Thema:

Konzeption und Entwicklung einer Erweiterung des Systems **BPM@SharePoint-QUAM 2.0** um Funktionalitäten zur Unterstützung der Modellierung und den Export in BPMN 2.0

Diplomarbeit

Arbeitsgruppe Wirtschaftsinformatik

Themensteller: LINTRA GmbH
Dipl.-Ing. Knut Köchli

Betreuer: Prof. Dr. rer. pol. habil. Hans-Knud Arndt

Vorgelegt von: Tihomir Todorov
E-Mail: todorov@st.ovgu.de

Abgabetermin: 07.02.2013

Inhaltsverzeichnis

Abbildungsverzeichnis

Listingverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BPM	Business Process Management
BPEL	Business Process Execution Language
BPMN	Business Process Model and Notation
BPMS	Business Process Management Suite
BPMI	Business Process Management Initiative
BPMA	Business Process Modeling and Analysis
EABPM	European Association of Business Process Management
EPK	Ereignisgesteuerte Prozesskette
IBM	International Business Machines
MOF	Meta Object Facility
OMG	Object Management Group
SDK	Software Development Kit
UML	Unified Modeling Language
WfMC	Workflow Management Coalition
WFMS	Workflow-Management-System
XPDL	XML Process Definition Language
XML	Extensible Markup Language
XMI	XML Model Interchange

1 Einleitung

1.1 Motivation und Ziel der Arbeit

Die rasante Entwicklung, die Business Process Management in der Vergangenheit erfahren hat, stellt die Hersteller von Software-Werkzeugen für Prozessmanagement vor die Herausforderung, den Funktionsumfang ihrer Produkte ständig zu überprüfen und ggf. zu erweitern. Prognosen des Marktforschungsanbieters Gartner zufolge werden im Jahr 2014 etwa 40% der Business-Manager und Wissensträger in Großunternehmen ein umfassendes grafisches Modell für die Darstellung ihrer Prozesse verwenden - fast eine Verzehnfachung gegenüber dem Jahr 2009, im dem es ca. 6% waren (Pettey, et al., 2010). Diese verbreitete Verwendung von Modellierungswerkzeugen wird laut Gartner zu einer deutlichen Verbesserung der Geschäftsprozesse und Leistungen führen. Laut Analysten wird daher empfohlen, die Prozessmodellierung als eigene Kompetenz in der Organisation zu etablieren (Pettey, et al., 2010).

Die vom Magdeburger Unternehmen LINTRA entwickelte BPM-Anwendung QUAM 2.0 basiert auf Microsoft SharePoint als Plattform zur Strukturierung von Unternehmensdaten und Microsoft Visio als Plattform für ihre Visualisierung. Die Abkürzung QUAM steht für Qualitätsmanagement, Umweltmanagement, Arbeitssicherheitsmanagement und Management-Regularien. QUAM-Systeme werden in Unternehmen hauptsächlich für die Dokumentation von Prozess- und Organisationsstrukturen eingesetzt; aber auch die Möglichkeit zur Definition einfacher Workflows, basierend auf Windows Workflow Foundation, ist in QUAM vorhanden.

Um die Bandbreite der Funktionen von QUAM 2.0 zu erweitern und gleichzeitig die QUAM Interoperabilität mit anderen Systemen zu erreichen, hat sich LINTRA zum Ziel gesetzt, einen weit verbreiteten Standard zu Modellierung, Modellaustausch und ggf. Workflowausführung zu implementieren. Ein anderer Hintergrund dieser Aufgabenstellung ist das Bedürfnis nach einer Flexibilisierung des QUAM-Systems bezüglich angebotener Modellierungselemente. Das QUAM-System verfügt über eine Symbolpalette,

mit der sich die Mehrheit der Unternehmensvorgänge aus fachlicher Sicht sehr gut abbilden lässt. Es gibt allerdings, nicht selten, große und komplexe Sachverhalte, die sich mit den derzeit benutzten Symbolen nicht interpretationsfrei darstellen lassen. Diese Prozesse ließen sich mit einer erweiterten Symbolpalette, die eine festgelegte Semantik anbietet, eindeutiger ausdrücken.

Im Rahmen dieser Abschlußarbeit sollen daher potentiell mögliche Modellierungssprachen nach bestimmten Kriterien untersucht werden. Nachdem die Auswahl einer Modellierungssprache erfolgt ist, soll diese in das QUAM 2.0 System integriert werden; auch die Implementierung einer Funktion zum XML-Export von Prozessmodellen soll erfolgen.

1.2 Gliederung

Im theoretischen Teil der Arbeit (Kapitel 2) werden Begriffe und Theorien des Geschäftsprozessmanagement und der Prozessmodellierung erläutert. Es werden derzeit weit verbreitete Modellierungssprachen, wie die EPK, BPMN 2.0, UML, XPDL und BPEL, dargestellt und nach Kriterien, wie Verbreitungsgrad, unterstützte Phasen der Modellierung usw., verglichen. Anschließend wird die zu implementierende Sprache ausgewählt und ausführlicher dargestellt.

In Kapitel 3 werden zunächst die XML-Sprache und das XML-Metamodell der ausgewählten Modellierungssprache behandelt. Darauf folgt eine Einführung in die Technologien, auf denen das System QUAM 2.0 basiert.

Der Entwurf und die prototypische Implementierung der Funktionserweiterung von QUAM 2.0 erfolgen in Kapitel 4. Dabei wird zu Beginn auf die Integration der ausgewählten Modellierungssprache in das QUAM 2.0 System eingegangen; danach wird auf die Implementierung der Export-Komponente eingegangen, die die Prozessmodelle in ein standardisiertes XML-Austauschformat überführt. Mit dem anschließenden Test eines exportierten Prozessmodells und einer Zusammenfassung der zentralen Erkenntnisse endet diese Arbeit.

2 Theoretische Grundlagen

2.1 Grundbegriffe

Mit diesem Kapitel wird eine Einführung in die Konzepte des Business Prozess Managements (BPM) gegeben. Es werden zentrale Begriffe und Vorgehensweisen des BPM diskutiert, z.B. Organisation, Prozess, Geschäftsprozess, End-to-End-Prozess, Prozessarten, Prozesselemente. Ferner wird auf den BPM-Lebenszyklus und dessen Phasen eingegangen. Diese Ausführungen sind Grundlagen für weitere Abschnitte, wie die zur Modellierungssprache BPMN 2.0. Im Abschnitt zur BPM-Technologie werden Informationstechnologien betrachtet, die die Managementdisziplin des BPM unterstützen.

2.1.1 Organisation und Business

Im Deutschen wird der Begriff Organisation häufig auch i.S.v. Struktur oder Ordnung verwandt. Um Doppeldeutigkeiten zu vermeiden, werden die Begriffe Unternehmen und Organisation im Rahmen der vorliegenden Arbeit synonym verwandt. Außerdem gelten alle Aussagen auch für nicht-gewinnorientierte Einrichtungen, wie z.B. gemeinnützige Vereine und öffentliche Institutionen.

Nach DIN EN ISO 14001 wird Organisation folgendermaßen definiert:

„Gesellschaft, Körperschaft, Betrieb, Unternehmen, Behörde oder Institution oder Teil oder Kombination davon, eingetragen oder nicht, öffentlich oder privat, mit eigenen Funktionen und eigener Verwaltung. ANMERKUNG: Bei Organisationen mit mehr als einer Betriebseinheit kann eine einzelne Betriebseinheit als Organisation definiert werden.“ (DIN EN ISO 14001:1996)

Im Rahmen dieser Arbeit umfasst der Begriff „Business" alle Formen von For- oder Non-Profit bzw. öffentlichen Organisationen, er kann allgemein mit dem Begriff Organisation gleichgesetzt werden.

Um die Konzepte des Business Process Managements darzulegen, muss auch definiert werden, mit welcher Bedeutung der Begriff Prozess in dieser Arbeit versehen ist, worauf im Folgenden eingegangen wird.

2.1.2 Prozess und Geschäftsprozess

In der Fachliteratur findet man vielfältige Definitionen für Prozess und Geschäftsprozess, auf zentrale Konzepte wird im Folgenden eingegangen.

Prozessdefinitionen

Rosemann, Kugeler und Becker verstehen als Prozess die inhaltlich abgeschlossene, zeitliche und sachlogische Folge von Aktivitäten, die zur Bearbeitung eines betriebswirtschaftlich relevanten Objektes notwendig ist (vgl. Rosemann, Kugeler & Becker, 2008, S. 8).

Nach der Definition des Deutschen Instituts für Normung ist ein Prozess ein „Satz von in Wechselbeziehung stehenden Mitteln und Tätigkeiten, die Eingaben in Ergebnisse umgestalten. ANMERKUNG: Zu den Mitteln können Personal, Finanzen, Anlagen, Einrichtungen, Techniken und Methoden gehören.“ (DIN EN ISO 8402:1995).

Nach der Definition von BPM Common Body of Knowledge wird unter Prozess eine Gruppe von Aktionen (auch Aktivitäten) verstanden, die entweder von Menschen oder von Maschinen erledigt wird und ein oder mehrere Ziele erreicht. Prozesse werden durch Ereignisse ausgelöst und erreichen ein Ergebnis oder mehrere Resultate. Letztere führen entweder zum Ende eines Prozesses oder fließen in einen weiteren Prozess hinein (vgl. EAPBM, 2011, S. 37).

Für Schmelzer besteht ein Prozess aus einer Folge von Aktivitäten, die aus einer Menge von Inputs bestimmte Leistungen erzeugen (vgl. Schmelzer & Sesselmann, 2010, S. 62).

Osterloh und Frost beschreiben ein Prozess als eine Abfolge, d.h. einen Fluss und die Transformation von Material, Informationen, Operationen und Entscheidungen (vgl. Osterloh & Frost, 2006, S. 33).

Definitionen für Geschäftsprozess

Nach Allweyer sind Geschäftsprozesse die zur Herstellung von Produkten und Leistungen notwendigen betrieblichen Abläufe (vgl. Allweyer, Geschäftsprozessmanagement, 2005).

Nach Schmelzer und Sesselmann schließt ein Geschäftsprozess funktions- und organisationsüberschreitende sowie wertschöpfende Aktivitäten ein, die für die Kunden Leistungen erzeugen und die aus der Organisationsstrategie abgeleitete Ziele umsetzen (vgl. Schmelzer&Sesselmann, 2010).

Rosemann, Kugeler und Becker verstehen unter Geschäftsprozess einen speziellen Prozess, der dem Erreichen der obersten Unternehmensziele dient und das Kerngeschäftsfeld beschreibt. Dieser Prozess hat auch Schnittstellen zu den Kunden und Lieferanten (vgl. Rosemann, Kugeler&Becker, 2008, S. 8).

Hammer und Champy verstehen unter Unternehmensprozess eine Menge von Aktivitäten, die Inputs benötigen und für den Kunden einen Nutzen generieren (vgl. Hammer&Champy, 1994).

Als Kernaussage aller Definitionen ist zu rekonstruieren, dass ein Geschäftsprozess ein besonderer Prozess ist, der einen Nutzen für die Kunden - und dadurch auch Unternehmensgewinn - generiert.

Dieser Arbeit wird die Geschäftsprozessdefinition von Gadatsch zu Grunde gelegt, da sie die grundlegenden Einzelelemente detaillierter beschreibt und Informations- und Kommunikationstechnologien berücksichtigt:

„Ein Geschäftsprozess ist eine zielgerichtete, zeitlich-logische Abfolge von Aufgaben, die arbeitsteilig von mehreren Organisationen oder Organisationseinheiten unter Nutzung von Informations- und Kommunikationstechnologien ausgeführt werden können. Er dient der Erstellung von Leistungen entsprechend der vorgegebenen, aus der Unternehmensstrategie abgeleiteten Prozessziele. Ein Geschäftsprozess kann formal auf unterschiedlichen Detaillierungsebenen und aus mehreren Sichten beschrieben werden. Ein maximaler Detaillierungsgrad der Beschreibung ist dann erreicht, wenn die ausgewiesenen Aufgaben je in einem Zug von einem Mitarbeiter ohne Wechsel des Arbeitsplatzes ausgeführt werden können“ (Gadatsch, 2005, S. 43).

In dieser Arbeit werden die Begriffe Prozess und Geschäftsprozess synonym verwendet, nicht zuletzt, weil mit Prozess ein Businessprozess gemeint ist.

„End-to-End“-Prozess

In der Fachliteratur wird zwischen einfachen Tätigkeiten bzw. Aufgaben und „End-to-end"-Prozessen unterschieden. Letztere haben eine zentrale Funktion, weil durch diese eine funktions- und abteilungsübergreifende Prozesssicht geschaffen werden kann, was auch für die Kunden von Nutzen ist. Ist der Verursacher eines Prozesses gleichzeitig der Empfänger des Prozessergebnisses, unabhängig davon welche und wie viele interne Stellen von diesem Prozess berührt sind, so spricht man von einem End-to-End-Prozess. Ein Erkennungskriterium für End-to-End Prozesse kann z.B. sein, dass der Anfang und das Ende eines Prozesses nicht an Abteilungs- oder Organisationsgrenzen liegen, sondern beim Prozessauslöser (vgl. Freund, 2010, S. 2). So können z.B. auch Mitarbeiter eines Unternehmens einen End-to-End Prozess auslösen, wenn sie bspw. einen Urlaub beantragen. Die Bezeichnung „End-to-End“ soll ein gewisses Fallen interner Bereichsgrenzen vermitteln.

2.1.3 Business Process Management (BPM)

Die Bezeichnung Business Process Management kommt aus dem englischen Sprachraum und wird auch als Synonym für Geschäftsprozessmanagement verwendet. Im vorhergehenden Abschnitt ist als Prozess eine Folge von bestimmten Aktivitäten definiert worden, die von Menschen oder Maschinen durchgeführt wird, um ein Ziel zu erreichen. Schließlich geht es darum, einen Kundennutzen zu schaffen und damit auch für das Unternehmen Wert zu schaffen. Business Process Management (BPM) stellt einen systematischen Ansatz zum Erfassen, Gestalten, Ausführen, Dokumentieren, Überwachen und Steuern von nicht-automatisierbaren und automatisierbaren Abläufen dar. BPM beinhaltet auch die bewusste und die IT-unterstützte Bestimmung und Verbesserung sowie den Erhalt von Prozessen (vgl. EAPBM, 2011). BPM ermöglicht es den Unternehmen, schneller und flexibler gute Ergebnisse zu erreichen.

BPM als Managementdisziplin. Abgrenzung zu den unterstützenden Technologien

Das Akronym BPM wird nicht selten, je nach Kontext, mit unterschiedlicher Bedeutung versehen. In der Informationstechnologie wird er von Softwarefirmen verwendet, um die Funktionen eines Produkts zu beschreiben; Manager und Berater nutzen den Begriff, wenn sie von Prozessen und der Managementdisziplin BPM sprechen.

BPM ist zunächst sowohl eine Disziplin des Managements als auch ein Ansatz, um Unternehmensprozesse zu managen. Der Einsatz von Informationstechnologien in BPM soll daher als unterstützend betrachtet werden. Viele Softwareanbieter haben Prozessmanagement-Suiten entwickelt, die das Management von Prozessen vereinfachen sollen. Dazu zählen Tools für die visuelle Modellierung, die Simulation, die Durchführung und die Automatisierung von Prozessen. Auf dem Markt existieren auch BPM-Plattformen, die alle genannten Funktionen beinhalten und als Business Process Management Suiten (BPMS) bezeichnet werden.

Kernaussagen des Business Process Managements

Nach EABPM umschreiben folgende Kernaussagen das Business Process Management allgemein (EAPBM, 2011):

- BPM stellt ein umfangreiches Managementkonzept dar, das durch eine Reihe von modernen IT-Technologien unterstützt wird.
- BPM unterscheidet zwischen Teilprozessen, Aufgaben, Aktivitäten und Funktionen.
- BPM umfasst die Analyse, das Design, die Modellierung, die Umsetzung und die Steuerung von Prozessen in einem Unternehmen.
- BPM erfordert Weisungsbefugnisse, da häufig neue Rollen, Zuständigkeiten und Strukturen eingeführt werden.

2.2 Business Process Management Lebenszyklus

In der Fachliteratur findet man mehrere Definitionen zum BPM-Lebenszyklus und zu den einzelnen Schritten, die jeder Geschäftsprozess in seinem

Lebenszyklus durchläuft. Als Grundlage für diese Arbeit dient die Definition der Europäischen Assoziation für Business Process Management (EAPBM, 2011). Grundsätzlich lassen sich danach folgende Phasen eines Geschäftsprozesses unterscheiden (EAPBM, 2011):

- strategische Prozessorientierung
- Prozessanalyse und Prozessmodellierung
- Prozessentwurf
- Prozesseinführung bzw. Prozessumsetzung
- fortdauernde Prozesssteuerung und -optimierung.

Diese fünf Phasen sind auf in Abbildung 1 grafisch dargestellt. Es sind zudem verschiedene Faktoren aufgeführt, die den Lebenszyklus von BPM unterstützend beeinflussen können. Insbesondere sind die vier Gesichtspunkte Führung, Werte, Kultur und Überzeugungen hervorzuheben. Anschließend wird genauer auf die einzelnen Phasen des BPM-Lebenszyklus eingegangen.

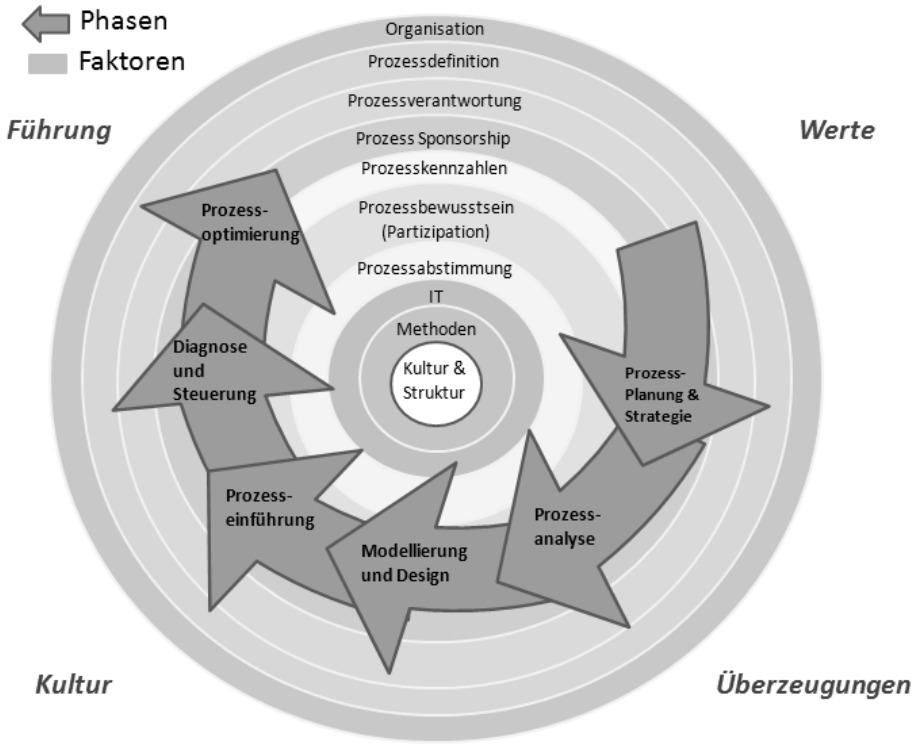


Abbildung 1 Phasen des Prozesslebenszyklus nach EABPM (EAPBM, 2011)

2.2.1 Prozessplanung und Strategie

Der BPM-Lebenszyklus beginnt mit dem Entwurf einer Strategie und Vorgehensplanung, der an einem ganzheitlichen Prozessdenken ausgerichtet ist. Dabei geht es vordergründig darum, dem Kunden durch die Prozessorientierung einen Vorteil anbieten zu können. Die Planung bietet eine gewisse Ordnung und damit eine Ausrichtung auf ein kontinuierliches und kundenorientiertes Prozessmanagement. Sie ist die Grundlage für ein gesamtheitliches BPM-Konzept, das die Orientierung auf die Unternehmensstrategie und die Zusammenführung von Strategie, Mensch, Prozess und System über die Bereichsgrenzen hinweg sichern soll. Ferner werden in diesem Schritt passende Rollen definiert und Zuständigkeiten geregelt sowie die Unterstützung durch die Unternehmensführung gewährleistet (EAPBM, 2011).

2.2.2 Prozessmodellierung und -analyse

Um einen Prozess verständlich zu machen, ist es hilfreich, ihn zu modellieren und die Faktoren zu ermitteln, die den Prozess positiv oder negativ beeinflussen können. Erfahrungsgemäß werden Unternehmen, die in Bezug auf BPM wenig erfahren sind, den gesamten Prozess dokumentieren. Dagegen werden sich Unternehmen, die in der Modellierung von Prozessen bereits Erfahrungen gesammelt haben, eher auf Sonderfälle oder Ausnahmen zu den Regelprozessen konzentrieren (Schmidt, Goetz, 2011). In der Untersuchung von Prozessen werden verschiedene Arten und Techniken mit dem Ziel eingesetzt, den IST-Zustand von Prozessen zu erfassen und diesen auf Konformität mit den erstrebten Zielen und Ergebnissen abzufragen. Das beinhaltet auch die Auseinandersetzung mit dem IST-Zustand und umfasst zudem, die Ansprüche aller Interessenvertreter zu ermitteln, bspw. Kunden, Mitarbeiter, Lieferanten, Manager. Hierzu bezieht man Informationen aus der Planung und aus Prozessmodellen, Kennzahlen aus Leistungsmessungen sowie Daten über Umweltveränderungen ein, um ein umfassendes Verständnis des Prozesses im Kontext des Unternehmens zu bekommen.

2.2.3 Prozessdesign

Beim Prozessdesign wird auf die planmäßige Gestaltung von Prozessen geachtet, wobei der zu erbringende Nutzen für den Kunden fokussiert wird. In dieser Phase werden die Abfolge von Prozess-Schritten und deren Inhalte festgelegt (Verantwortlichkeit, Zeit, Ort, verwendete Techniken). Es erfolgt eine Ausgestaltung des Prozesses hinsichtlich der Fragen wer agiert wann, wo, und wie im Prozess. Eine ebenfalls wichtige Komponente des Designs ist es, geeignete Prüfmechanismen zu definieren und Kennziffern für die Leistungsmessung zu erfassen.

2.2.4 Prozessumsetzung und -einführung

In der Phase der Prozessumsetzung und -einführung werden die im Design festgelegten Lösungen umgesetzt. Zu dieser Phase gehören daher die Entwicklung und der abschließende Test automatisierter bzw. manueller Prozesse aber bspw. auch Mitarbeiterschulungen und das Informieren aller Beteiligten. Auch kritische Aufgaben des Change Managements sind in dieser Phase zu bewältigen. Damit die Lösung erfolgreich funktioniert, ist die Überzeugung aller Interessenvertreter von der Qualität von großer Bedeutung.

2.2.5 Kontinuierliche Prozesssteuerung und -optimierung

Die letzte Phase des BPM-Lebenszyklus beinhaltet einerseits die Prozessverbesserung und andererseits die ständige Prozessoptimierung. Optimierte Prozesse erreichen langfristig die Zielsetzungen bezüglich Effizienz und Effektivität, unabhängig von Umwelteinflüssen. Das fortlaufende Messen und Monitoring von Prozessen liefert Prozessmanagern die nötigen Informationen, um Ressourcen anzupassen und Prozessziele zu erreichen. So werden im Laufe des BPM-Lebenszyklus wichtige Informationen über ein Unternehmen gewonnen, die zu einer kontinuierlichen Optimierung der Prozessgestaltung beitragen.

2.3 Prozessarten

Im Fachdiskurs werden vorrangig folgende Formen von End-to-end-Geschäftsprozessen unterschieden (vgl. Gadatsch, 2005, S. 48):

- Führungsprozesse
- Ausführungsprozesse
- Unterstützungsprozesse

2.3.1 Führungsprozesse

Führungsprozesse dienen der Umsetzung der Ausführungs- und Unterstützungsprozesse hinsichtlich der Erreichung gesetzter Ziele (Gadatsch, 2005). Sie tragen somit indirekt zur Wertschöpfung bei und stellen sicher, dass die Organisation effektiv und effizient funktioniert (EAPBM, 2011).

2.3.2 Ausführungsprozesse

Die Wertschöpfungskette eines Unternehmens ist eine Perspektive auf die Prozesse, die Nutzen für die Kunden generieren (Gadatsch, 2005). Gesamtprozesse bestehen aus Teilprozessen, die eigene, aus den übergeordneten Elternprozessen abgeleitete Leistungsziele haben. Wertschöpfungsketten reichen über mehrere Abteilungen; sie können auch mehrere Unternehmen überspannen und eine ganzheitliche Perspektive über die Wertschöpfung bieten. Die Ausführungsprozesse schaffen einen unmittelbaren Nutzen für den Kunden; sie sind funktionsübergreifend und nicht selten bereichs- und unternehmensübergreifend. Aus diesem Grund werden sie auch primäre Prozesse genannt, sie fassen die für den Unternehmenserfolg erforderlichen Tätigkeiten zusammen (Gadatsch, 2005). Jede Phase des Primärprozesses trägt dazu bei, Produkte oder Dienstleistungen zu erschaffen und damit auch einen Nutzen für den Kunden zu generieren.

2.3.3 Unterstützungsprozesse

Damit die Ausführungsprozesse erfolgreich und fortlaufend durchgeführt werden können, wird auf Ressourcen und eine Infrastruktur zurückgegriffen, die von den Unterstützungsprozessen bereitgestellt werden. Somit sind die Unterstützungsprozesse indirekt an einer Wertschöpfung für die Kunden beteiligt. Beispiele für Unterstützungsprozesse sind IT-Administration, Gebäudeverwaltung und Buchhaltung. Diese Unterstützungsprozesse haben ihre eigenen Lebenszyklen und sind an funktionale Organisationseinheiten gebunden. Dennoch überspannen diese Prozesse nicht selten die Grenzen einzelner Organisationseinheiten (Gadatsch, 2005).

Wenngleich sie nur mittelbar an der Wertschöpfung für die Kunden beteiligt sind, leisten Unterstützungsprozesse doch einen wesentlichen Beitrag für die Organisation, da sie einen unmittelbaren Einfluss auf die Effektivität von Primärprozessen haben.

2.4 Technologien zur Unterstützung des Business Process Managements

Für Nutzer des BPM gewinnen in den letzten Jahren zunehmend Anwendungen an Bedeutung, die bei der Analyse, dem Entwurf, der Umsetzung, der Ausführung, dem Management und dem Monitoring von Geschäftsprozessen hilfreich sein können (Schmidt, Goetz, 2011). Die neuen Konzepte des Geschäftsprozessmanagements haben auch zu neuen Anforderungen an die IT geführt. Dieser Abschnitt gibt einen Überblick über die Besonderheiten von Softwareanwendungen, die Unterstützungsfunktionen für die in den Prozess involvierten Mitarbeiter bieten. Der BPM-Lebenszyklus kann eine Reihe von aufwändigen Tätigkeiten beinhalten, entsprechend sind die Ansprüche an die Softwaresysteme, die unterstützend eingesetzt werden, größer geworden. Studien über den Erfolg im Geschäftsprozessmanagement haben die zentrale Rolle der IT-Anwendungen für BPM herausgestellt. Die Verwendung von IT-Technologien ist immer dann besonders wirksam, wenn die Komplexität der Aufgaben, die bewerkstelligt werden sollen, zu groß ist, um manuell erledigt zu werden. Prozessautomatisierung ist ein Thema, das immer mehr an Bedeutung bei mittleren bis großen Unternehmen gewinnt, insbesondere wenn es sich um

die Koordination und Kooperation zwischen Mitarbeitern, die geografisch weit voneinander entfernt sind, handelt. Die Automatisierung von Prozessen kann wichtige Effizienzsteigerungen schaffen, da dadurch Zeit und Kosten eingespart werden können und Zeitverspätungen, die zwischen den einzelnen Schritten in einem Prozess liegen, stark verringert werden.

Im Geschäftsprozessmanagement können Leistungen gemessen werden, die bei der Optimierung der Prozesswerte hilfreich sind und die eine Grundlage für Managemententscheidungen bieten können. Je komplexer diese Erfolgsmessgrößen werden, d.h. je größer die Datenmengen und die Zahl der Quellen sind, desto weniger kann auf IT-Anwendungen verzichtet werden.

2.4.1 Bestandteile der BPM-Technologie

In diesem Abschnitt werden Software-Systeme benannt und betrachtet, die zur Unterstützung von Prozessmanagement-Aktivitäten eingesetzt werden können.

Der Schwerpunkt liegt dabei auf Applikationen, die nachfolgende Leistungen ganz oder auch eingeschränkt unterstützen oder automatisieren:

- Modellierung, Analyse und Entwurf von Prozessen
- Umsetzung und Ausführung von Prozessen

Die Anwendungen können auf einzelne Teilaufgaben ausgerichtet sein, die die Leitung von Geschäftsprozessen unterstützen oder eine Reihe Geschäftsprozessmanagement-Aktivitäten gleichzeitig abdecken (oft auch als Software-Suiten bezeichnet). Die folgende Abbildung führt einige Software-Arten auf, die im Kontext von Prozessmanagement eingesetzt werden.

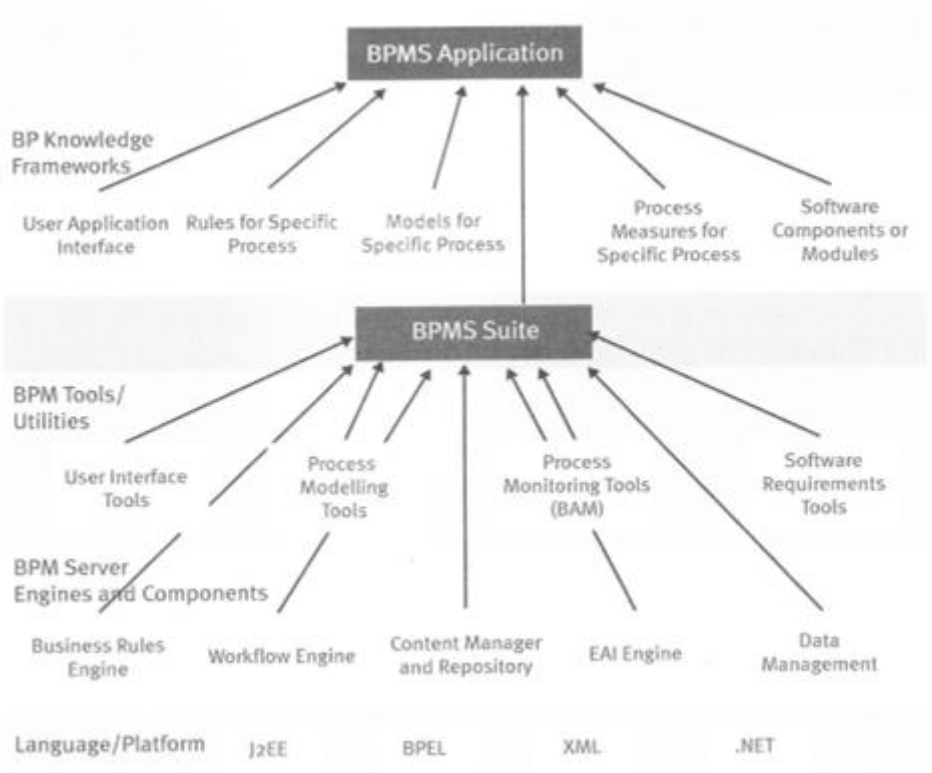


Abbildung 2 Software-Komponenten zur Unterstützung von Prozessmanagement-Aktivitäten (EAPBM, 2011)

Anwendern werden verschiedene Anwenderschnittstellen zur Verfügung gestellt (User Interface Tools); außerdem gibt es mehrere Ebenen von Server-Software, die von den Anwendern nur indirekt wahrgenommen werden können. Dazu zählen z. B. Workflow Engines, Business Rules Engines, Datenmanagement Systeme. Für die Entwicklung von Software-Systemen für das Geschäftsprozessmanagement werden Programmiersprachen, ganze Frameworks oder Plattformen eingesetzt. Beispiele dafür sind die Plattformen J2EE, NET, XML. In der Übersicht und den folgenden Ausführungen können nicht alle Entwicklungs- und Ausführungsplattformen und Werkzeuge aufgeführt werden. Es wurde wert auf einen repräsentativen Überblick gelegt, d.h. auf die Aufführung weitverbreiteter Anwendungen.

