagramm beschriebenen Prozess
dann erzeuge einen Inputparameter an der zum dem durch die EPK beschriebenen Prozess
(Name: *ERHÄLT KEINE BEZEICHNUNG*)

Regel CIM-26:

[Regel_CIM_AD_OrchOutput]
wenn durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Prozessschnittstelle repräsentiert die Antwortnachricht des in dem Aktivitätsdiagramm beschriebenen Prozesses an den übergeordneten Prozess
dann erzeuge einen Outputparameter an der zum dem durch die EPK beschriebenen Prozess
(Name: *ERHÄLT KEINE BEZEICHNUNG*)

Regel CIM-27:

[Regel_CIM_AD_Aktion]
wenn Funktion ist in einer EPK modelliert
dann füge eine Aktion dem Aktivitätsdiagramm hinzu
(Name: *‚<Funktionsname>'*)
und füge der Aktion einen Inputpin hinzu
(Name: ERHÄLT KEINE BEZEICHUNG**)**
und füge der Aktion einen Outputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG***)**

Regel CIM-28:

[Regel_CIM_AD_Kontrollfluss1]
wenn ein Ereignis verfügt über keine eingehende Kante

und die von dem Ereignis ausgehende Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm einen Startknoten hinzu
und füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: der gerade hinzugefügte Startknoten - Ziel: Eingangspin der zu der Funktion gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-29:

[Regel_CIM_AD_Kontrollfluss2]
wenn ein Ereignis verfügt über keine eingehende Kante
und die von dem Ereignis ausgehende Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm einen Startknoten hinzu
und füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: der gerade hinzugefügte Startknoten - Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-30:

[Regel_CIM_AD_Kontrollfluss3]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Prozessschnittstelle gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)

Regel CIM-31:

[Regel_CIM_AD_Kontrollfluss4]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Prozessschnittstelle gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-32:

[Regel_CIM_AD_Kontrollfluss5]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Prozessschnittstelle gehörenden Aktion – Ziel: Eingangspin der zu des Funktion gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-33:

[Regel_CIM_AD_Kontrollfluss6]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert

und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Ereignis der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-34:

[Regel_CIM_AD_Kontrollfluss7]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
und die Kante mündet in einer Funktion der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-35:

[Regel_CIM_AD_Kontrollfluss8]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Eingangsparameter der im Aktivitätsdiagramm modellierten Aktivität – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)

Regel CIM-36:

[Regel_CIM_AD_Kontrollfluss9]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Eingangsparameter der im Aktivitätsdiagramm modellierten Aktivität – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-37:

[Regel_CIM_AD_Kontrollfluss10]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Eingangsparameter der im Aktivitätsdiagramm modellierten Aktivität – Ziel: Eingangspin der zu des Funktion gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-38:

[Regel_CIM_AD_Kontrollfluss11]
wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Ereignis der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-39:

[Regel_CIM_AD_Kontrollfluss12]

wenn eine Prozessschnittstelle verfügt über eine ausgehende Kante
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
und die Kante mündet in einer Funktion der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-40:

[Regel_CIM_AD_Kontrollfluss13]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: Eingangspin der zu des Funktion gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-41:

[Regel_CIM_AD_Kontrollfluss14]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-42:

[Regel_CIM_AD_Kontrollfluss15]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: Eingangspin der zu der Prozessschnittstelle gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-43:

[Regel_CIM_AD_Kontrollfluss16]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: Ausgangsparameter der im Aktivitätsdiagramm modellierten Aktivität)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-44:

[Regel_CIM_AD_Kontrollfluss17]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und das Ereignis hat keine ausgehende Kante
dann füge dem Aktivitätsdiagramm einen Endknoten hinzu

und füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: der eben hinzugefügte Entknoten)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-45:

[Regel_CIM_AD_Kontrollfluss18]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu der Funktion gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)

Regel CIM-46:

[Regel_CIM_AD_Kontrollfluss19]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einer Funktion der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-47:

[Regel_CIM_AD_Kontrollfluss20]
wenn eine Funktion verfügt über eine ausgehende Kante
und die Kante mündet in einer Prozessschnittstelle der EPK
dann *FEHLER: Darf bei korrekt modellierter EPK nicht auftreten!*

Regel CIM-48:

[Regel_CIM_AD_Kontrollfluss21]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu der Funktion gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-49:

[Regel_CIM_AD_Kontrollfluss22]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-50:

[Regel_CIM_AD_Kontrollfluss23]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu der Prozessschnittstelle gehörenden Aktion)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-51:

[Regel_CIM_AD_Kontrollfluss24]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einem Ereignis der EPK
und die von dem Ereignis ausgehende Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Ausgangsparameter der im Aktivitätsdiagramm modellierten Aktivität)
und wende Regel **[Regel_CIM_AD_Bedingung]** auf die neu erzeugt Kante und das in der Bedingung dieser Regel beschriebene Ereignis an

Regel CIM-52:

[Regel_CIM_AD_Kontrollfluss25]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einer Funktion der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu der Funktion gehörenden Aktion)

Regel CIM-53:

[Regel_CIM_AD_Kontrollfluss26]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einem Kontrollelement der EPK
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu dem Kontrollelement gehörenden Aktion)

Regel CIM-54:

[Regel_CIM_AD_Kontrollfluss27]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Choreographyverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Eingangspin der zu der Prozessschnittstelle gehörenden Aktion)

Regel CIM-55:

[Regel_CIM_AD_Kontrollfluss28]
wenn ein Kontrollelement verfügt über eine ausgehende Kante
und die Kante mündet in einer Prozessschnittstelle der EPK
und durch die Prozessschnittstelle wird ein Orchestrationverhältnis modelliert
dann füge dem Aktivitätsdiagramm eine Kontrollflusskante hinzu (Quelle: Ausgangspin der zu dem Kontrollelement gehörenden Aktion – Ziel: Ausgangsparameter der im Aktivitätsdiagramm modellierten Aktivität)

Regel CIM-56:

[Regel_CIM_AD_Parametersatz1]
wenn Inputpins einer Aktion sollen gruppiert werden
dann ‚gruppiere' jeden Inputpin der Aktion in einem separaten Parametersatz

Regel CIM-57:

[Regel_CIM_AD_Parametersatz2]
wenn Outputpins einer Aktion sollen gruppiert werden
dann ‚gruppiere' jeden Outputpin der Aktion in einem separaten Parametersatz

Regel CIM-58:

[Regel_CIM_AD_ODERAktion1]
wenn in einer EPK ist ein exklusives ODER als Verzweigungsknoten modelliert
dann füge eine Aktion dem Aktivitätsdiagramm hinzu
(Name: *‚<ODER>'*)
und füge der Aktion einen Inputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG*)
und füge der Aktion Outputpins für jede am Verzeigungsknoten ausgehende Kante hinzu
(Namen: *ERHALTEN KEINE BEZEICHUNG*)
und wende die Regel **[Regel_CIM_AD_Parametersatz2]** auf die Outputpins an

Regel CIM-59:

[Regel_CIM_AD_ODERAktion2]
wenn in einer EPK ist ein exklusives ODER als Verbindungsknoten modelliert
dann füge eine Aktion dem Aktivitätsdiagramm hinzu
(Name: *‚<ODER>'*)
und füge der Aktion einen Outputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG*)
und füge der Aktion Inputpins für jede am Verbindungsknoten eingehende Kante hinzu
(Namen: *ERHALTEN KEINE BEZEICHUNG*)
und wende die Regel **[Regel_CIM_AD_Parametersatz1]** auf die Inputpins an

Regel CIM-60:

[Regel_CIM_AD_UNDAktion1]
wenn in einer EPK ist ein UND als Parallelisierungsknoten modelliert
dann füge eine Aktion dem Aktivitätsdiagramm hinzu
(Name: *‚<UND>'*)
und füge der Aktion einen Inputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG*)
und füge der Aktion Outputpins für jede am Parallelisierungsknoten ausgehende Kante hinzu
(Namen: *ERHALTEN KEINE BEZEICHUNG*)

Regel CIM-61:

[Regel_CIM_AD_UNDAktion2]
wenn in einer EPK ist ein UND als Synchronisationsknoten modelliert
dann füge eine Aktion dem Aktivitätsdiagramm hinzu
(Name: *‚<UND>'*)
und füge der Aktion einen Outputpin hinzu
(Name: *ERHÄLT KEINE BEZEICHUNG*)
und füge der Aktion Inputpins für jede am Synchronisationsknoten eingehende Kante hinzu
(Namen: *ERHALTEN KEINE BEZEICHUNG*)

Regel CIM-62:

[Regel_CIM_AD_Bedingung]
wenn eine Kontrollflusskante ist in einem Aktivitätsdiagramm modelliert
und die Kanten der EPK, aus denen diese Kante im Aktivitätsdiagramm generiert wurde, sind Eingangs- und/oder Ausgangskante eines Ereignisses
dann füge der Kante im Aktivitätsdiagramm eine Bedingung hinzu
(Bezeichnung: *‚<Ereignisname>'*)

Transformationsregeln Interaktionsübersichtsdiagramme

Regel CIM-63:

[Regel_CIM_IOD_Interaktionsübersichtsdiagramm]
wenn TRUE

dann erzeuge Interaktionsübersichtsdiagramm
(Name: *‚<NameBeschriebenerProzess>-AbstractProcess'*)
und füge alle dem beschriebenen Prozess zugeordneten Sequenzdiagramme als Interaktionreferenz hinzu
und füge eine Aktion mit dem Namen des beschriebenen Prozess hinzu
und wende Regel **[Regel_CIM_IOD_EingehendeNachricht]** auf alle Interaktionsreferenzen an
und wende Regel **[Regel_CIM_IOD_AusgehendeNachricht]** auf alle Interaktionsreferenzen an
und wende Regel **[Regel_CIM_IOD_OhneNachricht1]** an
und wende Regel **[Regel_CIM_IOD_OhneNachricht2]** an

Regel CIM-64:

[Regel_CIM_IOD_EigehendeNachricht]
wenn das durch die Interaktionsreferenz referenzierte Sequenzdiagramm modelliert eine für den beschriebenen Prozess eingehende Nachricht
dann füge dem Interaktionsübersichtsdiagramm eine Kontrollflusskante von der Interaktionsreferenz zu der Aktion der Interaktionsübersichtsdiagramms hinzu
und füge Startknoten dem Interaktionsübersichtsdiagramm hinzu
und füge Kontrollflusskante von dem gerade hinzugefügten Startknoten zu der Interaktionsreferenz hinzu

Regel CIM-65:

[Regel_CIM_IOD_AusgehendeNachricht]
wenn das durch die Interaktionsreferenz referenzierte Sequenzdiagramm modelliert eine für den beschriebenen Prozess ausgehende Nachricht
dann füge dem Interaktionsübersichtsdiagramm eine Kontrollflusskante von der Aktion der Interaktionsübersichtsdiagramms zu der Interaktionsreferenz hinzu
und füge Endknoten dem Interaktionsübersichtsdiagramm hinzu
und füge Kontrollflusskante von der Interaktionsreferenz zu dem gerade hinzugefügten Endknoten hinzu

Regel CIM-66:

[Regel_CIM_OID_OhneNachricht1]
wenn Interaktionsübersichtsdiagramm enthält keine Interaktionsreferenz dessen Sequenzdiagramm eine eingehende Nachricht modelliert
dann füge Startknoten dem Interaktionsübersichtsdiagramm hinzu
und füge Kontrollflusskante von dem gerade hinzugefügten Startknoten zu der Aktion des Interaktionsübersichtgramms hinzu

Regel CIM-67:

[Regel_CIM_OID_OhneNachricht2]
wenn Interaktionsübersichtsdiagramm enthält keine Interaktionsreferenz dessen Sequenzdiagramm eine ausgehende Nachricht modelliert
dann füge Endkonten dem Interaktionsübersichtsdiagramm hinzu
und füge Kontrollflusskante von der Aktion des Interaktionsübersichtgramms zu dem gerade hinzugefügten Endknoten hinzu

Anhang B

In diesem Anhang finden sich die Transformationsregeln zur Generierung eines PIMs aus einem CIM. Die Regeln haben dieselben Bezeichnungen und Nummerierung wie in Kapitel 4 dieser Arbeit. Die Regeln sind in einer *‚wenn-dann'*-Form formuliert: Gilt die unter *‚wenn'* formulierte Bedingung, so werden die unter *‚dann'* angegebenen Aktionen ausgeführt.