2.4.1.1 Modellierung, Analyse und Entwurf

Business Process Modeling and Analysis (BPMA) beinhaltet die Konzeption und Beschreibung eines Prozesses, wobei Prozessmodelle und verschiedene Szenarien oder Alternativprozesse entworfen werden. Software-Tools, die der

BPMA-Unterstützung dienen, beinhalten Werkzeuge, die eine grafische Darstellung einschließlich der detaillierten Beschreibungen der Ziele der Prozesse unterstützen (EAPBM, 2011). Das Skizzieren des Ablaufs eines Prozesses gehört zu den ersten Schritten bei der Prozessentwicklung. Die Darstellung der End-to-End-Prozesse ist ein wesentlicher Teil des BPM-Lebenszyklus, der notwendig ist, wenn Prozesse entwickelt werden sollen, die die betrieblichen Anforderungen erfüllen und die ferner in der Lage sind, die Anforderungen an die Umsetzung detailliert abzubilden. Auf einer einfachen Ebene sind sämtliche Grafikanwendungen sinnvoll, die eine Visualisierung und Beschreibung des Prozessablaufs möglich machen. Ablaufdiagramme bieten die Möglichkeit, einzelne Prozess-Schritte in Form von Feldern darzustellen. Wichtig ist die Unterstützung von Anmerkungen, die z.B. Rollen und detaillierte Beschreibungen zu jedem einzelnen Prozess-Schritt beschreiben. Obwohl Flussdiagramme auch per Hand auf Papier erstellt werden können, bieten Computerprogramme die bessere Alternative für eine einfache Bearbeitung, Korrektur und elektronische Verteilung der dokumentierten Prozesse. Die Modellierung von Prozessen liefert gleichzeitig die Dokumentation, die für die Weiterkommunikation der Prozesse an die Unternehmensführung, Mitarbeiter, Prozessdesigner und Entwickler nützlich ist. Einige Anwendungen behandeln Diagrammelemente als intelligente Objekte, so dass durch einen Mausklick für einen Prozessschritt relevante Daten dargestellt werden. Das können unter anderem detaillierte Aktivitätsbeschreibungen sein aber auch bei der Bearbeitung erforderliche Informationsquellen, Regeln, Einweisungen usw. Das im Rahmen dieser Arbeit betrachtete und erweiterte System QUAM 2.0 ist ein Beispiel für eine solche Anwendung, da QUAM eine nahezu unbegrenzte Attributierung einzelner Prozesselemente anbietet. Alle Informationen über Elemente und Attribute, die durch den Mausklick abgerufen werden, werden in QUAM in einer zentralen Datenbank abgespeichert und mit den Objekten auf dem Diagramm verknüpft. Dies wird durch die zwei Technologien ermöglicht, die dem QUAM System zugrunde liegen. So erlaubt bspw. Visio die objektorientierte Behandlung einzelner Diagrammobjekte, zu denen Eigenschaften und Funktionen hinzugefügt und visualisiert werden können. Auf

der anderen Seite erlaubt SharePoint eine umfangreiche Darstellung des Unternehmensdatenmodells.

Eine weitere Stufe der Evolution der BPM-Technologie sind die IT-Anwendungen, die eine Simulation von Prozessmodellen erlauben. Die Messwerte, die zur Leistungsmessung entwickelt werden, werden in einen Simulationslauf einbezogen, damit die Effizienz von Prozessen gemessen werden kann. Die Modellierung und Simulation stellen immer wiederkehrende Tätigkeiten dar, so dass für eine Reihe von Annahmen ständig ermittelt wird, ob sie gültig sind. Im Laufe der Simulation werden Kennzahlen (wie z. B. die benötigte Durchlaufzeit oder die Kosten) bestimmt, um letztendlich Indikatoren für Verbesserungen zu ermitteln.

Zu den grundlegenden Funktionen einer typischen Modellierungs- und Simulationsanwendung gehören:

• Die grafische Darstellung der Prozesse als Prozesslandkarte und die Darstellung der einzelnen Prozesse

• Die Definition des Informationsflusses zwischen den einzelnen Prozessschritten und die Bestimmung der Bedingungen, unter denen sich dieser Fluss ändern kann.

2.4.1.2 Umsetzung von Geschäftsprozessen

Ist ein Prozess bereits entworfen, kann zur Prozessimplementierung eine Reihe von Anwendungen genutzt werden. Zu den wichtigsten Anwendungen zählen nach EABPM insbesondere:

- Dokumentenmanagementsysteme, die der Erfassung, Organisation und Verwaltung von Informationen dienen
- Workflowmanagement-Systeme
- Informationssysteme, die eine Zusammenarbeit von Arbeitsgruppen ermöglichen

Dokumentenmanagementsysteme

Alle Geschäftsprozesse nutzen Informationen aus Dokumenten, die im System gespeichert sind, sei es als Office-Dokumente im Dateisystem oder in einer

Datenbank. Anwendungen zur Unterstützung der Erfassung und der Verwaltung von Informationen in elektronischer Form schaffen eine wichtige Voraussetzung für die Entwicklung von Geschäftsprozessmanagement-Systemen. Die oben betrachteten Systeme zur Unterstützung von Prozessen ermöglichen den Mitarbeitern eine Nutzung durch „Pushing“ oder „Pulling“. „Push“-Methoden schließen das Senden von Informationen an eine Person ein, wobei dadurch eine Aufgabe erzeugt wird. Ein Beispiel dafür ist das Senden einer E-Mail an einen Mitarbeiter, durch die eine Aufgabe ausgelöst wird. Bei den „Pull"-Methoden wählen die Mitarbeiter die Aufgaben aus einem Aufgabenpool aus. Hier spielt die korrekte Klassifizierung der Informationen eine wichtige Rolle, damit die Mitarbeiter schnell die benötigten Informationen finden. Anwendungen zur Metadatenverwaltung und Informationssuche, einschließlich Volltextsuche, sind hier besonders zweckmäßig.

Applikationen, die eine umfassende Verwaltung von Unternehmensinformationen ermöglichen, sind auch für die Verwendung als Prozess Repositories hilfreich. Dabei werden Prozessrichtlinien, Regelungen und Kompetenzen in elektronischer Form erfasst und zur Verfügung gestellt.

Workflow Systeme

Liegen die Daten aus einem Prozess in strukturierter Form vor, können diese von einem anderen System benutzt werden, z.B. von einem Workflow-System (Gadatsch, 2005). Bei der Automatisierung von Arbeitsabläufen werden Systeme involviert, die für die einzelnen Tätigkeiten in einem Prozess die notwendigen Informationen bereitstellen. Das Workflow-System steuert den Prozess und die Regeln zur Nutzung und Weitergabe von Informationen. Dabei kann es sich um komplexe oder einfache Regeln handeln, bspw. um das Weiterleiten von Rechnungs-, und Inventurinformationen an eine autorisierte Stelle, z.B. die Geschäftsführung. Abbildung 3 zeigt das Prinzip der Workflowautomatisierung.

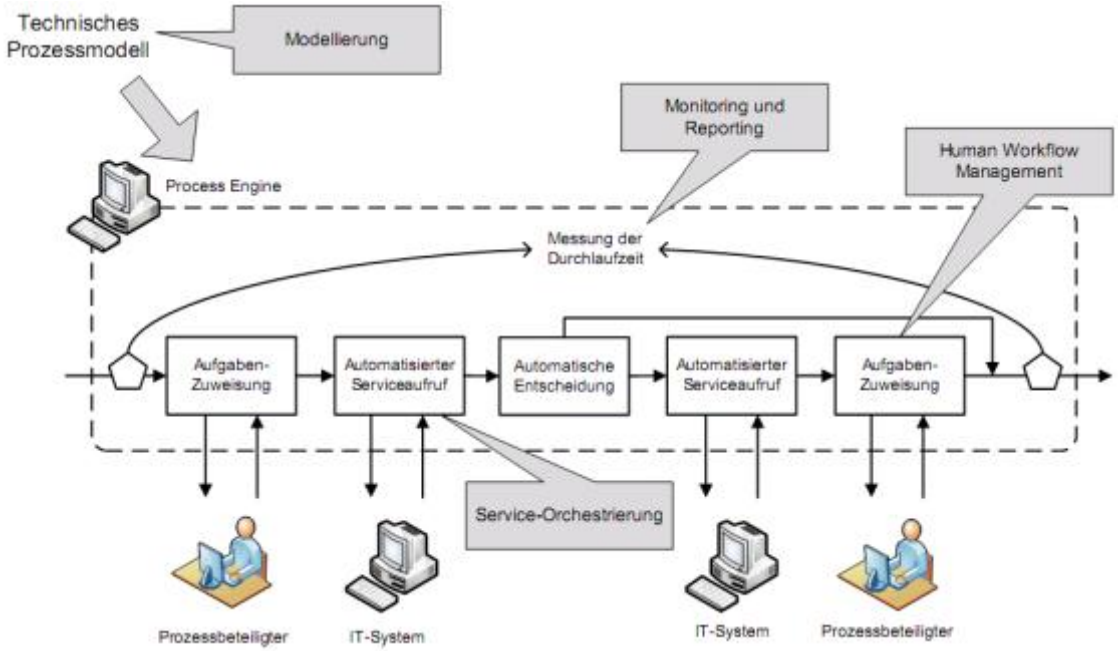


Abbildung 3 Prinzip der Process Engine (Freund & Götzer, 2008)

Viele Systeme zur Automatisierung von Arbeitsabläufen bauen auf elektronische Dokumentenmanagementsysteme auf, um Daten, die von diesen Systemen verwaltet werden, an die in einen Prozess involvierten Mitarbeiter weiter zu leiten.

Wichtige Merkmale eines Workflow-Systems sind nachfolgend aufgelistet:

• Das Zur-Verfügung-Stellen einer „Inbox" und „Outbox" für den Anwender. Alle Prozessbeteiligten finden ihre Aufgaben in einer elektronischen Inbox, in der sämtliche Informationen abgespeichert werden, die für die Aufgabe von Relevanz sind. Wenn die Aufgabe erledigt ist, werden die aus diesem Schritt resultierenden Informationen in eine Outbox gesetzt und an einen anderen Mitarbeiter oder eine andere Maschine weitergeleitet, um den nächsten Prozessschritt auszuführen. Werden bestimmte kritische Aufgaben bis zu einem Termin nicht erledigt, kann z.B. ein Alarm ausgelöst und an zuständige Mitarbeiter weitergeleitet werden. Die Möglichkeit zur Rollenfestlegung sichert eine Verwaltung von Zuständigkeiten und Befugnissen. Typische Anwenderrollen sind z. B. Dateneingaben, Dokumentenüberprüfung, Einholung von Genehmigungen (EAPBM, 2011).

• Workflow-Systeme können Regeln bestimmen, mit denen der Prozess gesteuert wird, diese sind auch unter den Namen „rule Engines“ bekannt.

Solche Systeme können zum einen einfache Informationen weiterleiten, die auf spezifischen Kennzahlen basieren (z. B. Rechnungs- oder Umsatzbeträge). Zum anderen bieten sie eine komplexe „Regelsprache", die eine Fülle von Bedingungen überprüfen kann und einen Algorithmus ausführt (EAPBM, 2011).

Anwendungen für Arbeitsgruppen-Zusammenarbeit

Erfahrungen zeigen, dass einer der wichtigsten Erfolgsfaktoren für Projekte die Beteiligung bzw. das Zusammenwirken aller involvierten Mitarbeiter ist. Anwendungen zur Unterstützung der Zusammenarbeit bei der Entwicklung, Ausführung und Verwaltung von Prozessen werden oft unter dem Oberbegriff kollaborative Systeme zusammengefasst. Besonders wichtig sind solche Anwendungen aufgrund der geografischen Verteilung der an dem Prozess beteiligten Mitarbeiter. Sie stellen Funktionen bereit, die die Zusammenarbeit verschiedener Anwender auf unterschiedlichen Rechnern ermöglichen. Typische Workgroup-Applikationen sind Tools zur elektronischen, meistens webbasierten, Kommunikation. Konferenz-Tools und Management-Tools gehören auch zu der Kategorie der Workgroup-Applikationen (Allweyer, 2009).

2.4.2 Vorteile der Prozessautomatisierung

Die Automatisierung des gesamten Prozesses oder der Teile eines Prozesses kann bedeutsame Effizienzsteigerungen bewirken. Diese lassen sich durch die Unterstützung von Aktivitäten erreichen, wie z. B. der Verwaltung großer Mengen von Dokumenten und Daten oder der Reduzierung von Wartezeiten bei Ablauf kritischer Aktivitäten. Effizienzsteigerungen führen auch zu einer Verringerung der Kosten. Außerdem können einfache Prozesse mit Hilfe von grafischen Tools umgesetzt werden, ohne dass dabei ein zusätzlicher Programmieraufwand entsteht. Beispiel dafür ist die Definition von einfachen Workflows im SharePoint Designer 2010, bei denen ein Mitarbeiter einer Fachabteilung einen Ablauf definieren kann. Prozessmanagement-Systeme haben zudem den Vorteil, dass durch ihren Einsatz gesetzliche Regelungen eingehalten werden können. Zum Beispiel lässt sich durch ein solches System sicherstellen, dass regelmäßige Kontrollen durchgeführt wurden.

In diesem Kapitel wurde eine Einleitung zu den Konzepten des Business Prozess Managements gegeben. Zentrale Begriffe und Konzepte des BPM, wie End-to-End-Prozess, Prozessarten und Prozesselemente wurden eingeführt und diskutiert. Anschließend wurde eine Vorstellung des BPM-Lebenszyklus und seiner Phasen gegeben.

Im nächsten Abschnitt findet eine Vorstellung verschiedener Modellierungssprachen. Diese werden anschliessend nach bestimmten Kriterien untersucht. Daraufhin wird die Auswahl einer Modellierungssprache, die implementiert werden soll getroffen. Das darauffolgende Kapitel beschäftigt sich mit der ausgewählten Modellierungssprache, die BPMN 2.0. Dabei wird sie aus Anwenderperspektive betrachtet - eine Betrachtung der BPMN aus implementierungstechnischer Perspektive erfolgt im Kapitel über die technischen Grundlagen.

2.5 Auswahl einer zu implementierenden Modellierungssprache

Gegenwärtig existiert eine Reihe unterschiedlicher Sprachen zur Prozessmodellierung. Zu den Sprachen mit großer Verbreitung zählen - neben BPMN - EPK, UML, XPDL und BPEL. Dieser Abschnitt beschäftigt sich mit der Darstellung und der anschließenden Auswahl einer dieser Sprachen, die in QUAM 2.0 implementiert werden soll. Dabei werden die Sprachen nach bestimmten Kriterien, wie z.B. Verbreitungsgrad, unterstützte Modellierungsphasen, Standardisierung usw. bewertet.

2.5.1 Einzelne Modellierungssprachen

2.5.1.1 Ereignisgesteuerte Prozesskette (EPK)

Die ereignisgesteuerte Prozesskette wurde im Jahr 1992 entwickelt und ist ein Teil der ARIS-Methodik. Auf Basis von EPK wurde das ARIS-Tool-Set entwickelt, das eng in die ERP-Lösungen von SAP integriert wird. Dies hat dazu geführt, dass in SAP-Produkten umgesetzte Prozesse als EPK dokumentiert wurden, was als ein wichtiger Grund der großen Verbreitung angesehen wird

(Freund, et al., 2010). Inzwischen bietet auch das Toolset ARIS eine Prozessmodellierung auf Basis von BPMN 2.0.

Die EPK besteht aus den drei Grundsymbolen Funktion, Ereignis und Konnektor. Konnektoren können als exklusive Verzweigung (XOR), Und-Oder-Verzweigung (OR) und als Parallelisierung (AND) wirken. Eine Unterscheidung zwischen daten- und ereignisbasierten Verzweigungen gibt es in der EPK nicht. In einer erweiterten EPK-Fassung sind weitere Symbole vorgesehen, z.B. zur Darstellung von Organisationseinheiten, Daten und Anwendungssystemen. Über sog. Prozesspfade wird auf Teilprozesse verwiesen (Freund, et al., 2010). Die im Bereich von fachlich orientierter Prozessmodellierung eingesetzten ereignisgesteuerten Prozessketten werden insbesondere in dem Modellierungswerkzeug ARIS und im Umfeld von SAP-Einführung angewendet. Dabei ist anzumerken, dass es sich bei der EPK nicht um einen wirklichen Standard handelt und sie hauptsächlich im deutschsprachigen Raum Verbreitung findet (Freund, et al., 2010). Zusammenfassend lässt sich folgende Einschätzung treffen:

- Die EPK ist sehr ausdrucksstark und umfassend. Es stehen sehr viele Modellierungsartefakte zur Auswahl, mit denen die verschiedenen Gesichtspunkte von Prozessen dargestellt werden können.
- Die EPK enthält Symbole für organisatorische und für prozessrelevante Aspekte und ermöglicht somit eine durchgehende Betrachtung von Prozessen.
- Die EPK erlaubt die Integration mit anderen wichtigen Notationen
- Die EPK weist eine weite Verbreitung in der Praxis auf.

2.5.1.2 UML-Aktivitätsdiagramm

Die UML-Notation bietet insgesamt 14 Diagrammarten an, von denen das Aktivitätsdiagramm zur Beschreibung von Prozessen verwendet werden kann. Die UML wird, wie BPMN, auch von OMG verwaltet und wurde als eine graphische Modellierungssprache zur Spezifikation, Konstruktion und

Dokumentation von Software-Teilen und anderen Systemen entwickelt (vgl. Drawehn & Schneider, 2010, S. 14).

Die Aktivitätsdiagramme der UML wurden von der OMG als Standard für die Software-Architekten zur Spezifikation der zu entwickelnden Anwendungen vorgesehen. Nicht selten werden sie für die Prozessmodellierung verwendet, insbesondere im Kontext von IT-Projekten, z.B. in der Erarbeitung von Soll-Prozessen. Die Diagramme sind aber weiterhin relativ technisch, so dass sie für Mitarbeiter der Fachabteilungen noch immer kompliziert sind (vgl. Bartonitz, 2009). Im Vergleich zu Modellierungskonzepten wie UML sind die EPK „auf eine wohltuende Weise abgehoben. Sie konzentrieren sich auf die Metaebene, den Gesamtzusammenhang des jeweiligen Geschäftsprozesses und sind doch so detailiert, dass mit ihnen ohne Schwierigkeit der Bezug zu weiteren wichtigen Elementen der Unternehmensmodellierung (v.a. Datenbanken, aber auch Oranisationsstrukturen u.a.) hergestellt werden kann" (Staud, 2006). Aufgrund ihres technischen Schwerpunkts wird UML für Mitarbeiter aus Fachabteilungen als ungeeignet angesehen und wird in dieser Diplomarbeit nicht weiter betrachtet.

2.5.1.3 XML Process Definition Language (XPDL)

Die XPDL ist eine XML-basierte Sprache zur Beschreibung von Arbeitsabläufen, die durch menschliche Personen ausgeführt werden (Workflow). Sie ist auch maschinell lesbar und dementsprechend auch ausführbar. Die Sprache wird seit 1993 von der Workflow Management Coalition (WfMC) vorangetrieben und standardisiert. XPDL eignet sich als Austauschformat zwischen verschiedenen Prozessmodellierungswerkzeugen; aber auch in Simulationswerkzeugen und Workflow Management Systemen kann XPDL Verwendung finden. Da der Standard durch seine Erweiterungsmöglichkeiten weniger restriktiv ist, z.B. im Vergleich zu BPEL, können proprietäre Erweiterungen (extensions) entstehen, die den Austausch von XPDL-Modellen stark einschränken (vgl. Rowley, 2009). Der Akzent von XPDL liegt in der speicherbaren Repräsentation von Prozessmodellen. Zu diesem Zweck ist XPDL graphenorientiert, während der nächstbetrachtete

Standard - BPEL - einen blockorientierten Ansatz verfolgt. Die graphenorientierten BPMN-Modelle können zudem besser in XPDL gespeichert werden als in BPEL (vgl. Bartonitz, 2009).

2.5.1.4 Business Process Execution Language

Die Business Process Execution Language (BPEL) ist eine XML-basierte Sprache zur Modellierung von Prozessen. Die Beschreibung wird als Webservice bereitgestellt und kann als solcher verwendet werden. Die Sprache ist im Jahr 2002 von IBM, BEA Systems und Microsoft eingeführt worden (vgl. Fischer, 2007). XPDL und BPEL haben unterschiedliche Schwächen. Während der XPDL Komponenten fehlen, die für eine vollständige Ausführung sorgen (z.B. Exception Handling, Asynchronität), fehlen der BPEL Komponenten der grafischen Positionen und Werte für die Simulation. Sowohl XPDL als auch BPEL gelten, genauso wie UML, als rein technische Standards und sind für die fachliche Geschäftsprozessmodellierung ungeeignet. Diese drei Standards werden deshalb in dieser Arbeit nicht weiter betrachtet.

2.5.1.5 Die Modellierungssprache BPMN 2.0

In diesem Abschnitt wird die Business Process Model and Notation (BPMN) näher erläutert. Sie ist derzeit sehr aktuell, da sie zwei wichtige Aufgaben in der Entwicklung von BPM-Systemen zu erfüllen versucht. Auf der einen Seite soll sie die Kommunikationslücke zwischen der Fach- und der IT-Welt schließen, auf der anderen Seite bietet sie in ihrer letzten Version Elemente, die die direkte Ausführung von Modellen ermöglichen sollen. BPMN ist ein Standard, der durch die Business Process Management Initiative (BPMI) verabschiedet wurde. Die BPMI ist eine Vereinigung von Toolanbietern, die mit der IT-Standardisierungsorganisation Object Management Group fusioniert hat (vgl. BPMI, 2006). Neben der Ereignisgesteuerten Prozesskette ist BPMN momentan eine der am weitesten akzeptierten Prozessmodellierungssprachen. Nach Ansicht des Fraunhofer Instituts wird die Bedeutung von BPMN weiter zunehmen (vgl. Drawehn & Schneider, 2010, S. 220). Die mit der neuen

Version hinzugekommenen Änderungen der BPMN sind (vgl. OMG, Business Process Model and Notation, 2010):

- Entwicklung eines XML-Austauschformats
- Entwicklung einer Ausführungssemantik
- Erweiterung der Symbolmenge
- Erweiterung der verfügbaren Diagramme

Eine Gegenüberstellung der betrachteten Modellierungsstandards erfolgt in Tabelle 1.

	Verbreitung	**Austauschformat**	**Ausführungs-semantik**	**Weiterentwick-lung vorgesehen**	**Eignung für fachliche Modellierung**
EPK	Sehr hoch	EPML	Unklarheit, ob EPML ausgeführt werden kann	Ja	Ja
UML	Vorwiegend im technischen Umfeld	Ja	Nein	Ja	Nein
XPDL	Vorgesehen für Modellierung von Workflows	Ja	JA, mit Einschränkungen	Ja	Nein
BPEL	Vorgesehen für Orchestrierung von WebServices	Nein	Ja, mit Einschränkung	Nein, der BPEL Standard hat den Status „complete“	Nein
BPMN 2.0	Sehr hoch	Ja	Ja	Ja, Vorschläge für BPMN 2.1 liegen vor	Ja

Tabelle 1 Gegenüberstellung einzelner Modellierungsstandards

Aufgrund der Überlegenheit der EPK und BPMN gegenüber den anderen Modellierungssprachen wird der Vergleich zwischen diesen beiden Sprachen fortgeführt.

2.5.2 Gegenüberstellung der Modellierungssprachen anhand von Vergleichskriterien

2.5.2.1 Unterstützte Phasen der Modellierung

In diesem Abschnitt werden die unterschiedlichen Ebenen bei der Prozessmodellierung abgehandelt.

2.5.2.1.1 Ebene der Geschäftsprozessmodellierung

In der Phase der Geschäftsprozessmodellierung werden diejenigen Informationen erhoben und dargestellt, die zur Identifizierung, Optimierung und Ausführung von betrieblichen Vorgängen notwendig sind. Dazu werden vorwiegend semi-formale grafische Beschreibungssprachen, kombiniert mit textuellen Ergänzungen, verwendet, die eine gemeinsame Diskussionsbasis für Berater, Mitarbeiter der Fachabteilungen und Unternehmensführung bilden und die Kommunikation zwischen diesen ermöglichen (Wittges, 2005). In dieser Phase liegt der Fokus auf betrieblichen Arbeitsvorgängen. Diese reichen von manuell ausgeführten Aufgaben (z.B. Ware im Lager sortieren) über teilautomatisierte Vorgänge (z.B. Wareneingang erfassen) bis hin zu vollautomatisierten Prozessen (Benachrichtigung, wenn Lagerbestand zu niedrig ist). Somit hat die Geschäftsprozessmodellierung die Aufgabe, ein gemeinsames Verständnis für den Geschäftsprozess aus fachlicher Perspektive zu schaffen (Wittges, 2005).

Fokus bei der Geschäftsprozessmodellierung sind „normale“ Prozessabläufe (sog. Happy Paths), um bereichsübergreifende Beziehungen deutlich zu machen und ggf. hinsichtlich Ausgaben, Zeit und Qualität verbessern zu können (Morelli, 2010). Typisch für die in dieser Phase erzeugten Modelle ist ein geringer Detaillierungsgrad. Gespräche über übergeordnete, relevante Fachbegriffe sind erfolgreicher als Verbindungen mit Datenstrukturen und Modellen (Morelli, 2010).

Sowohl die EPK als auch die BPMN 2.0 sind für die Phase der Geschäftsprozessmodellierung geeignet. Insbesondere die EPK werden in Fachkreisen als intuitiver für die Modellierung von Geschäftsprozessen angesehen, nicht zuletzt deshalb, weil die BPMN mit ihren über 150 möglichen Ausprägungen der Modellierungssymbole als sehr komplex gilt. Um gegen diese Meinung argumen-

tieren zu können und weil aktuelle Workflow Systeme noch nicht alle Neuerungen beherrschen, hat die Trägerorganisation OMG die sogenannten Konformitätsklassen ins Leben gerufen. Die Klasse „SIMPLE“ z.B. soll mit seinen sieben Modellierungssymbolen besonders an die Bedürfnisse der Fachabteilungen angepasst sein (Bartonitz, 2010).

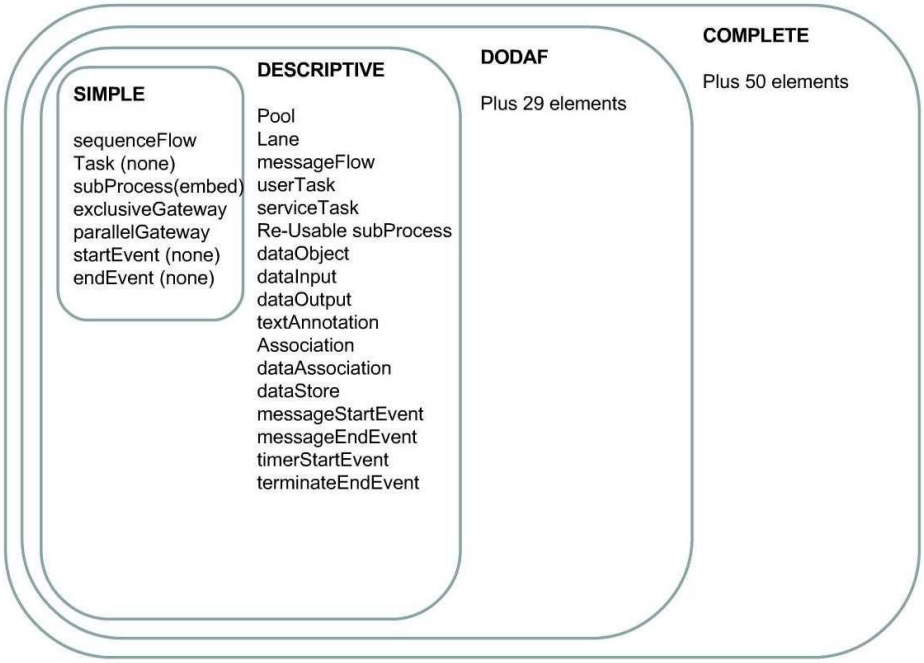


Abbildung 4 Konformitätsklassen von BPMN 2.0 (OMG, 2010)

2.5.2.1.2 Ebene der Workflowmodellierung

In dieser Phase wird das Workflow-Modell erzeugt. Hierbei handelt es sich um eine abstrakte, für die automatisierte Ausführung optimierte Abbildung des betrieblichen Prozesses. Ausgangspunkt dafür ist das Geschäftsprozessmodell der vorhergehenden Phase. Auf dessen Basis werden die Aktivitäten des Workflows definiert und mit zusätzlichen Informationen versehen (Wittges, 2005).

Die in dieser Phase erzeugten Dokumente und Daten sollen den Workflow-Entwickler in die Lage versetzen, den Workflow zu implementieren. Workflow-Entwickler verfügen im Wesentlichen über informationstechnologisches Wissen. Der Fokus liegt häufig einzig und allein auf Prozessen, die sich automatisieren lassen. Dabei lassen in der Praxis Workflow-Entwickler häufig organisatorische

Fragestellungen (z.B. die Dauerhaftigkeit der betroffenen Aufbau- und Ablauforganisation) außer Acht. Informationen über Fehler oder relevante Ausnahmen und Sonderfälle werden als wichtig für die Machbarkeit bewertet.

In der Phase der Workflow-Modellierung ist der Detaillierungsgrad der Modelle hoch und reicht bis hin zur Handhabung von Service Calls, Erzeugen von Workflow-Masken oder Datenmappings. Folglich handelt es sich dabei um eine Fülle von Detailinformationen, wie z.B.:

- Konkrete Anwendungssysteme und ihre Schnittstellen
- Organisationseinheiten und Personen, die am Workflow beteiligt sind
- Im Workflow benötigten Daten
- Erforderliche Masken

Freiheitsgrade existieren bei einem Workflow-Modell nur in geringem Maße (Morelli, 2010). Als notwendig für die Umsetzung der Modelle erweisen sich deren Richtigkeit und Ganzheit. Verbunden mit dem Ziel, eine Wiederverwendbarkeit zu erreichen, folgt der Anspruch auf ein hohes Maß an Formalismus und Konsistenz.

Charakteristisch für die Einstellung von Workflow-Entwicklern ist ein analytisches Verständnis für die Lösung eines Problems. Die Intention ist die Schaffung eines optimalen Workflows-Modells, das in einen ausführbaren Code konvertiert werden kann. Ein Erschwernis für Workflow-Modelle liegt in deren Erklärungsbedürftigkeit; betreffende Erklärungen müssen in einer Sprache verfasst werden, die für die Mitarbeiter aus den Fachabteilungen unmissverständlich ist. EPK-Modelle sind in der Regel für eine Überführung in Workflow Management Systeme nicht geeignet, da sie meist stark überarbeitet werden müssen. Auch ist der Schulungs- und Einarbeitungsaufwand vergleichsweise hoch. Die BPMN bietet für die Ebene der Workflowmodellierung die nächste Klasse „DESCRIPTIVE“. Darin enthalten sind zum Beispiel (OMG, 2010):

- Pools und Lanes für das Referenzieren auf Organisationsstrukturen
- Die Unterscheidung der Tasks (Aktivitäten) in Human und Service
- Datenobjekte mit In- und Output-Beziehungen
- Annotation für Beschriftungen

- Datenquellen
- Detaillierung der Events bzgl. Austausch Nachrichten (Message)

2.5.2.1.3 Ebene der Workflow-Implementierung

Workflow-Implementierungen beschreiben die Workflow-Modelle der vorangegangenen Phase in Bezug auf ein konkretes Workflow-Management-System. Überdies werden in dieser Phase auch technische Besonderheiten definiert, wie etwa bestimmte Rechnernamen, Pfadnamen der Workflow-Applikationen sowie deren Parameter, d.h. in dieser Phase findet neben der Übersetzung des fachlichen Workflow Modells in ein technisches Modell auch die Einbettung des Workflows in die System-Umgebung statt. Hier werden außer der Spezifikation der einzelnen Workflow-Aktivitäten auch der Datenfluss sowie der Kontrollfluss beschrieben. Die Verwirklichung dieses Abschnitts wird von Workflow-Entwicklern realisiert. Dabei werden formale Sprachen eingesetzt, die von einer Ausführungsengine interpretiert werden können. Diese Ausführungsengine spielt eine zentrale Rolle im Prozess der Workflow-Implementierung. Alle Daten, die für die Abwicklung benötigt werden, müssen in der Workflow-Implementierung erfasst werden. Charakteristische Informationsobjekte für diese Phase sind (Wittges, 2005):

- Aktivitäten
- Verantwortliche
- Aufgaben
- Masken
- Verbindungen (Kontrollfluss)
- Weiterleitungsregeln
- Bedingungen
- Beschreibungen
- Start- und Endereignisse

Für diese Phase ist die BPMN 2.0 aufgrund des umfangreichen Detaillierungsgrads und der Ausführungssemantik geeigneter als die EPK. Mittlerweile gibt es die ersten BPM-Suiten, die den kompletten BPM-Lebenszyklus, von Geschäftsprozessmodellierung bis Workflowimplementierung und Ausführung, allein mit BPMN 2.0 unterstützen (Götz, 2011).

Zusammenfassend lässt sich sagen, dass es grundlegende konzeptionelle Unterschiede zwischen den EPK und BPMN gibt. Während die EPK die Philosophie verfolgen, dass für die IT- und Fachabteilungen unterschiedliche Modellierungssprachen notwendig sind, versucht die OMG durch die Konformitätsklassen der BPMN 2.0 „eine Sprache für alles“, also für beide Abteilungen, zu liefern.

2.5.2.2 Verbreitungsgrad

Im Folgenden wird der Verbreitungsgrad der beiden Standards betrachtet; er spielt eine wesentliche Rolle bei der Auswahl des Modellierungsstandards. Einige Autoren im deutschsprachigen Raum weisen auf einen höheren Verbreitungsgrad von EPK gegenüber der BPMN hin, gleichzeitig aber auf „extrem starke Wachstumsraten“ bei der Nutzung der BPMN (Morelli, 2010). Bezüglich der Verbreitung von Modellierungswerkzeugen ist seit einigen Jahren eine starke BPMN-Bewegung seitens großer Hersteller wie z.B. IBM (BlueWorks, Business Modeler, Integration Developer) und SAP (SAP Netweaver BPM und „Gravity“) festzustellen (Morelli, 2010).

Wie auf den untenstehenden Abbildungen zu erkennen ist, widerspiegeln sich die obengenannten Tendenzen auch auf der Plattform BPM-Netzwerk.de. Laut Statistiken hat sich die Anzahl der Plattformteilnehmer mit BPMN-Praxiserfahrung in weniger als einem Jahr von September 2009 bis Juli 2010 um fast die Hälfte erhöht. Bei den anderen Notationen - EPK und UML - sind es jeweils nur rund 25% für den gleichen Zeitraum (Freund, et al., 2010).

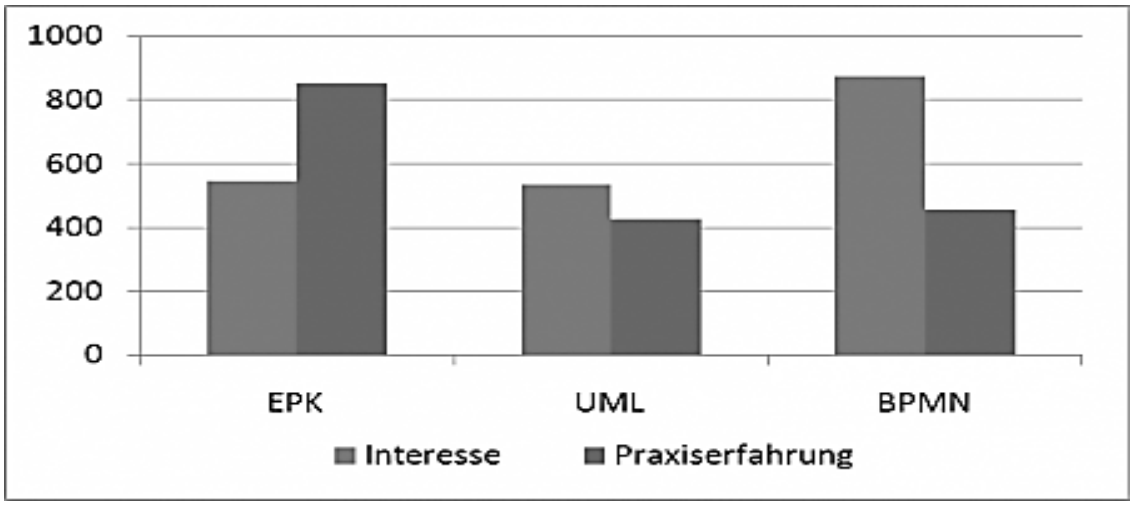


Abbildung 5 Popularität von Prozessnotationen auf BPM-Netzwerk.de (Stand September 2009) (Freund, et al., 2010)

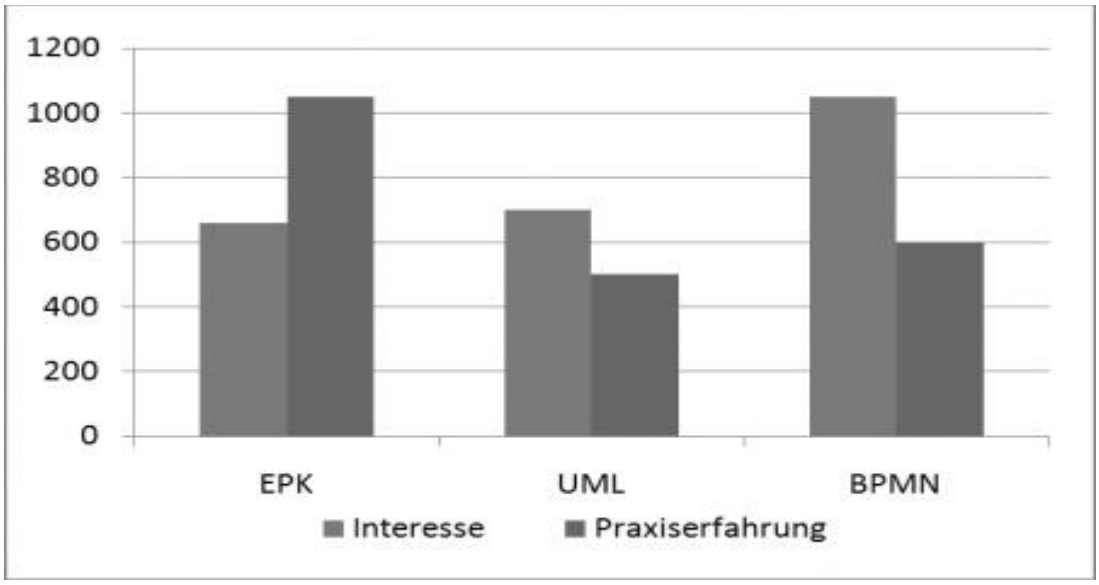


Abbildung 6 Popularität von Prozessnotationen auf BPM-Netzwerk.de (Stand Juli 2010) (Freund, et al., 2010)

Als Hauptzwecke beim Einsatz des Standards BPMN werden in einer Umfrage der Hochschule Bonn-Rhein-Sieg hauptsächlich die Dokumentation und die Optimierung von Prozessen genannt. Insgesamt gaben 39% der Befragten an, die BPMN einzusetzen; mehr als diejenigen, die die EPK einsetzen. Häufiger wurden nur nicht standardisierte, proprietäre Methoden genannt (Gadatsch, et al., 2012) (Allweyer, 2012). Anhand dieser Informationen lässt sich die stets steigende Popularität des BPMN-Standards erkennen.

2.5.2.3 Studie zur Modellierung mit BPMN und EPK

Eine mit dem "Preis für Wirtschaftsinformatik 2008" an der Hochschule Leipzig ausgezeichnete Befragung beschäftigt sich mit der Modellierung mit EPK und BPMN. Es wurden insgesamt achtzehn Studenten ohne Vorkenntnisse in der Prozessmodellierung befragt, wobei sowohl Multiple-Choice- als auch Fragen mit freier Antwortmöglichkeit gestellt wurden. Bei den Multiple-Choice-Fragen wurde den Befragten die Möglichkeit gegeben, die Modellierungsstandards auf einer Skala von 1 bis 6 zu bewerten, während bei den Fragen mit freier Antwortmöglichkeit die Studenten die Antworten selbst formulieren durften (Steudel, et al., 2008). Die Ergebnisse dieser Befragung sind nachfolgend tabellarisch dargestellt.

	1	**2**	**3**	**4**	**5**	**6**	**ø**
EPK	1	11	4	2	0	0	2,61
BPMN	1	7	5	4	1	0	2,72

Tabelle 2 Wie kommen Sie mit der Umsetzung eines Textes in einem Business-Modell zurecht (1 sehr gut - 6 sehr schlecht)? (Steudel, et al., 2008)

	1	**2**	**3**	**4**	**5**	**6**	**ø**
EPK	4	8	6	0	0	0	2,11
BPMN	3	6	6	2	1	0	2,56

Tabelle 3 Wie wird Ihrer Meinung nach die Realität eines Prozesses in einem Business-Modell widergespiegelt (1 sehr gut – 6 sehr schlecht)? (Steudel, et al., 2008)

Modellierungsstandard	**Anzahl Studenten**	**Prozent**
EPK	12	67%
BPMN	6	33%

Tabelle 4 Mit welchem Modell kommen Sie besser zurecht? (Steudel, et al., 2008)

Nachfolgend wird auf die Fragen mit freier Antwortmöglichkeit eingegangen, die sich auf die Favorisierung des BPMN bzw. EPK durch die Befragten bezogen. Für die Wahl des BPMN wurden folgende Gründe genannt:

- Bessere grafische Darstellungsmöglichkeit 2
- Übersichtlicher 4
- Komplexer 5
- Verständlicher 1

Der Entscheidung für das EPK lagen folgende Einschätzungen zugrunde:

- Übersichtlicher 5
- Logischer 5
- Verständlicher 3
- Einfacher umzusetzen 3
- Einfachere Symbolik 2

Testperson	EPK	BPMN
1	12	15
2	15	12
3	17	17
4	17	16,5
5	3	16
6	9	7
7	11	18
8	13	15
9	11	13
10	12,5	13
11	15	17
12	14	16
13	17	7,5
14	10	17
15	15	17
16	13	16
17	13	20
18	16	19
⌀ Punkte	12,97	15,11

Tabelle 5 Insgesamt vergebene Punkte

Testperson	EPK	BPMN
1	9	7
2	5	7
3	9	8
4	24	11
5	10	7
6	7	11
7	5	8
8	10	10
9	6	5
10	7	4
11	8	5
12	9	10
13	17	5
14	10	10
15	15	k. A.
16	12	10
17	13	11
18	11	5
⌀ (min)	10,39	7,8

Tabelle 6 Benötigte Zeit je Modell

Festzuhalten ist, dass 33% der Befragten die BPMN favorisierten und sich 67% für die EPK entschieden; als Gründe wurden die Übersichtlichkeit, der logischere Aufbau und die bessere Handhabung genannt (Steudel, et al., 2008). Insgesamt aber erzielte die BPMN laut Befragung eine bessere Punktbewertung und wurde deutlich schneller umgesetzt als die EPK (Steudel, et al., 2008). Fazit ist demnach, dass die BPMN - aufgrund der kürzeren Modellierungszeit - eine effektivere Anwendung anbietet, die EPK dagegen anwenderfreundlicher erscheint.

In diesem Abschnitt wurden einige Modellierungssprachen vorgestellt und anhand bestimmter Kriterien verglichen. Die EPK und die BPMN sind die zwei Modellierungssprachen, die derzeit am weitesten verbreitet sind. Besonders die BPMN zeigt eine rasante Entwicklung und ist immer mehr im Fokus größerer Softwarehersteller, wie SAP, IBM und Microsoft. Obwohl die EPK als intuitiver für die Modellierung von Geschäftsprozessen gelten, bietet die BPMN 2.0 mit einem ganzheitlichen Konzept; es wird eine einzige Modellierungssprache für alle Phasen der Modellierung angeboten - von der Geschäftsprozessmodellierung bis hin zur Workflowimplementierung. Inwieweit sich die BPMN als eine Ausführungssprache etablieren wird, bleibt abzuwarten. Erste BPM-Suiten, die alle Phasen der Modellierung mit BPMN 2.0 unterstützen, sind bereits implementiert worden (Götz, 2011). Die Entscheidung, welche Modellierungssprache in QUAM 2.0 implementiert wird, fällt aufgrund der o.g. Ausführungen zu Gunsten der BPMN 2.0. Im nächsten Abschnitt wird auf die anwendungsrelevanten Grundlagen der BPMN eingegangen.

2.6 Grundlagen von BPMN 2.0

Die Prozessmodellierung nach BPMN geht davon aus, dass in Organisationen Ereignisse (Events) stattfinden und daraufhin bestimmte Aufgaben (Tasks) erledigt werden müssen, wobei Einschränkungen zu beachten sind bzw. Entscheidungen getroffen werden müssen. Zur Darstellung von Verantwortlichkeiten werden in BPMN die so genannten Swimlanes und Pools verwendet. Befinden sich Ereignisse, Aktivitäten oder Gateways innerhalb eines

Pools, so werden sie mittels der Sequenzflüsse miteinander verbunden. Werden sie über Schwimmbahngrenzen verbunden, so muss dies über Nachrichtenflüsse (MessageFlows) geschehen. Weitere Informationen zum Prozess können durch Artefakte angegeben werden, wobei Artefakte keinen Einfluss auf die Reihenfolge der Flusselemente haben. Die Verbinder, die die Artefakte mit einem Flussobjekt verknüpfen, werden Assoziationen genannt (Freund, 2010).

Bei der BPMN 2.0 handelt es sich um eine Modellierungssprache bzw. ein Metamodell für die Modellierung von Geschäftsprozessen. Dennoch wird häufig von Fachkreisen bemängelt, dass die BPMN z.B. Prozesslandkarten, Organisationsaufbau, Geschäftsregeln nicht abbilden kann. Die BPMN konzentriert sich auf Prozesse, was die Abbildung einer zeitlich logischen Abfolge von Aktivitäten beinhaltet. Für die Beschreibung von anderen Sachverhalten einer Organisation gibt es andere, zum Teil standardisierte, Notationen. BPMN-Modelle können sehr gut mit anderen Diagrammtypen verknüpft werden, beispielsweise mit Prozesslandkarten oder Organigrammen. Außerdem bietet BPMN die Möglichkeit, den Standard mit eigenen Symbolen zu erweitern (vgl. Freund, 2010, S. 14).

Prozessmodelle, Prozessinstanzen und Token

Um die Business Process Diagramme (BPD) zu verstehen, sind nachfolgend aufgeführte Grundkategorien und Prinzipien von Bedeutung.

Prozessmodell: Ein Prozessmodell ist die Beschreibung eines Prozesses, wobei zu beachten ist, dass bei der BPMN ein Business Process Diagramm mehrere Prozessmodelle beinhalten kann.

Prozessinstanz: Eine Prozessinstanz ist die konkrete Ausprägung eines Prozessmodells, d.h. es handelt sich um einen Prozess aus der Realität.

Token: Das Token ist ein Konzept, das auch in den Petri Netzen Verwendung findet. Möchte man sich für ein gegebenes Prozessmodell vorstellen, welche Prozesswege während der Ausführung infrage kämen, ist das mittels Token Konzept möglich. Allweyer vergleicht das Token Konzept mit den Spielmarken bei einem Gesellschaftsspiel, die entsprechend der Regeln durch den Spielplan geschoben werden (vgl. Allweyer, 2009). Dabei erzeugt ein Start Ereignis

immer eine Marke (Token), der das Prozessdiagramm durchläuft. Für das Verständnis komplexer Diagramme ist das Token Konzept von besonderer Bedeutung.

Die sichtbaren Diagrammzeichnungen sind nicht die einzigen Informationen, die ein BPMN-Modell beinhaltet. Zusätzlich können zu jedem Prozesselement Attribute hinterlegt werden. Diese Element-Attribute sind in der Spezifikation von BPMN 2.0 beschrieben und dienen hauptsächlich der Prozessausführung oder Simulation. Für die Hersteller von BPM Tools gibt es die Möglichkeit, auch andere (proprietäre) Attribute zu jedem Symbol zu hinterlegen.

Ein durchdachtes und mit zusätzlichen prozessausführungsrelevanten Attributen versehenes Prozessmodell kann direkt einer Prozessausführungskomponente übergeben werden, die das Modell ausführt.

2.6.1 Aktivitäten

Abbildung 7 BPMN Aktivität (OMG, 2010)

Eines der wichtigsten Elemente eines Prozessmodells ist die Aktivität (Task). Man unterscheidet dabei nach

- Tasks: Arbeitsschritte, die nicht unterteilt sind
- Unterprozesse (Sub-Process): Prozesse, die in weitere Schritten unterteilt werden

Jede Aktivität sollte mit einem selbsterklärenden Namen versehen werden, z.B. nach dem Objekt-Verrichtungsprinzip, d.h. [Objekt] + [Verb]. In der Praxis werden die Begriffe Aktivität und Task synonym verwendet.

2.6.1.1 Spezielle Aufgaben

Die BPMN Spezifikation bietet die Möglichkeit, mit unterschiedlichen Aktivitätstypen zu arbeiten. Damit lassen sich vor allem technische

Prozessmodelle detaillierter beschreiben, was für eine Prozessausführungskomponente (auch Prozess-Engine) von Vorteil ist. Im Folgenden werden die einzelnen Aktivitätstypen kurz dargestellt.

Manuell:

Abbildung 8 Manuelle Aktivität in BPMN (OMG, 2010)

Hierbei handelt es sich um eine Aktivität, die vom Benutzer ausschließlich manuell verrichtet wird, ohne dabei Unterstützung von einem Softwaresystem zu bekommen.

Benutzer:

Abbildung 9 Benutzer Aktivität in BPMN (OMG, 2010)

Wird eine Aktivität von einer Prozess-Engine automatisch, z.B. durch das Auslösen bestimmter Ereignisse, erstellt und einem Menschen zugeteilt, handelt es sich dabei um eine Benutzer-Aktivität. Hier wartet die Prozess-Engine auf die Erledigung der Aufgabe und erhält Informationen über den Aufgabenstatus (vgl. Rademakers, 2012), z.B. in Form einer entsprechenden Bestätigung. Somit gehört diese Aufgabe zum „Human Workflow Management“.

Service:

Abbildung 10 Service Aktivität in BPMN (OMG, 2010)

Bei der Service-Aufgabe erledigt eine Softwarekomponente automatisch eine Aktivität, das kann z.B. eine Teilfunktion sein, die in der Prozessausführung genutzt wird. Standardmäßig ist diese Funktion ein Web-Dienst, es kann sich aber auch um eine andere Realisierung handeln.

Empfangen und Senden:

Abbildung 11 Empfangen und Senden Aktivität in BPMN (OMG, 2010)

Das Empfangen und Senden von Nachrichten kann ebenfalls als Aktivität modelliert werden. Je nachdem, ob es sich um das Empfangen oder um das Senden handelt, wird diese Aktivität entweder durch einen nichtausgefüllten oder ausgefüllten Umschlag visualisiert. Ein Beispiel für eine solche Aktivität ist die automatische Generierung einer E-Mail beim Überschreiten eines bestimmten Projektfists.

Skript:

Abbildung 12 Skript Aktivität in BPMN (OMG, 2010)

Ein Script-Task wird in der Process Engine ausgeführt und kann unterschiedlichen Zwecken dienen.

Geschäftsregel:

Abbildung 13 Geschäftsregel Aktivität in BPMN (OMG, 2010)

Diese, in der neuen Version hinzugekommene, Aktivität dient ausschließlich der Anwendung von Geschäftsregeln (vgl. Freund, 2010, S. 76).

Proprietäre Aufgabentypen: Die BPMN 2.0 Spezifikation erlaubt die individuelle Entwicklung von Aufgabentypen bzw. Aufgabensymbolen, damit eine genauere Anpassung der Notation an die Unternehmensgegebenheiten erfolgen kann. Mögliche Erweiterungen wären z.B. Aufgaben für Telefonkonferenzen, Meetings usw.

2.6.1.2 Markierungen

Zusätzlich zu den Aktivitätstypen können Aufgaben auch mit verschiedenen Markierungen versehen werden, z.B. Markierungen für Schleifen, Mehrfach-Instanzen oder Kompensationen. Diese Markierungen sollen nachfolgend kurz erläutert werden.

Schleife

Abbildung 14 Schleife in BPMN (OMG, 2010)

Hierbei entscheidet eine für die Aufgabe definierte Bedingung darüber, ob die Aufgabe wiederholt werden muss oder abgeschlossen ist.

Mehrfachaufgabe

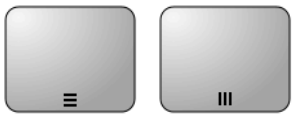


Abbildung 15 Mehrfachaufgabe in BPMN (OMG, 2010)

Eine Mehrfachaufgabe wird, wie die Schleife, mehrfach instanziiert, die Instanzen können aber nicht nur sequenziell, sondern auch parallel ausgeführt werden.

2.6.2 Gateways

Gateways steuern den Prozessfluss und sind die Prozessknoten an denen entschieden wird, was als Nächstes getan werden muss. Folgende Gateway-Typen werden in der BPMN Spezifikation eingesetzt.

2.6.2.1 XOR-Gateway (datenbasiertes Gateway)

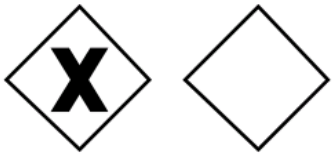


Abbildung 16 XOR Gateway in BPMN (OMG, 2010)

Bei einem datenbasierten, exklusiven Gateway handelt es sich um eine Entweder-oder-Entscheidung. Die Entscheidung wird aufgrund verfügbarer Daten getroffen, wobei nur einer der ausgehenden Pfade durchlaufen werden kann.

2.6.2.2 Paralleles Gateway

Abbildung 17 Paralleles Gateway in BPMN (OMG, 2010)

Das parallele Gateway oder auch "AND Split" hat einen eingehenden Sequenzfluss und mehrere ausgehenden Sequenzflüsse. Dieses Gatewaysymbol drückt aus, dass alle von ihm ausgehenden Sequenzflüsse gleichzeitig ausgeführt werden.

Durch die Parallelisierung muss auch jeder separate Pfad mit einem Endereignis enden, damit der Gesamtprozess terminieren kann. Mit einem parallelen inklusiven Gateway kann man Prozesspfade wieder zusammenführen. Der Gateway hat mehrere eingehende und einen ausgehenden Sequenzfluss. Dadurch wird eine Synchronisierung erreicht.

2.6.2.3 Datenbasiertes inklusives Gateway (OR-Gateway)

Abbildung 18 OR-Gateway in BPMN (OMG, 2010)

Datenbasierte inklusive Gateways werden verwendet, wenn es darum geht, eine kompakte Darstellung zu ermöglichen und Komplexität zu reduzieren. Damit lassen sich Und/Oder-Situationen beschreiben, bei denen mindestens ein Pfad und höchstens alle ausgehenden Pfade gleichzeitig durchlaufen werden können.

2.6.3 Ereignisse

Ereignisse drücken aus, dass vor, während oder am Ende eines Prozesses etwas *geschieht*. Man unterscheidet grundsätzlich zwischen Start-, Zwischen- und Endereignissen.

2.6.3.1 Startereignis, Zwischenereignis, Endereignis

Startereignisse sind die Auslöser eines Prozesses. Zwischenereignisse zeigen einen Zustand, der im Prozess erreicht wird. Sie können für Simulationszwecke benutzt werden, um z.B. ein Ereignis als Prozessmeilenstein zu markieren und die Zeit bis zur Erreichung dieses Meilensteins zu messen. Der Zustand am Ende eines Prozesses wird durch Endereignisse dargestellt.

2.6.3.2 Eingetretene und ausgelöste Ereignisse

Da die o.g. Unterteilung der Ereignisse aus Implementierungsperspektive zu ungenau ist, werden sie zusätzlich nach eingetreten und ausgelöst unterschieden. Eingetretene Ereignisse sind solche Ereignisse, die auf einen bestimmten Auslöser („trigger“) bezogen sind und als eingetreten gelten, sobald dieser Auslöser „gefeuert“ wurde. Sie können dazu führen, dass ein Prozess startet oder fortgesetzt wird. Sie können auch verursachen, dass die Aufgabe, die gerade bearbeitet wird abgebrochen wird.

Ausgelöste Ereignisse sind solche Ereignisse, die selbst einen Auslöser feuern, statt darauf zu reagieren, dass ein Auslöser gefeuert wurde. Sie können entweder während des Prozesses ausgelöst oder am Ende des Prozesses ausgelöst werden.

2.6.3.3 Ereignistypen

Wie bei den Aktivitäten ist auch hier eine Unterteilung nach Ereignistypen möglich, die einzelnen Ereignistypen werden im Folgenden dargestellt:

Nachricht

Abbildung 19 Nachrichtereignis (OMG, 2010)

Mit Hilfe des Nachrichtenereignisses lässt sich in BPMN eine Kommunikation abbilden. In BPMN ist jeder Vorgang, der sich auf einen spezifischen Addressaten bezieht und für diesen eine Information enthält, eine Nachricht.

Zeit

Abbildung 20 Zeitereignis (OMG, 2010)

Das Zeitereignis findet aufgrund seiner Flexibilität sehr oft Anwendung bei der Prozessmodellierung. Dieses Ereignis kann z.B. genutzt werden, wenn ein Prozess in Intervallen oder regelmäßig zu bestimmten Zeitpunkten gestartet werden muss. Wird das Ereignis als Zwischenereignis verwendet, kann es z.B. einen Prozess anhalten, bis eine Zeitspanne verstrichen ist oder ein Zeitpunkt erreicht ist. Außerdem kann dieses Ereignis den Prozess stoppen, solange ein definierter Zeitpunkt nicht erreicht ist oder bis eine Zeitspanne verstrichen ist. Als angeheftetes Zeitereignis drückt es z.B. ein Timeout aus, d.h. eine zeitliche Frist, die die maximale Bearbeitungsdauer einer Tätigkeit festlegt.

Fehlerereignis

Abbildung 21 Fehlerereignis in BPMN (OMG, 2010)

Mit dem Fehlerereignis lassen sich eventuell auftretende Fehler berücksichtigen, um z.B. eine Eskalation oder Behebung darzustellen. Fehlereignisse werden durch einen kleinen Blitz dargestellt.

Was genau ein solcher Fehler sein kann, ist dem Modellierer überlassen und ist nicht durch die BPMN 2.0 Spezifikation geregelt. Da ein Fehler in einem Prozess ein schwerwiegendes Ereignis ist, kann das Fehlerereignis nur als angeheftetes Ereignis modelliert werden, um also zu zeigen, dass eine Fehlerbehandlung vorzunehmen ist.

Bedingungsereignis

Abbildung 22 Bedingungsereignis in BPMN (OMG, 2010)

Über ein Bedingungsereignis wird eine Situation ausgedrückt, bei der ein Prozess nur dann starten oder weiterlaufen kann, wenn eine bestimmte Bedingung erfüllt ist. Die Bedingung ist neben dem Zeitereignis der einzige Typ, der nur als eingetretenes Ereignis existiert.

Signalereignis

Abbildung 23 Signalereignis in BPMN (OMG, 2010)

Das Signalereignis wird durch ein kleines Dreieck in einem Ereigniskreis dargestellt. In der BPMN können Signale in allen Ereignisausprägungen modelliert werden. Ein Signal ist, im Gegensatz zu einer Nachricht, nicht adressiert; es kann sozusagen als ein Broadcastingsignal gesendet werden, auf das jeder oder niemand reagieren kann.

Terminierungen

Abbildung 24 Terminierungsereignis in BPMN (OMG, 2010)

Terminierungsereignisse führen zu dem Ende einer Prozessinstanz und können als Konsequenz nur als Endereignis verwendet werden.

Kompensation

Abbildung 25 Kompensationsereignis in BPMN (OMG, 2010)

Bei einer Kompensation geht es um das Rückgängigmachen ausgeführter Aufgaben, z.B. Ticketbuchungen, Kartenbelastung usw. Kompensationen werden nur in Bezug auf Transaktionen eingesetzt.

Mehrfachereignis

Abbildung 26 Mehrfachereignis in BPMN (OMG, 2010)

Mit einem Mehrfachereignis können mehrere Ereignisse in einem Symbol zusammengefasst werden. Als eingetretenes Ereignis bedeutet das Symbol, dass nur eines der enthaltenen Ereignisse eintritt, um den Prozess zu starten, fortzusetzen oder die Aufgabe abzubrechen. Handelt es sich dagegen um ein ausgelöstes Ereignis, führt dies dazu, dass alle enthaltenen Ereignisse ausgelöst werden.

Abbruch

Abbildung 27 Abbruchsereignis (OMG, 2010)

Das Abbruchereignis kann nur im Kontext von Transaktionen verwendet werden und unterbricht eine Transaktionsinstanz.

2.6.4 Swimlanes

Die Flusselemente, die erläutert wurden, geben dem Betrachter eine Übersicht darüber, was in einem Prozess getan werden muss. Die Lanes geben die Information, wer für die Durchführung dieser Aufgaben oder Teilprozesse innerhalb eines Prozesses verantwortlich ist. Man kann dadurch erkennen, welche Aufgaben welchen Personen, Abteilungen oder Organisationen zugeordnet wurden. Die BPMN-Spezifikation gibt keine Auskunft darüber, ob es sich bei den Verantwortlichen um Menschen, Abteilungen, Organisationseinheiten oder IT-Systeme handelt (vgl. Freund, 2010, S. 95). Eine nützliche Eigenschaft der BPMN-Lanes ist die, dass sie auch verschachtelt werden können und dadurch eine gewisse Hierarchisierung der Verantwortlichkeiten visualisiert werden kann.

Die Lanes werden immer von Pools umrandet, die gleichzeitig die Grenzen des Prozesses umfassen. Außerdem drücken die Pools eine für die Lanes übergeordnete Instanz aus, die die Steuerung des Prozesses übernimmt und z.B. eine Aufgabenzuordnung vornimmt.

2.6.5 Sequenzfluss und Nachrichtenfluss

2.6.5.1 Sequenzfluss

Der Sequenzfluss verbindet immer zwei Flusselemente miteinander und beschreibt damit die zeitlich-logische Beziehung zwischen ihnen. Das ist auch der Pfad, über den ein Prozesstoken läuft.

2.6.5.2 Nachrichtenfluss

Der Nachrichtenfluss verbindet Nachrichtenereignisse miteinander und symbolisiert eine Kommunikation zwischen ihnen. Der Austausch von Nachrichten kann zwischen unterschiedlichen Pools erfolgen, nicht aber zwischen verschiedenen Lanes eines Pools (vgl. Silver, 2011).

2.6.6 Artefakte

Artefakte können keinen direkten Einfluss auf die Ablaufsemantik ihres Prozessmodells ausüben. In der BPMN 2.0 sind folgende Artefakte vorgesehen:

- Datenobjekte
- Anmerkungen und Gruppierungen
- Eigene Artefakte

In der BPMN ist es erlaubt, eigene Artefakte zu definieren, die z.B. unternehmensspezifische Sachverhalte abbilden und weitere Informationen zum Prozessmodell beinhalten.

2.6.7 Datenobjekte

In der BPMN werden Aspekte, die die zeitlich-logische Reihenfolge der Elemente indirekt betreffen, nachrangig behandelt. Das können z.B. Dokumente oder Daten sein, die innerhalb des Prozesses verwendet und erzeugt werden. Datenobjekte können alle möglichen Informationen repräsentieren, unabhängig von ihrer physischen Form (Papierdokumente, elektronische Datensätze). Die Verbindung mit anderen Flussobjekten erfolgt über Assoziationen, sie können mit einer Bezeichnung versehen werden. Datenobjekte können auch einen Status erhalten, der mit eckigen Klammern ausgedruckt wird. Beispiele für den Status von Datenobjekten sind bspw. geprüft, erzeugt, abgelehnt, freigegeben.

3 Technische Grundlagen

Da der BPMN-Standard ein Austauschformat und Ausführungsemantik in XML-Format anbietet, werden als erstes Grundlegende Themen der XML-Sprache vorgestellt. Um das Metamodell der BPMN-Modellierungssprache erläutern zu können, wird im Abschnitt 3.1.2 auf den W3C-Standard XML Schema eingegangen. Abschnitt 3.1.3 behandelt kurz das Meta Object Facility(MOF) einen anderen Standard der OMG, der zur Serialisierung von BPMN-Modelle verwendet werden kann. Im Abschnitt

3.1 Die XML-Sprache

Die eXtensible Markup Language (XML) ist eine Auszeichnungssprache zur Speicherung von Informationen in einer strukturierten Form. Sie stellt ein Vokabular dar, das aus Symbolen und der ihnen zugewiesenen Semantik besteht, ergänzt um Regeln zur Kombination der Symbole (vgl. Jeckle, 2004). Diese Regeln beziehen sich auf die grammatikalische Struktur und den Inhalt eines Dokuments. Abbildung 28 zeigt den Zusammenhang einiger Basistechniken der XML-Sprache.

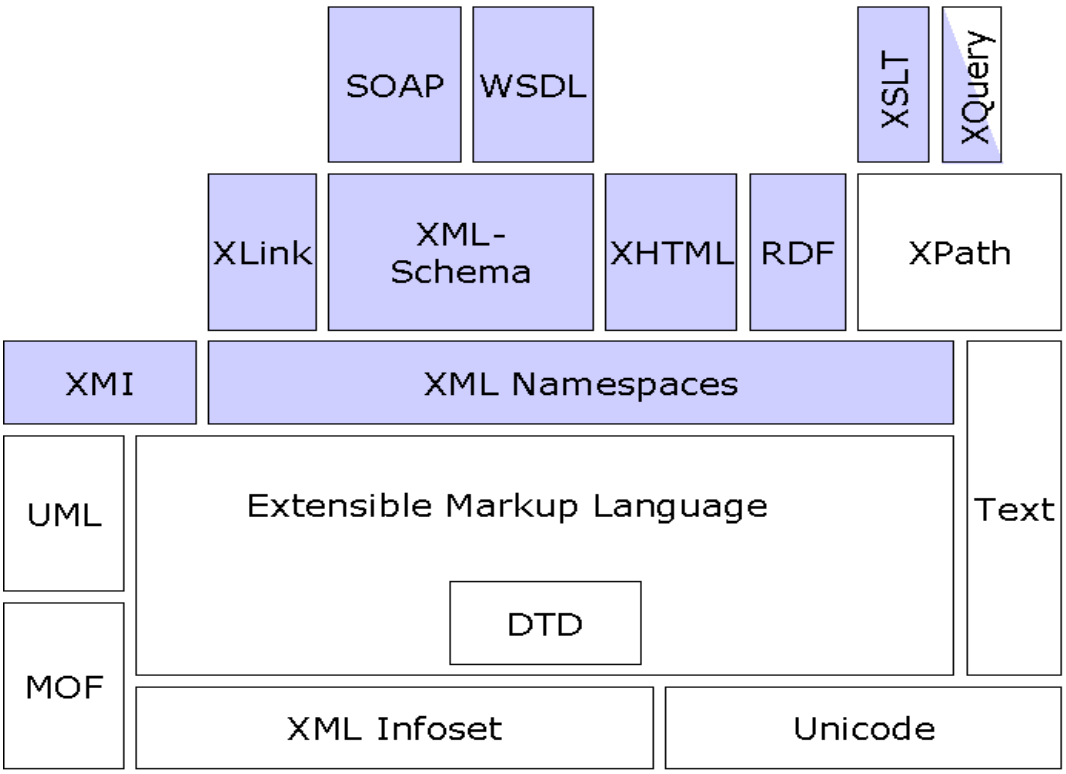


Abbildung 28 Zusammenhang einiger Basistechniken der XML-Sprache (Jeckle, 2004)

Ursprünglich wurde XML als eine Teilmenge der Standard Generalized Markup Language (SGML) definiert. Dadurch erbt XML bestimmte Regeln der SGML, wie etwa die Document Type Definitions (DTD), die zur Bestimmung des Vokabulars einer XML-Sprache dienen. Auf der Grafik sind auch die XML-Namensräume abgebildet, die zur Abgrenzung von gleichen Bezeichnern in unterschiedlichen Kontexten dienen. Damit lassen sich Strukturen mit beliebigen Element- und Attributnamen entwerfen, ohne auf Namenskonflikte achten zu müssen, wenn z.B. verschiedene Dokumente gleichzeitig benutzt werden, in denen gleichnamige Elemente existieren.