Transformationsregeln

Regel PIM-1:

[Regel_PIM_Identiät]
wenn TRUE
dann übernimm die Diagramme und Konzepte, auf die die Regel angewandt wird, aus dem CIM-Modell in das PIM-Modell

Regel PIM-2:

[Regel_PIM_CD_Prozesse]
wenn in einem Klassendiagramm auf CIM-Ebene sind Prozesse mit Beziehungen wie Prozesshierarchie und Prozessanordnungen modelliert
dann wende Regel **[Regel_PIM_Identität]** auf das Klassendiagramm an
und wende Regel **[Regel_PIM_CD_Port1]** an
und wende Regel **[Regel_PIM_CD_Port2]** an
und wende Regel **[Regel_PIM_CD_Port3]** an
und wende Regel **[Regel_PIM_CD_Port4]** an

Regel PIM-3:

[Regel_PIM_CD_Port1]
wenn das interne Verhalten eines Prozesses ist in einem Aktivitätsdiagramm modelliert
und das Aktivitätsdiagramm enthält Aktionen mit dem Stereotyp *«receive»*
dann füge der *«process»* Klasse, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem Partnerprozess hinzu (Name: *‚<Prozessname>_<Partnerprozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleChor1]** auf den Port und die Aktion mit dem Stereotyp *«receive»* an

Regel PIM-4:

[Regel_PIM_CD_Port2]
wenn das interne Verhalten eines Prozesses ist in einem Aktivitätsdiagramm modelliert
und das Aktivitätsdiagramm enthält Aktionen mit dem Stereotyp *«send»*
dann füge der *«process»* Klasse, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem Partnerprozess hinzu (Name: *‚<Prozessname>_<Partnerprozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleChor2]** auf den Port und die Aktion mit dem Stereotyp *«send»* an

Regel PIM-5:

[Regel_PIM_CD_Port3]
wenn das interne Verhalten eines Prozesses ist in einem Aktivitätsdiagramm modelliert
und das Aktivitätsdiagramm enthält Aktionen die Subprozesse referenzieren (und zu denen somit ein Orchestrationverhältnis besteht)
und es liegt eine asynchrone Kommunikation vor

dann füge der *«process»* Klasse, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem Subprozess hinzu
(Name: *‚<Prozessname>_<Subprozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleOrch1]** auf den Port des Prozesses und die beschriebene Aktion an
und füge der *«process»* Klasse des Subprozesses, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem im Aktivitätsdiagramm beschriebenen Prozess hinzu
(Name: *‚<Subprozessname>_<Prozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleOrch2]** auf den Port des Subprozesses und die beschriebene Aktion an

Regel PIM-6:

[Regel_PIM_CD_Port4]
wenn das interne Verhalten eines Prozesses ist in einem Aktivitätsdiagramm modelliert
und das Aktivitätsdiagramm enthält Aktionen die Subprozesse referenzieren (und zu denen somit ein Orchestrationverhältnis besteht)
und es liegt eine synchrone Kommunikation vor
dann füge der *«process»* Klasse, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem Subprozess hinzu
(Name: *‚<Prozessname>_<Subprozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleOrch3]** auf den Port des Prozesses und die beschriebene Aktion an
und füge der *«process»* Klasse des Subprozesses, falls nicht schon vorhanden, einen Port für die Kommunikation mit dem im Aktivitätsdiagramm beschriebenen Prozess hinzu
(Name: *‚<Subprozessname>_<Prozessname>'*)
und wende Regel **[Regel_PIM_CD_SchnittstelleOrch4]** auf den Port des Subprozesses und die beschriebene Aktion an

Regel PIM-7:

[Regel_PIM_CD_SchnittstelleOrch1]
wenn es liegt der Port eines Prozesses, der aus einer Aktion, die eine Subprozessaufruf darstellt, generiert wurde, vor
und es liegt eine asynchrone Kommunikation vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_ORCH*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *required Interface* hinzu
und erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_CB*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *provided Interface* hinzu

Regel PIM-8:

[Regel_PIM_CD_SchnittstelleOrch2]
wenn es liegt der Port eines Subprozesses, der aus einer Aktion, die eine Subprozessaufruf darstellt, generiert wurde, vor
und es liegt eine asynchrone Kommunikation vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_ORCH*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *provided Interface* hinzu
und erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_CB*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *required Interface* hinzu

Regel PIM-9:

[Regel_PIM_CD_SchnittstelleOrch3]

wenn es liegt der Port eines Prozesses, der aus einer Aktion, die eine Subprozessaufruf darstellt, generiert wurde, vor
und es liegt eine synchrone Kommunikation vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_ORCH*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *required Interface* hinzu

Regel PIM-10:

[Regel_PIM_CD_SchnittstelleOrch4]
wenn es liegt der Port eines Subprozesses, der aus einer Aktion, die eine Subprozessaufruf darstellt, generiert wurde, vor
und es liegt eine synchrone Kommunikation vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<SubprozessName>_ORCH*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *provided Interface* hinzu

Regel PIM-11:

[Regel_PIM_CD_SchnittstelleChor1]
wenn es liegt der Port eines Prozesses, der aus einer Aktion mit dem Stereotyp *«receive»* generiert wurde, vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<NameNachfolgerprozess>_<NameVorgängerprozess>_CHOR‘*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *provided Interface* hinzu

Regel PIM-12:

[Regel_PIM_CD_SchnittstelleChor2]
wenn es liegt der Port eines Prozesses, der aus einer Aktion mit dem Stereotyp *«send»* generiert wurde, vor
dann erzeuge, falls nicht schon vorhanden, eine Klasse mit dem Stereotyp *«interface»*
(Name: *‚<NameNachfolgerprozess>_<NameVorgängerprozess>_CHOR‘*)
und füge dem Port der *«process»* Klasse, falls nicht schon vorhanden, die eben generierte Schnittstelle als *required Interface* hinzu

Regel PIM-13:

[Regel_PIM_CD_Organisation]
wenn in einem Klassendiagramm auf CIM-Ebene sind Organsationen bzw. Personentypen als Rollen definiert
dann wende Regel **[Regel_PIM_Identität]** auf das Klassendiagramm bzw. die Klassen mit dem Stereotyp *«role»* an

Regel PIM-14:

[Regel_PIM_CD_Daten]
wenn in einem Klassendiagramm auf CIM-Ebene sind Daten(typen) als Klassen mit dem Stereotyp *«entity»* definiert
dann wende Regel **[Regel_PIM_Identität]** auf das Klassendiagramm bzw. die Klassen mit dem Stereotyp *«entity»* an
und wende Regel **[Regel_PIM_CD_BDocument1]** auf alle Klassen mit dem Stereotyp *«entity»* an
und wende Regel **[Regel_PIM_CD_BDocument2]** auf alle Klassen mit dem Stereotyp *«document»* an

Regel PIM-15:

[Regel_PIM_CD_BDocument1]
wenn eine Klasse mit dem Stereotyp *«entity»* ist in einem Klassendiagramm einem oder mehreren *«process»* Klassen als Input- oder Outputparameter zugeordnet

dann erzeuge für jede *«process»* Klasse, die der *«entity»* zugeordnet ist, eine Klasse mit dem Stereotyp *«document»* (Name: *‚<Entityname>_<Prozessname>'*)

Regel PIM-16:

[Regel_PIM_CD_BDocument2]
wenn es wurde eine Klasse mit dem Stereotyp *«document»* erzeugt
dann erzeuge eine gerichtete Abhängigkeitsbeziehung zwischen der *«document»* Klasse und der zugehörigen *«entity»* Klasse (Quelle: Klasse mit dem Stereotyp *«document»* – Ziel: zugehörige *«entity»* Klasse)

Regel PIM-17:

[Regel_PIM_Verhaltensdiagramme]
wenn es liegt ein Verhaltensdiagramm vor
dann wende Regel **[Regel_PIM_Identität]** auf das Verhaltensdiagramm an
und wende die Regeln **[Regel_PIM_AD_Datenfluss1-28]** auf das Verhaltendiagramm an

Regel PIM-18:

[Regel_PIM_AD_Datenfluss1]
wenn ein Aktivitätsdiagramm enthält Aktionen die Referenzen auf Subprozesse darstellen
und den Subprozessen sind auf CIM-Ebene *«entity»* Klassen als Inputparameter zugeordnet
dann erzeuge zu jeder solchen Aktion für alle *«entity»* Klassen, die Inputparameter für den zur Aktion gehörigen Prozess darstellen, Eingangspins mit dem Namen der zugehörigen *«document»* Klasse

Regel PIM-19:

[Regel_PIM_AD_Datenfluss2]
wenn ein Aktivitätsdiagramm enthält Aktionen die Referenzen auf Subprozesse darstellen
und den Subprozessen sind auf CIM-Ebene *«entity»* Klassen als Outputparameter zugeordnet
dann erzeuge zu jeder solchen Aktion für alle *«entity»* Klassen, die Outputparameter für den zur Aktion gehörigen Prozess darstellen, Ausgangspins mit dem Namen der zugehörigen *«document»* Klasse

Regel PIM-20:

[Regel_PIM_AD_Datenfluss3]
wenn ein Aktivitätsdiagramm enthält Aktionen mit dem Stereotyp *«receive»*
und dem im Aktivitätsdiagramm modellierten Prozessen sind auf CIM-Ebene *«entity»* Klassen als Inputparameter zugeordnet
dann erzeuge zu jeder solche Aktion mit dem Stereotyp *«receive»* einen Outputpin für jede *«entity»* die dem im Aktivitätsdiagramm modellierten Prozess als Inputparameter zugeordnet ist; der Pin trägt den Namen der zugehörigen *«document»* Klasse

Regel PIM-21:

[Regel_PIM_AD_Datenfluss4]
wenn ein Aktivitätsdiagramm enthält Aktionen mit dem Stereotyp *«send»*
und dem im Aktivitätsdiagramm modellierten Prozessen sind auf CIM-Ebene *«entity»* Klassen als Outputparameter zugeordnet
dann erzeuge zu jeder solche Aktion mit dem Stereotyp *«send»* einen Inputpin für jede *«entity»* die dem im Aktivitätsdiagramm modellierten Prozess als Outputparameter zugeordnet ist; der Pin trägt den Namen der zugehörigen *«document»* Klasse

Regel PIM-22:

[Regel_PIM_AD_Datenfluss5]
wenn die im Aktivitätsdiagramm modellierte Aktivität enthält Eingangsparameter für den Kontrollfluss (d.h. der Prozess steht mit einem übergeordneten Prozess in einem Orchestrationverhältnis)

und dem durch das Aktivitätsdiagramm modellierten Prozess sind auf CIM-Ebene *«entity»* Klassen als Inputparameter zugeordnet
dann erzeuge zu jeder solchen *«entity»* einen Eingangsparameter; die Bezeichnung der Eingangsparameter ist jeweils der Name der zur *«entity»* und zum Prozess gehörigen *«document»* Klasse

Regel PIM-23:

[Regel_PIM_AD_Datenfluss6]
wenn die im Aktivitätsdiagramm modellierte Aktivität enthält Ausgangsparameter für den Kontrollfluss (d.h. der Prozess steht mit einem übergeordneten Prozess in einem Orchestrationverhältnis)
und dem durch das Aktivitätsdiagramm modellierten Prozess sind auf CIM-Ebene *«entity»* Klassen als Outputparameter zugeordnet
dann erzeuge zu jeder solchen *«entity»* einen Ausgangsparameter; die Bezeichnung der Ausgangsparameter ist jeweils der Name der zur *«entity»* und zum Prozess gehörigen *«document»* Klasse

Regel PIM-24:

[Regel_PIM_AD_Datenfluss7]
wenn zwischen zwei Aktionen (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die beiden Aktionen verfügen über Pins deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören
und an der Aktion von der die Kontrollflusskante ausgeht werden nur Outputpins betrachtet
und an der Aktion an der die Kontrollflusskante eingeht werden nur Inputpins betrachtet
dann erzeuge eine Datenflusskante zwischen zusammengehörigen Pins der beiden Aktionen; die Richtung der Datenflusskante entspricht der der Kontrollflusskante

Regel PIM-25:

[Regel_PIM_AD_Datenfluss8]
wenn zwischen einem Eingangsparameter der im Aktivitätsdiagramm modellierten Aktivität und einer Aktion (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die Aktivität und die Aktion verfügen über Eingangsparameter bzw. Inputpins deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören
und an der Aktivität werden nur Eingangsparameter der Aktivität betrachtet
und an der Aktion an der die Kontrollflusskante eingeht werden nur Inputpins betrachtet
dann erzeuge jeweils eine Datenflusskante zwischen zusammengehörigen Eingangsparameter der Aktivität und Inputpins der Aktion; die Richtung der Datenflusskante entspricht der der Kontrollflusskante

Regel PIM-26:

[Regel_PIM_AD_Datenfluss9]
wenn zwischen einem Ausgangsparameter der im Aktivitätsdiagramm modellierten Aktivität und einer Aktion (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die Aktivität und die Aktion verfügen über Ausgangsparameter bzw. Outputpins deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören
und an der Aktivität werden nur Ausgangsparameter der Aktivität betrachtet
und an der Aktion an der die Kontrollflusskante ausgeht werden nur Outputpins betrachtet
dann erzeuge jeweils eine Datenflusskante zwischen zusammengehörigen Ausgangsparameter der Aktivität und Outputpins der Aktion; die Richtung der Datenflusskante entspricht der der Kontrollflusskante

Regel PIM-27:

[Regel_PIM_AD_Datenfluss10]

wenn zwischen zwei Aktionen (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die beiden Aktionen verfügen **nicht** über Pins deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören
und an der Aktion von der die Kontrollflusskante ausgeht werden nur Outputpins betrachtet
und an der Aktion an der die Kontrollflusskante eingeht werden nur Inputpins betrachtet
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion mit der ausgehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)
und füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin der Aktion mit der eingehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-28:

[Regel_PIM_AD_Datenfluss11]
wenn zwischen einem Eingangsparameter der im Aktivitätsdiagramm modellierten Aktivität und einer Aktion (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die Aktivität und die Aktion verfügen über Eingangsparameter bzw. Inputpins deren Objektknotentypen (*«document»*) **nicht** zu demselben *«entity»* gehören
und an der Aktivität werden nur Eingangsparameter der Aktivität betrachtet
und an der Aktion an der die Kontrollflusskante eingeht werden nur Inputpins betrachtet
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Eingangsparameter der Aktivität hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten von den Eingangsparametern zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)
und füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin der Aktion mit der eingehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-29:

[Regel_PIM_AD_Datenfluss12]
wenn zwischen einem Ausgangsparameter der im Aktivitätsdiagramm modellierten Aktivität und einer Aktion (mit oder ohne Stereotyp) ist auf CIM-Ebene eine Kontrollflusskanten modelliert
und die Aktivität und die Aktion verfügen über Ausgangsparameter bzw. Outputpins deren Objektknotentypen (*«document»*) **nicht** zu demselben *«entity»* gehören
und an der Aktivität werden nur Ausgangsparameter der Aktivität betrachtet
und an der Aktion an der die Kontrollflusskante ausgeht werden nur Outputpins betrachtet
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Ausgangsparameter der Aktivität hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Ausgangsparametern und von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)
und füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion mit der ausgehenden Kante hinzu

und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-30:

[Regel_PIM_AD_Datenfluss13]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verzweigungsknoten

und alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante von dem *‚ODER'*-Verzweigungsknoten haben, verfügen über Inputpins (bzw. Ausgangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge für alle solchen Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), von der die Kontrollflusskante ausgeht, der *‚ODER'*-Kontrollaktion einen Inputpin hinzu, der dieselbe Bezeichnung wie der Outputpin trägt

und erzeuge eine Datenflusskante zwischen zusammengehörigen Outputpins der Aktion (bzw. Eingangsparametern der Aktivität) und den neu erzeugten Inputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-31:

[Regel_PIM_AD_Datenfluss14]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verzweigungsknoten

und alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante vom oder ausgehende Kontrollflusskante zum *‚ODER'*-Verzweigungsknoten haben, verfügen über Input- oder Outputpins (bzw. Ausgangs- oder Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge für alle solchen Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), in die die Kontrollflusskante eingeht, der *‚ODER'*-Kontrollaktion einen Outputpin hinzu, der dieselbe Bezeichnung wie der Inputpin trägt; die neu hinzugefügten Outputpins gehören zum selben Parametersatz, wie der Outputpin von dem die in der Bedingung erwähnte Kontrollflusskante ausgeht

und erzeuge eine Datenflusskante zwischen zusammengehörigen Inputpins der Aktion (bzw. Ausgangsparametern der Aktivität) und den neu erzeugte Outputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-32:

[Regel_PIM_AD_Datenfluss15]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verbindungsknoten

und alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante vom oder ausgehende Kontrollflusskante zum *‚ODER'*-Verbindungsknoten haben, verfügen über Input- oder Outputpins (bzw. Ausgangs- oder Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge für alle solchen Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), von der die Kontrollflusskante ausgeht, der *‚ODER'*-Kontrollaktion einen Inputpin hinzu, der dieselbe Bezeichnung wie

der Outputpin trägt; die neu hinzugefügten Inputpins gehören zum selben Parametersatz, wie der Inputpin von den die in der Bedingung erwähnte Kontrollflusskante eingeht

und erzeuge eine Datenflusskante zwischen zusammengehörigen Outputpins der Aktion (bzw. Eingangsparametern der Aktivität) und den neu erzeugten Inputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-33:

[Regel_PIM_AD_Datenfluss16]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verbindungsknoten

und alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine ausgehende Kontrollflusskante zum *‚ODER'*-Verbindungsknoten haben, verfügen über Outputpins (bzw. Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge für alle solchen Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), in die die Kontrollflusskante eingeht, der *‚ODER'*-Kontrollaktion einen Outputpin hinzu, der dieselbe Bezeichnung wie der Inputpin trägt

und erzeuge eine Datenflusskante zwischen zusammengehörigen Inputpins der Aktion (bzw. Ausgangsparametern der Aktivität) und den neu erzeugten Outputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-34:

[Regel_PIM_AD_Datenfluss17]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verzweigungsknoten

und **nicht** alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante von dem *‚ODER'*-Verzweigungsknoten haben, verfügen über Inputpins (bzw. Ausgangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion (bzw. Eingangsparameter der Aktivität) mit der ausgehenden Kante hinzu

und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins (bzw. Eingangsparametern) zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-35:

[Regel_PIM_AD_Datenfluss18]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚ODER'*-Verzweigungsknoten

und **nicht** alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante vom oder ausgehende Kontrollflusskante zum *‚ODER'*-Verzweigungsknoten haben, verfügen über Input- oder Outputpins (bzw. Ausgangs- oder Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin

der Aktion (bzw. Ausgangsparameter der Aktivität) mit der eingehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins (bzw. Ausgangsparametern) von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-36:

[Regel_PIM_AD_Datenfluss19]
wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert
und das Kontrollelement ist ein *‚ODER'*-Verbindungsknoten
und **nicht** alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine eingehende Kontrollflusskante vom oder ausgehende Kontrollflusskante zum *‚ODER'*-Verbindungsknoten haben, verfügen über Input- oder Outputpins (bzw. Ausgangs- oder Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion (bzw. Eingangsparameter der Aktivität) mit der ausgehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins (bzw. Eingangsparametern) zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-37:

[Regel_PIM_AD_Datenfluss20]
wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert
und das Kontrollelement ist ein *‚ODER'*-Verbindungsknoten
und **nicht** alle Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine ausgehende Kontrollflusskante zum *‚ODER'*-Verbindungsknoten haben, verfügen über Outputpins (bzw. Eingangsparameter) deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin der Aktion (bzw. Ausgangsparameter der Aktivität) mit der eingehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins (bzw. Ausgangsparametern) von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-38:

[Regel_PIM_AD_Datenfluss21]
wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert
und das Kontrollelement ist ein *‚UND'*-Parallelisierungsknoten
und es gibt eine Aktion (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine eingehende Kontrollflusskante von dem *‚UND'*-Parallelisierungsknoten hat und über Inputpins (bzw. Ausgangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante
dann füge für alle solchen Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), von der die Kontrollflusskante ausgeht, der *‚UND'*-Kontrollaktion einen Inputpin hinzu, der dieselbe Bezeichnung wie der Outputpin trägt
und erzeuge eine Datenflusskante zwischen zusammengehörigen Outputpins der Aktion (bzw. Eingangsparametern der Aktivität) und den neu erzeugten Inputpins der Kontrollaktion; die Richtung der Datenfluss-

kante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-39:

[Regel_PIM_AD_Datenfluss22]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Parallelisierungsknoten

und es gibt eine Aktion (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine ausgehende Kontrollflusskante zum *‚UND'*-Parallelisierungsknoten hat und über Outputpins (bzw. Eingangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge für alle solchen Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), in die die Kontrollflusskante eingeht, der *‚UND'*-Kontrollaktion einen Outputpin hinzu, der dieselbe Bezeichnung wie der Inputpin trägt

und erzeuge eine Datenflusskante zwischen zusammengehörigen Inputpins der Aktion (bzw. Ausgangsparametern der Aktivität) und den neu erzeugte Outputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-40:

[Regel_PIM_AD_Datenfluss23]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Sychonisierungsknoten

und es gibt eine Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine eingehende Kontrollflusskante vom *‚UND'*-Synchronisierungsknoten hat und über Inputpins (bzw. Ausgangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge für alle solchen Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), von der die Kontrollflusskante ausgeht, der *‚UND'*-Kontrollaktion einen Inputpin hinzu, der dieselbe Bezeichnung wie der Outputpin trägt

und erzeuge eine Datenflusskante zwischen zusammengehörigen Outputpins der Aktion (bzw. Eingangsparametern der Aktivität) und den neu erzeugten Inputpins der Kontrollaktion; die Richtung der Datenflusskante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-41:

[Regel_PIM_AD_Datenfluss24]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Synchronisierungsknoten

und es gibt eine Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine ausgehende Kontrollflusskante zum *‚UND'*-Synchronisierungsknoten hat und über Outputpins (bzw. Eingangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge für alle solchen Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), in die die Kontrollflusskante eingeht, der *‚UND'*-Kontrollaktion einen Outputpin hinzu, der dieselbe Bezeichnung wie der Inputpin trägt

und erzeuge eine Datenflusskante zwischen zusammengehörigen Inputpins der Aktion (bzw. Ausgangsparametern der Aktivität) und den neu erzeugte Outputpins der Kontrollaktion; die Richtung der Datenfluss-

kante entspricht der der in der Bedingung der Regel erwähnten Kontrollflusskante

Regel PIM-42:

[Regel_PIM_AD_Datenfluss25]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Parallelisierungsknoten

und es gibt **keine** Aktion (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine eingehende Kontrollflusskante von dem *‚UND'*-Parallelisierungsknoten hat und über Inputpins (bzw. Ausgangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion (bzw. Eingangsparameter der Aktivität) mit der ausgehenden Kante hinzu

und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins (bzw. Eingangsparametern) zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-43:

[Regel_PIM_AD_Datenfluss26]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Parallelisierungsknoten

und es gibt **keine** Aktion (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine ausgehende Kontrollflusskante zum *‚UND'*-Parallelisierungsknoten hat und über Outputpins (bzw. Eingangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante

dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin der Aktion (bzw. Ausgangsparameter der Aktivität) mit der eingehenden Kante hinzu

und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins (bzw. Ausgangsparametern) von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-44:

[Regel_PIM_AD_Datenfluss27]

wenn zwischen einer Aktion (mit oder ohne Stereotyp) (bzw. einem Eingangsparameter der Aktivität) und einer Kontrollaktion ist auf CIM-Ebene eine Kontrollflusskante modelliert

und das Kontrollelement ist ein *‚UND'*-Sychonisierungsknoten

und es gibt **keine** Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. Aktivität), die eine eingehende Kontrollflusskante vom *‚UND'*-Synchronisierungsknoten hat und über Inputpins (bzw. Ausgangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Outputpins der Aktion (bzw. Eingangsparameter der Aktivität), mit der ausgehenden Kante

dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Outputpin der Aktion (bzw. Eingangsparameter der Aktivität) mit der ausgehenden Kante hinzu

und füge dem Aktivitätsdiagramm Datenflusskanten von den Outputpins (bzw. Eingangsparametern) zu den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-45:

[Regel_PIM_AD_Datenfluss28]

wenn zwischen einer Kontrollaktion und einer Aktion (mit oder ohne Stereotyp) (bzw. einem Ausgangsparameter der Aktivität) ist auf CIM-Ebene eine Kontrollflusskante modelliert
und das Kontrollelement ist ein *‚UND'*-Synchronisierungsknoten
und es gibt **keine** Aktionen (mit oder ohne Stereotyp, Kontrollaktionen) (bzw. die Aktivität), die eine ausgehende Kontrollflusskante zum *‚UND'*-Synchronisierungsknoten hat und über Outputpins (bzw. Eingangsparameter) verfügt, deren Objektknotentypen (*«document»*) zu demselben *«entity»* gehören wie die Objektknotentypen von Inputpins der Aktion (bzw. Ausgangsparameter der Aktivität), mit der eingehenden Kante
dann füge, falls nicht schon vorhanden, dem Aktivitätsdiagramm einen Objektknoten mit dem Stereotyp *«datastore»* für jeden solchen Inputpin der Aktion (bzw. Ausgangsparameter der Aktivität) mit der eingehenden Kante hinzu
und füge dem Aktivitätsdiagramm Datenflusskanten zu den Inputpins (bzw. Ausgangsparametern) von den zugehörigen Objektknoten hinzu (Bezeichnung der Kanten: *‚{copy}'*)

Regel PIM-46:

[Regel_PIM_CD_Inputparameter1]
wenn in einem Klassendiagramm auf CIM-Ebene ist eine *«entity»* Klasse einer *«process»* Klasse als Inputparameter zugeordnet (Assoziation *‚isInput'*)
dann erstelle eine Assoziation mit der Bezeichnung *‚isInput'* zwischen der *«process»* Klasse und der *«document»* Klasse, die aus der *«entity»* Klasse für die *«process»* Klasse generiert wurde
und lösche die Assoziation *‚isInput'* zwischen der *«entity»* und der *«process»* Klasse

Regel PIM-47:

[Regel_PIM_CD_Outputparameter1]
wenn in einem Klassendiagramm auf CIM-Ebene ist eine *«entity»* Klasse einer *«process»* Klasse als Outputparameter zugeordnet (Assoziation *‚hasOutput'*)
dann erstelle eine Assoziation mit der Bezeichnung *‚hasOutput'* zwischen der *«process»* Klasse und der *«document»* Klasse, die aus der *«entity»* Klasse für die *«process»* Klasse generiert wurde
und lösche die Assoziation *‚hasOutput'* zwischen der *«entity»* und der *«process»* Klasse

Regel PIM-48:

[Regel_PIM_CD_Inputparameter2]
wenn das interne Verhalten eines Prozesses (*«process»* Klasse) ist in einem Aktivitätsdiagramm beschrieben
und das Aktivitätsdiagramm enthält Aktionen die Subprozessaufrufe repräsentieren
und den Subprozessen sind in Klassendiagrammen der CIM-Ebene *«entity»* Klassen als Inputparameter zugeordnet (Assoziation *‚isInput'*)
dann erstelle für jeden Subprozess eine Assoziation mit der Bezeichnung *‚isInput'* zwischen der *«process»* Klasse und *«document»* Klasse(n); die *«document»* Klasse(n) wurde(n) spezifisch für den jeweiligen Subprozess aus der zugehörigen *«entity»* Klasse generiert

Regel PIM-49:

[Regel_PIM_CD_Outputparameter2]
wenn das interne Verhalten eines Prozesses (*«process»* Klasse) ist in einem Aktivitätsdiagramm beschrieben
und das Aktivitätsdiagramm enthält Aktionen die Subprozessaufrufe repräsentieren
und den Subprozessen sind in Klassendiagrammen der CIM-Ebene *«entity»* Klassen als Outputparameter zugeordnet (Assoziation *‚hasOutput'*)
dann erstelle für jeden Subprozess eine Assoziation mit der Bezeichnung *‚hasOutput'* zwischen der *«process»* Klasse und *«document»* Klasse(n); die *«document»* Klasse(n) wurde(n) spezifisch für den jeweiligen Subprozess aus der zugehörigen *«entity»* Klasse generiert

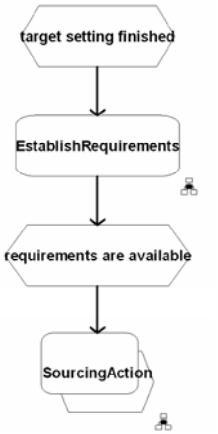
Anhang C

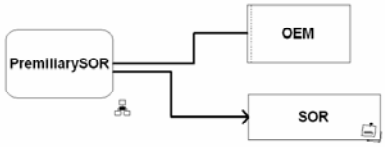
In diesem Anhang finden sich alle Diagramme zu dem Beispiel aus der Fallstudie. Die Diagramme sind jeweils einem der drei Modelle (ARIS-Modell, BPDM-Modell auf CIM-Ebene und BPDM-Modell auf PIM-Ebene) zugeordnet. Innerhalb eines Modells sind die zu einem Prozess gehörigen Diagramme zusammen gruppiert. Der jeweilige Prozess(name) wird mit folgender Notation angegeben: *‚<NameÜbergeordneterProzess>::<NameProzess>'*.