Im Gegensatz zu anderen Auszeichnungssprachen, wie zum Beispiel die HTML, hat XML den Vorteil, dass Sprachelemente frei definiert werden können. HTML hat eine fest vordefinierte Menge an Tags und zugehörigen Attributen, dazu zählen <table>, <i>, , <p>,
 usw. In XML dagegen sind beliebige Elemente definierbar, wie z.B. <Kunde>, <Auftrag>, <Ereignis>, <Task> usw. Ein einfaches Beispiel hierfür ist in **Listing 1** angegeben.

XML-Code

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Ein XML Beispiel -->
<Kunde>
        <Geschlecht=“männlich“>
        <Name>Beispielkunde</Name>
        <Ort>Magdeburg</Ort>
        <Plz>39104</Plz>
</Kunde>
```

Listing 1 Einfaches XML-Beispiel

Desweiteren ist XML im Gegensatz zu anderen Auszeichnungssprachen „case sensitiv“, d.h. bei XML spielt die Groß- und Kleinschreibung eine Rolle. Ein anderes Charakteristikum der Sprache ist das, dass sie plattformneutral ist, was bei der Kommunikation zwischen Systemen unterschiedlicher Betriebssystemwelten und Programmiersprachen von großer Bedeutung ist. Anderes charakteristisches Merkmal der Sprache ist die Trennung von Struktur

und Darstellung. Auch diesbezüglich unterscheidet sie sich von der HTML, die vielmehr eine Sprache zur visuellen Gestaltung von Inhalten ist.

3.1.1 Strukturelle Grundlagen der XML-Sprache, Bestandteile von XML-Dokumenten

In diesem Abschnitt sollen die wesentlichen Komponenten der XML-Sprache anhand einfacher Beispiele erläutert werden. Die strukturelle Grundlage der XML wird durch das sog. XML *„Information Set“* dargestellt. Ein *„Information Set“* definiert wichtige Begriffe einer XML-Struktur.

Die Syntax von XML ist in einer Empfehlung des World Wide Web Consortium (W3C) definiert, während die Semantik durch den W3C-Standard des XML *Information Set* definiert wird (Jeckle, 2004). In diesen Spezifikationen werden wichtige Definitionen in Bezug auf Begrifflichkeiten und Beziehungen der verschiedenen Elemente eines XML-Dokuments festgelegt. Nachfolgend wird sowohl auf grundlegende Begriffe als auch auf die Beziehungen von Elementen zueinander eingegangen.

Definition von einem XML-Dokument

Unter einem XML-Dokument ist ein Datenfluss zu verstehen, der die Strukturierungsanforderungen der eXtensible Markup Language erfüllt (vgl. Jeckle, 2004).

Jedes XML-Dokument beinhaltet ein sog. *„Information Set“*, das alle Informationen eines Dokuments, bspw. Hinweise zu Elementen, Attributen, Kommentaren, enthält (vgl. Jeckle, 2004). Listing 1 stellt ein einfaches Dokument dar, das die am häufigsten verwendeten Sprachelemente der XML beinhaltet.

Document Information Item

Zu jedem „*Information Set*“ gibt es genau ein sog. „*Document Information Item“*. Dadurch wird der Grenzbereich des XML-Dokuments gekennzeichnet. Das Element enthält wichtige Informationen über die XML-Version und das eingesetzte Codierungsschema.

XML-Code
<?xml version="1.0" encoding="UTF-8"?>

Listing 2 Ein Document Information Item

Das *Document Information Item* beinhaltet u.a. Informationen des Dokumentprologs. Wird der Prolog ausgelassen, so sind die darin enthaltenen Informationen im *Document Information Item* nicht *gesetzt* (Jeckle, 2004). Weiterhin besteht das *Document Information Item* aus einer Liste von Kindknoten. In dieser Liste existiert genau ein Element vom Typ *Information Item*, das den Startknoten (auch Wurzelknoten) eines XML-Dokuments darstellt. Auf Grund seiner großen Bedeutung als Wurzel des XML-Baumes wird dieses Element als Document Element bezeichnet. Andere wichtige Kindknoten der Liste sind die Elemente vom Typ *Processing Instruction Information Item*. Sie sind für den XML-Prozessor bestimmt und beinhalten Verarbeitungs-anweisungen, die dieser interpretiert. Die Liste beinhaltet auch das *Document Type Declaration Item*, falls das XML-Dokument eine Dokumenttypdeklaration enthält.

Element Information Item

XML-Code
<Gebäude> ** <Raum>Meetingsraum</Raum>** ** <Raum>Pausenraum</Raum>** **</Gebäude>**

Listing 3 Ein Element Information Item

Einen weiteren Bestandteil eines Dokuments stellt das *Element Information Item* dar. Seine Rahmen werden durch die Start- und Endtags gekennzeichnet. Bei leeren Elementen ist der Starttag gleichzeitig auch Endtag, was zur Folge hat, dass leere Elemente keine Unterelemente haben können.

Durch mehrere ineinander verschachtelte *Element Information Items* wird die Struktur eines XML-Dokuments bestimmt. Die Typen eines XML-Dokuments

werden über den Tag-Namen definiert, sie werden auch zur Gültigkeitsprüfung eines Dokuments verwendet. Ähnlich wie in den Programmiersprachen werden die Elementnamen gebildet (Vonhoegen, 2007). Der Elementname muss mit einem Buchstaben, Unterstrich oder Doppelpunkt beginnen, worauf beliebige Zeichen folgen können. Nicht erlaubt sind Leerzeichen, Tabulatoren oder Zeilenvorschübe. Außerdem darf der Elementname nicht mit sich öffnender oder sich schließender Winkelklammer beginnen.

Attribute Information Item

XML-Code
<Person Geschlecht="maennlich"> **<Name>Steve Jobs</Name>** **</Person>**

Listing 4 Ein Attribute Information Item

Listing 4 zeigt ein *Attribute Information Item*. Hauptzweck des *Attribute Information Sets* ist es, das Element mit gewissen Eigenschaften zu versehen. Die Syntax besagt, dass Attribute in einem Start-Tag definiert werden müssen und über Namen-Wert-Paare Wertzuweisungen bekommen. Im Gegensatz zu den *Element Information Items* sind für ein *Attribute Information Item* keine weiteren Verschachtelungen erlaubt. Somit dürfen innerhalb eines Attributs keine Symbole auftreten, die vom XML-Prozessor als Starttags interpretiert werden können. Auch die Anordnung des Erscheinens der Attribute in einem Element unterscheidet sich von der Anordnung des Kindelements innerhalb eines Elements. Kindelemente werden in einer geordneten Liste definiert, während Attribute in beliebiger Ordnung auftreten können, ohne dabei die Semantik des Dokuments zu beeinflussen. Der Einsatz von Attributen bietet sich an, wenn z.B. deskriptive Informationen über das Element dargestellt werden sollen. Hier kann es sich um Informationen höherer Ordnung, sog. Metainformationen, handeln (Jeckle, 2004).

Elemente werden dagegen immer eingesetzt, wenn eine weitere Unterstrukturierung der darzustellenden Inhalte möglich ist. Da Attribute syntaktisch an das Elternelement gebunden sind, ist ihre Verwendung in

anderen Kontexten unmöglich (vgl. Jeckle, 2004). Entschärft wird diese Einschränkung durch die Einführung des XML-Schema-Standards.

Eine grafische Gegenüberstellung von Element- und Attributdeklaration ist in Abbildung 29 dargestellt.

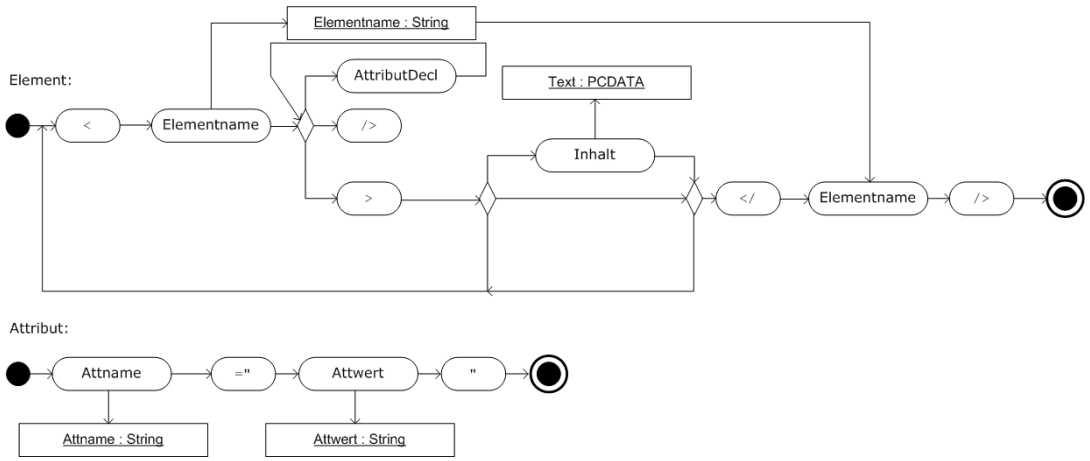


Abbildung 29 Element- und Attributdeklaration in XML (Jeckle, 2004)

Comment Information Item

XML-Code
<! -- Das ist ein Kommentar -->

Listing 5 Ein Comment Information Item

Die XML-Sprache erlaubt das Definieren von Kommentaren (*Comment Information Item)* innerhalb eines XML-Dokuments. Ein Beispiel für einen Kommentar stellt Listing 5 dar.

Konzept der Namensräume

Die Entwicklung von neuen XML-Sprachen trägt dazu bei, dass es für ähnliche Problemstellungen zu Mehrfachentwicklungen kommt. Dies äußert sich in dem Gebrauch identischer Bezeichner in unterschiedlichen XML-Sprachen. Möchte man existierende Sprachsegmente anderer XML-Sprachen in neue Sprachen integrieren, tritt das Problem der Eindeutigkeit der Elemente auf.

Semantik und Syntax der Namensräume werden durch die Recommendation Namespaces definiert. Das Konzept ist kurz nach dem Veröffentlichen der ersten XML-Version vorgestellt worden. Kerngedanke der Namensräume ist es, Element und Attributnamen so zu erweitern, dass zum Schluss, auch wenn

mehrere Dokumente vereinigt werden, eindeutige Namen entstehen. Hauptziel von W3C ist es gewesen, ein Konzept der Identifikation zu entwerfen, das sowohl mächtig als auch leicht zu administrieren ist.

Document Type Definition

Unter Document Type Definition (DTD) versteht man eine Auflistung von Tag-Deklarationen oder Regeln, die bestimmen, welche XML-Elemente in einem Dokument vorkommen dürfen, wie sie in Beziehung zueinanderstehen und welche Attribute sie beinhalten können. Ob ein XML-Dokument valide ist, wird anhand der Dokumenttypdefinition geprüft. Mittlerweile hat DTD gegenüber dem neueren XML-Schema-Standard an Bedeutung verloren. Im Gegensatz zu DTDs sind XML-Schemata komplexer und mächtiger in der Ausdruckskraft. Ein XML-Schema ist selbst ein XML-Dokument. Es werden viele vordefinierte einfache Datentypen angeboten, aber es können auch komplexe Datentypen definiert werden.

3.1.2 XML-Schema-Definition (XSD)

Schema-Definitionen stellen einen wesentlichen Bestandteil von XML dar. Sie sind in dieser Arbeit auch von besonderer Bedeutung, da der BPMN 2.0-Standard auch in Form von XML-Schema-Definitionen veröffentlicht ist, die gleichzeitig zur Serialisierung von BPMN-Modellen benutzt werden.

Mittels einer Schemasprache lassen sich eigene kontextspezifische Datentypen definieren. In XSD wird zwischen einfachen und komplexen Datentypen unterschieden. Leitet sich ein XML-Dokument aus einer Schema-Definition ab, so muss es die Regeln einhalten, die in dem XSD-Dokument definiert sind.

3.1.2.1 Aufbau eines XSD-Dokuments

Ein XSD-Dokument ist selbst ein XML-Dokument, das auch ein *Document Information Item* enthält. Zusätzlich beinhaltet es das <schema> Element, innerhalb dessen das XSD-Dokument definiert wird. Das <schema> Element umschließt alle weiteren Regeln oder Definitionen weiterer Elemente. In Listing 6 ist dieser Bereich durch <!-- Hier sind eigene Definitionen möglich -->

gekennzeichnet. Innerhalb des Elements wird mit der Anweisung xmlns=„http://www.w3.org/2001/XMLSchema“ ein Standard-Namensraum zugewiesen. Diese Definition ist für jedes XSD-Dokument erforderlich. Mittels xmlns:xsd wird auf einen definierten Namensraum verwiesen. Somit wird sichergestellt, dass lediglich Instanzen eines Namensraums verwendet werden (Jeckle, 2004). Das ist dann von wichtig, wenn in einem Dokument mehrere Elemente mit identischen Namen auftreten können.

XSD-Beispiel

```
<? xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <!--Hier sind eigene Definitionen möglich -->
</xsd:schema>
```

Listing 6 Beispiel für ein einfaches XSD-Dokument

Zur Prüfung von XSD- und XML-Dokumenten bietet Visual Studio 2010 hilfreiche Werkzeuge. Es werden nach der Überprüfung eines oder mehrerer Dokumente dazu Meldungen ausgegeben, ob die erstellten Inhalte nach einem vorgegebenen Schema gültig sind.

3.1.2.2 Einfache Datentypen in XSD

Einfache Datentypen können im Unterschied zu komplexen Datentypen keine Kindelemente oder Attribute enthalten, sie können also nicht weiter strukturiert werden. Listing 7 zeigt einige Beispiele für einfache Datentypen, die in XSD standardmäßig angeboten werden. Definiert werden diese mit dem Element *simpleType*. In Abhängigkeit davon, welche Datentypen für ein Element erwartet werden, kann der Typ explizit angegeben werden, was ein Vorteil gegenüber dem Document Type Definition darstellt. Deklariert wird ein Typ durch die Wertepaare „Namensraum: Typ“. Im Listing 7 werden einige wichtige Datentypen in XSD angegeben (Vonhoegen, 2007).

XSD-Code

```
xsd:string, xsd:decimal, xsd:integer, xsd:float, xsd:Boolean, xsd:date, xsd:time
```

Listing 7 Eine Auflistung aller wichtigen einfachen Dateitypen in XSD

Ein einfacher Datentyp leitet sich immer von einem Basis-Typ ab, wobei drei Formen der Ableitung möglich sind: Beschränkung, Liste und Union. Als Beispiel wird hier der Datentyp double angegeben, der eine beschränkte Ableitung des Basistyps decimal ist, da er über einen kleineren Gültigkeitsbereich verfügt. Veranschaulicht wird das Konzept durch ein etwas komplexeres Beispiel aus der BPMN 2.0 Spezifikation:

XSD-und XML-Code

XML-Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<process>
        <processType>Public</processType>
</process>
```

XSD-Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
                xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                targetNamespace="http://www.omg.org/spec/BPMN/20100524/MODEL">
 <xsd:element name="process" type="tProcess" substitutionGroup="rootElement"/>
        <xsd:complexType name="tProcess">…</xsd:complexType>
        <xsd:simpleType name="tProcessType">
                <xsd:restriction base="xsd:string">
                        <xsd:enumeration value="None"/>
                        <xsd:enumeration value="Public"/>
                        <xsd:enumeration value="Private"/>
                </xsd:restriction>
        </xsd:simpleType>
```

Listing 8 Einfacher Datentyp

Im oberen Bereich von Listing 8 befindet sich ein mögliches XML-Dokument, das eine valide Instanz des unteren XSD-Dokuments darstellt. Das XSD-Dokument definiert ein Element vom Typ process mit einem Kindelement *processType*, deklariert als einfacher Elementtyp vom Typ Enumeration. Die möglichen Werte None, Public und Private sind durch den Datentyp Enumeration festgelegt.

3.1.2.3 Komplexe Datentypen

Komplexe Datentypen stellen logische Gruppierungen von Elementen oder Attributen dar und werden mit dem Element *complexType* definiert. Anders als bei einfachen Datentypen hat man hier die Möglichkeit, Elemente ineinander zu verschachteln und somit gewisse Dokumenthierarchien zu bilden. Listing 10 zeigt am Beispiel die Definition des komplexen Datentyps process der BPMN 2.0 Spezifikation sowie eine mögliche Ausprägung des XSD-Schemata dazu.

XML-Dokument

```
  <process id="_4.0.process" name="Funktionsname ">
   <laneSet>
     <lane id="_1.0" name="Funktionsname ">
      <flowNodeRef>_23.0</flowNodeRef>
      <flowNodeRef>_50.0</flowNodeRef>
     </lane>
   </laneSet>
   <endEvent id="_65.0" name=""/>
   <task id="_50.0" name=""/>
   <intermediateCatchEvent id="_38.0" name=""/>
   <task id="_23.0" name=""/>
   <startEvent id="_12.0" name=""/>
   <sequenceFlow id="_37.0" sourceRef="_12.0" targetRef="_23.0"/>
  </process>
```

XSD-Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
          xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
          xmlns:xsd="http://www.w3.org/2001/XMLSchema"
          targetNamespace="http://www.omg.org/spec/BPMN/20100524/MODEL">
<xsd:element name="process" type="tProcess" substitutionGroup="rootElement"/>
     <xsd:complexType name="tProcess">
      <xsd:complexContent>
       <xsd:extension base="tCallableElement">
         <xsd:sequence>
           <xsd:element ref="auditing" minOccurs="0" maxOccurs="1"/>
           <xsd:element ref="monitoring" minOccurs="0" maxOccurs="1"/>
<xsd:element ref="property" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="laneSet" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="flowElement" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="artifact" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
<xsd:attribute name="processType" type="tProcessType" default="None"/>
<xsd:attribute name="isClosed" type="xsd:boolean" default="false"/>
        <xsd:attribute name="isExecutable" type="xsd:boolean"/>
<xsd:attribute name="definitionalCollaborationRef" type="xsd:QName" use="optional"/>
       </xsd:extension>
     </xsd:complexContent>
     </xsd:complexType>
     <xsd:simpleType name="tProcessType"> ... </xsd:simpleType>
</xsd:schema>
```

Listing 9 Definition eines komplexen Datentyps am Beispiel der BPMN 2.0

Im Folgenden wird ein Überblick über die Metaobjekt Facility gegeben, ein Standard, der ebenso von der Object Management Group definiert und weiterentwickelt wird und mit dem der BPMN 2.0 konform ist.

3.1.3 Meta Object Facility (MOF)

Meta Object Facility (MOF) ist, wie auch BPMN und UML, ein Standard, der von der Object Management Group gehostet wird und mit dessen Hilfe sich Informationen beliebiger Natur modellieren lassen. Der Standard beinhaltet mehrere Bestandteile zur Metamodellierung. Ziel ist es, eine Interoperabilität und Portabilität von Modellen und Systemen zu erreichen, die auf MOF basieren. MOF definiert außerdem die Möglichkeit zur Reflection-Entwicklung, bei der eine Anwendung zur Laufzeit ein Modell abrufen kann.

Mit Hilfe des vom MOF zur Verfügung gestellten Metameta-Modells können neue Metamodelle erstellt werden. Charakteristisch für MOF sind die Standardisierung der Modellierungskonzepte und die Wiederverwendung der UML2 in der Spezifikation von MOF2, was das Metamodell der UML2 automatisch MOF-konform macht. Mittels Metadata Object Facility können Metadaten-Repositories umgesetzt werden (vgl. Brodsky, 2002, S. 12). Sowohl die UML-Spezifikation als auch BPMN-Spezifikation basieren auf MOF. Grundsätzlich wird zwischen Essential MOF (EMOF) und Complete MOF (CMOF) unterschieden, wobei EMOF eine Teilmenge von MOF ist und dazu dient, MOF-konforme Modelle auf eine unkomplizierte Art und Weise zu erstellen, ohne sich dabei mit dem kompletten MOF auszukennen. Die MOF Architektur hat insgesamt vier Schichten, die nachfolgend erläutert werden.

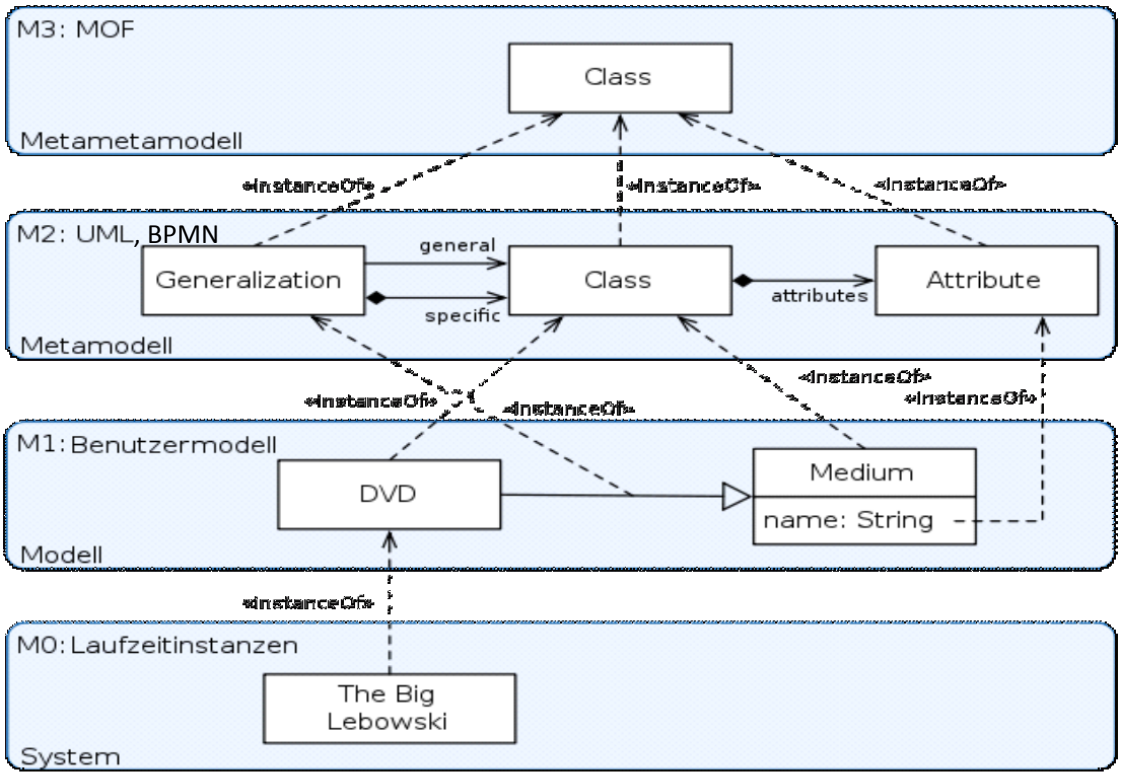


Abbildung 30 Vierschichten-Architektur von MOF (Wikipedia, 2012)

In der Architektur (Abbildung 30 Vierschichten-Architektur von MOF) bildet das MOF-Modell die M3-Schicht, also die Meta-Metasprache mit der die Metamodelle auf der M2-Ebene erstellt werden. Auf der M2-Ebene sind beliebige Metamodelle, wie z.B. von BPMN 2.0, UML, EPK, einzuordnen, die mit der MOF-Spezifikation konform sind. Die M1-Ebene beschreibt eine Modellausprägung der M2-Ebene und ist somit ein konkretes Modell, zum Beispiel ein BPMN 2.0 Prozessmodell. Auf der untersten M0-Ebene befindet sich das Objekt der realen Welt (vgl. Object Management Group, 2007).

Mittels XMI (XML Model Interchange) können aus MOF-konformen Metamodellen XML-Schemata erstellt werden (vgl. Brodsky, 2002, S. 12). Für die Transformation eines Metamodells in das XMI-Schema können Softwaretools eingesetzt werden; die Instanzen dieses Metamodells resultieren dann in konkreten XML-Dokumenten, die konform mit den XMI-Schemata sind. Soll also ein Modell mittels XMI serialisiert werden können, muss das Modell erst in MOF definiert werden. Das XMI-Modell ist eine Modellinstanz des MOF-Modells. Eine Serialisierung über XMI ist demnach eine Modelltransformation, bei der das eine Modell in ein anderes überführt wird. Da der Standard BPMN

2.0 MOF-konform ist, werden dazu von der Object Management Group entsprechende XMI-Dateien angeboten. Aus dem XMI des MOF-konformen BPMN 2.0 Metamodells ergeben sich dann Regeln zur Serialisierung von BPMN 2.0 Modellen. Die Implementierung von BPMN 2.0 per Hand oder durch Codegeneratoren ermöglicht dann den Austausch von validen Modellen zwischen den Softwareprodukten. Viele gängige XML-Tools unterstützen beim Modellaustausch eher XSD und nicht XMI bzw. unterstützen sie es in neueren Versionen. Dies hat dazu geführt, dass OMG BPMN 2.0 nicht nur als XMI, sondern auch als XSD zur Verfügung stellt. Im Rahmen dieser Arbeit werden XSD-Modelle für die Deserialisierung angewandt.

Nachfolgend werden zuerst grundlegende Konzepte der Serialisierung und Deserialisierung erläutert und danach die Serialisierung mit .NET betrachtet. Anschließend wird das BPMN 2.0 Metamodell aus Implementierungsperspektive betrachtet, d.h. es werden implementierungstechnische Aspekte des Metamodells fokussiert.

3.2 Serialisierung

Unter Serialisierung versteht man in der Informatik die Überführung des Zustands eines Objekts in ein Format, das den Transport dieses Objektes, z.B. mittels HTTP, über das Internet möglich macht (

Abbildung 31).

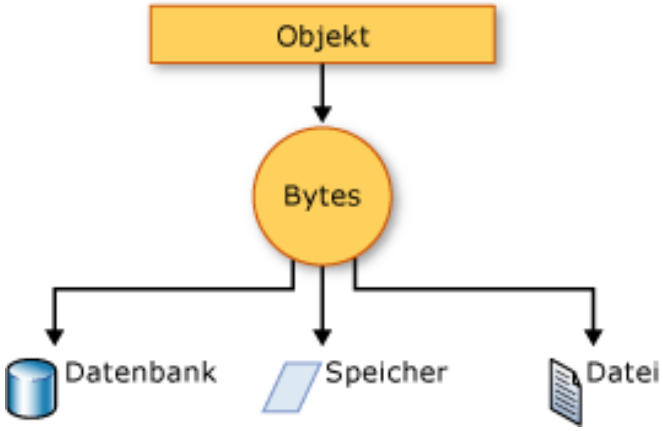


Abbildung 31 Serialisierung eines Objekts (Microsoft, 2005)

Die Deserialisierung dagegen stellt den umgekehrten Vorgang dar, d.h. das Objekt wird aus einem Stream oder einer Datei rekonstruiert (Abbildung 32).

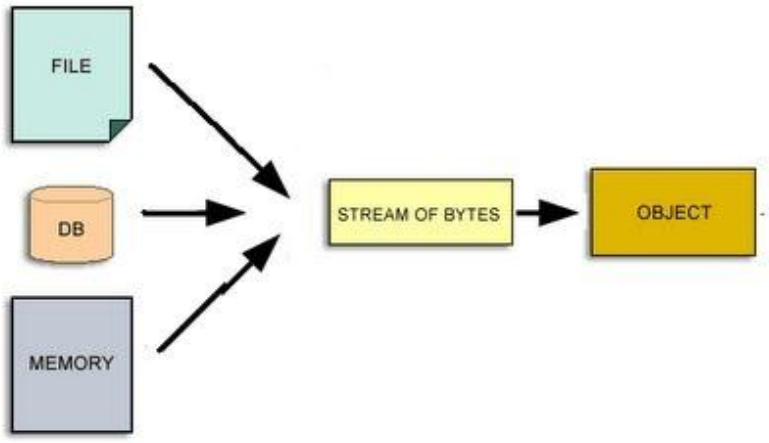


Abbildung 32 Deserialisierung eines Objektes (Microsoft, 2005)

XML-Serialisierung

Bei der XML-Serialisierung werden die Objekte, statt in binärer Form, als XML-Stream persistiert. Dabei kann man nur diejenigen Felder eines Objekts serialisieren, die als “public” gekennzeichnet sind. Da bei der Serialisierung keine Informationen zu dem Objekttyp angegeben werden, muss ggf. schon bei der Deserialisierung eines Objektmodells eine Typkonvertierung (Casting) stattfinden. Damit wird sichergestellt, dass das später serialisierte Objekt vom „richtigen“ Typ ist.

Da die Object Management Group mit der Version 2.0 der BPMN auch ein XML-Schema anbietet, ist das Erzeugen eines vollständigen Objektmodells über die Deserialisierung der Schema-Dateien möglich.

Verschiedene objektorientierte Programmiersprachen bieten eine direkte Objekt-Serialisierung an, entweder durch sogenannte syntaktische Zucker-Elemente oder durch die Bereitstellung einer Standard-Schnittstelle dafür. Einige dieser Programmiersprachen sind Ruby, Smalltalk, Python, PHP, Java und die .NET Sprachfamilie. Da das QUAM 2.0 System auf SharePoint bzw. auf der .NET Technologie basiert und Visio nur die Programmiersprachen der .NET Technologie und VBA unterstützt, wurde die Modelldeserialisierung in C# umgesetzt. Daher werden nachfolgend einige Grundlagen der Serialisierung in C# vorgestellt.

Serialisierung mit C#

Eine wesentliche Rolle bei der Serialisierung in C# spielt die Klasse XmlSerializer mit ihren zwei wichtigsten Methoden serialize() und deserialize(). Die von XmlSerializer erstellten XML-Inhalte sind XML 1.0-konform, wobei damit generierte Datentypen der Datentypdefinition von “XML-Schema Part 2” entsprechen. Objektdaten werden mittels Klassen, Klassenfeldern, Eigenschaften, Arrays und primitiver Typen beschrieben.

Bei Vorhandensein eines XML-Schemata ist es möglich, daraus entweder manuell oder mittels Codegenerierungstools Klassendateien oder ganze Objektmodelle zu deserialisieren, die für dieses Schema streng typisiert und mit Attributen versehen sind. Sind solche Klassen gegeben, sind ein einfach zu bearbeitendes Objektmodell und eine Konformität mit den XML-Schema-Definitionen (zum Beispiel mit dem BPMN 2.0 XSD Modell) erreicht (Microsoft, 2005).

Mit der XMLSerializer-Klasse kann die Serialisierung eines Objekts als XML-Stream nahezu uneingeschränkt gesteuert werden. Zum Beispiel kann ein Klassenfeld als XML-Elementattribut serialisiert werden und dabei der Name bestimmt werden, den das Attribut bekommen soll. Für einfache Datentypen der XSD-Language gibt es entsprechende Datentypen im .NET-Framework. Für Datentypen, die kein Gegenstück im .NET-Framework haben, werden mit dem XSD-Tool anhand eines Schemas Klassen erzeugt. Attribute werden auf Eigenschaften vom Typ String abgebildet; mit der Eigenschaft DataType wird der Name des XML-Datentyps festgelegt.

3.3 Das BPMN 2.0 Metamodell

Seit der Version 2.0 des BPMN-Standards ist auch die XML-Serialisierung von BPMN-Modellen definiert. Dazu werden zum einen Beschreibungen in der BPMN-Spezifikation geliefert, zum anderen werden seitens OMG zwei Dokumente angeboten (mit den Erweiterungen CMOF und XSD), welche eine XML-Repräsentation des gesamten BPMN-Metamodells darstellen. Ursprünglich wurde von der Object Management Group nur die Serialisierung mittels XMI angeboten. Viele XML-Tools bieten allerdings nur die Serialisierung

über XSD, was auch als einer der Gründe angesehen werden kann, weshalb XSD als der industrierelevante Serialisierungsstandard angesehen wird (vgl. Silver, BPMS Watch, 2012). Aus diesem Grund haben sich einige der Gremienmitglieder dafür eingesetzt, dass die BPMN 2.0 auch als XSD-Dokument zur Verfügung gestellt wird. Auch im Rahmen dieser Abschlussarbeit ist der XSD-Standard zur Deserialisierung und Serialisierung gewählt worden.

Der Vorgang der XML-Serialisierung eines BPMN-Modells besteht aus einer Abfolge von Klassen- und Eigenschaftsdefinitionen, die anschließend der *XmlSerializer*-Klasse als Argument der *serialize()*-Methode übergeben werden. Das so serialisierte BPMN-Modell beinhaltet zwei Definitionsteile, zum einen die Prozess-Definition (in BPMN-XML als *Process* Tag) und zum anderen die Diagramm-Definition (in BPMN-XML als *BPMNDiagram* Tag). Die Prozess-Definition stellt die Semantik und die Beziehungen zwischen den Modellelementen dar und enthält keine grafischen Informationen über die Diagramm-Elemente.

Der Teil „Diagramm-Definition“ beinhaltet dagegen ausschließlich grafische Informationen, zum Beispiel die Anordnung der Objekte auf dem Diagramm, ihre Position auf der X-, Y-Koordinate usw. Um das ganze BPMN-Diagramm darzustellen, benötigt ein Diagrammtool sowohl das Prozessdefinitionsteil (*ProcessDefinition)* als auch das *DiagramDefinition* Teil.

Um die Konsistenz zu gewährleisten, ist es nicht erlaubt, dass ein Element aus dem Prozessdefinitionsteil auf mehr als ein Element aus dem Diagrammteil referenziert. Das ergibt sich aus den obengenannten Tatsachen, nämlich, dass ein XML-Modell aus zwei Hauptteilen besteht, die aber jeweils unterschiedliche Perspektiven auf das Modell darstellen.

Insgesamt besteht das BPMN 2.0 Metamodell aus fünf Dokumenten, die jeweils unterschiedliche Teilbereiche abdecken. Sowohl für den semantischen Teil als auch für den grafischen Teil der Spezifikation wurde von OMG ein extra XML-Schema entwickelt und als Dokument freigegeben. Das sind zum einen das Semantic.xsd-Dokument und zum anderen das BPMNDI.xsd-Dokument.

Das BPMN20.xsd Schema fügt alle semantischen und grafischen Informationen in ein Elternelement zusammen, genannt *definitions* (vgl. Object Management Group, 2007).

Der Diagram Definition (DD) Standard, der ebenfalls von OMG entwickelt wird und verschiedene Arten von grafischen Elementen wie Knoten, Kanten und Labels darstellt, wurde bisher nur in der BPMN-Definition eingesetzt, ist aber generisch gehalten, so dass er für die Serialisierung anderer OMG-Modellierungssprachen benutzt werden kann.

Das Dokument BPMNDI.xsd greift auf Definitionen von DI.xsd und DC.xsd zurück, wobei DI im XML-Schema für Diagramaustausch („Diagram Interchange“) steht und DC für „Diagram Common“, was ein Teil von DD ist und Elemente davon definiert. Die Dokumente DC.xsd und DI.xsd gehören zu einer vorläufigen Version von Diagram Definition und sind der aktuellen Version von BPMN 2.0 hinzugefügt (vgl. Object Management Group, 2007). Das Dokument BPMNDI.xsd erweitert die in DI.xsd und DC.xsd definierten Strukturen um zusätzliche, BPMN-spezifische Attribute.

Jede BPMN-Schema-Datei ist mit einem Namensraum (target namespace) assoziiert. Elemente der XML-Struktur, die in diesem Namensraum definiert sind, werden mit einem entsprechenden Präfix versehen. Syntaktisch gesehen ist das eine URI, deren Zweck es lediglich ist, dem in dem Schema definierten Element einen global identifizierbaren Namen zu geben. Somit ist die eindeutige Verwendung von Tags in XML-Dokumentinstanzen möglich, die in unterschiedlichen Schema-Dokumenten definiert sind.

Das BPMN 2.0 Metamodell verwendet für die Serialisierung unterschiedliche Namensräume. Elemente und Typen sind einem Namensraum, in Abhängigkeit des Kontextes, zu dem sie gehören, zugeordnet. Was zum Beispiel sehr spezifisch für den BPMN 2.0 Kontext ist, wird dem BPMN 2.0 Namensraum http://www.omg.org/spec/BPMN/20100524/MODEL zugeordnet. Dazu gehört alles, was in Semantic.xsd definiert ist. Andererseits sind alle Typen und Elemente, die in der BPMNDI.xsd definiert sind, Teil des Namensraums BPMN 2.0 Diagram Interchange: http://www.omg.org/spec/BPMN/20100524/DI.

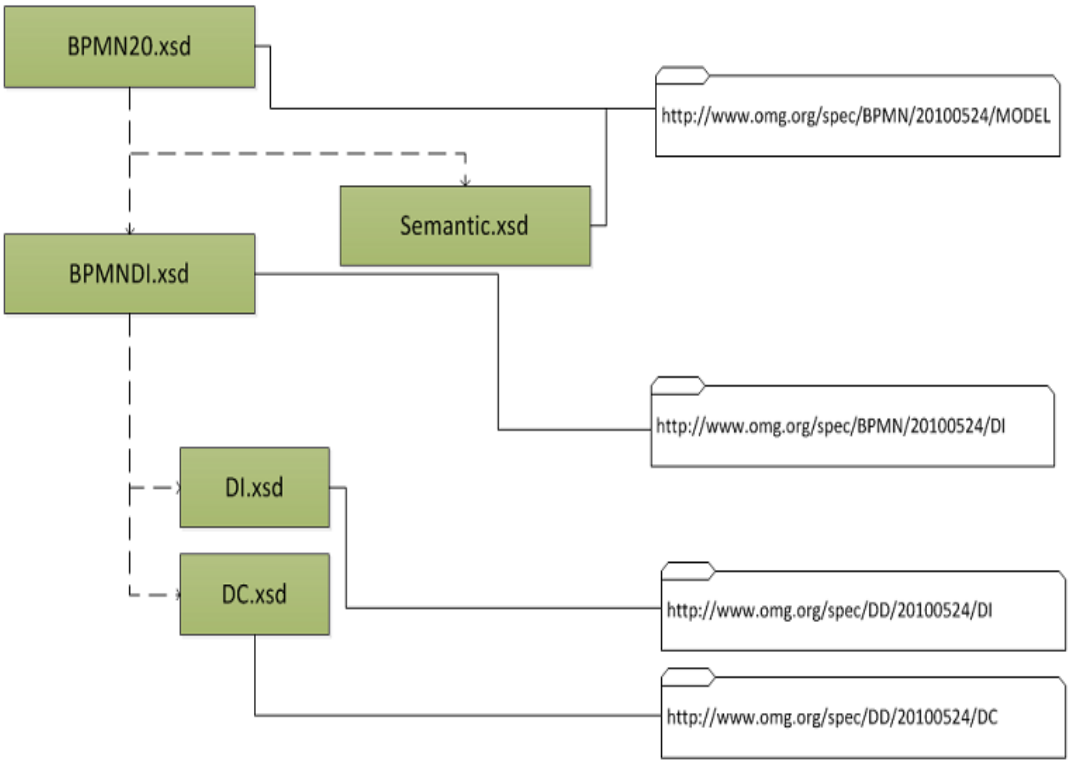


Abbildung 33 BPMN 2.0 Schema-Dokumente und deren Beziehungen untereinander (Silver, 2011)

Die Abbildung 33 zeigt zusammenfassend die BPMN 2.0 XML-Schema-Dateien mit ihren Abhängigkeiten und entsprechenden Namensräumen. Jede einzelne XSD-Datei stellt ein eigenständiges Modell dar. Es ist zu erkennen, dass die Modelle hierarchisch aufgebaut sind. Ein Pfeil von BPMN20.xsd zu BPMNDI.xsd bedeutet, dass das Dokument BPMN20.xsd Elemente verwendet, die im Dokument BPMNDI.xsd definiert sind. Die Dokumente DI.xsd und DD.xsd haben ihre eigenen Namensräume; sie referenzieren nicht auf BPMN, stattdessen sind sie Teil einer anderen Spezifikation DD Diagram Definition.

Listing 10 zeigt das Beispiel eines einfachen BPMN-Prozesses, serialisiert in BPMN-XML. Der *definitions* Tag umschließt alle anderen Elemente und enthält, wie oben beschrieben, sowohl den Semantikbereich als auch den Präsentationsbereich. Der Semantikbereich ist in diesem Fall innerhalb des *process* Tags ausgedrückt, was aber nicht immer der Fall sein muss. Werden z.B. zusätzlich Swimlanes zum Diagramm hinzugefügt, so existiert aus Sicht der BPMN eine Kollaboration. Dementsprechend wird vor dem Prozess-Tag ein *collaboration* Tag mit allen Teilnehmern als Kindelemente erstellt.

Der Präsentationsbereich, durch den Tag *BPMNDiagram* umschlossen, befindet sich in der unteren Hälfte der XML-Struktur und beinhaltet ausschließlich die grafischen Informationen aller im Diagramm befindlichen Symbole, wie Position, Größe usw. Jedes Element des Präsentations-Bereichs verweist über das Attribut *bpmnElement* auf ein Element aus dem semantischen Teil.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL">
   <process id="569c9c96b316" isExecutable="false">
      <startEvent id="1794C094019F" name="">
         <outgoing>87E98A1367D0</outgoing>
      </startEvent>
      <task completionQuantity="1" id="DB2EECE54D11"
      isForCompensation="false" name="" startQuantity="1">
         <incoming>87E98A1367D0</incoming>
         <outgoing>A3E43BF067BB</outgoing>
      </task>
      <endEvent id="87E956D1AFED" name="">
         <incoming>A3E43BF067BB</incoming>
      </endEvent>
      <sequenceFlow id="87E98A1367D0" name=""
        sourceRef="1794C094019F" targetRef="DB2EECE54D11"/>
      <sequenceFlow id="A3E43BF067BB" name=""
        sourceRef="DB2EECE54D11" targetRef="87E956D1AFED"/>
   </process>
   <bpmndi:BPMNDiagram id="b1ca38d81fde">
     <bpmndi:BPMNPlane bpmnElement="569c9c96b316" id="b2104eca84b2">
        <bpmndi:BPMNShape bpmnElement="1794C094019F" id="1794C094019F_gui">
           <omgdc:Bounds height="30.0" width="30.0" x="315.0" y="240.0"/>
        </bpmndi:BPMNShape>
        <bpmndi:BPMNShape bpmnElement="DB2EECE54D11" id="DB2EECE54D11_gui">
           <omgdc:Bounds height="80.0" width="100.0" x="390.0" y="215.0"/>
        </bpmndi:BPMNShape>
        <bpmndi:BPMNShape bpmnElement="87E956D1AFED" id="87E956D1AFED_gui">
           <omgdc:Bounds height="28.0" width="28.0" x="535.0" y="241.0"/>
        </bpmndi:BPMNShape>
        <bpmndi:BPMNEdge bpmnElement="87E98A1367D0" id="87E98A1367D0_gui">
           <omgdi:waypoint x="345.0" y="255.0"/>
           <omgdi:waypoint x="390.0" y="255.0"/>
        </bpmndi:BPMNEdge>
        <bpmndi:BPMNEdge bpmnElement="A3E43BF067BB" id="A3E43BF067BB_gui">
           <omgdi:waypoint x="490.0" y="255.0"/>
           <omgdi:waypoint x="535.0" y="255.0"/>
       </bpmndi:BPMNEdge>
    </bpmndi:BPMNPlane>
   </bpmndi:BPMNDiagram>
</definitions>
```

Listing 10 Ein BPMN 2.0 Prozessmodell

Mit diesem Kapitel sind die Grundlagen der XML-Sprache und der Meta Object Facility (MOF) sowie das BPMN 2.0 Metamodell erläutert worden. Außerdem

wurde einen Überblick über die Serialisierung von Objekten und die C# Serialisierung in XML gegeben. Der folgende Abschnitt widmet sich einer der Technologien, die QUAM zugrunde liegen, der SharePoint-Technologie.

3.4 Grundlagen von SharePoint

SharePoint ist eine Technologie von Microsoft für jegliche Aspekte des Informationsmanagements. Sowohl das Speichern, Bereitstellen und Suchen als auch die Integration und Konsolidierung von Informationen werden von SharePoint unterstützt.

Gegenwärtig beinhaltet die SharePoint Produktpalette die Microsoft SharePoint Foundation (Nachfolger von Windows SharePoint Services) und den Microsoft SharePoint Server (MOSS) 2010, der in verschiedenen Varianten angeboten wird. Bereits die kostenfreie SharePoint Foundation beinhaltet zahlreiche Funktionen, die ohne zusätzlichen Programmieraufwand eingesetzt werden können. Beispiele dafür sind das Dokumentenmanagement, das Content-Management, das Daten-Management und die SharePoint Foundation Workflows (vgl. Boddenberg, 2011, S. 17).

Die Technologie stellt verschiedene Schnittstellen zu Produkten von Microsoft oder Drittanbietern (wie z.B. SAP) zur Verfügung. Dadurch können Daten in andere Produkte importiert, in diesen weiterverarbeitet und aus ihnen exportiert werden. Wenn der Funktionsumfang von SharePoint nicht ausreicht, werden sog. Webparts eingesetzt, die mittels des SharePoint Designers oder über Microsoft Visual Studio entwickelt werden.

SharePoint verzeichnet mittlerweile einen weltweit hohen Verbreitungsgrad, was sich in zahlreichen Studien namhafter Marktanalyse-Unternehmen widerspiegelt (vgl. Miles, 2011; Koplowitz, 2010). Abbildung 34 zeigt einen Funktionsüberblick der SharePoint Technologie.

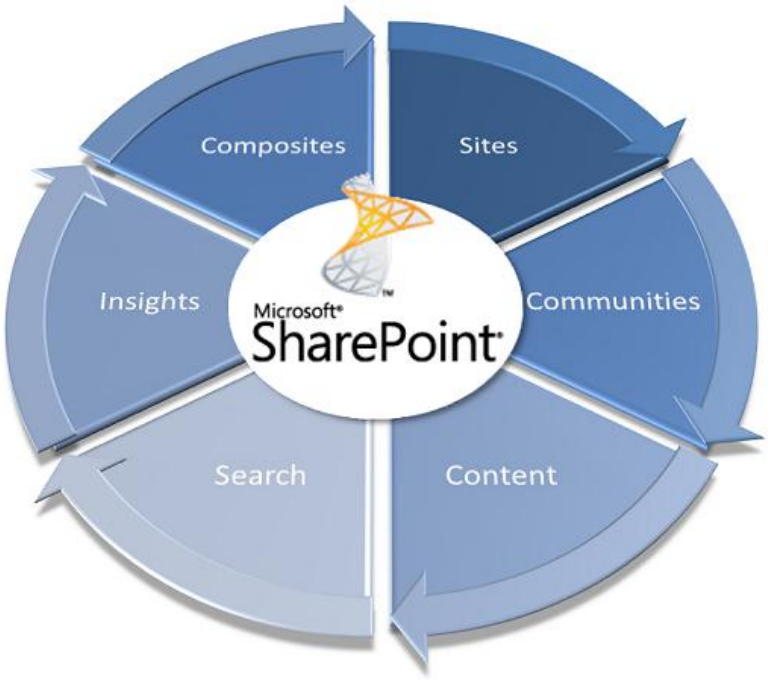


Abbildung 34 Funktionsüberblick der SharePoint Technologie (Microsoft, 2010)

Die sechs dargestellten Bereiche, die SharePoint abdeckt, werden im Folgenden skizziert (vgl. EVIATEC):

- Speichern und Teilen von Informationen ("Sites")
- Zusammenarbeit ("Communities")
- Inhaltsverwaltung über den gesamten Lebenszyklus ("Content")
- Informationen suchen und finden ("Search")
- Entscheidungstreffen auf Basis relevanter Informationen ("Insights")
- Geschäftsanwendungen einfach bereitstellen ("Composites")

Ein kennzeichnendes Merkmal von SharePoint ist, dass es sich dabei um eine Art Baukastenlösung handelt, auf deren Grundlage verschiedenste Informationssysteme entwickelt werden können.

Bei der Erstellung eines SharePoint-Systems sollte der Entwickler zunächst die SharePoint Vorlagen identifizieren, die der formulierten Aufgabenstellung am ehesten entsprechen. Zur Ergänzung dieser Vorlagen können anschließend vorgefertigte Komponenten (wie z.B. Listen, Webparts, Bibliotheken) ausgewählt und hinzugefügt werden. Diese Schritte finden webbasiert statt und erfordern keinen Entwicklungsaufwand.

3.4.1 SharePoint Foundation

SharePoint Foundation bietet ein integriertes Angebot von Diensten, das die Kommunikation und Kollaboration in Unternehmen unterstützt. Ziel ist es, über eine Weboberfläche Menschen und Informationen mit Prozessen und Systemen zu verbinden. Bspw. hat der Anwender die Möglichkeit, für eine Projektdokumentation Dokumente mit einer mehrstufigen Versionierung zu versehen, um mehrere Haupt- und Nebenversionen davon zu erzeugen. Weiterhin bieten Funktionen wie Wikis, Blogs, E-Mail Integration, Synchronisierung von Outlook Kalendereinträgen eine weitere Unterstützung der Teamarbeit (vgl. Microsoft, 2010).

3.4.2 Die SharePoint Architektur

SharePoint baut auf die Microsoft ASP.NET Technologie als Plattform für Webapplikationen auf, wobei zur Speicherung von Daten der Microsoft SQL Server verwendet wird (vgl. Krause, 2010). Eine SharePoint Webanwendung besteht aus einer oder mehreren Websitensammlungen. Jede Webseite besteht aus mehreren Unterwebseiten, welche über eine zentrale Navigationsleiste erreicht werden können. Webseiten bestehen in erster Linie aus Listen, die Informationselemente enthalten. Diese Informationselemente werden durch Felder mit Eigenschaften attributiert (vgl. Microsoft, 2010). Abbildung 35 zeigt eine Übersicht über die SharePoint Foundation Websitearchitektur. Die einzelnen Punkte werden anschließend kurz erläutert.

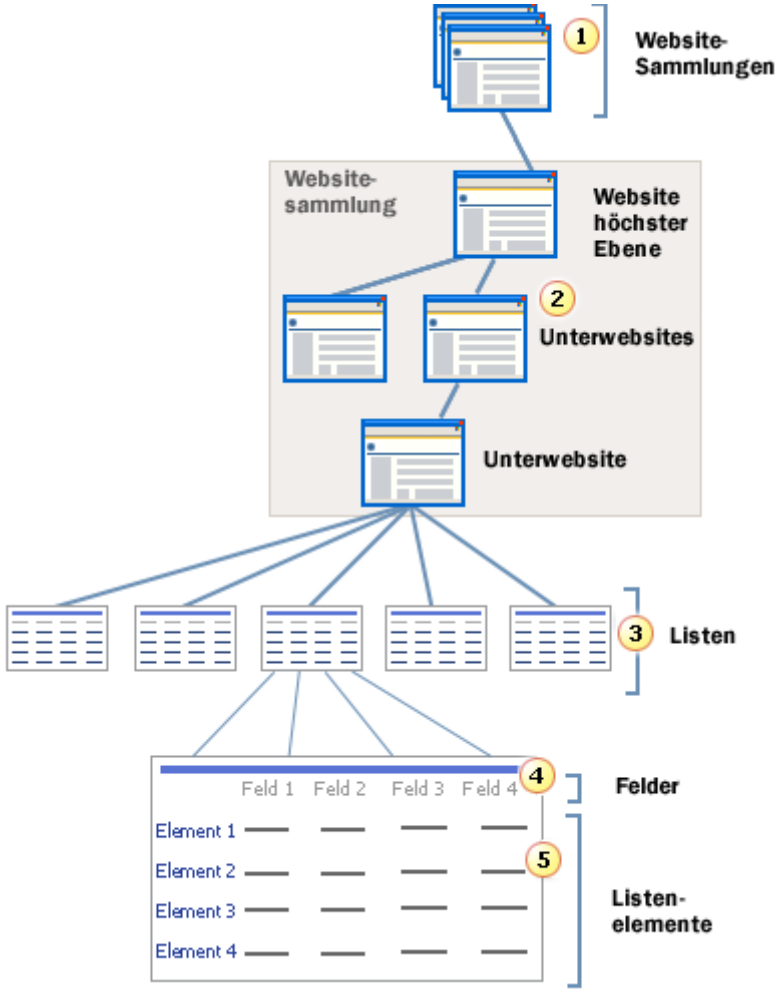


Abbildung 35 SharePoint Architektur (vgl. Microsoft, 2010)

1. Website-Sammlungen

Jede Klasse vom Typ *SPSite* stellt eine Gruppe von logisch zusammengehörenden *SPWeb*-Objekten dar. Eine solche Gruppe heißt allgemein "Websitesammlung“ und enthält Eigenschaften, die zur Verwaltung der Websitesammlung verwendet werden können. Beispielsweise bietet die Eigenschaft „*AllWebs*“ den Zugriff auf eine Auflistung aller Websites innerhalb der Websitesammlung, einschließlich der Website auf oberster Ebene. Auch der Zugriff auf eine konkrete Website ist möglich (Microsoft, 2010).

2. Die Klasse *SPWeb*

Zu jeder Websitesammlung existiert eine beliebige Anzahl von „*SPWeb“*-Objekten, von denen jedes Eigenschaften enthält, mit denen eine Website einschließlich der zugehörigen Vorlage verwaltet wird und auf Dateien und Ordner in der Website zugegriffen werden kann. Beispielsweise gibt die Webs-

Eigenschaft ein Objekt zurück, das alle Unterwebsites einer angegebenen Website darstellt. Die Eigenschaft „*Lists*“ gibt eine Aufzählung aller Listen der aktuellen Website zurück.

3. **SharePoint Liste**

Ein Objekt vom Typ *SPList* enthält Eigenschaften, die zum Verwalten der Liste oder für den Zugriff auf Listenelemente verwendet werden. Auf einer Liste können Abfragen ausgeführt werden, die bestimmte Listenelemente zurückgeben (Microsoft, 2010).

4. **Listenspalten**

Anders als in der Datenbanktheorie, in der die Zeilen einer Tabelle als Felder bezeichnet werden, sind bei SharePoint mit Feldern die Spalten der Tabelle gemeint. Die *SPField* ist die Basisklasse jeder Spalte in SharePoint. Über die Felder der Klasse kann die Spalte programmatisch angepasst werden.

5. ***SPListItem*-Objekt:** Das *SPListItem*-Objekt stellt eine einzelne Zeile in der Liste dar.

SharePoint-Webseiten sind aus einzelnen Bausteinen zusammengestellt, wobei Webparts die eigentlichen Bausteine darstellen. Webparts können aus einem WebPart-Katalog in Webseiten eingebunden werden.

Listen dienen in SharePoint zur Informationsspeicherung und werden als Tabellen visualisiert. Sie sind jedoch keine herkömmlichen Tabellen, wie in der Datenbankentheorie, sondern virtuelle Strukturen, die zur Laufzeit aus einer Menge von Daten aufgebaut werden (vgl. Microsoft, 2010). SharePoint bietet die Möglichkeit, Listen aus einem Katalog vordefinierter Listentypen zu erstellen oder eigene Listentypen zu definieren und daraus Listeninstanzen zu erzeugen. Vordefinierte Listentypen existieren z.B. für die Darstellung von Ankündigungen, Kalender, Kontakte, Hyperlinks.

Für das Speichern gemeinsamer Dokumente oder allgemeiner Dateien bietet SharePoint die sogenannten Dokumentenbibliotheken. Eine Dokumentenbibliothek stellt eine spezielle Liste mit zusätzlichen Funktionen (Dokumentenupload, Versionierung usw.) dar. SharePoint Listen können in Microsoft Excel, Access oder Outlook (bei Listen mit Kontaktdaten) geöffnet,

bearbeitet und weiterverwendet werden. Jeder SharePoint Anwender kann über E-Mail benachrichtigt werden, wenn sich eine Liste, Einträge oder beliebige Eigenschaften der Liste ändern.

3.4.3 SharePoint als BPM Plattform

Geschäftsanwender und Mitarbeiter der IT-Abteilungen unterliegen der Pflicht, Dokumente und Daten effizient zu verwalten. Aktuelle Studien zeigen eine wachsende Akzeptanz von SharePoint als Plattform im Bereich der Kollaborationssoftware, sowohl bei IT-Leitern als bei Mitarbeitern aus den Fachabteilungen (vgl. Miles, 2011).

Den Einsatz von SharePoint als BPM Plattform wird aber dann als zweckmäßig angesehen, wenn eine Einbeziehung von Drittanbietern mit vorhandenem Knowhow in diesem Bereich stattfindet (vgl. Miers, 2010).

Mit SharePoint 2010 wurden folgende Neuerungen und Werkzeuge eingeführt, die den Entwicklungsprozess von BPM-Lösungen für SharePoint unterstützen sollen:

1. **Neuerungen SharePoint Designer**: Mit SharePoint Designer wird es Mitarbeitern aus den Fachabteilungen ermöglicht, einfache Workflows zu erstellen. Darüber hinaus können Administratoren, im Unterschied zu früheren SharePoint Designer Versionen, bestimmte Funktionen oder den ganzen Zugriff für bestimmte Anwender ausschalten.
2. **Direkter Import von Visio-Workflow-Diagrammen**: Der Anwender hat die Möglichkeit, Workflows in Visio 2010 zu modellieren und sie dann in den SharePoint Designer importieren. Von dort aus können die Workflow-Diagramme mit prozessspezifischen Informationen versehen werden. Anschließend können diese Workflows in SharePoint publiziert werden.
3. **Neuerungen Visual Studio:** Die neue Version Visual Studio 2010 integriert Funktionen und Vorlagen zur Anpassung und Entwicklung von SharePoint Lösungen.

Im nächsten Abschnitt wird eine weitere Technologie betrachtet, auf die QUAM als BPM-System aufsetzt - Microsoft Visio. Im Anschluss daran wird das QUAM System mit seinen Eigenschaften betrachtet.

3.5 Visio und sein Objektmodell

Microsoft Visio wird in Organisationen als Plattform für die einheitliche Modellierung und den Tausch von Diagrammen eingesetzt. Dabei fokussiert die Anwendung die standardisierte Erstellung von Geschäftsdiagrammen, dieses Merkmal unterscheidet sie von klassischen Zeichnungsprogrammen (vgl. Lelic, 2003). Die Geschäftsdiagramme werden in Visio erstellt, indem der Benutzer aus einer Menge von vordefinierten Zeichnungskomponenten per Drag&Drop-Verfahren ein Modell erstellt. Auf Grund der Ähnlichkeit mit dem Baukastenprinzip wird Visio z.T. als eine Art Software-Lego bezeichnet (vgl. Lelic, 2003).

Visio stellt nicht die Zeichnungsfunktionalitäten in den Vordergrund, sondern die Möglichkeit, vorgefertigte Zeichnungsobjekte zu kombinieren, Informationen nonvisueller Art in Shapes zu speichern und diese zu visualisieren. Das Ergebnis ist ein Abbild der Realität (Modell) statt einer klassischen Zeichnung. Die Diagrammelemente können mit einer gewissen „Intelligenz“ ausgestattet werden und bspw. auf Aktionen der Modellierer reagieren (Lelic, 2003).

Derzeit wird Visio in drei unterschiedlichen Versionen angeboten, Visio 2010 Standard, Visio 2010 Professional und Visio 2010 Premium. Die Unterschiede liegen in den angebotenen Diagrammvorlagen, Anwendungsfunktionalitäten, Programmbibliotheken usw.

Grundlagen der vorliegenden Arbeit sind das Programmiermodell von Visio Standard und die Diagrammvorlagen von Visio Premium.

3.5.1 Grundlegende Visio-Begriffe

Shape, SmartShape und MasterShape

Jedes Objekt auf einem Zeichenblatt in Visio wird Shape genannt; unabhängig davon, um welches Objekt es sich handelt (Zeichnungsblatt, Zeichnungsseite

oder Symbol auf Zeichnungsblatt). Shapes können programmatisch erweitert werden. Wird ein Shape mit Shape-Programmierung derart verändert, dass es mehr als die rein visuellen Informationen enthält (zum Beispiel auf Aktionen des Anwenders reagieren), so wird aus dem Shape ein SmartShape (vgl. Lelic, 2003).

Alle Shapes der BPMN 1.2 Vorlage in Visio sind SmartShapes, da sie eine erhebliche Menge an vielfältigen Daten beinhalten.

Befindet sich ein Shape in einer Schablone, so handelt es sich dabei um ein MasterShape. Von einem Shape spricht man erst dann, wenn es sich auf der Zeichnungsfläche befindet. Jedes MasterShape hat eine Menge von Eigenschaften, die von den Shapes geerbt werden. Diese Eigenschaften können später geändert werden; auch andere Eigenschaften können definiert werden (vgl. Martin, 2011).

Schablone (Stencil)

Visio Shapes sind nicht als Einzeldateien abgespeichert, sondern werden in Containerdateien abgelegt, die Schablonen genannt werden. Jede Schablone beinhaltet eine bestimmte Menge von Shapes, die logisch zu einem bestimmten Themengebiet gehören.

Vorlage (Template)

Ähnlich wie in Textverarbeitungsanwendungen können auch Visio-Zeichnungen auf Vorlagen basieren. Jede Vorlage kann mit einer bestimmten Menge von Schablonen versehen werden, die thematisch zusammenpassen.

Das ShapeSheet

Das ShapeSheet spielt bei der Shape-Programmierung in Visio eine wesentliche Rolle. Es stellt eine Art Behälter dar, in dem die gesamten Shape-Parameter abgelegt werden, die von der Visio-Engine für das Diagrammerzeugen ausgewertet werden (vgl. Martin, 2011).

Dokumenttypen in Visio

Visio verwendet ein eigenes Binärformat, in der Regel mit der Erweiterung *.vsd. Mit der weiten Verbreitung der XML-Sprache wurde das XML-Format eingeführt. Dokumente, die in diesem Format abgespeichert sind, haben die

Endung *.vdx. Das VDX-Format benötigt erheblich mehr Speicherplatz als das *.vsd Format, da es die Struktur der verwendeten Grafiken und Vererbungsbeziehungen zwischen den Elementen innerhalb des Dokuments beschreibt (Parker, 2010). Die *.vdw Dateien wurden in Visio 2010 eingeführt; sie sind Visio Web-Zeichnungen, die in SharePoint 2010 publiziert werden können. Diese Web-Zeichnungen können z.B. SharePoint Elemente visualisieren, wobei sie auch dynamisch in dem Sinne sind, dass eine automatische Aktualisierung und Anpassung an die Daten ermöglicht wird. Visio Dokumente können ihren Arbeitsbereich innerhalb der Datei speichern, d.h. es existiert nahezu immer eine Sammlung von angedockten Schablonen, die die Master-Shapes enthalten. Eine Visio Schablone (oft auch Visio-Stencil) ist ein Visio Dokument mit versteckten Seiten; es wird üblicherweise mit der Erweiterung *.vss oder *.vsx (XML-Schablone) abgespeichert (Parker, 2010).

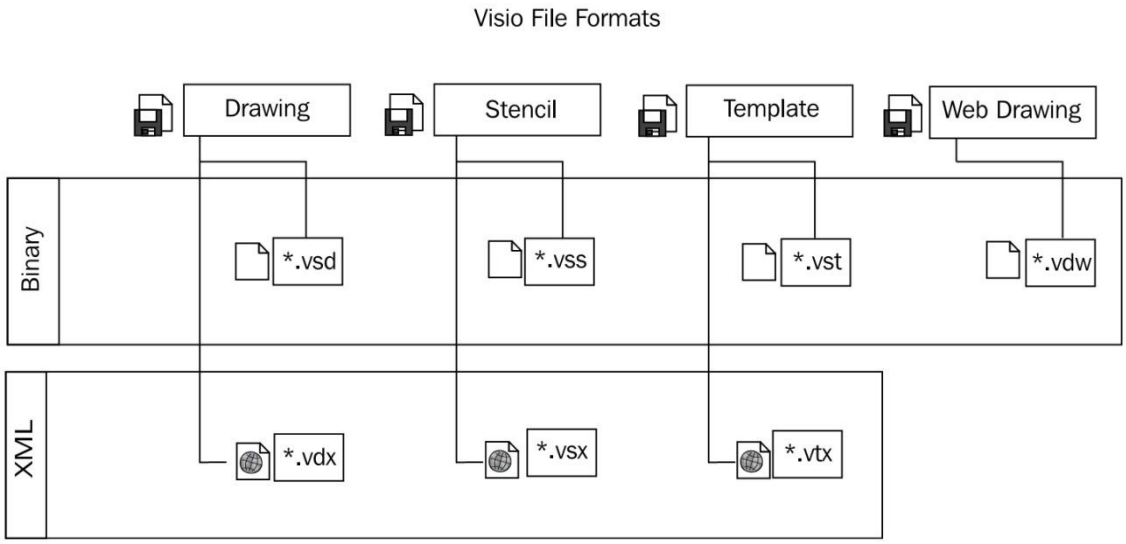


Abbildung 36 Visio Formate (vgl. Parker, 2010)

3.5.2 Das Objektmodell von Visio

Visio verfügt über eine Engine für die Diagrammerstellung, die Möglichkeiten zur Verfügung stellt, Daten und Funktionen zu jedem Diagrammobjekt (Diagrammobjekte werden in Visio Shapes genannt) zu kapseln. Außerdem besteht die Möglichkeit zur Vererbung von bestimmten Objekttypen. Um diese

ganze Funktionalität zu liefern, entsteht eine äußerst komplexe Struktur des Objektmodells (Parker, 2010 S. 28).

Grundsätzlich installiert Visio eine Vielzahl von Programmbibliotheken, von denen hier nur diejenigen betrachtet werden, die für diese Arbeit von Bedeutung sind.

Wichtige Teile des Visio Objektmodells

Die für die Umsetzung des praktischen Teils dieser Abschlussarbeit zentralen Komponenten des Visio Objektmodells werden nachfolgend dargestellt.

Das *Application* Object

Die Mehrheit der Objekte in Visio stammt aus dem *Application* Objekt. Dazu zählen alle aktiven Diagrammobjekte, von denen zwei für diese Arbeit, speziell für das Durchlaufen von Prozessdiagrammen, von besonderer Bedeutung sind: *ActiveDocument* und *ActivePage*.

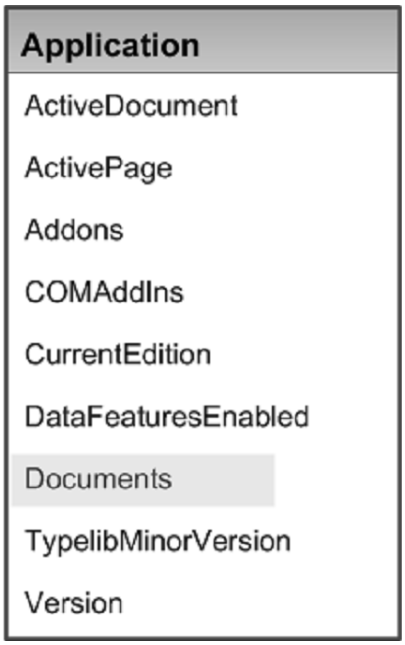


Abbildung 37 Das Application Objekt mit seinen Eigenschaften (vgl. Parker, 2010)

ActiveDocument und *ActivePage*

Die zwei Objekte ***ActiveDocument*** und ***ActivePage*** sind über das *Globals*-Objekt referenzierbar. Greift man über eine Office Erweiterung (z.B. über das Add-In dieser Arbeit) auf die zwei Objekte zu, so erfolgt der Zugriff über das Application Objekt (Parker, 2010 S. 37).