Fallstudie ARIS-Modelle

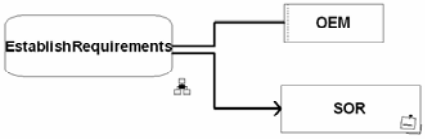
OEM::WKD

OEM::PreliminarySOR

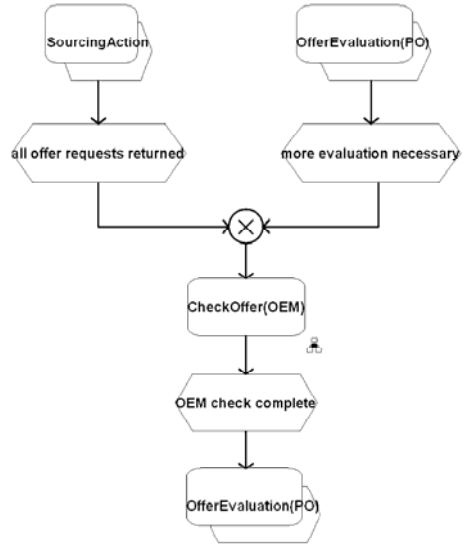


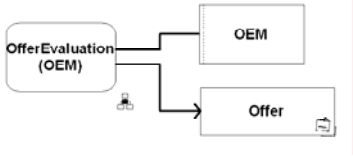


OEM::PreliminarySOR::EstablishRequirements



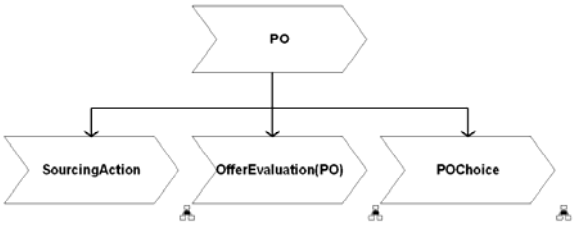
OEM::OfferEvaluation



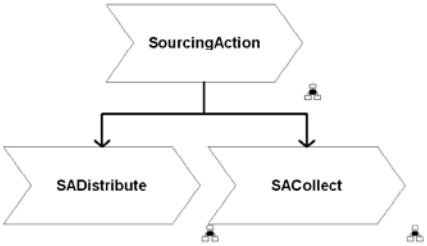


OEM::OfferEvaluation::CheckOffer

PO



PO::SourcingAction

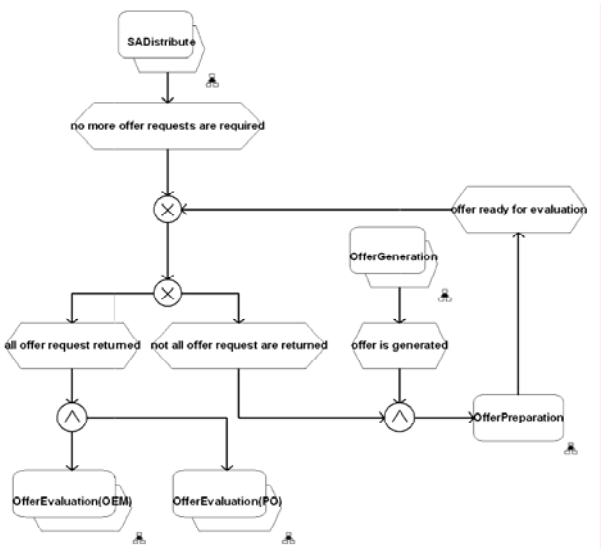


PO::SourcingAction::SADistribute

PO::SourcingAction::SADistribute::GenerateSourcingAction

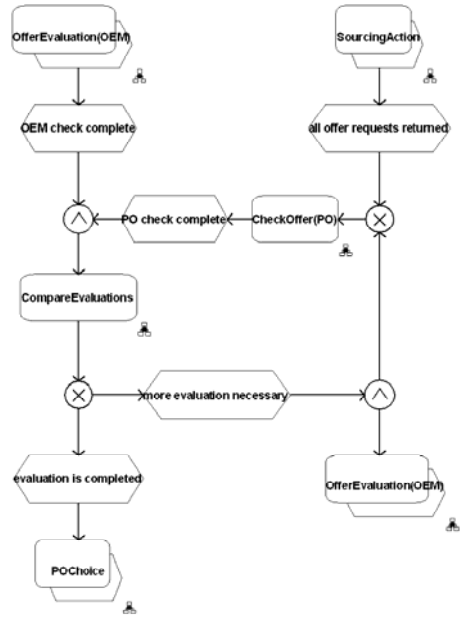
PO::SourcingAction::SADistribute::PrepareOfferRequest

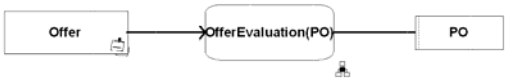
PO::SourcingAction::SACollect



PO::SourcingAction::SACollect::OfferPreparation

PO::OfferEvaluation(PO)

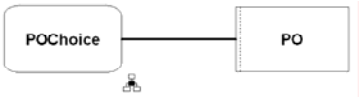




PO::OfferEvaluation(PO)::CheckOffer(PO)

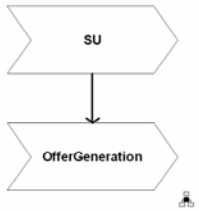
PO::OfferEvaluation(PO)::CompareEvaluations

PO::POChoice

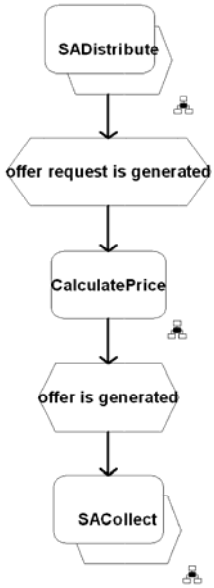


PO::POChoice::ChoosePartners

SU



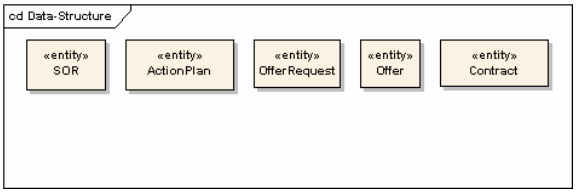
SU::OfferGeneration



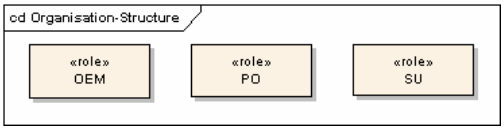
SU::OfferGeneration::CalculatePrice

Fallstudie BPDM-Modelle auf CIM-Ebene

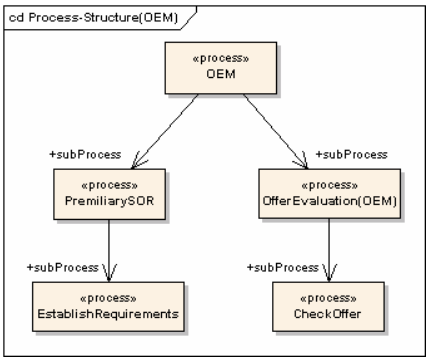
Datentypen



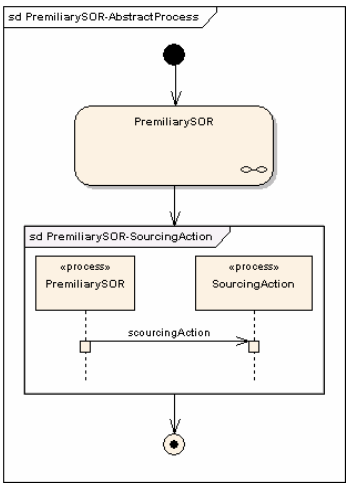
Rollen

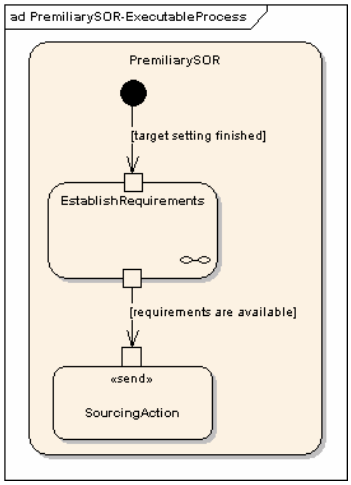


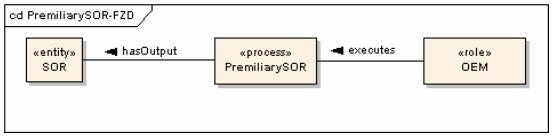
OEM



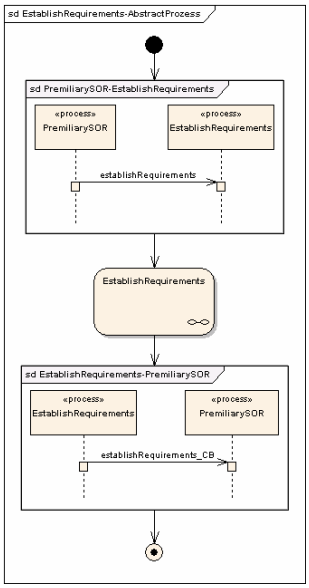
OEM::PreliminarySOR

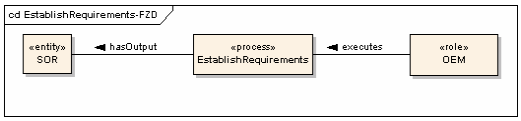




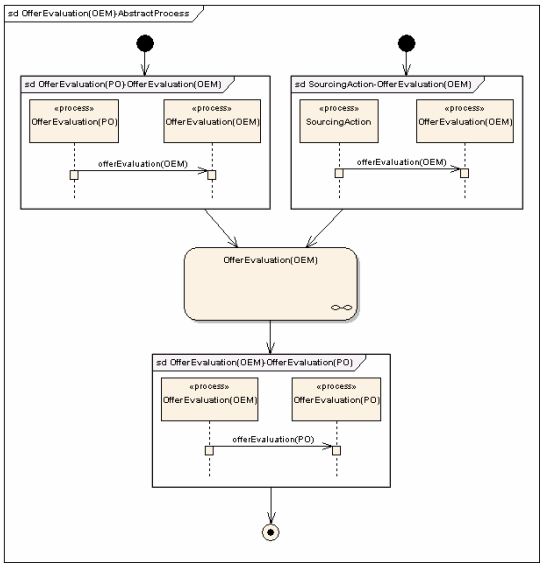


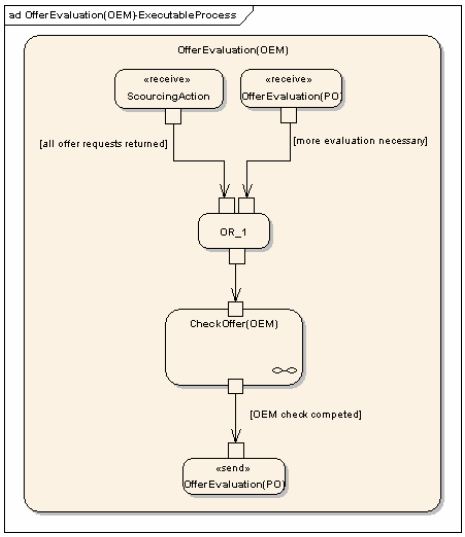
OEM::PreliminarySOR::EstablishRequirements

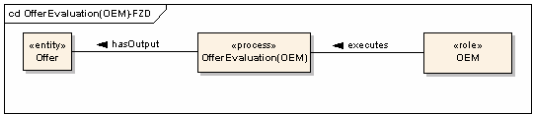




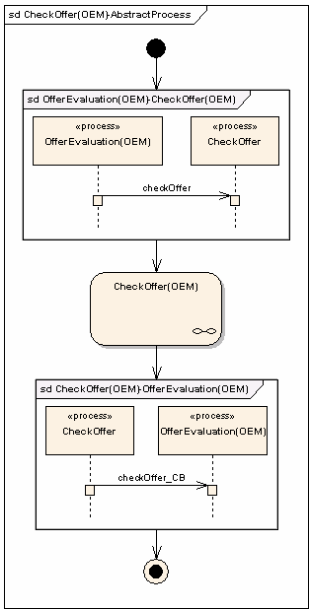
OEM::OfferEvaluation(OEM)

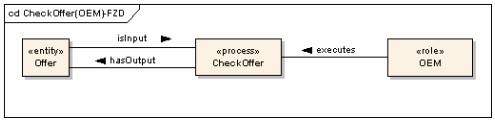




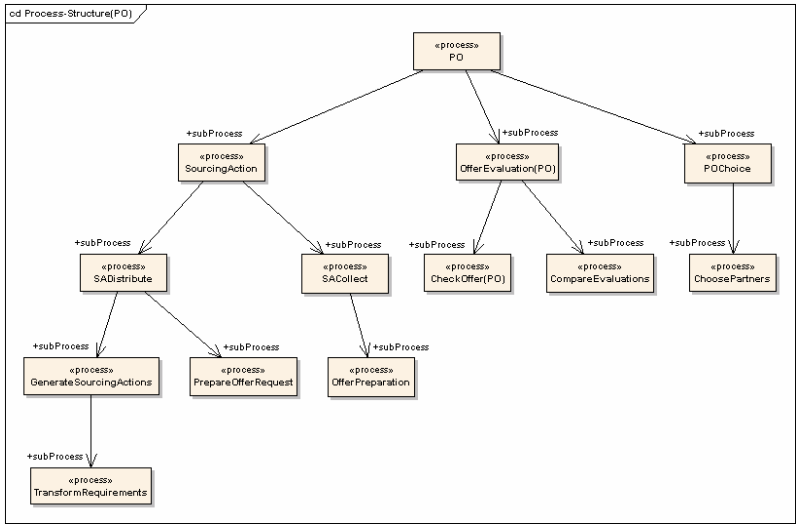


OEM::OfferEvaluation(OEM)::CheckOffer(OEM)

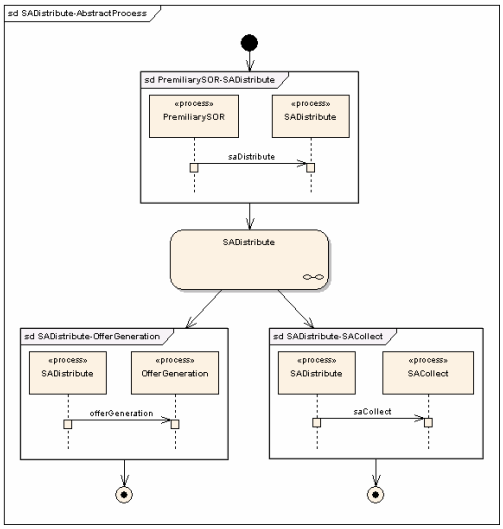


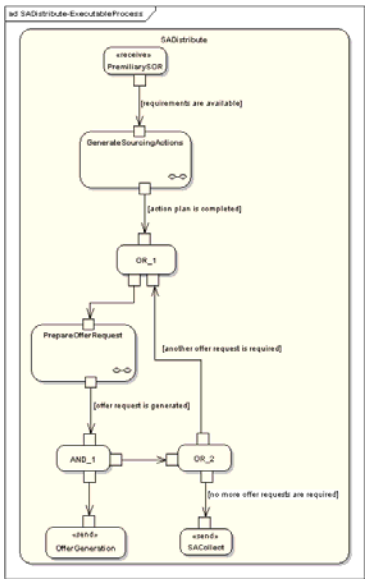


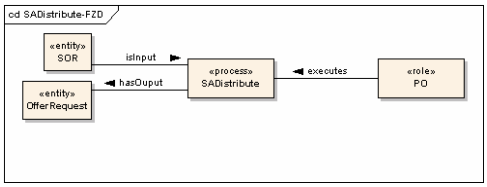
PO



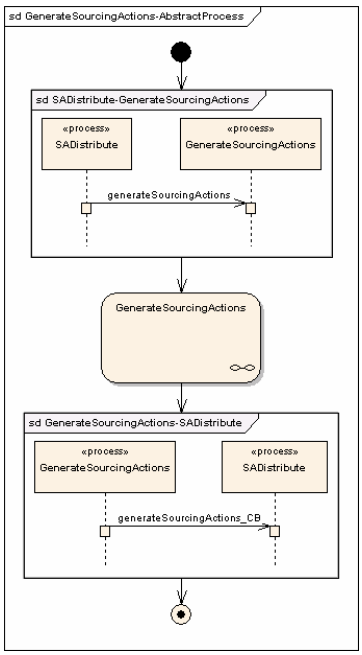
PO::SourcingAction::SADistribute

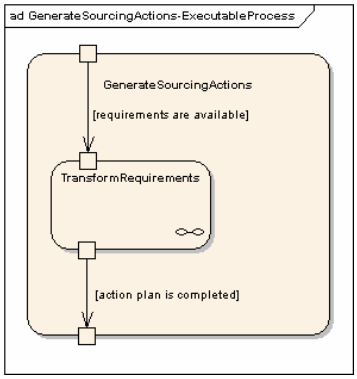


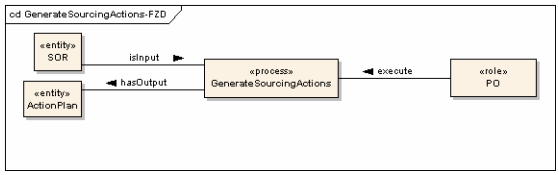




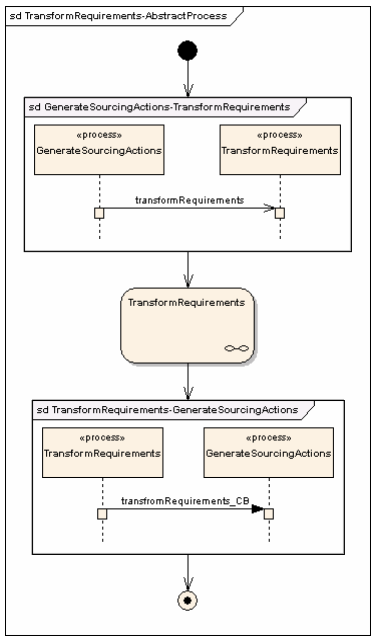
PO::SourcingAction::SADistribute::GenerateSourcingActions

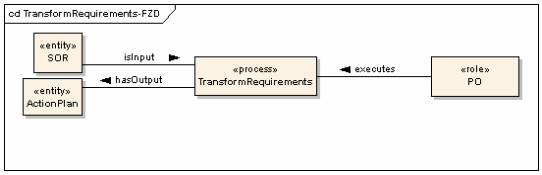




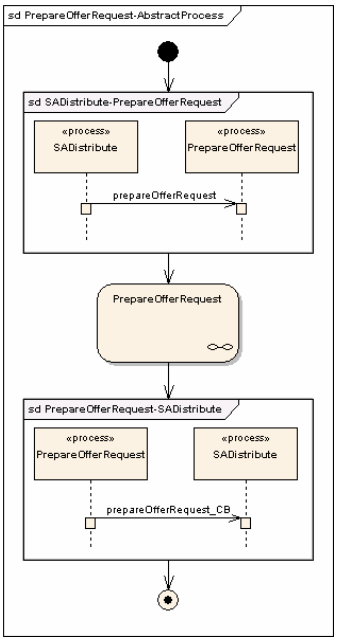


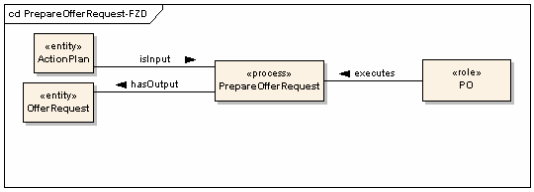
PO::SourcingAction::SADistribute::GenerateSourcingActions::TransformRequirements



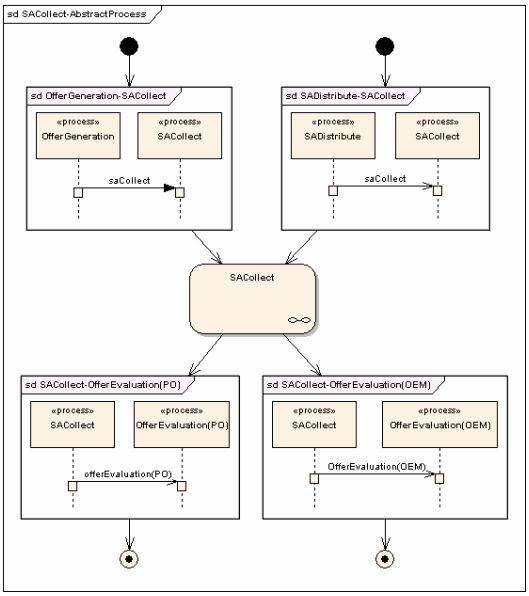


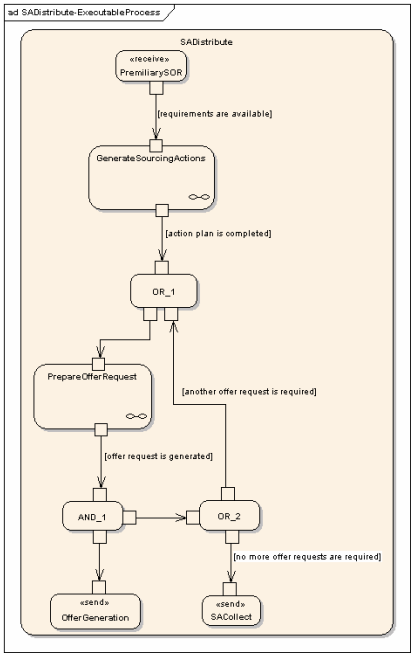
PO::SourcingAction::SADistribute::PrepareOfferRequest

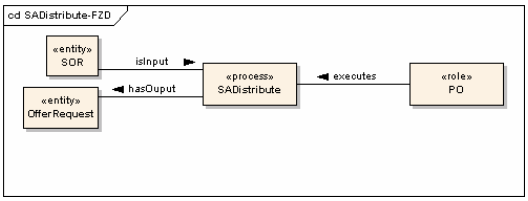




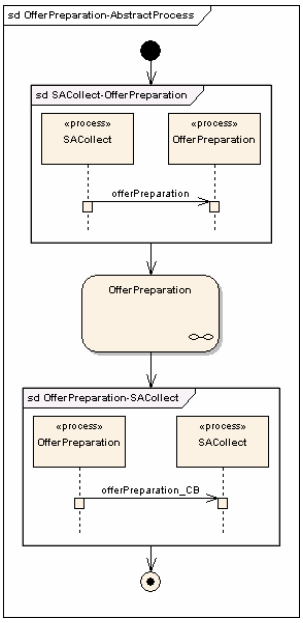
PO::SourcingAction::SACollect

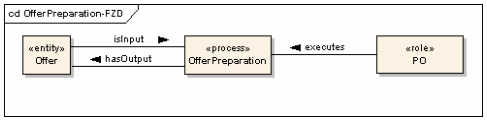




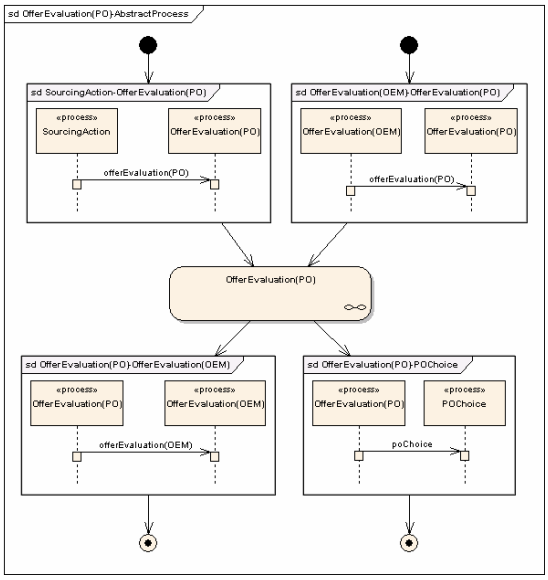


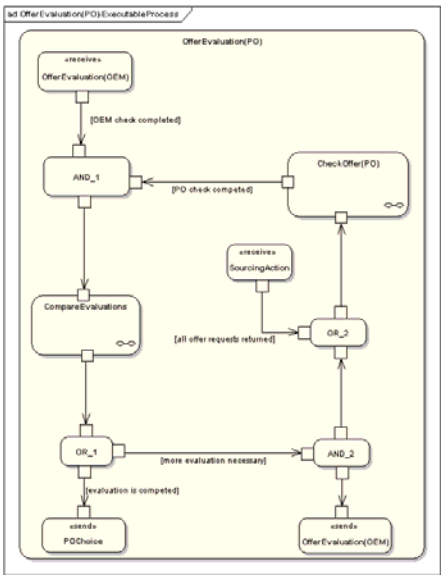
PO::SourcingAction::SACollect::OfferPreparation

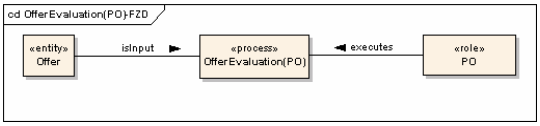




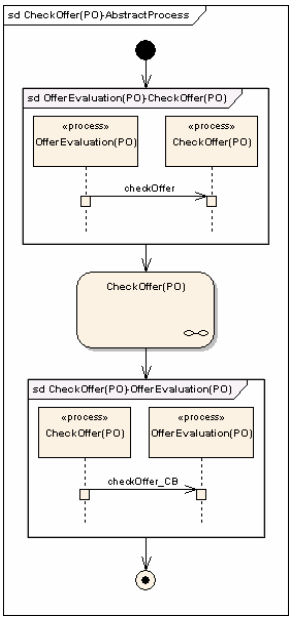
PO:OfferEvaluation(PO)

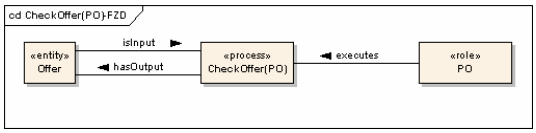




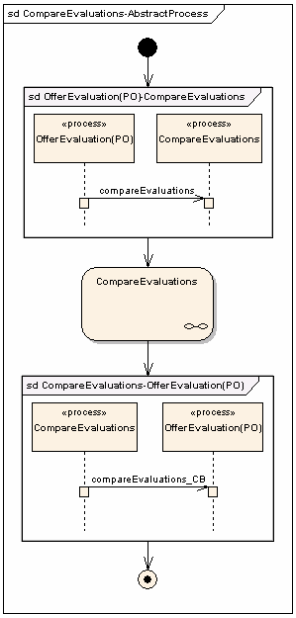


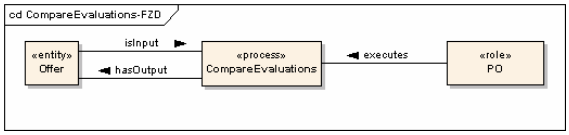
PO:OfferEvaluation(PO)::CheckOffer(PO)



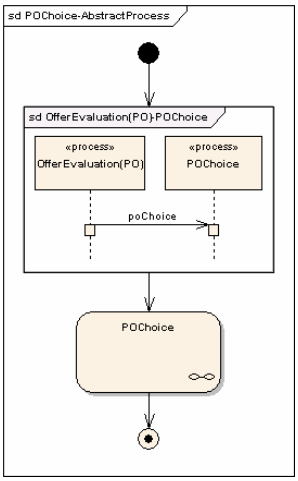


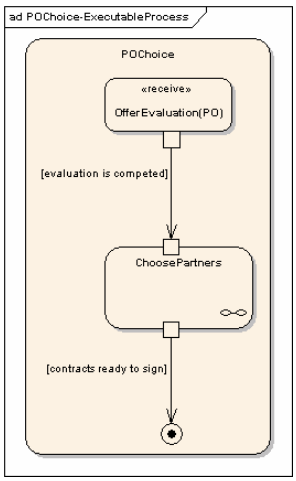
PO:OfferEvaluation(PO)::CompareEvaluations