```
var m_vsoPage = Globals.ActivePage;
```

Listing 11 Zugriff aus ActivePage über das Globals Objekt

```
var m_vsoPage = VisioAddIn.Globals.ThisAddIn.Application.ActivePage;
```

Listing 12 Zugriff auf ActivePage über das VisioAddIn Objekt

Die *ComAddIns* Sammlung

Die Sammlung *COMAddIns* ist Teil der Microsoft Office 14.0 Objektbibliothek und sollte im Entwicklungsprojekt von Visual Studio referenziert werden, damit die Objekte zur Verfügung stehen und *IntelliSense* benutzt werden kann.

Die *Documents*-Sammlung und das *Document* Objekt

Die ***Documents***-Sammlung enthält alle Schablonen und Diagramme, die aktuell geöffnet sind. Die Sammlung Application.***Documents*** umfasst mehrere Objekte vom Typ ***Document***. Ein Objekt vom Typ ***Document*** beinhaltet eine Sammlung von ***DataRecordsets***, ***Masters***, ***Pages*** und anderen Eigenschaften, die für die Arbeit mit Prozessdiagrammen nützlich sind Abbildung 38.

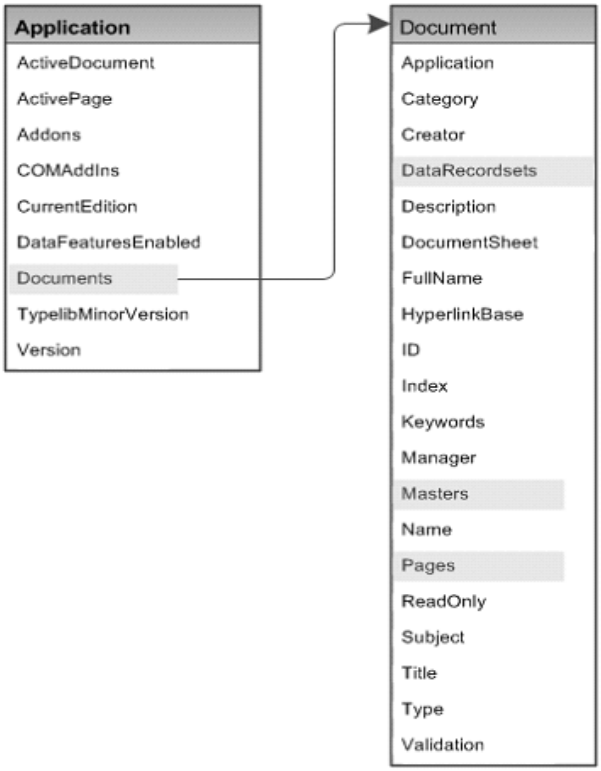


Abbildung 38 Das *Document*-Objekt mit seinen Eigenschaften (vgl. Parker, 2010)

Die Eigenschaften *Name und NameU*

Mit der Eigenschaft ***Name*** wird auf den angezeigten Shape-Namen zugegriffen, während ***NameU*** den internen Namen darstellt.

Das ***Page***-Objekt

Das *Page*-Objekt beinhaltet das wichtigste Objekt des Visio Objektmodells - das *Shape*. Darin sind auch *Connects* und *Layers Objekte* enthalten, die aber für diese Arbeit keine Relevanz hatten.

Das *Shape*-Objekt

Das ***Shape*** Objekt ist das wichtigste aller Objekte des Visio Objektmodells; mit seinen zugehörigen Abschnitten, Zeilen und Zellen ist es zugleich auch das umfangreichste. Die Shape-Objekte werden in ein- und zweidimensionale Shapes unterteilt. Dies wird durch die ***OneD***-Eigenschaft eines Shapes ermöglicht; sie liefert wichtige Informationen über die Art eines Shapes. OneD drückt aus, dass das Shape nur eine Dimension hat; es handelt sich dabei um eine Linie. Für die Erstellung der in dieser Abschlussarbeit angestrebten Lösung wurde auf diese Eigenschaft mehrmals zugegriffen, um bspw. zu überprüfen, ob es sich beim aktuell verarbeiteten Shape um einen Prozessknoten oder eine Prozesskante handelt.

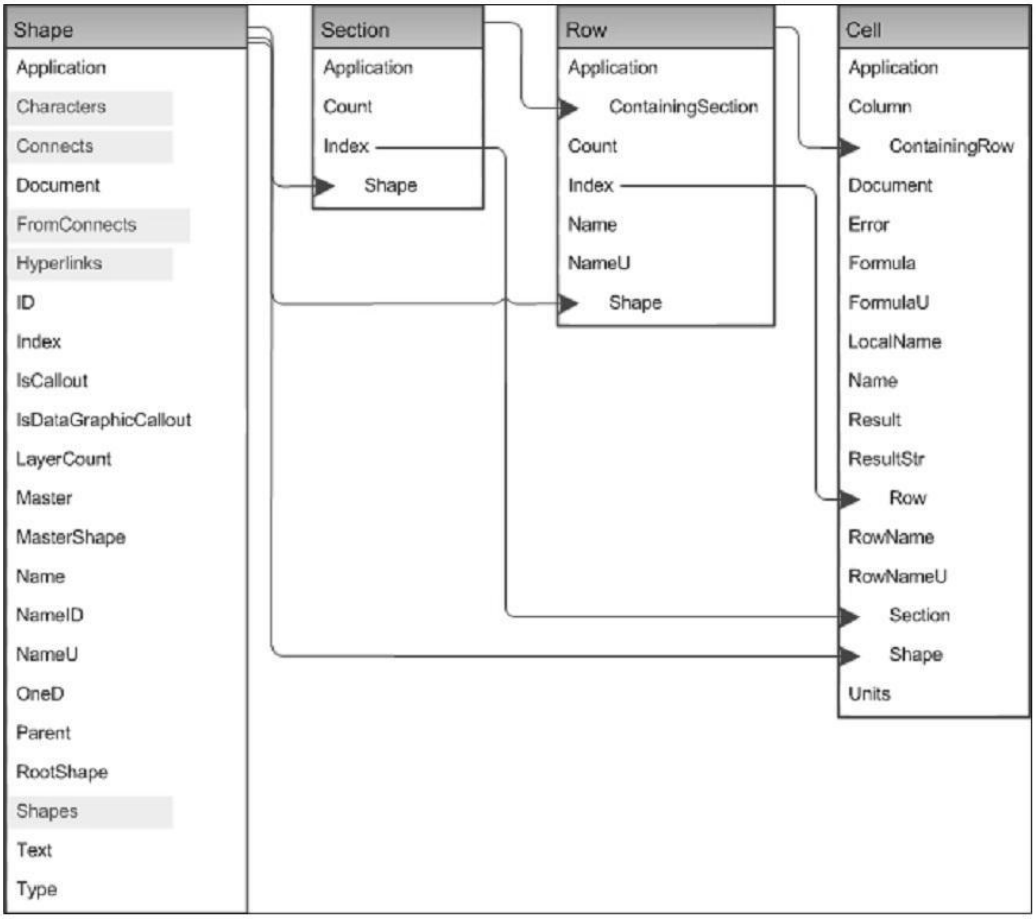


Abbildung 39 Das Shape Objekt mit seinen Eigenschaften (vgl. Parker, 2010)

Das *ShapeSheet*-Objekt

Im Gegensatz zu anderen Grafikanwendungen existieren für ein Shape keine Vektorinformationen, sondern es gibt eine Sammlung von Eigenschaften, die das Aussehen und Verhalten definiert. Diese Eigenschaften sind in einer tabellenartigen Struktur zusammengefasst, dem so genannten ShapeSheet (vgl. Martin, 2011). Alle Informationen aus dem ShapeSheet werden von Visio zur Laufzeit ausgewertet und als Ergebnis grafisch dargestellt. Shape und ShapeSheet sind also identisch; sie repräsentieren ein und dieselbe Datenmenge.

Jede ShapeSheet beinhaltet bestimmte Abschnitte, in Visio „Sections“ genannt. Einige davon sind immer präsent sind; es gibt auch jene, die nur für bestimmte Shapes zur Verfügung stehen. Die fixen Abschnitte repräsentieren generische Daten, die für alle Shapes existieren müssen, wie z.B. Breite, Höhe usw. Die variablen Abschnitte dagegen sind nur bei bestimmten Shapes präsent und von dem Kontext abhängig, in dem das Shape eingesetzt wird. Deshalb ist es ein

wichtiger Schritt - bevor überhaupt auf die Informationen eines Abschnitts zugegriffen wird - zu überprüfen, ob der Abschnitt existiert.

Die in dieser Arbeit am häufigsten benutzten Abschnitte ShapeData und UserDefined sind ein Beispiel für nicht immer vorhandene Abschnitte. Sie beinhalten äußerst wichtige Informationen über BPMN Shapes, wie z.B. Elementtyp, grafische Markierungen, Anzahl von Iterationen einer Aufgabe.

Jeder ShapeSheet-Abschnitt besteht aus einzelnen Zeilen, die wiederum ihre Repräsentation als Objekte haben. Ein Beispiel für einen ShapeSheet Bereich mit BPMN-spezifischen Informationen ist der Abbildung 40 zu entnehmen. Wie auf der Abbildung zu sehen, umfassen Abschnitte Zeilen. Jeder Zeile wiederum sind mehrere Zellen zugeordnet, in denen die eigentlichen Informationen abgespeichert werden. Durch die einzelnen Zellen kann iteriert werden, um zu dem gewünschten Dateneintrag zu gelangen.

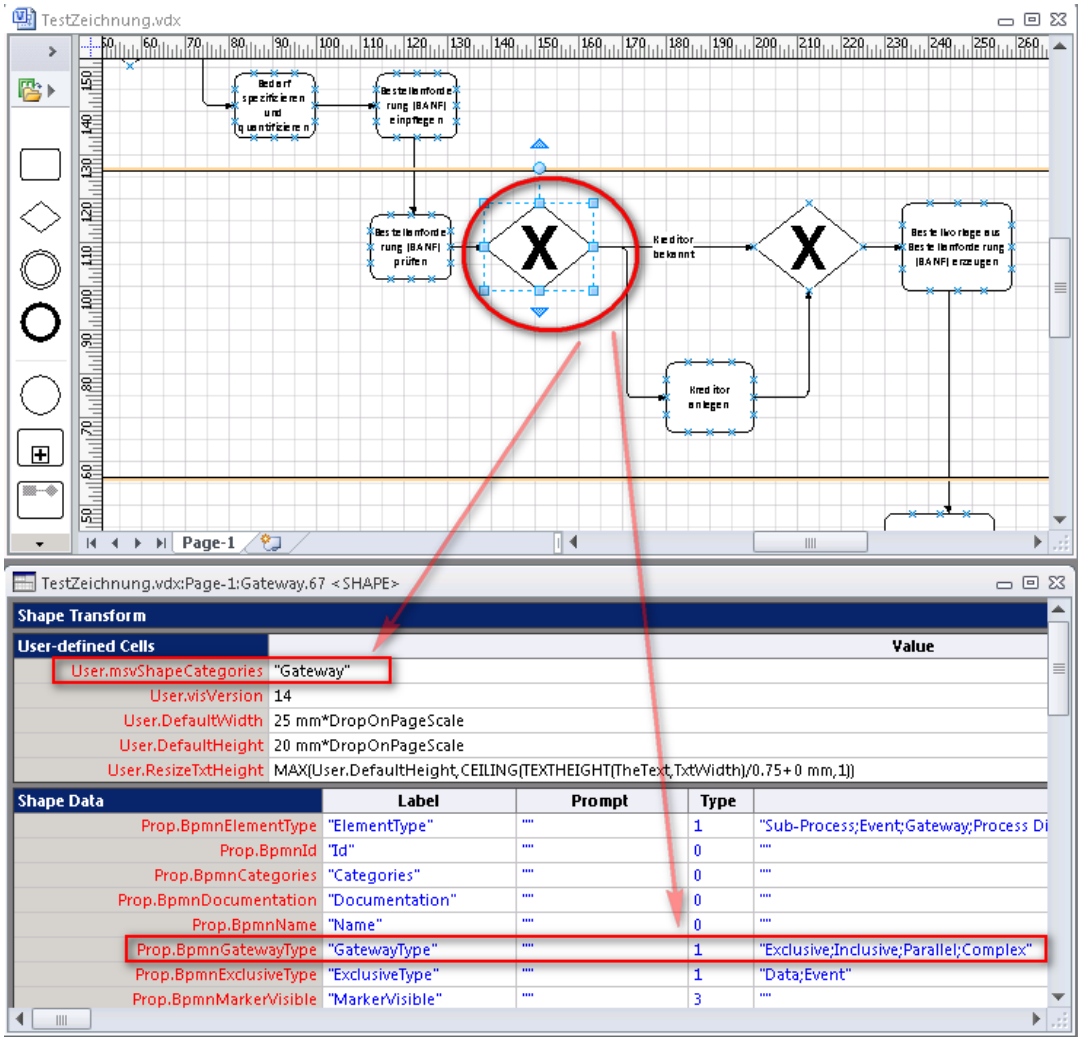


Abbildung 40 BPMN-Diagramm mit dem dazugehörigen ShapeSheet

3.5.3 Das Connectivity API von Visio

Mit der im Jahr 2010 eingeführten **Connectivity API** bietet Visio neue Möglichkeiten zur programmatischen Bearbeitung von Diagrammen. Die einzelnen Shapes von Ablaufdiagrammen können benutzerfreundlich durchlaufen und nach Containereigeschaften abgefragt werden.

*Die Methode **Shape.ConnectedShapes***

Die Methode ConnectedShapes hat in der Connectivity API von Visio einen zentralen Stellenwert. Sie gibt ein Feld mit IDs von Shapes zurück, die durch einen Verbinder mit dem aktuellen Shape verknüpft sind.

Die Methode hat zwei Parameter, die Flags und den CategoryFilter:

- **Flags:** Mit Hilfe von Flags werden die zurückgelieferten Shapes gefiltert. Mittels All, Incoming oder Outgoing wird bestimmt, ob alle eingehenden oder ausgehenden Kanten berücksichtigt werden.
- **CategoryFilter:** Dieser Parameter filtert die zurückgelieferten Shapes nach Kategorien. Auf die Kategorie eines Shapes wird durch das ShapeSheet, mittels User.msvShapeCategories, zugegriffen.

*Die Methode **Shape.GluedShapes***

Eine andere Funktion der Connectivity API, die in der Arbeit von großem praktischen Nutzen war, ist die ***Shape.GluedShapes*** Funktion.

Sie liefert ein Feld mit den ShapeIDs aller direkt mit dem aktuellen Shape verbundenen Shapes. Diese Methode hat folgende Parameter:

- Flags: All1D, All2D, Incoming1D, Incoming2D, Outgoing1D, Outgoing2D
- CategoryFilter: In den CategoryFilter werden nur die ShapeIDs angegeben, die einer bestimmten Kategorie angehören. Die Kategorie wird über das ShapeSheet ermittelt.

Die Eigenschaft „MemberOfContainers“

Mit dieser Eigenschaft wird die Möglichkeit gegeben, bequem mit den sog. Containereigenschaften zu arbeiten. Bestimmte Shapes sind gleichzeitig Container-Shapes, die andere Shapes zu “beheimaten”. Fragt man ein Shape mit der besonderen Eigenschaft Container nach den Shapes ab, die es

beherbergt, so wird ein Feld mit deren IDs zurückgegeben. Man kann auch abfragen, zu welchen Containern ein Shape gehört.

3.5.4 Visio und SharePoint in der Prozessmanagementpalette von Microsoft

Nachdem alle relevanten Konzepte, sowohl aus betriebswirtschaftlicher als auch aus technischer Sicht, erläutert wurden, wird das QUAM als BPM-System mit seinen Funktionen betrachtet. Abschließend wird eine Einordnung von Prozessmanagement Technologien von Microsoft gegeben.

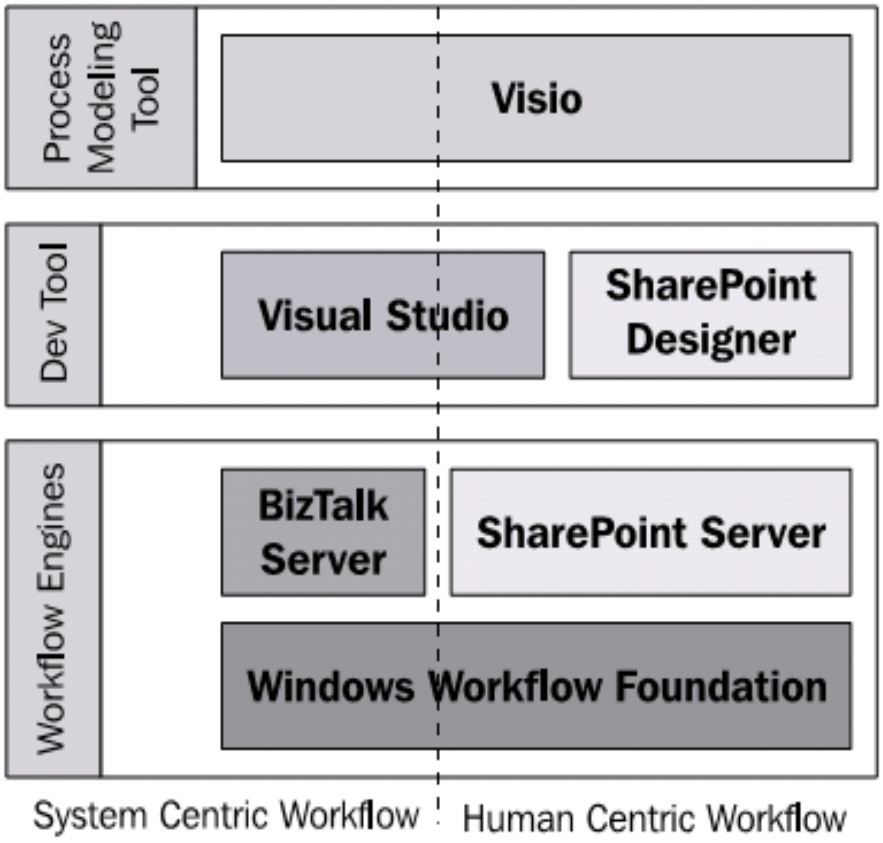


Abbildung 41 Microsoft Process Management Product Stack (vgl. Parker, 2010)

Auf der obersten Ebene der Anwendungen zum Prozessmanagement von Microsoft ist Visio als Modellierungswerkzeug angesiedelt. Mit der ständigen Weiterentwicklung der Visio-Funktionen werden nicht mehr nur Prozessmodellierungsfelder abgedeckt, sondern Möglichkeiten zur Simulation und Gestaltung von Workflows angeboten. Über den Einsatz von Entwicklungswerkzeugen wie Visual Studio und SharePoint Designer lässt sich außerdem der Funktionsumfang von Visio nahezu beliebig erweitern. Dies kann

zum Beispiel notwendig sein, wenn bestimmte Schnittstellen zu Technologien tieferer Ebenen (etwa zum SharePoint Server oder BizTalk Server) geschaffen werden sollen, was beim QUAM-System der Fall ist. Auf der Ebene „Workflow-Engines“ werden zum einen der BizTalk-Server und zum anderen der SharePoint Server positioniert.

3.6 Das SharePoint System QUAM

Das Akronym QUAM steht für Qualitätsmanagement, Umweltmanagement, Arbeitssicherheitsmanagement und Managementregularien. QUAM 2.0 ist eine SharePoint-basierte Anwendung, die von der Firma LINTRA entwickelt wurde. Sie nutzt das Diagrammtool Microsoft Visio, um Organisationsstrukturen, Prozess- und Ablaufmodelle sowie Unternehmensregelwerke zu modellieren, zu visualisieren und miteinander in Beziehung zu setzen. Im QUAM greifen vorhandene Management-Systeme zur Unternehmens- und Projektsteuerung auf ein einziges konsistentes Prozess- und Strukturmodell zurück (vgl. Computerwoche, 2012). QUAM 2.0 organisiert über SharePoint die Zusammenarbeit und die Beziehungen zwischen den Ressourcen und den Kapitalanlagen des Unternehmens; es beachtet dabei automatisch die geltenden Regeln und prüft die Konformität mit vorhandenen Qualitäts-, Risiko- oder Umweltmanagementsystemen. Mit QUAM 2.0 lassen sich automatisch verschiedene Sichten auf die Unternehmensprozesse erzeugen.

Da QUAM auf SharePoint basiert, ist es vollständig webbasiert und damit plattform- und ortsunabhängig. Dadurch können Mitarbeiter auf Informationen über Prozesse und Organisationsstrukturen eines Unternehmens zugreifen. Durch die Flexibilität der SharePoint Technologie kann außerdem eine Anwendung wie QUAM nahezu beliebige funktionale Erweiterungen erfahren. Die Benutzergruppenverwaltung in QUAM erlaubt auch die Anbindung externer Nutzer, etwa Kunden, Kapitalgeber usw. Das wird durch die Erteilung verschiedener Zugriffsberechtigungen ermöglicht.

3.6.1 QUAM Aufbau

Der Aufbau und die Funktionen von QUAM werden nachfolgend anhand von Beispielen dargestellt; dabei werden Screenshots eines QUAM-Demosystems verwendet. Abbildung 42 zeigt die Oberfläche der Startseite des QUAM. Die Navigationselemente und die Inhalte der Startseite können beliebig angepasst werden, wobei erfahrungsgemäß ein Gesamtüberblick der Unternehmensprozesse, z.B. in Form von Prozesslandkarten, als Einstieg angezeigt wird. Zusätzlich kann der Startbereich aber für jeden einzelnen Anwender so angepasst werden, dass mitarbeiterspezifische Informationen (z.B. relevante Prozesse bzw. Prozessschritte, Kalender, Termine, anstehende Ereignisse) enthalten sind.

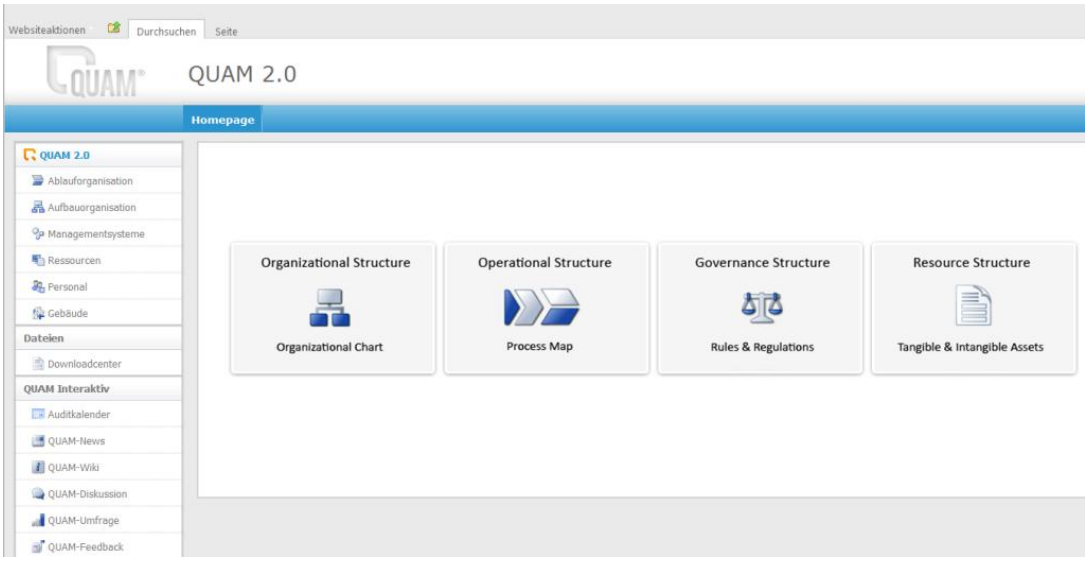


Abbildung 42 Einstieg in das QUAM System

Aus Anwendersicht besteht QUAM aus den folgenden zentralen Bestandteilen:

- Ablauforganisation für die Gestaltung aller Unternehmensprozesse
- Aufbauorganisation für die Gestaltung der Organisationsstrukturen
- Managementsysteme der Organisation
- Ressourcen
- Personal
- Dokumente und Bibliotheken

Außerdem besteht in QUAM die Möglichkeit zur Definition von einfachen oder aber komplexen Workflows auf Basis von Microsoft SharePoint.

3.6.1.1 Liste „Ablauforganisation“

Die Ablauforganisation bietet die grafische Darstellung von Prozesslandkarten, sowie die Abbildung der untergeordneten Prozesse und eine Auflistung der einzelnen Prozessschritte mit den dazugehörigen Informationen. Zentral für diese Liste ist daher die Gestaltung der im Unternehmen ablaufenden Geschäftsprozesse, die analysiert und definiert werden sollen. Der QUAM-Anwender kann (in Abhängigkeit von der Position, die er im Unternehmen besitzt) die Unternehmensprozesse betrachten oder selbst modellieren. Auch die Versionierung der modellierten Prozesse ist möglich.

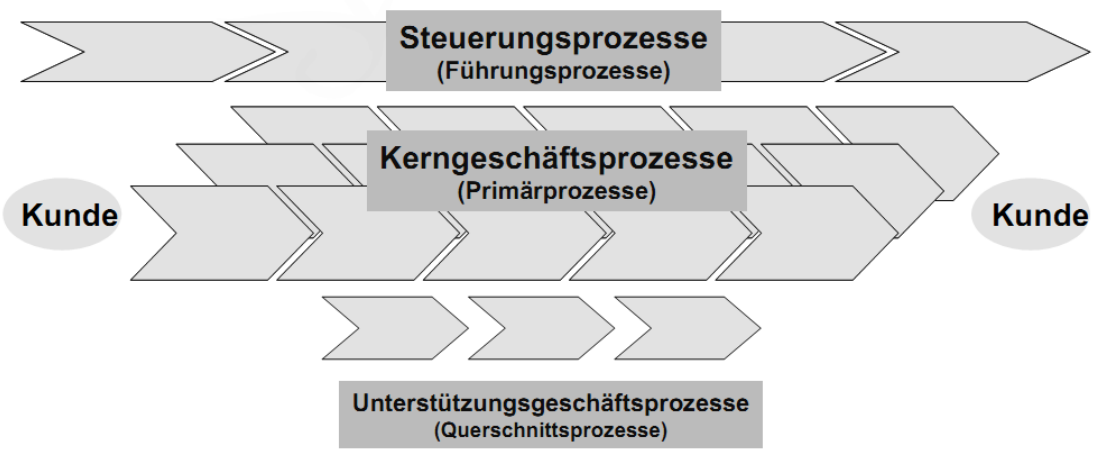


Abbildung 43 Kern- und Unterstützungsprozesse (vgl. Gadatsch, 2005, S. 49)

Eine mögliche Visualisierung der Prozesse in einem Unternehmen ist durch die Verwendung von Prozesslandkarten gegeben. Prozesse können, z.B. in Abhängigkeit von der Nähe zum Kerngeschäft des Unternehmens, in folgende Gruppen unterteilt und durch Prozesslandkarten visualisiert werden Abbildung (vgl. Gadatsch, 2005):

- Steuerungsprozesse
- Kernprozesse
- Unterstützungsprozesse

Jede Prozessgruppe kann mehrere einzelne Prozesse als Bestandteile beinhalten, die logisch miteinander in Verbindung stehen. Jeder Prozess wiederum besteht aus einer Abfolge von Prozess-Schritten, die einzeln abrufbar und editierbar sind. In QUAM wird jeder Prozess als SharePoint Element in der Liste Ablauforganisation gespeichert und kann mit nahezu unbegrenzt vielen Attributen versehen werden. Auch eine umfangreiche Verknüpfung mit anderen QUAM Elementen kann erfolgen, so können bspw. für einen bestimmten Schritt verantwortliche Personen oder Abteilungen dargestellt werden. Änderungen der Elementattribute, wie z.B. Verantwortlichkeitsmodifikationen bzgl. einer Aufgabe, müssen nur einmalig erfolgen.

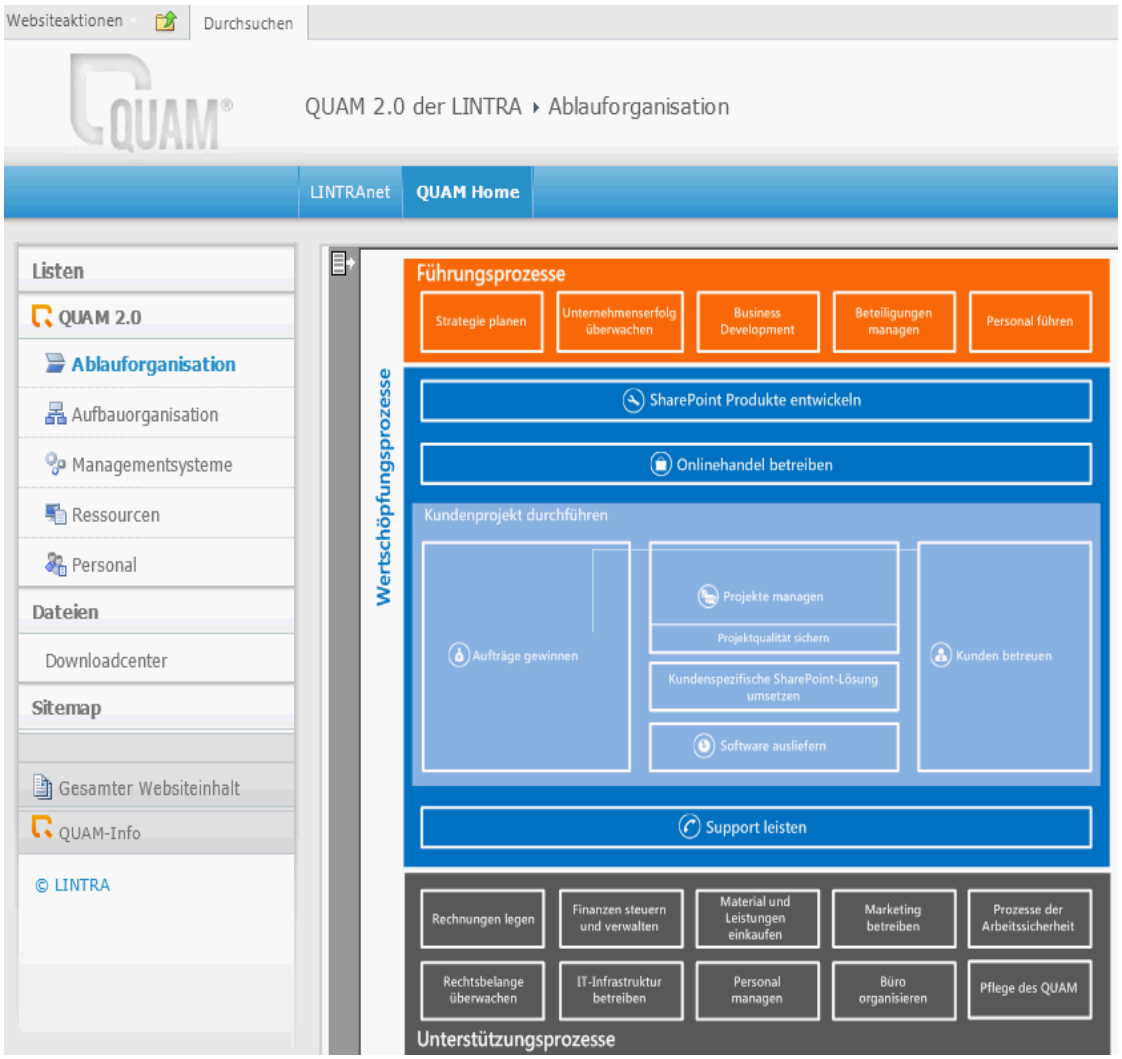


Abbildung 44 Ablauforganisation in QUAM

Abbildung 44 zeigt die zentrale Prozesslandkarte, über die der Einstieg in die Unternehmensprozesse erfolgt. Alle Symbole auf der Prozesslandkarte stellen einzelne SharePoint Elemente dar, die wiederum hierarchisch untergliedert

sind. So sind zum Beispiel drei Prozessgruppen erkennbar, von denen jede Gruppe aus einzelnen, logisch zusammengehörenden Prozessen besteht. Die grafische Modellierung findet mit dem QUAM Visio Modeler statt, der eine Erweiterung der Funktionen von Microsoft Visio darstellt.

Ein großer Vorteil dieser Lösung ist die redundanzfreie Haltung der Daten. Änderungen, die beim Modellieren eines Unternehmensablaufs stattfinden, werden an einer zentralen Stelle in der Datenbank geändert und sind sofort für alle Beteiligten sichtbar.

Jeder Prozess in der Ablauforganisation kann mit einer sehr großen Anzahl zusätzlicher Informationen attributiert werden. Exemplarisch seien hier folgende Attribute genannt:

- **Prozessbeschreibung**

  Dadurch kann der Prozess ausführlicher beschrieben werden. Die Beschreibung gibt dem Betrachter die Möglichkeit, nähere Informationen über den Prozess zu erfahren, falls das Diagramm nicht selbsterklärend ist.

- **Prozessverantwortlicher**

  Hierdurch wird die Möglichkeit gegeben, einen Prozessverantwortlichen oder mehrere Prozessverantwortliche anzugeben. Dabei wird aus der Menge der existierenden QUAM-Benutzer einer ausgewählt.

- **Mitwirkende, entscheidende, informativ beteiligte Organisationseinheiten**

- **Mitgeltende Unterlagen, angewandte Systeme und Regularien**

  Durch diese Angaben werden existierende Systeme oder Unterlagen mit dem Prozess verknüpft.

Die Modellierung der einzelnen Prozesse erfolgt mittels einfacher Flussdiagramme. Dabei werden insgesamt sechs Symbole zur Verfügung gestellt, mit denen der Prozess modelliert werden kann. Aufgrund der hohen Nachfrage einerseits und der mittlerweile relativ großen Verbreitung der BPMN 2.0 andererseits wird eine Erweiterung der Symbolmenge angestrebt, was die Aufgabenstellung dieser Abschlussarbeit ist. Zudem ist auch die Evolution der BPMN ein Grund dafür, dass die Modellierungssprache für die Gestaltung von