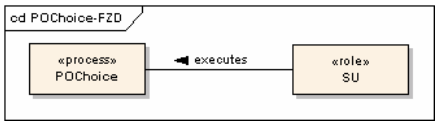




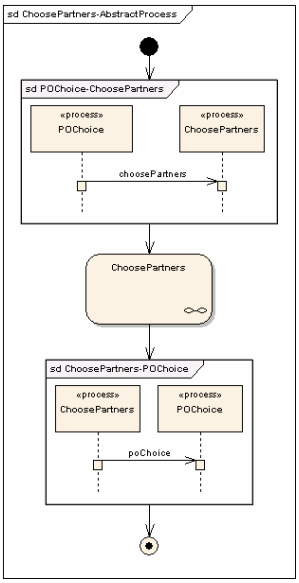
PO::POChoice

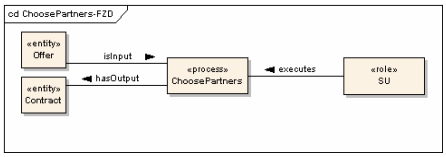




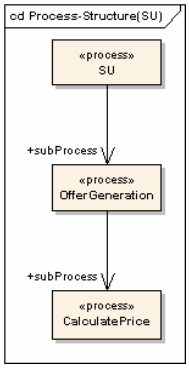


PO::POChoice::ChoosePartners

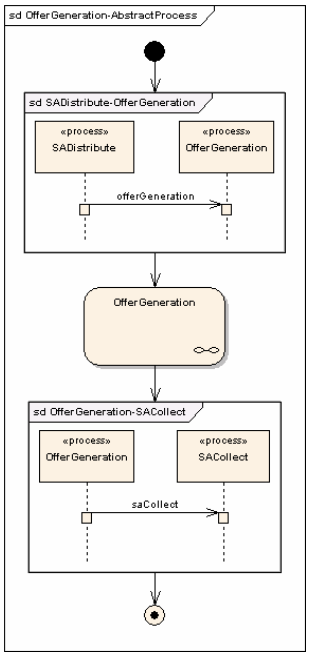


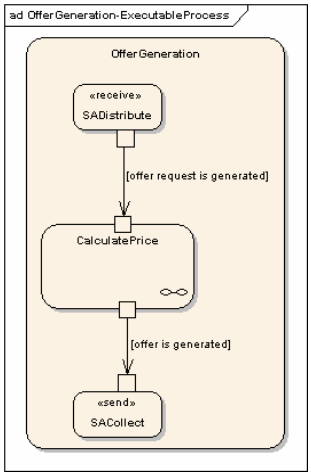


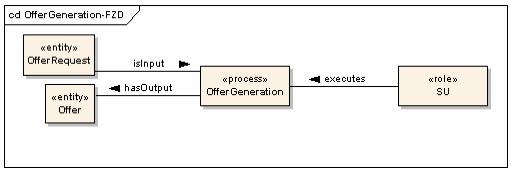
SU



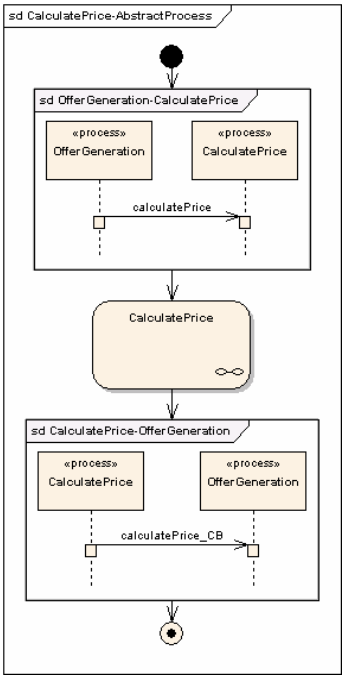
SU::OfferGeneration

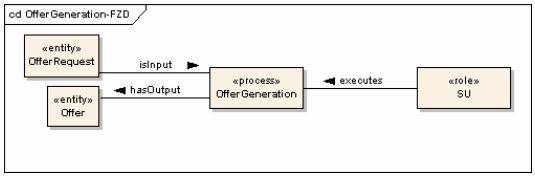






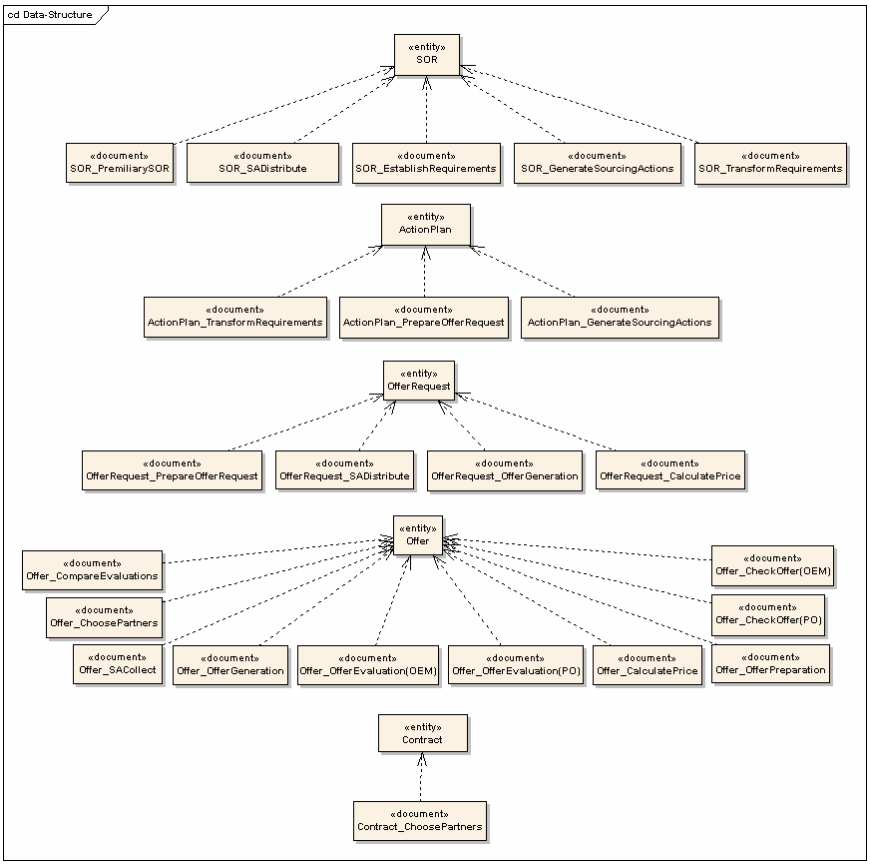
SU::OfferGeneration::CalculatePrice



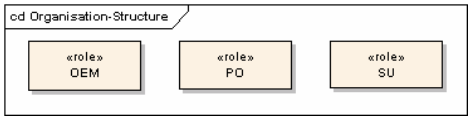


Fallstudie BPDM-Modelle auf PIM-Ebene

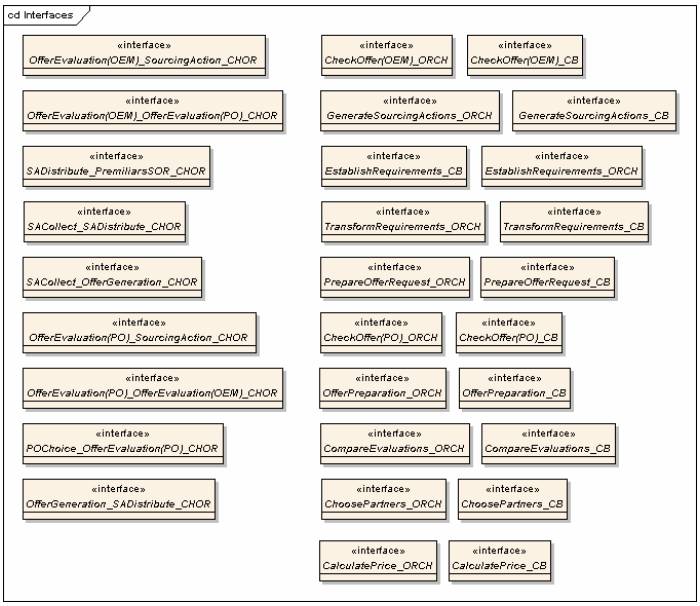
Datentypen



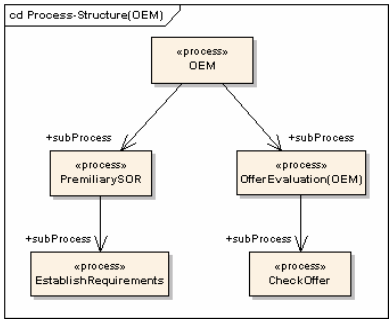
Rollen



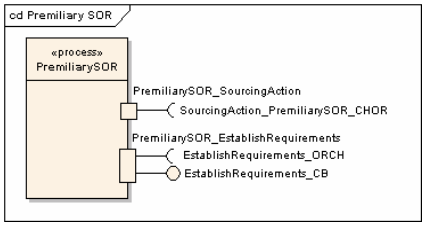
Schnittstellen

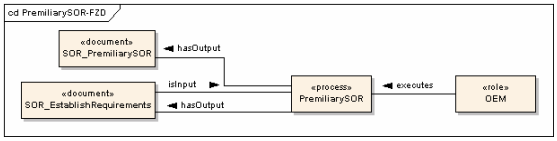


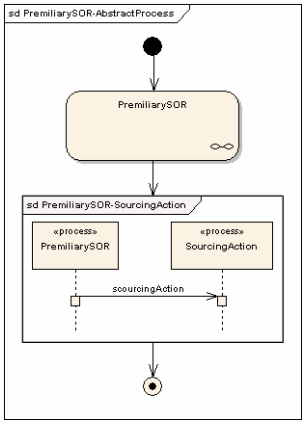
OEM

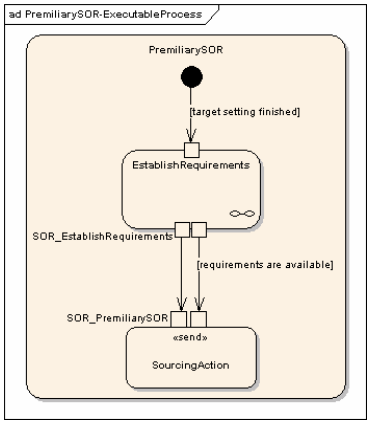


OEM::PreliminarySOR

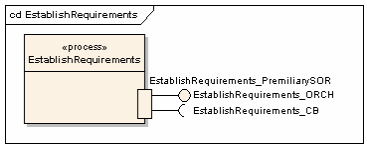


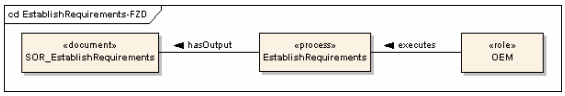


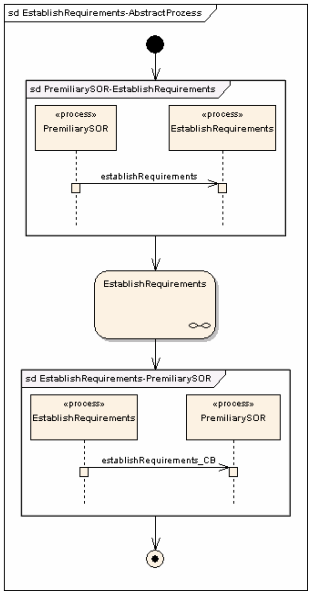




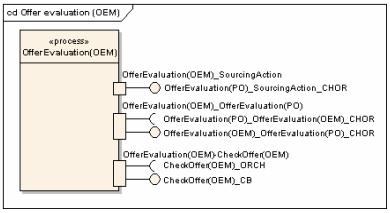
OEM::PreliminarySOR::EstablishRequirements

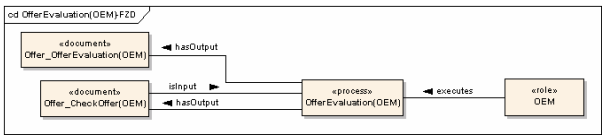


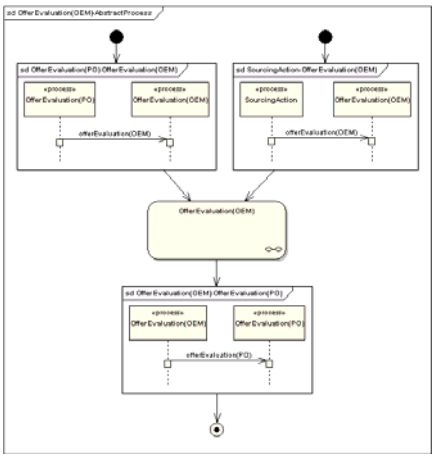


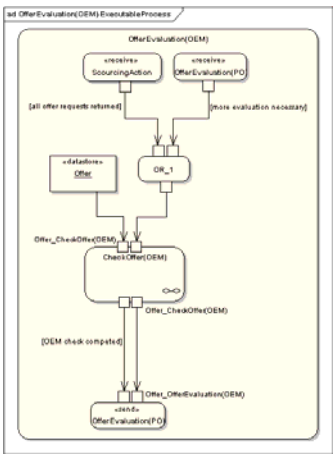


OEM::OfferEvaluation(OEM)

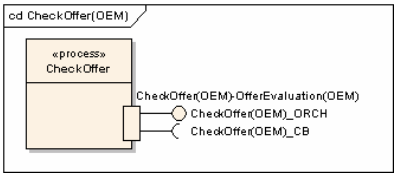


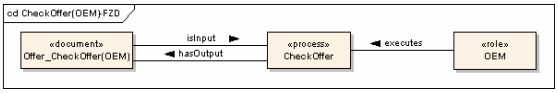


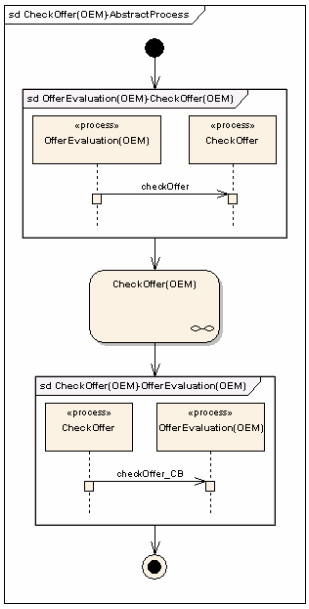




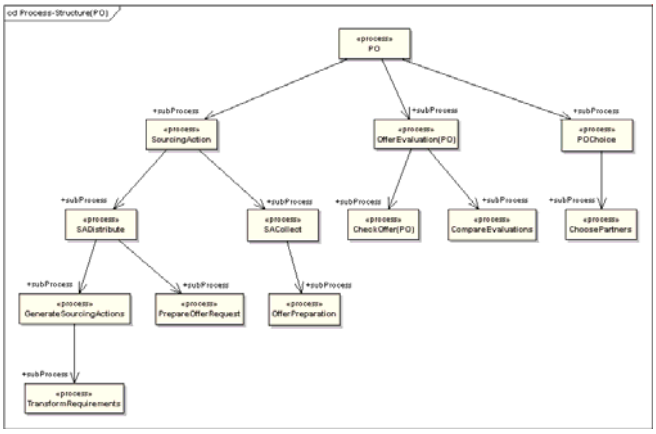
OEM::OfferEvaluation(OEM)::CheckOffer(OEM)



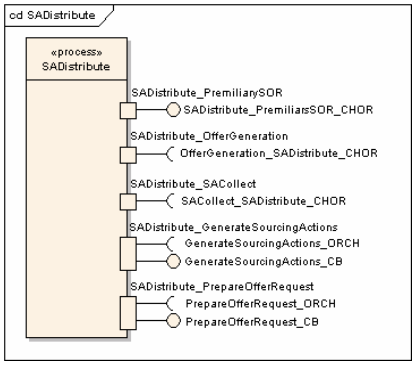


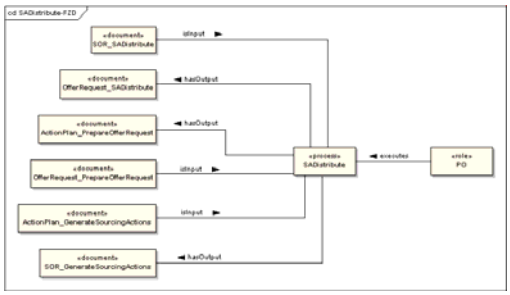


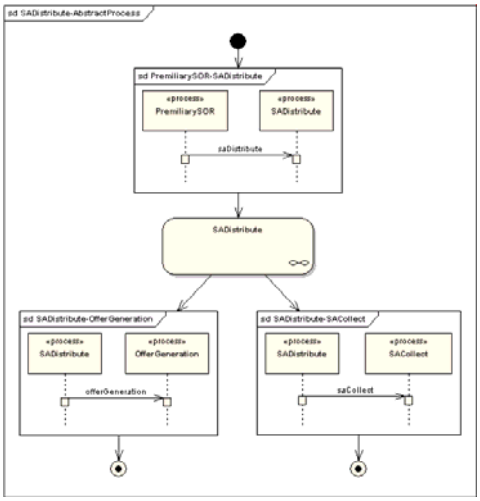
PO

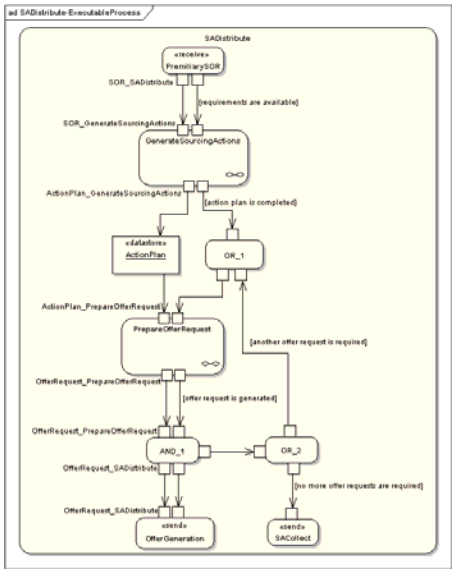


PO::SourcingAction::SADistribute

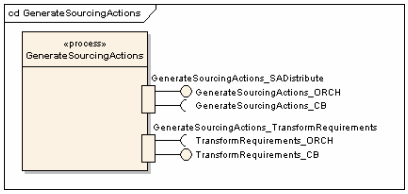


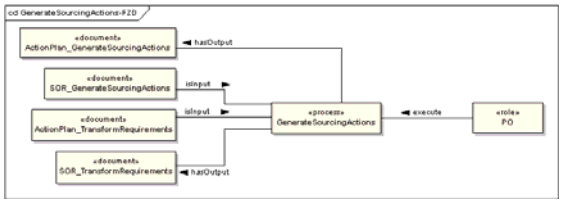


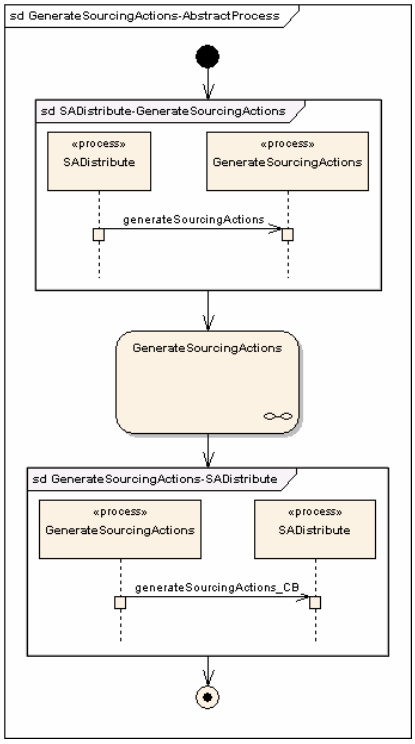


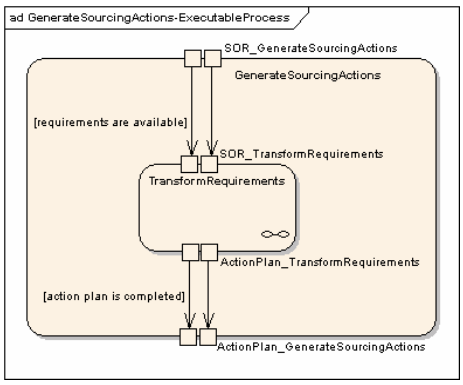


PO::SourcingAction::SADistribute::GenerateSourcingActions

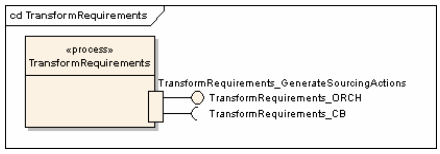


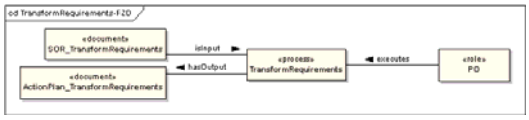


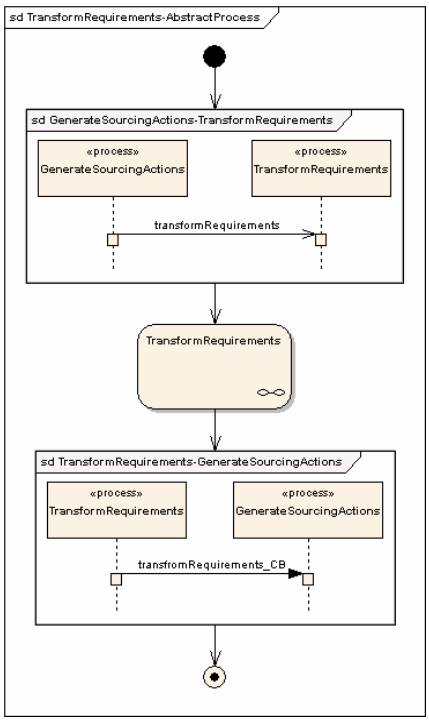




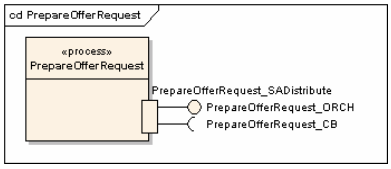
PO::SourcingAction::SADistribute::GenerateSourcingActions::TransformRequirements

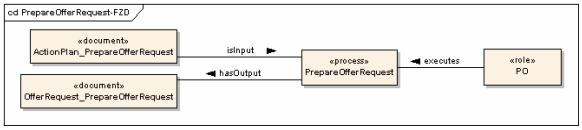


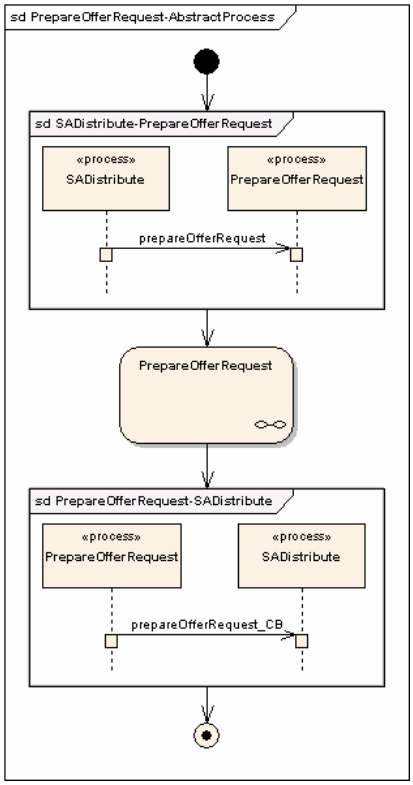




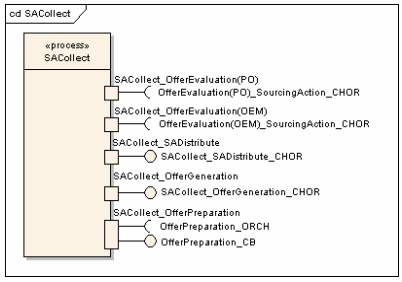
PO::SourcingAction::SADistribute::PrepareOfferRequest

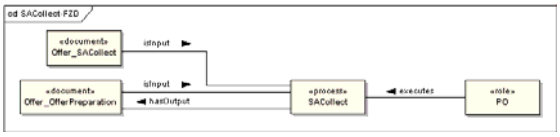


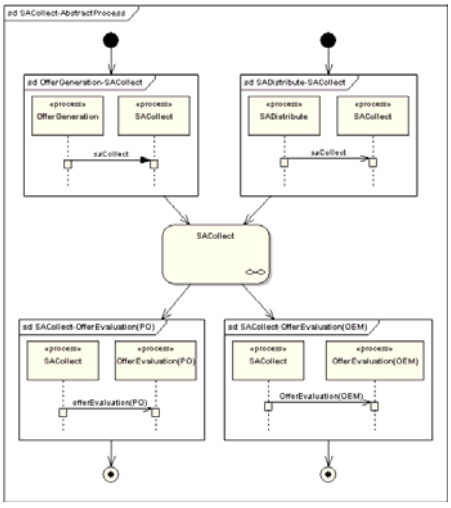


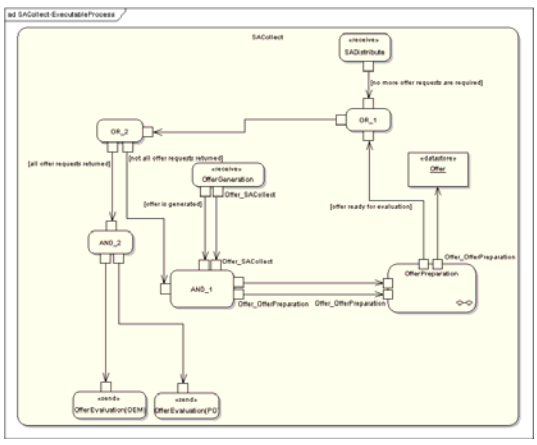


PO::SourcingAction::SACollect

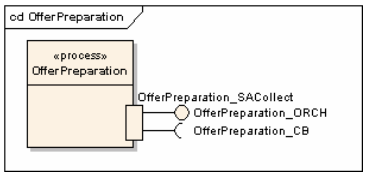


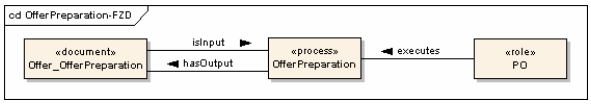


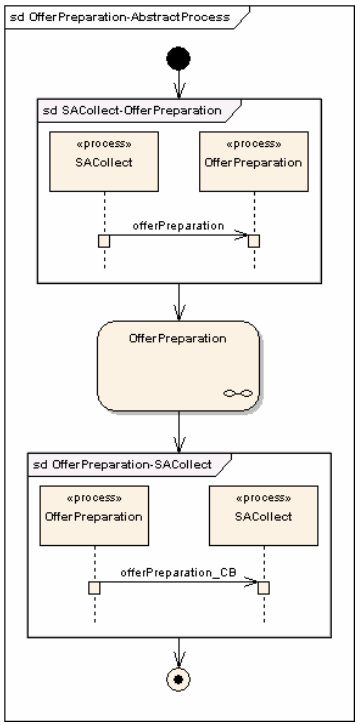




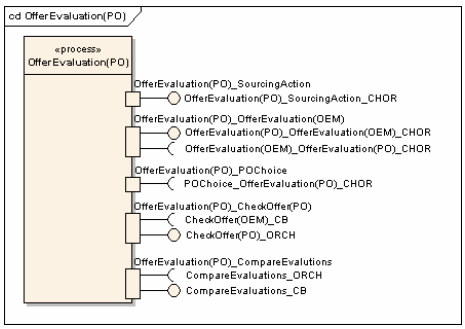
PO::SourcingAction::SACollect::OfferPreparation