Prozessen in QUAM von Interesse ist. Mittlerweile bietet BPMN 2.0 nicht nur eine sehr ausführliche Notation an, sondern auch die Möglichkeit zum Prozessaustausch und Prozessausführung.

3.6.1.2 Liste „Aufbauorganisation“

Die Aufbauorganisation stellt die Organisationsstruktur eines Unternehmens dar. Das umfasst alle Organisationseinheiten und Rollen des Unternehmens aber auch die verfügbaren Mitarbeiter und anderen Ressourcen. In der Aufbauorganisation wird dargestellt, wie die einzelnen Organisationseinheiten und deren Hierarchie, die vorhandenen Rollen und Teams sowie die externen Partner des Unternehmens in Beziehung zueinanderstehen.

Zu jedem Element der Aufbauorganisation wird standardmäßig eine Rolle hinterlegt, was unter den Punkten „Verantwortung“, „Mitwirkung“ oder „Information“ geschehen kann. Änderungen in Elementen der Aufbauorganisation werden auch in anderen Listen übernommen, bspw. in der Ablauforganisation. Umgekehrt haben auch Änderungen in anderen Listen Auswirkung in der Liste Aufbauorganisation, bspw. wenn zu einer Prozessaufgabe verantwortliche Personen eingetragen werden. Die grafische Modellierung und Darstellung der Organisationsstrukturen als Organigramme findet auch unter dem Einsatz von QUAM Modeler Add-In statt. Dadurch, dass Organigramme untereinander verlinkt werden können, wird eine Modellierung auf unterschiedlichen Ebenen ermöglicht, was zu einer Übersichtlichkeit der Aufbauorganisation beiträgt. Abbildung 45 zeigt die Liste Aufbauorganisation der QUAM-Demoanwendung.

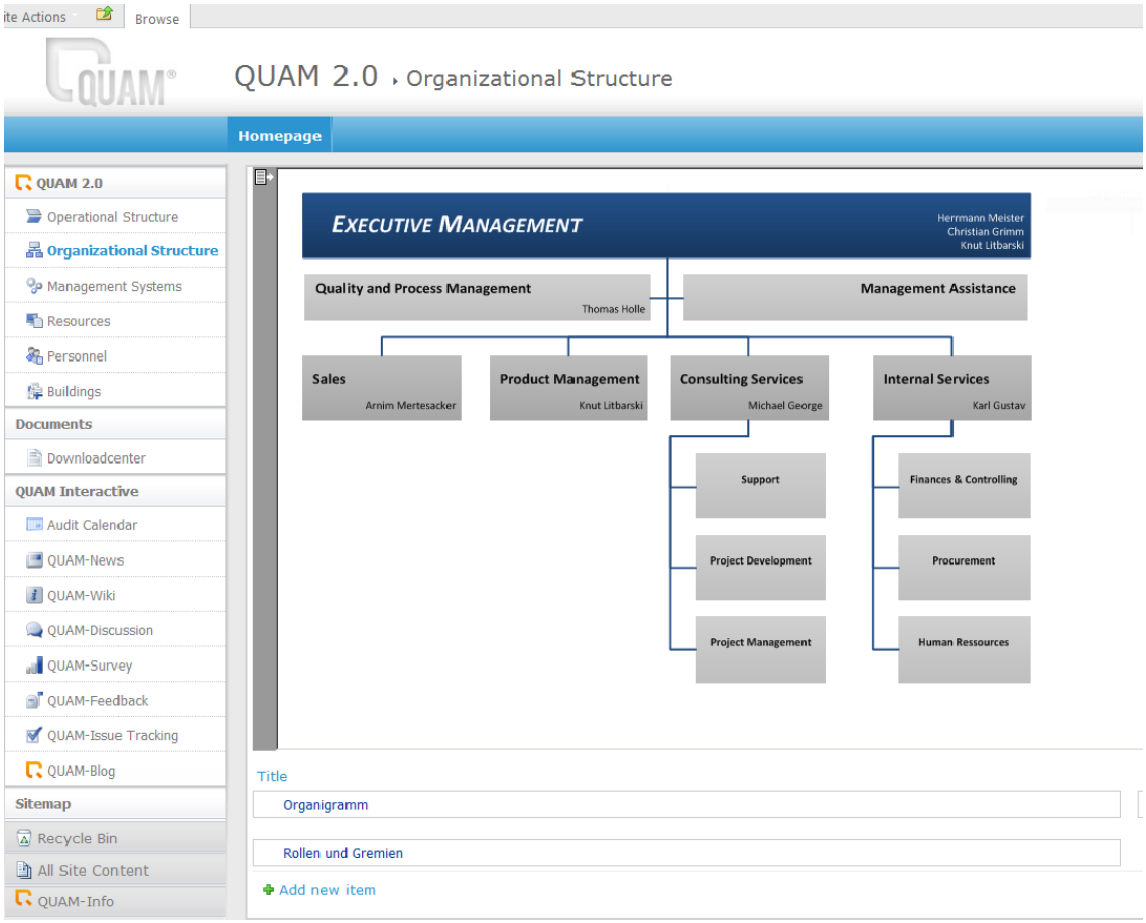


Abbildung 45 QUAM Organisationsstruktur

3.6.1.3 Liste „Personal“

In der Liste Personal werden alle Mitarbeiter eines Unternehmens mit ihrem Vor- und Nachnamen abgespeichert. Weitere Informationen beziehen sich z.B. auf Position, Organisationseinheit, Verantwortlichkeiten, Raumnummer, Telefonnummer, Foto, E-Mail-Adresse. Zu jedem Mitarbeiter einer Organisation steht somit eine Vielzahl von Kontaktdaten zur Verfügung. Die Beziehung eines Mitarbeiters zu bestimmten Organisationseinheiten des Unternehmens wird durch die Spalten „Manager der Organisationseinheit“ und „Mitarbeiter der Organisationseinheit“ ausgedrückt. Wird ein Mitarbeiter einer anderen Organisationseinheit zugeteilt und diese Information in der Personalliste vermerkt, so werden alle davon betroffenen Listen (z.B. Organigramme der Aufbauorganisation) aktualisiert. Abbildung 46 zeigt die Liste „Personal“ einer QUAM-Demoanwendung.

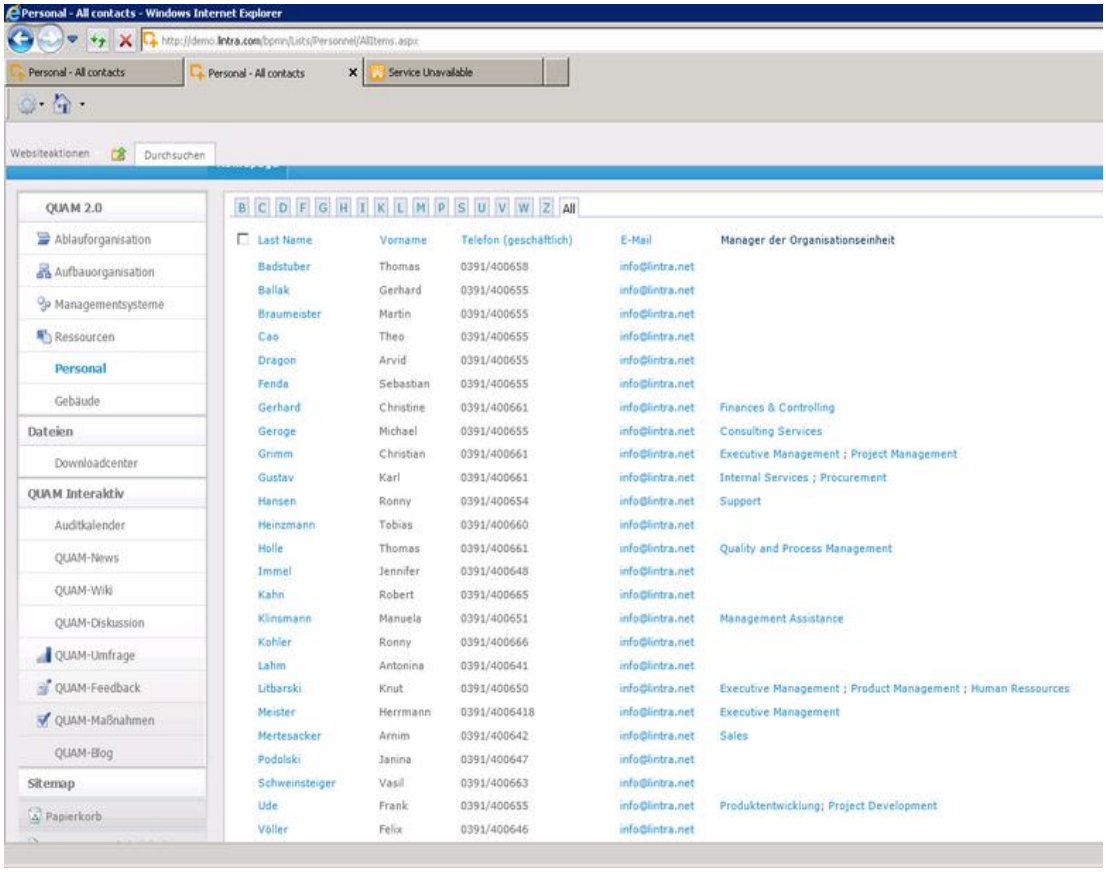


Abbildung 46 Liste „Personal“ in QUAM

3.6.1.4 Liste „Downloadcenter“

In der Liste Downloadcenter werden von allen Mitarbeitern häufig verwendete Dokumente, Formulare, Bilder und jegliche andere Dateien zentral abgelegt. Die Ordner sind thematisch nach Organisationseinheiten untergliedert und die Verknüpfung der im Downloadcenter gespeicherten Dateien mit den Listen „Managementsysteme“ und „Ressourcen“ ist möglich.

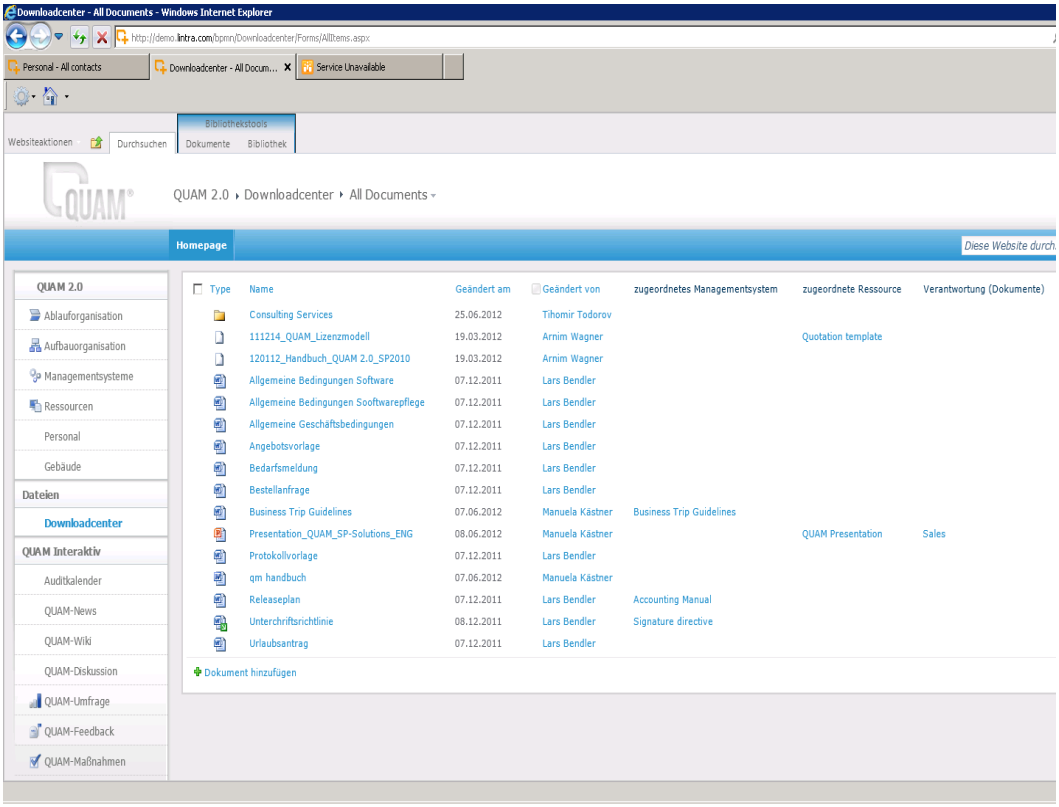


Abbildung 47 QUAM-Downloadcenter

3.6.1.5 Liste „Ressourcen“

Ressourcen gehören zur Aufbauorganisation des Unternehmens und stellen die "Hilfsmittel" dar, mit denen Mitarbeiter Produkte und Dienstleistungen erstellen. Dabei wird im QUAM zwischen IT-Systemen, Formularen sowie Sachmitteln unterschieden.

Durch das Anlegen neuer Untergruppen können die Ressourcen feiner gegliedert und übersichtlicher dargestellt werden (vgl. LINTRA, 2010). In QUAM werden zunächst folgende drei Ressourcengruppen unterschieden:

- IT-Systeme
- Dokumente und Aufzeichnungen
- Sachmittel

Die verwendeten Dokumente und Aufzeichnungen könnten bspw. darüber hinaus danach unterschieden werden, ob sie der Kommunikation mit den Kunden, mit den Lieferanten oder der internen Kommunikation dienen. Dies

wird ausgedrückt, indem unterhalb der Gruppe „Dokumente und Aufzeichnungen“ drei Untergruppen mit den entsprechenden Bezeichnungen angelegt werden. Um die neu erstellten Gruppen entsprechend zuzuordnen, wird ihnen die Gruppe "Dokumente und Aufzeichnungen" als Elternobjekt zugewiesen (Abbildung 48).

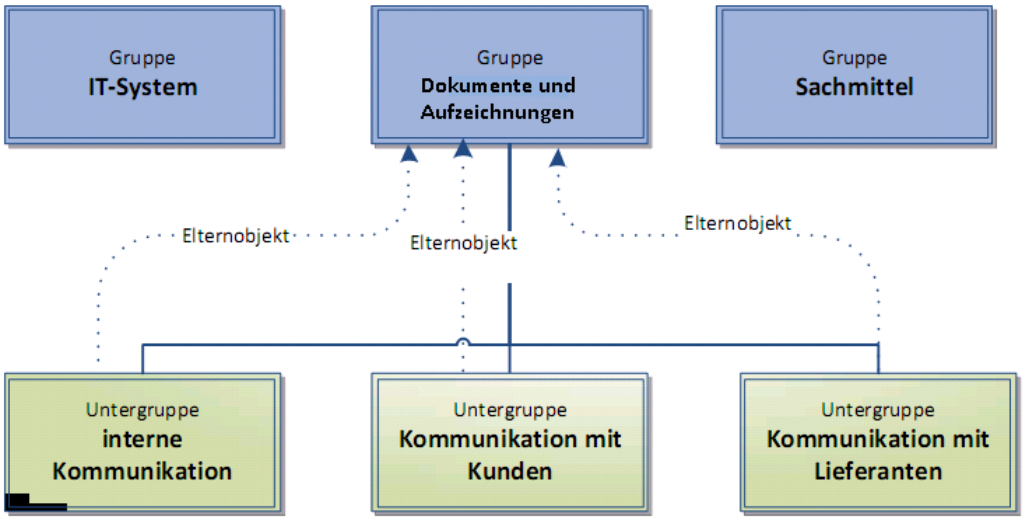


Abbildung 48 Beziehungen in der Ressourcen-Liste (vgl. QUAM Anwenderhandbuch)

3.6.1.6 Liste „Managementsysteme“

Managementsysteme beschreiben notwendige Rahmenbedingungen, denen das Unternehmen per Gesetz unterliegt. Das Qualitätsmanagement, das Umweltmanagement, die Arbeitssicherheit sowie die Managementregularien bilden so das umfassende Regelwerk des Unternehmens, welches durch interne Vorgaben der Geschäftsleitung sowie externe Forderungen der Kunden, Banken, Gesetzgeber, normgebenden Stellen usw. definiert wird. QUAM bietet die Möglichkeit, die Anforderungen dieser Regelwerke über entsprechende Handbücher zu strukturieren.

Abbildung 49 Abschnitt eines Managementhandbuches

Nachdem das QUAM-System vorgestellt wurde, werden in Tabelle 7 Eigenschaften von QUAM die im Kapitel über BPM-Technologien beschriebenen Merkmale eines BPM-Systems aufgeführt und QUAM gegenübergestellt.

Merkmale eines BPM-Systems	**QUAM**
Grafische Darstellung der Prozesse als Prozesslandkarte und die Darstellung der einzelnen Prozesse	Ja
Definition des Informationsflusses zwischen den einzelnen Prozess-Schritten	Ja
Dokumentenmanagementsystem	Ja
Prozessrepository (Prozessrichtlinien, Regelungen, Kompetenzen usw.)	Ja
Workflow-System	SharePoint Workflows können definiert werden
Prozesssimulation	Nein
Prozessausführung	Nein

Tabelle 7 Eigenschaften von QUAM

4 Entwurf und prototypische Implementierung

4.1 Erweiterung der Notation von QUAM

Hierbei handelt es sich um eine Erweiterung der Symbolpalette von QUAM. Nach der Erweiterung kann der Modellierer Prozessmodelle in der BPMN 2.0 Notation gestalten.

4.1.1 Anforderungsanalyse

Die Liste Ablauforganisation in QUAM umfasst alle Aktivitäten des Unternehmens. Sie stellt die Prozesswerkstatt dar, in der die Prozessmodelle des Unternehmens entworfen und freigegeben werden. Die Modellierung der Unternehmensprozesse erfolgt auf zwei unterschiedliche Weisen.

Zum einen kann eine textuelle Modellierung ohne Verwendung grafischer Hilfsmittel vorgenommen werden, indem SharePoint-Elemente durch Zuweisungen von Elternobjekten strukturiert werden. Dazu wird, wie in Abbildung 50 veranschaulicht, in der Liste Ablauforganisation zuerst das zu erstellende Prozesselement gewählt und anschließend seine Eltern- oder Kindelemente.

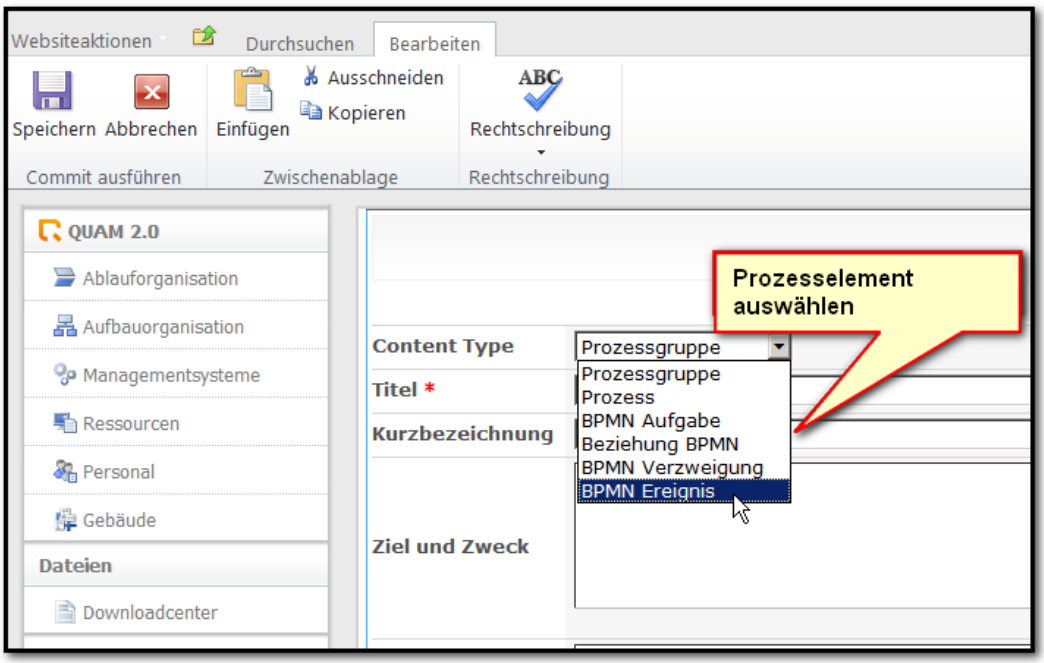


Abbildung 50 Textuelle Modellierung mit QUAM

Zum anderen ist die Gestaltung der Unternehmensabläufe grafisch mit Hilfe vom QUAM-Visio Modeler möglich. Abbildung 50 zeigt das geöffnete QUAM-Modeler-Fenster:

4.1.2 Umsetzung der Symbolerweiterung

Um die Erweiterung der Symbolpalette auf BPMN 2.0 umzusetzen, müssen sowohl Konfigurationen im QUAM-Modeler als auch in den QUAM-Inhaltstypen vorgenommen werden. Nachfolgend werden alle notwendigen Änderungen zunächst in ihrer logischen Reihenfolge aufgelistet und anschließend erläutert:

- **Erweiterung der QUAM-Modeler Schablone**
- **Anpassung der betroffenen QUAM-Inhaltstypen**
- **Anpassung der XML-Konfigurationsdatei der Modeler-Komponente**

Zu beachten ist, dass die Erweiterung der Symbol-Palette der Liste Ablauforganisation keine Änderung der QUAM-Inhaltstypenmenge bewirken soll, vielmehr sind die betroffenen QUAM 2.0 Inhaltstypen soweit wie möglich auf den BPMN 2.0 Symbolen abzubilden.

Erweiterung der QUAM-Modeler Schablone

Die BPMN-Vorlage in Visio 2010 unterstützt die Version 1.2 der Spezifikation. Die Änderungen der Symbolpalette von BPMN 1.2 auf BPMN 2.0 sind eher gering und werden nachfolgend aufgezählt.

- Erweiterung um nicht-unterbrechende Ereignisse
- Ergänzung um Event-Subprozesse
- Einführung von Choreografie- und Konversationsdiagrammen
- Eliminierung von Referenz-Tasks

Eine BPMN 2.0 Schablone kann erstellt werden, indem die BPMN 1.2 Schablone um die fehlenden Shapes erweitert wird. Die so veränderte Schablone muss in die entsprechende QUAM-Dokumentenbibliothek hochgeladen werden, von wo sie vom QUAM-Modeler beim Modellieren geladen wird.

Zum BPMN 2.0 Standard gehören inzwischen auch die Choreografie- und Konversationsdiagramme, die allerdings noch eine eher geringe praktische Relevanz aufweisen (vgl. Silver, 2011). Aus diesem Grund wird vorerst keine Unterstützung der BPMN 2.0 Choreografiediagramme und Konversationsdiagramme angestrebt.

Im Folgenden werden die Modellierungselemente und Inhaltstypen von QUAM 2.0 tabellarisch dargestellt (Abbildung 51). Zu jedem QUAM-Inhaltstyp und Symbol wird das korrespondierende BPMN 2.0 Shape dargestellt. Aus Gründen der Übersichtlichkeit wird auf die die Darstellung der kompletten BPMN 2.0 Symbolpalette verzichtet.

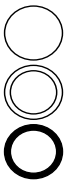
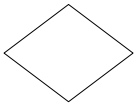
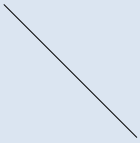
QUAM-Inhaltstyp	QUAM Shape	BPMN 2.0 Shape	Erklärung zum QUAM-Inhaltstyp
Ereignis			Geschehen, das während eines Prozesses auftreten kann und den Start- und Endpunkt eines Prozesses darstellt. Dies kann z.B. die Anfrage eines Kunden oder das Eintreten einer bestimmten Situation sein.
Aufgabe			Tätigkeit, die innerhalb eines Prozesses durchgeführt wird.
Entscheidung			Ein Entscheidungspunkt innerhalb eines Prozesses.
Verbindung			Um Aufgaben, Ereignisse und Verzweigungen miteinander zu verknüpfen, müssen Verbindungen definiert werden. Diese werden im QUAM mit Hilfe von Beziehungen modelliert.
Prozess			Ein Prozess in QUAM ist ein Ablauf, der in mehrere einzelne Objekte zerlegt werden kann.
Prozessgruppe			Gruppe von Prozessen, die logisch zusammengehören

Abbildung 51 Gegenüberstellung von QUAM 2.0 und BPMN 2.0 Modellierungselemente

Das Abbilden der QUAM-Inhaltstypen auf den BPMN 2.0 Shapes sieht folgendermaßen aus:

- **Grundinhaltstypen**

  Für die Inhaltstypen Ereignis, Aufgabe und Entscheidung kann eine direkte Zuordnung zu den Grundelementen der BPMN 2.0 Event, Task und Gateway erfolgen. Da das laut der BPMN-Spezifikation sog. generische Elemente sind (von denen also keine Instanzen erstellt werden sollten), sollte zu jedem Element noch eine Unterteilung bzw. Konkretisierung dahingehend vorgenommen werden, um welches spezifische Element es sich handelt.

- **Prozess und Prozessgruppe**

  QUAM verwendet die Inhaltstypen Prozess und Prozessgruppe, die in der BPMN 2.0 Spezifikation keine direkte Entsprechung haben. Beide Inhaltstypen sind vielmehr auf einer höheren Modellierungsebene anzusiedeln, auf der z.B. Prozesslandkarten das Zusammenwirken von Teilprozessen visualisieren. Der Inhaltstyp Prozess drückt einen abgeschlossenen Ablauf aus, er kann auf das BPMN 2.0 Symbol für einen zugeklappten Teilprozess gemappt werden. Dieser wird unter anderem genutzt, um eine Wiederverwendung von Abläufen zu erreichen.

- **Swimlanes**

  Swimlanes sind eine häufig verwendete Methode, um Verantwortlichkeiten für bestimmte Prozesse oder Aufgaben anzugeben. In QUAM 2.0 ist der Inhaltstyp *Rolle* vorhanden. Dieser wird für die Zuordnung von Organisationseinheiten/ Personen zu Aufgaben verwendet.

- **Artefakte**

  Als Artefakte werden in die Schablone die nachfolgend aufgeführten Elemente hinzugefügt, ohne dabei neue Inhaltstypen zu erstellen.

  - Datenspeicher, Dateninput, Datenoutput
  - Gruppierung
  - Anmerkung

Anpassung der QUAM-Inhaltstypen

Die beschriebene Anpassung der Visio-Schablone ist der erste Schritt, um die QUAM-Symbolpallette zu erweitern. Damit einzelne BPMN 2.0 Elementausprägungen nicht nur grafisch visualisiert werden, sondern auch in das QUAM-Datenmodell übernommen werden, müssen Anpassungen der QUAM-Inhaltstypen sowie der QUAM-Modeler Komponente bzw. an ihrer Konfiguration vorgenommen werden.

Jeder Inhaltstyp der Ablauforganisation wird um eine zusätzliche Spalte für Elementausprägung bzw. Spezialisierung erweitert. Der Inhaltstyp Ereignis wird z.B. um die Spalte EventType erweitert. Gleiches gilt auch für die Inhaltstypen Entscheidung, Aufgabe und Beziehung.

Ziel und Zweck dieser Erweiterung ist es, keine Informationen zu den spezifischen BPMN-Elementen verloren gehen zu lassen. Wird z.B. ein StartEreignis auf die Zeichnungsfläche gezogen, so wird ein Ereignis-Inhaltstyp erstellt und die Spalte für Ereignistyp auf StartEvent gesetzt.

Anpassung der XML-Konfigurationsdatei für die Modeler Komponente

Einen weiteren Schritt in Richtung Erweiterung der Symbolpalette stellt die Anpassung der XML-Konfigurationsstruktur dar, die von der Modeler-Komponente geladen wird und für die korrekte Zuordnung von Visio Shapes zu Inhaltstypen sorgt. Im Folgenden wird diese Struktur erläutert, wobei nur auf ihre wichtigsten Elemente eingegangen wird.

Folgende Elemente des XML-Konfigurationsdokuments sind für die Zuordnung von Visio-Shapes zu QUAM-Inhaltstypen von Bedeutung:

- ***<Quamvisiosettings>*** stellt das oberste Element der Konfigurationsstruktur dar. Das einzige Attribut dieses Elements gibt den Namen der Konfigurationsdatei an.
- ***<ContentType>*** ist Kindelement von *Quamvisiosettings*. Zu jedem Inhaltstyp der Liste Ablauforganisation wird genau ein korrespondierendes XML-Element *ContentType* erstellt. Über die Attribute des Elements lassen sich folgende Informationen angeben: Inhaltstyp, eindeutige ID (GUID) des Inhaltstyps sowie der Ordner in SharePoint.

- ***<MasterShape>*** ist ein Kindelement vom *ContentType*-Element. Über seine Attribute wird der eindeutige Name des Visio Master-Shapes angegeben und der Fakt, ob der Inhaltstyp von anderen Shapes repräsentiert wird.
- ***<SPField>*** ist ein optionales Element und es wird dann eingesetzt, wenn ein Inhaltstyp von mehr als einem Visio-Shape repräsentiert wird (was in der vorliegenden Arbeit der Fall ist). In diesem Fall wird der Inhaltstyp um eine zusätzliche Spalte erweitert, die die *Spezialisierung des* Inhaltstyps angibt. Beispiele dafür sind der Inhaltstyp Ereignis und seine Spezialisierung Startereignis sowie der Inhaltstyp Entscheidung und seine Spezialisierung Entscheidungstyp.

Eine Beispielkonfiguration ist in Listing 13 dargestellt, wobei aus Gründen der Übersichtlichkeit auf die Auflistung der gesamten Struktur verzichtet wurde. Der Auszug enthält die Konfiguration der Inhaltstypen Entscheidung und Verbindung.