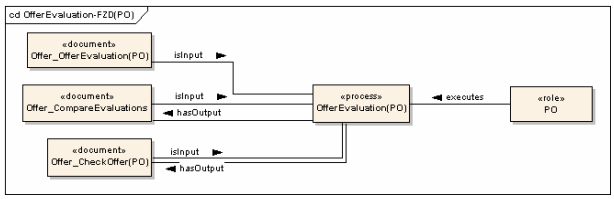


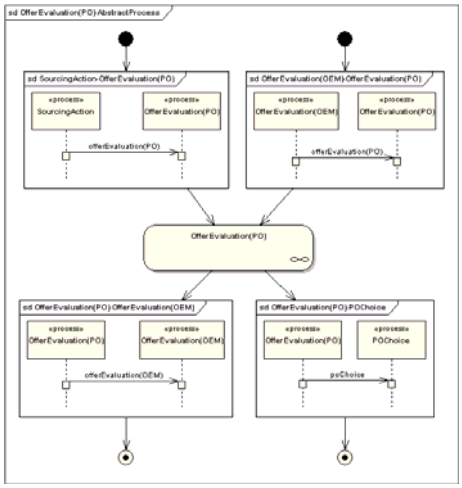


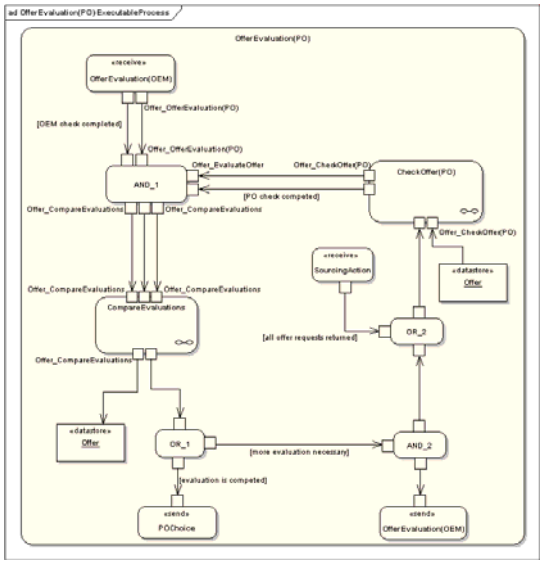


PO:OfferEvaluation(PO)

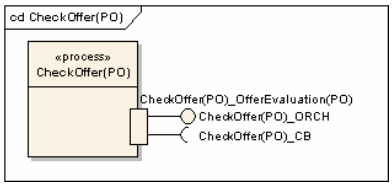


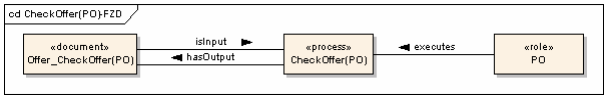


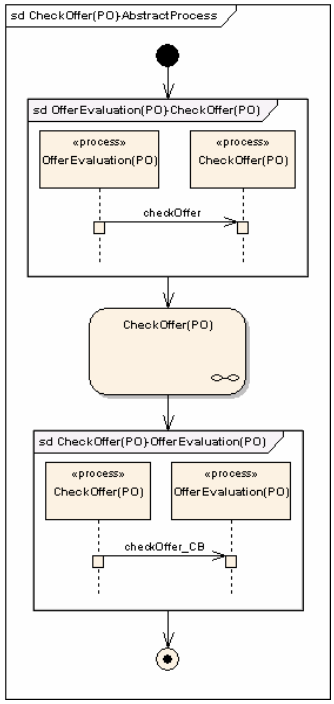

sd OfferEvaluation(PO):AbstractProcess
sd SourcingAction-OfferEvaluation(PO)
«process»
SourcingAction
«process»
OfferEvaluation(PO)
offerEvaluation(PO)
sd OfferEvaluation(OEM):OfferEvaluation(PO)
«process»
OfferEvaluation(OEM)
«process»
OfferEvaluation(PO)
offerEvaluation(PO)
OfferEvaluation(PO)
sd OfferEvaluation(PO):OfferEvaluation(OEM)
«process»
OfferEvaluation(PO)
«process»
OfferEvaluation(OEM)
offerEvaluation(OEM)
sd OfferEvaluation(PO):POChoice
«process»
OfferEvaluation(PO)
«process»
POChoice
poChoice


sd OfferEvaluation(PO):ExecutableProcess
OfferEvaluation(PO)
«receive»
OfferEvaluation(OEM)
Offer_OfferEvaluation(PO)
[OEM check completed]
Offer_OfferEvaluation(PO)
AND_1
Offer_EvaluateOffer
Offer_CheckOffer(PO)
[PO check competed]
CheckOffer(PO)
Offer_CompareEvaluations
Offer_CompareEvaluations
Offer_CheckOffer(PO)
«receive»
SourcingAction
«datastore»
Offer
Offer_CompareEvaluations
Offer_CompareEvaluations
CompareEvaluations
[all offer requests returned]
OR_2
Offer_CompareEvaluations
«datastore»
Offer
OR_1
[more evaluation necessary]
AND_2
[evaluation is competed]
«send»
POChoice
«send»
OfferEvaluation(OEM)

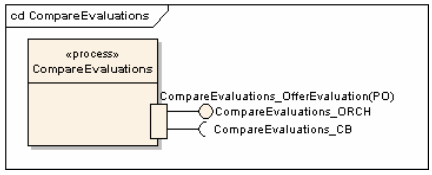
PO:OfferEvaluation(PO)::CheckOffer(PO)

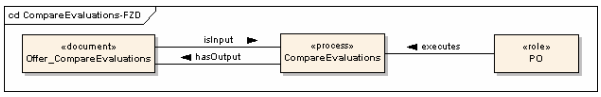


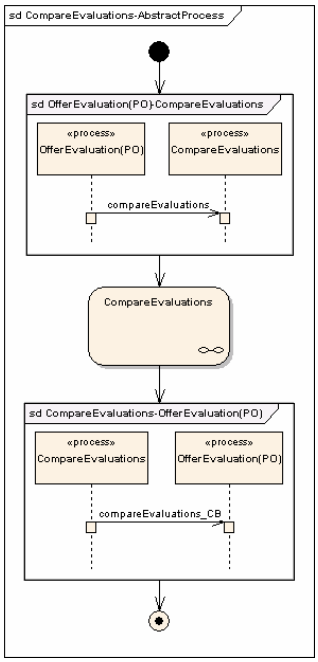




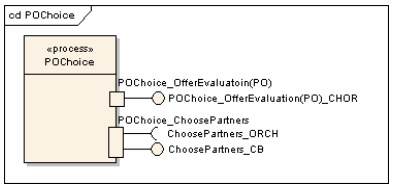
PO:OfferEvaluation(PO)::CompareEvaluations

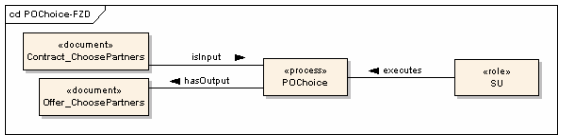


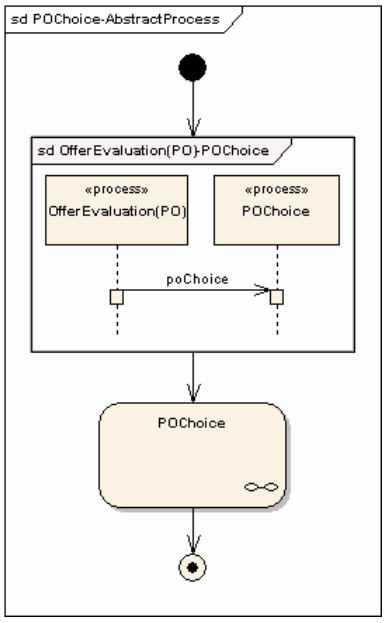


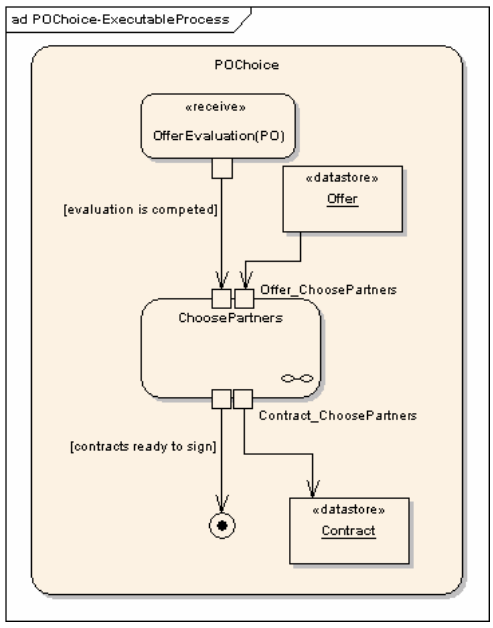


PO::POChoice

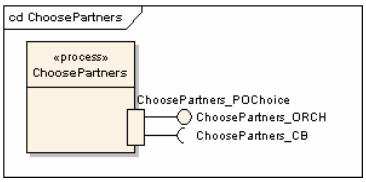


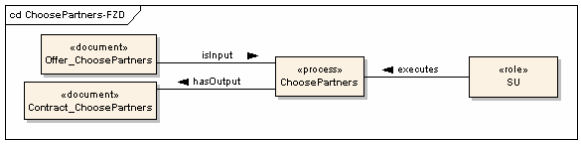


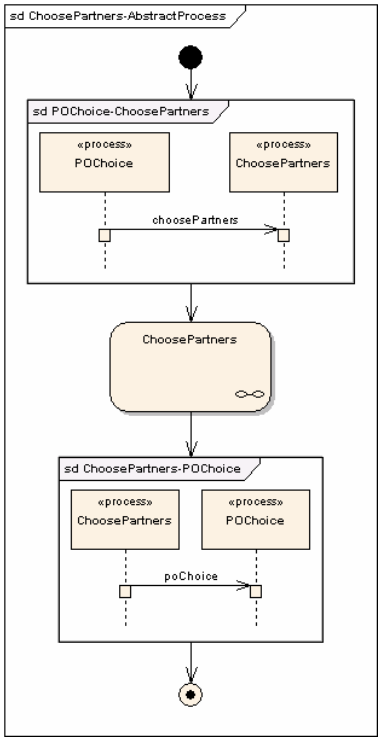




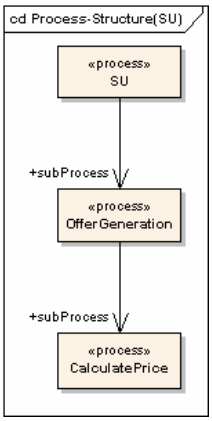
PO::POChoice::ChoosePartners



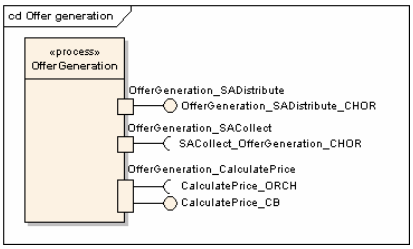


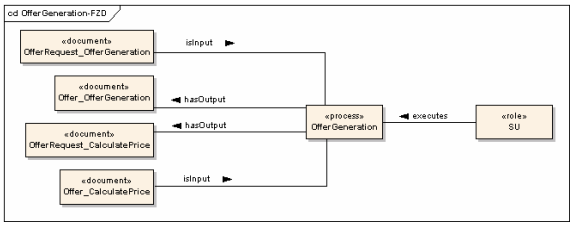


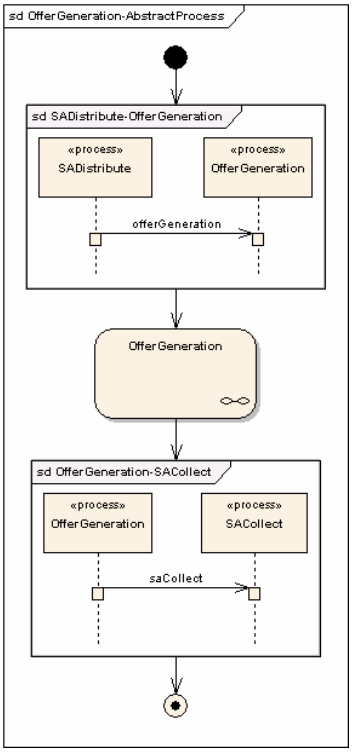
SU

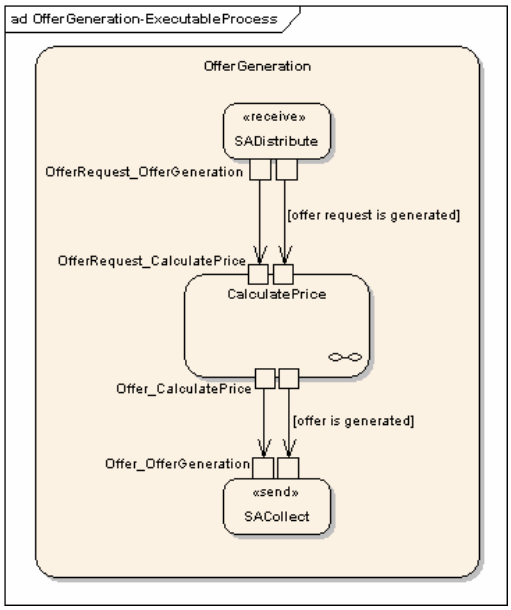


SU::OfferGeneration

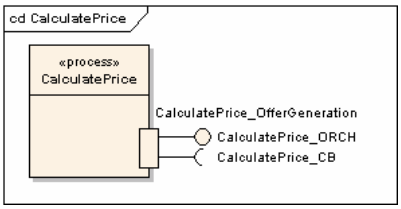


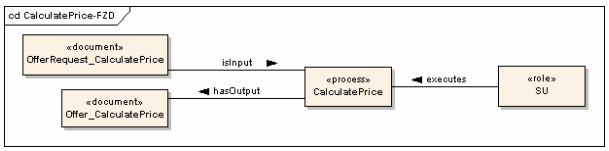


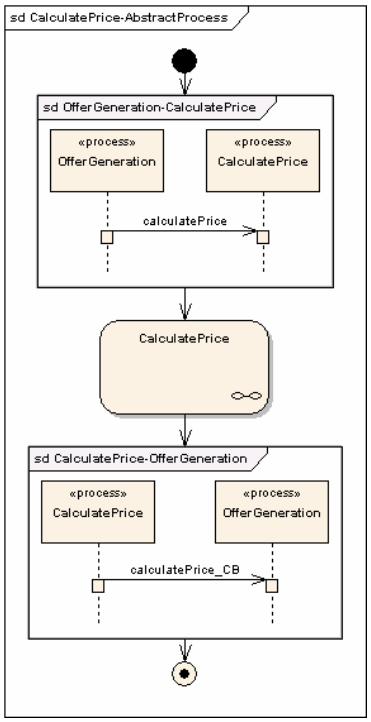




SU::OfferGeneration::CalculatePrice






sd CalculatePrice-AbstractProcess
sd OfferGeneration-CalculatePrice
«process»
OfferGeneration
«process»
CalculatePrice
calculatePrice
CalculatePrice
sd CalculatePrice-OfferGeneration
«process»
CalculatePrice
«process»
OfferGeneration
calculatePrice_CB