```
<quamvisiosettings vssFile="QUAM_OperationalStructure">
 <ContentType       Name="Gateway" ID="1" CTSFolder="BPMN Gateway">
            <MasterShape Name="Exclusive Data Gateway" Single="False" Verbinder="False"
                    StandardMaster="True">
                              <SPField    Name="GatewayType"
                                                  InternalName="GatewayType"
                                                  Value="Exclusive Data Gateway"
                                                  Type="Choice"/>
                    </MasterShape>
                    <MasterShape   Name="Exclusive Event Gateway"
                                              Single="False"
                                              Verbinder="False"
                                              StandardMaster="True">
                              <SPField   Name="GatewayType"
                                                  InternalName="GatewayType"
                                                  Value="Exclusive Event Gateway"
                                                  Type="Choice" />
                    </MasterShape>
             </ContentType>
</quamvisiosettings>
```

Listing 13 XML-Konfigurationsstruktur für das DiagramEditControl WebPart

Wenn die angepasste XML-Konfigurationsstruktur der Modeler-Komponente übergeben wird, ist die Erweiterung der Symbolpalette von QUAM 2.0

abgeschlossen. Alle bisher vorgenommenen Veränderungen am QUAM dienen zum einen dazu, dem Benutzer die Modellierung mit der BPMN 2.0 Sprache zu ermöglichen. Zum anderen stellen sie auch Vorbereitungsmaßnamen zur Umsetzung des Exports in das BPMN-XML-Format dar. Diese Umsetzung ist Gegenstand des folgenden Abschnitts.

4.2 Überführung eines Prozessdiagramms in das BPMN-XML-Format

4.2.1 Entwurf des Systems

Nachdem im vorangegangenen Abschnitt auf die Umsetzung der notationellen Erweiterung eingegangen wurde, sollen im Folgenden architekturelle und implementierungsrelevante Details der Arbeit betrachtet werden.

Beim Entwurf eines Softwaresystems werden die systembezogenen Anforderungen umgesetzt. Ergebnis dieser Umsetzung ist eine Beschreibung der Systemarchitektur des Prototyps. Die Architektur besteht aus den einzelnen Prototypkomponenten sowie den extern sichtbaren Schnittstellen und Beziehungen zwischen den Komponenten (vgl. Dumke, 2003). Die Ergebnisse dieser Phase sind wiederum Ausgangspunkte für die nächste Phase der Implementierung.

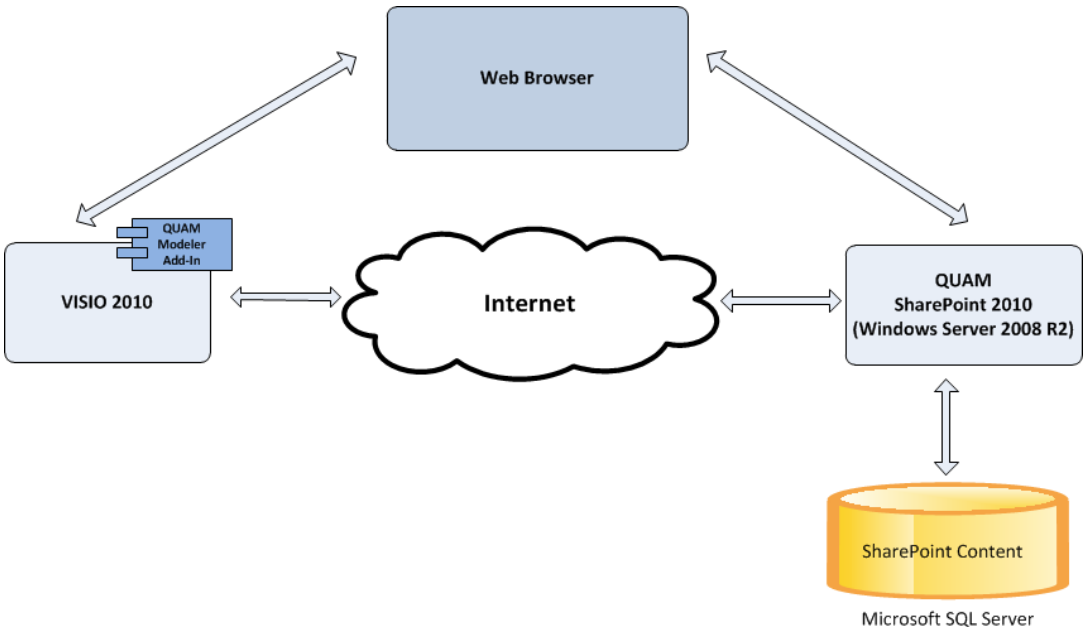


Abbildung 52 QUAM-Grobarchitektur

In Abbildung 52 ist die Grobarchitektur eines QUAM-Systems dargestellt. Das QUAM samt SharePoint 2010 benötigt als Grundplattform einen Webserver, der wiederum entweder auf einem Windows-Server in der Version 2003, 2005 oder 2008 oder Windows 7 installiert wird, wobei letzterer nicht zum produktiven Betrieb, sondern nur zu Entwicklungszwecken empfohlen wird. Für die Speicherung der SharePoint-Daten sowie der Konfigurationseinstellung der QUAM-Anwendung wird eine Datenbank benötigt, die auf einen externen Server ausgelagert werden kann. Das Auslesen und Speichern von Daten erfolgt über HTTP-Requests und ASP-Seiten. Der Anwender greift über Web-Browser auf die SharePoint-Elemente zu und startet gegebenenfalls die QUAM-Modeler-Komponente, wenn er eine Modellierung vornehmen möchte. Dadurch wird ein Visio-Fenster geöffnet, in dem die Inhalte modelliert werden können. Der Unterschied zu der reinen Visio-Modellierung ist der QUAM-Modeler. Diese Komponente integriert eine zusätzliche Ribbon-Oberfläche in Visio, über die quam-spezifische Befehle zum SharePoint-Server geschickt werden.

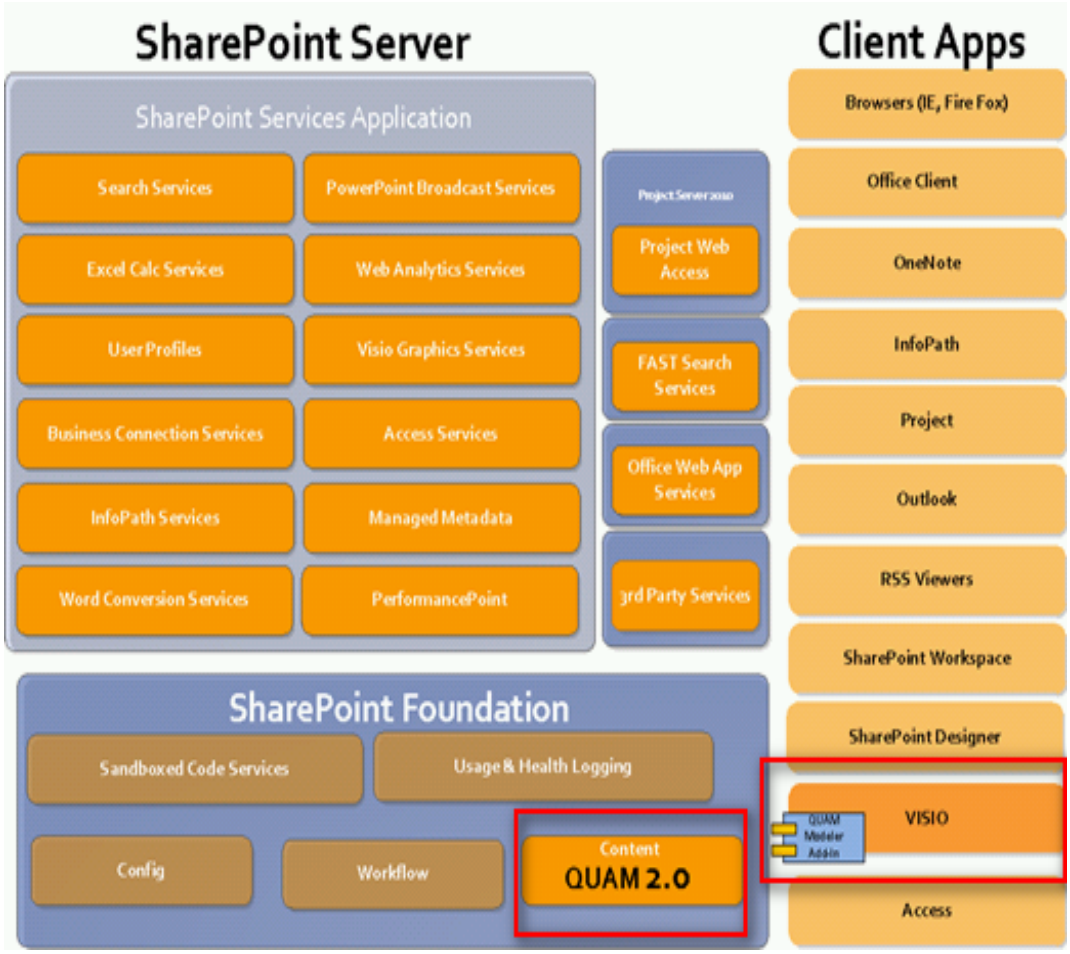


Abbildung 53 QUAM im Kontext von SharePoint

Eine andere Perspektive auf das QUAM-System zeigt die obenstehende Abbildung. Darin ist die Architektur von QUAM im Kontext von SharePoint noch einmal schematisch dargestellt. Zum einen ist zu entnehmen, dass die QUAM-Anwendung auf dem SharePoint Foundation Server im Bereich „Content“ angesiedelt ist, zum anderen erweitert die Modeler-Komponente als eine Office-Add-In die Funktionalität von Visio.

Das Konvertierungs-Add-In setzt sich aus den folgenden Komponenten zusammen, welche auf drei Schichten verteilt sind. Die Ebene ***MS Visio*** beschreibt die Systemumgebung von Visio und beinhaltet das Visio-BPMN-Diagramm. Auf der Ebene ***QUAM Converter Add-In*** befindet sich das BPMN 2.0 Objektmodell zur Erstellung des XML-Formats. Diese Ebene wird unter Verwendung des Visio SDK, des .NET Framework und des BPMN 2.0 Modells der OMG verwirklicht. Die letzte Ebene, ***XML,*** beinhaltet das exportfähige BPMN 2.0 XML.

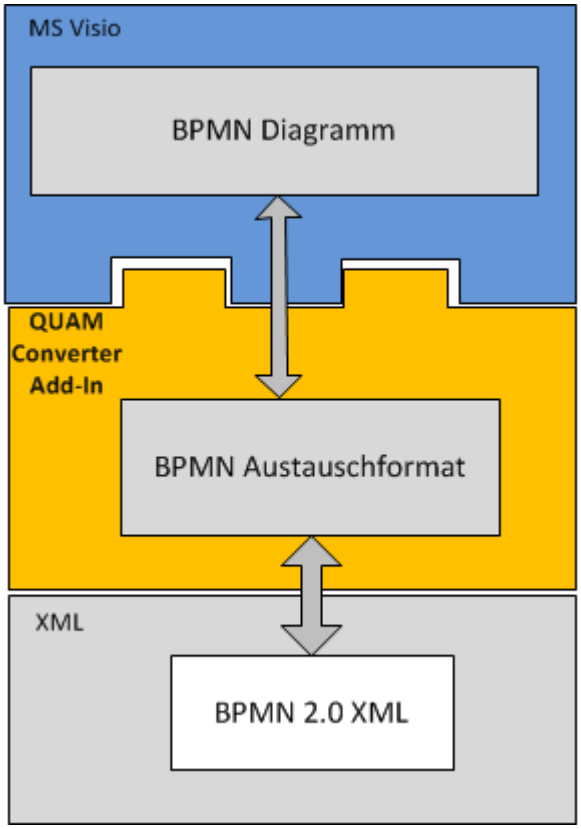


Abbildung 54 Grobarchitektur des Export-Add-Ins

4.2.2 Prototypische Implementierung

Wie in den vorhergehenden Kapiteln diskutiert, spielt die Interoperabilität von Prozessmodellen eine wesentliche Rolle. Diese Interoperabilität kann im Fall von QUAM erreicht werden, indem die Prozesse in dem BPMN 2.0 Austauschformat abgespeichert werden. Die QUAM-Funktionalität kann so erweitert werden, dass sie eine Export-Funktion unterstützt, deren Umsetzung Gegenstand der nachfolgenden Ausführungen ist. Zur Überprüfung der Funktionalität kann schließlich ein Prozessdiagramm von QUAM in das *.bpmn Format exportiert werden und in ein fremdes Modellierungswerkzeug importiert werden, das das BPMN-XML-Format unterstützt. Das Diagramm soll dann nach dem Import identisch aussehen - wie das im QUAM 2.0.

Generell bestehen zwei Möglichkeiten, die Inhalte eines QUAM-Ablaufdiagramms in das BPMN-XML-Format zu überführen; zum einen durch eine Konvertierung bereits abgespeicherter Diagramme, zum anderen durch eine Umwandlung und den Export von Diagramminhalten zur Laufzeit. In den

folgenden Abschnitten werden die einzelnen Varianten detaillierter vorgestellt. Eine Gegenüberstellung der Vor- und Nachteile der Alternativen erfolgt im Anschluss daran.

4.2.2.1 Integrationsalternativen

4.2.2.1.1 Variante Dateikonvertierung

Bei dieser Variante handelt es sich um die Umwandlung von Visio-Dateien in das BPMN-XML-Format. Die Modeler-Komponente von QUAM speichert alle Diagramme in einer SharePoint-Dokumentenbibliothek ab. Von dort aus können sie programmatisch geöffnet und ggf. verarbeitet werden. Mögliche Dateiformate des Diagramms wären vsd, vdw oder vdx; letzteres ist das Microsoft-XML-Dateiformat für Visio Diagramme(siehe Grundlegende Visio-Begriffe). Liegen die Ablaufdiagramme in einem XML-Format vor, kann der Ablauf der Konvertierung folgendermaßen aussehen:

Zunächst wird die XML-Struktur des abgespeicherten Diagramms mittels DOM oder LINQ durchlaufen und die darin befindlichen Elemente analysiert. Daraus werden nur die Informationen extrahiert, die für das Mapping von Relevanz sind. Die im VDX-Format gespeicherte Zeichnung ist besonders speicherintensiv, da sie sehr viele Informationen beinhaltet, von denen nicht alle für das Umwandeln von Relevanz sind. Relevante Informationen, die extrahiert werden sollen, sind z.B.:

- das Visio-XML-Shape-Element mit den dazugehörigen Informationen, wie ElementTyp, ElementName, Koordinaten, Größe, Vorgänger- und Nachfolgerelemente
- die Verbindungen zwischen den einzelnen Visio-Diagrammelementen
- andere Beziehungen, z.B. Containments-Beziehungen

Sind alle relevanten Informationen zu den BPMN-Elementen extrahiert, muss zu jedem Element ein korrespondierendes BPMN-XML-Element mit den entsprechenden Attributierungen und Kindelementen erzeugt werden.

Microsoft bietet zum Visio-XML-Format eine sehr ausführliche Dokumentation, jedoch sind die BPMN-Schablone und die spezifischen BPMN-ShapeSheets

dazu undokumentiert, so dass eine Verarbeitung der XML-Strukturen eine erstmalige und aufwendige Analyse der Schablonen- und Shapesheetlogik erfordert, was sich wiederum nachteilig auf dem Implementierungsaufwand auswirkt.

Eine andere wichtige Problematik ergibt sich aus der Spezifik des QUAM-Modelers. Ähnlich wie Visio kann er konfiguriert werden, so dass die Diagramme in unterschiedlichen Formaten in der Diagramm-Dokumentenbibliothek *ChartLib* abgelegt werden. Exemplarisch werden hier das Standardformat VSD, VDX oder VDW genannt. Bei der Konvertierung ist eine erstmalige Abfrage notwendig, um zu überprüfen, in welchem Format die Diagramme vorliegen. Erst danach kann ein für die jeweilige Dateistruktur geeigneter Konvertierungsalgorithmus angewendet werden.

Das wirkt sich zwangsläufig auf den Implementierungsaufwand aus, da die Umsetzung unterschiedliche Umwandlungsalgorithmen für alle möglichen Speicherformate erfordert, seien es vsd, vdx oder vdw.

Vorteil dieser Lösung ist, dass der Benutzer auch zu einem späteren Zeitpunkt, nachdem er die Modellierung abgeschlossen hat, den Konvertiervorgang anstoßen kann. Zudem muss zum Start des Konvertiervorgangs nicht die Modeler-Komponente gestartet werden.

4.2.2.1.2 Integration in die Funktionalität von QUAM

Eine andere Möglichkeit zum Erzeugen einer BPMN-XML-Struktur aus einem QUAM-Prozessdiagramm ist die Entwicklung einer Erweiterung für Visio, deren Funktion die Funktionalität von QUAM-Modeler erweitert. Für den Prozessmodellierer besteht die Möglichkeit, nach dem Modellieren das Ablaufdiagramm in das BPMN-XML-Format zu exportieren.

Microsoft bietet eine Programmierschnittstelle für Add-Ins auf Anwendungsebene, mit denen Visio um zusätzliche Funktionen erweitert wird. Eine solche Erweiterung kann z.B. sein, dass das Prozessdiagramm zur Laufzeit bearbeitet und in eine andere Struktur exportiert wird. Voraussetzung dafür ist, dass die Diagrammstruktur vollständig und erfolgreich analysiert werden kann. Die Diagrammanalyse beinhaltet das erstmalige Durchlaufen aller sich auf der Zeichnung befindenden Objekte. Nachfolgend oder aber auch

synchron zum Durchlaufen erfolgt ein Zuordnen bzw. Erkennen der Diagrammsymbole samt aller relevanten Daten. Anschließend muss daraus eine gültige BPMN 2.0 XML-Struktur erstellt werden. Bei dieser Lösungsvariante musste im Rahmen dieser Abschlussarbeit zwischen zwei weiteren Implementierungsalternativen abgewogen werden. Es bezog sich darauf, wie die BPMN-XML-Strukturen erstellt werden. Zum einen können sie „per Hand“ mit LINQ to XML erstellt werden, zum anderen stehen dem Programmierer Werkzeuge zur Verfügung, mit denen ein XSD-Modell zuerst serialisiert werden kann.

- **Erzeugen der XML-Strukturen mittels LINQ to XML**

Hierbei wird initial das Diagramm durchlaufen; dabei werden relevante Daten gesammelt. Die XML-Strukturen werden anschließend „per Hand“ über die .NET Komponente LINQ erstellt.

LINQ (Language-Integrated Query) ist eine neue Gruppe von Eigenschaften in .NET, durch die die Abfragefunktionen auf die Sprachsyntax von C-Sharp und Visual Basic erweitert werden. LINQ ermöglicht neue standardisierte Muster zum Abfragen und Aktualisieren von Daten; die Technologie kann potenziell zur Unterstützung beliebiger Arten von Datenspeichern erweitert werden (vgl. Microsoft, LINQ Language-Integrated Query, 2012).

LINQ to XML stellt eine XML-Programmierschnittstelle im Arbeitsspeicher bereit, die LINQ nutzt. LINQ to XML verwendet die neuesten .NET Framework-Sprachfunktionen und kann als aktualisierte und neu gestaltete DOM-XML-Programmierschnittstelle angesehen werden (vgl. Microsoft, LINQ to XML, 2012). Dieser Lösungsweg wurde im Rahmen der vorliegenden Arbeit nicht ausgewählt. Da er jedoch nahezu die gesamte Bearbeitungszeit über als einzig mögliche Lösung angesehen wurde, wird er kurz aufgeführt. Er beinhaltet, grob skizziert, folgende Umsetzungsschritte:

- Programmatische Analyse des Diagramms
- Herausfiltern der relevanten Daten
- Abspeichern der Informationen in einer Datenklasse
- Abschließende Generierung einer XML-Struktur mittels LINQ to XML

Die programmatische Analyse erfolgt mit Hilfe von .NET und Visio SDK. Dabei werden alle sich auf dem Diagramm befindenden Shapes durchlaufen und nach ihren Eigenschaften durchsucht. Da sich auf dem Diagramm auch Shapes befinden können, die nicht zu der BPMN gehören, muss diesbezüglich eine Unterscheidung stattfinden. Damit diese Unterscheidung ermöglicht wird, wird zu jedem Shape, das zur BPMN 2.0 gehört, ein Eintrag in ShapeSheet erstellt. Dieser Wert kann beim Durchlaufen abgefragt werden; danach kann entschieden werden, ob es sich um ein BPMN-Shape handelt oder nicht.

Zu jedem auf dem Diagramm gefundenen BPMN-Shape wird eine Instanz einer speziell dafür definierten Datenklasse erzeugt. Darin werden alle für das Mapping notwendigen allgemeinen und speziellen Eigenschaften des BPMN-Shapes gespeichert. Wurden alle Shapes durchlaufen und die entsprechenden Datenobjekte erstellt, so können die zu den Shapes korrespondierenden BPMN-XML-Elemente per LINQ to XML erzeugt werden. Die Definition eines einzelnen BPMN-XML-Elements mittels LINQ to XML kann z.B. folgendermaßen aussehen:

```
var createFlowNodeRef = from shapeinfo in bpmnInfoList
where (shapeinfo.ShapeType != “SequenceFlow”)
select new XElement(“flowNodeRef”, shapeinfo.ShapeID);
```

Listing 14 Definition eines XML-Elements mittels LINQ to XML

Dabei geht es um das Erzeugen des XML-Elements `flowNodeRef`, das in einer BPMN-XML-Struktur die Rolle einer Referenz auf andere Diagrammelemente spielt.

Vor allem der letzte Schritt ist als Nachteil dieser Variante zu sehen, da es um die manuelle Erstellung von zum Teil über 300 XML-Elementdefinitionen geht. Das erhöht nicht nur den Implementierungsaufwand, sondern führt zu einem schwer wartbaren Code.

- **Erstellen der XML-Strukturen durch Serialisierung des BPMN Diagramms**

Bei dieser Umsetzungsvariante wird zuerst das von der Object Management Group frei gegebene BPMN 2.0 Metamodell (als XSD-Dokument vorliegend)

samt aller Elemente und Elementattribute deserialisiert, so dass der Programmierer im Idealfall mit einem vollständigen und konsistenten BPMN-Objektmodell arbeiten kann. Ein Überblick über die Serialisierung im Allgemeinen und im Speziellen sowie über das BPMN 2.0 Metamodell erfolgten im Kapitel 3 dieser Arbeit.

Vorteile der Variante

Im Gegensatz zu den oben dargestellten (aber auch anderen) Lösungen, die z.B. über XSLT eine Transformation von „Visio-Dateien“ zu BPMN 2.0 XML-Strukturen anbieten, weist die Variante der De- und Serialisierung folgende Vorteile auf:

- Durch den Einsatz eines stark typisierten Objektmodells ist garantiert, dass die erzeugten XML-Strukturen schemakonform sind.
- Erhebliche Zeitersparnisse durch den Wegfall der aufwendigen LINQ to XML Generierung von XML-Strukturen.
- Diese Lösungsvariante erlaubt die Integration der Funktionalität in die Oberfläche von QUAM-Modeler, da es sich bei beiden um Visio-Add-Ins handelt.
- Der Entwickler kann direkt mit dem Visio-Objektmodell arbeiten; es sind keine Kenntnisse zu VDX, VDW oder VDS notwendig.
- Es besteht eine Konformität zu den Entwicklungsrichtlinien des Auftraggebers. Die LINTRA GmbH setzt bei der Entwicklung ihrer Softwareprodukte ausschließlich auf die .NET Technologie und die C-Sharp als Entwicklungssprache.

Auswahl eines Verfahrens

Die Vor- und Nachteile der Lösungsvorschläge Dateikonvertierung und Export sind in folgender Tabelle kurz zusammengefasst:

Mögliche Lösungen zur Erzeugung von BPMN-XML-Inhalten	**Vorteile**	**Nachteile**
Dateikonvertierung Visio-BPMN- Diagramm zu BPMN-XML-Struktur	• Kein Starten des Visio Modeler durch den Nutzer • Hohe Flexibilität, falls mehrere Dateiumwandlungvarianten unterstützt werden	• sehr hoher Umsetzungsaufwand bei mehreren Konvertierungsalgorithmen • Keine Integration in QUAM-Modeler • Visio-BPMN-XML-Vorlage undokumentiert
Export von BPMN-Inhalten zur Laufzeit	• Integration in QUAM Modeler • Geringer Umsetzungsaufwand (zeitlich, personal)	• Ausführen von QUAM-Modeler notwendig

Tabelle 8 Gegenüberstellung der Lösungsalternativen zur Erzeugung von BPMN-XML-Inhalten

Aufgrund der Möglichkeit der Integration in den QUAM-Modeler und die Stabilität der Umsetzung durch Deserialisierung wurde für die Diplomarbeit die zweite Variante, bei der ein Export mittels Serialisierung eines Prozessdiagramms erfolgt, gewählt.

Auswahl eines Werkzeugs zur Codegenerierung

Das Resultat der Serialisierung ist ein komplexes Objekmodell mit über 150 einzelnen C-Sharp-Klassen, von denen jede teilweise sehr umfangreich attributiert ist. Außerdem erlaubt das Metamodell die Mehrfachvererbung, was die Abbildung auf C-Sharp-Klassenhierarchien zusätzlich erschwert. Aufgrund dieser Merkmale des Metamodells und um menschliche Fehler bei der manuellen Erstellung der einzelnen C-Sharp-Klassen zu vermeiden, ist für diese Arbeit eine Deserialisierung unter Einsatz von Codegenerierungstools vorgezogen worden.

Im Rahmen dieser Arbeit wurden einzelne Tools zur Codegenerierung evaluiert und daraus das Werkzeug XSD2Code Tool ausgewählt; Auswahlkriterien waren folgende:

- Das Tool muss aus XSD-Dokumenten den C-Sharp-Quellcode in der Version C-Sharp 3.5 oder höher generieren.
- Das Tool muss Kommentare aus den XSD-Dokumenten generieren können.
- Der generierte Quellcode muss direkt (d.h. ohne zusätzliche Anpassungen) serialisierbar sein.
- Das Tool sollte die konfliktfreie Generierung von Basistypen von mehreren XSD-Dokumenten unterstützen.

Dabei wurden folgende Tools auf die oben genannten Eigenschaften getestet:

1. XSD.exe von Microsoft (supplied with the SDK and Visual Studio)
2. XSDCodeGen
3. Xsd2Code
4. XsdObjectGen von Microsoft
5. CodeXS

Die Tools XSD.exe, XSDCodeGen und XsdObjectGen unterstützen ältere Versionen der C-Sharp-Sprache und wurden somit ausgeschlossen. Außerdem bieten sie keine Generierung von Kommentaren aus den XSD-Dokumenten an. CodeXS bietet keine Integration in Visual Studio 2010 und wird nicht mehr gepflegt.

Das Tool Xsd2Code wird gepflegt und weiterentwickelt, außerdem ist es gut in Visual Studio 2010 integrierbar, da der Entwickler nur per Mausklick ein Objektmodell aus den XSD-Dokumenten generieren kann.

Zwar erfüllt Xsd2Code die oben genannten Anforderungen, bei der Nutzung im Rahmen der Diplomarbeit zeigten sich dennoch Schwierigkeiten, die zum Teil auf die von der Object Management Group freigegebenen XSD-Dokumente zurückzuführen sind, worauf nachfolgend eingegangen wird.

Kritische Anmerkung

Die BPMN 2.0 Spezifikation stellt ein objektorientiertes Metamodell dar, das in UML-Form vorliegt. Die Darstellung in XML und speziell in XSD wird erschwert, da die XML-Sprache z.B. nicht objektorientiert ist. Das Modell ist sehr komplex, was allein schon durch die Mehrfachvererbung von Elementen zustande kommt. Die Mehrfachvererbung ist ein wichtiges Merkmal objektorientierter

Systeme, das in UML sehr gut realisiert werden kann. Um die Mehrfachvererbung zu unterstützen, werden in den XSD-Definitionen die sog. substitutionGroup-Elementattribute[1] verwendet. Dies kann sich nachteilig - vor allem bei größeren Modellen, wie dem BPMN-Metamodell - auf die Deserialisierung auswirken, da der vom Xsd2Code-Tool erstellte Quellcode nicht vollständig konsistent ist. Er muss manuell nachbearbeitet werden, damit die Serialisierung korrekt funktioniert.

4.2.2.2 Bestandteile des Converter-Add-Ins

Die Themen dieses Abschnitts sind die Bestandteile und Arbeitsweise des Visio-Add-Ins. Abbildung 55 zeigt die Benutzeroberfläche des Konvertierungs-Add-Ins vor und nach der Integration in das QUAM-Modeler-Add-In.

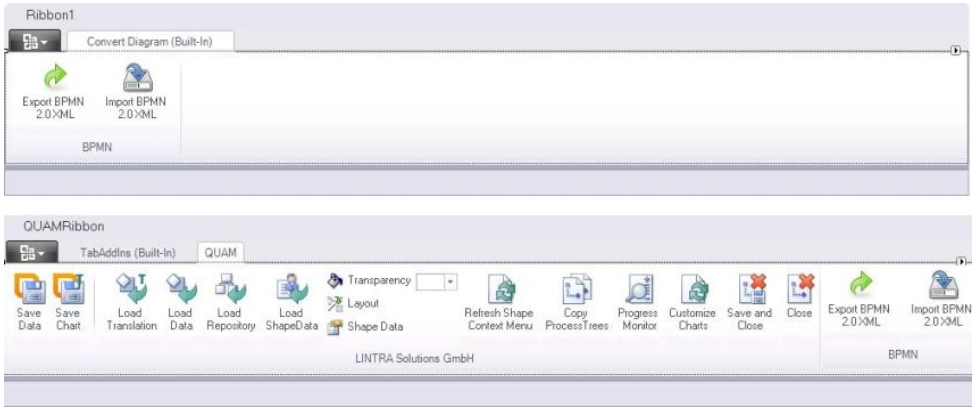


Abbildung 55 Benutzeroberfläche der Export Erweiterung

Basis für die Entwicklung der Konvertierungserweiterung sind die Visual Studio Tools for Office, auch als VSTO bekannt. Wie es der Name bereits aussagt, handelt es sich um Werkzeuge zum Entwickeln von Geschäftsanwendungen für Office (auch unter der Bezeichnung Office Business Applications/ OBA bekannt). VSTO stellt eine Reihe von Projektvorlagen für Visual Studio bereit, mit deren Hilfe Office-Anwendungen unter dem Einsatz der Sprache C-Sharp um neue Funktionalitäten erweitert werden (vgl. Microsoft, 2012). Der Grund für die Auswahl von VSTO als Werkzeug im Rahmen dieser Arbeit war zum einen

[1] Dadurch kann innerhalb einer Elementdeklaration angegeben werden anstelle welcher anderen Elemente das gerade deklarierte Element verwendet werden darf.

die Tatsache, dass die von der Firma LINTRA angebotenen Anwendungen alle in .NET und C-Sharp implementiert worden sind. Zum anderen ist VSTO in die Programmierumgebung Visual Studio 2010 integriert, daher müssen keine neue Entwicklungsumgebung angeeignet und Sprache erlernt werden. Ein anderer Grund für die Auswahl ist, dass durch die VSTO auch komplexe Lösungen in eine Office Anwendung und speziell in Visio integriert werden können, ohne dass der Anwender die ihm bekannte Umgebung verlassen muss.

Die Entwicklungsumgebung

Als Entwicklungsumgebung wurde für diese Arbeit Visual Studio 2010 Ultimate Edition eingesetzt, wodurch die für die VSTO-Entwicklung erforderlichen Komponenten mitinstalliert wurden. Dazu gehören das .NET Framework, die VSTO-Laufzeit und die primären Interop-Assemblies, auf die im nächsten Abschnitt eingegangen wird. Für die VSTO-Entwicklung ist auch die Installation der entsprechenden Version von Microsoft Office notwendig.

Das Objektmodell von Visio basiert auf COM (Component Object Model), aus diesem Grund sind für die Programmierung die entsprechenden Interop-Assemblies notwendig. Sie stellen eine Art Wrapperklassen dar und ermöglichen den Zugriff auf .NET auf COM-Komponenten. Die Interop-Assemblies liegen in vorkompilierter Form bereit und werden von Microsoft als Primary Interop Assemblies verteilt (vgl. Titel, 2011).

Die Primary Interop Assemblies (PIAs)

Die PIAs beinhalten einerseits die Beschreibungen der Datentypen, die in Office und speziell Visio verwendet werden. Andererseits enthalten sie eine Reihe von Wrapperklassen, mit denen aus .NET auf nicht verwalteten Code der Objektmodelle in Office zugegriffen werden kann.

Das Visio-Add-In

Das im Rahmen dieser Arbeit erstellte Add-In stellt eine Erweiterung für Visio dar, das automatisch mit dem Start von Visio bzw. QUAM-Modeler ausgeführt wird. Das Add-In besteht aus einer Assembly, die auch automatisch geladen wird, sobald Visio gestartet wird. Zentraler Anlaufpunkt für den Code ist die Klasse ThisAddIn.cs, das ist das Add-In selbst. Diese Klasse stellt Eigenschaften bereit, mit denen der Zugriff auf das Objektmodell von Visio

ermöglicht wird sowie auf Ereignisse zum Ausführen des Codes beim Starten oder Beenden des Add-Ins.

Arbeitsweise des Visio Add-Ins

Wird Microsoft Visio gestartet, so wird auch die Assembly des Add-Ins geladen, wodurch die Konvertierungsfunktionen sofort der Visio-Anwendung zur Verfügung stehen. Der Dialog zwischen den COM-Komponenten von Visio und der Assembly erfolgt über die PIAs. Aus Sicherheits- und Stabilitätsgründen wird für das Add-In eine eigene Anwendungsdomäne erstellt, in die die Assembly geladen wird. Somit haben Fehler im Add-In-Code keinen Einfluss auf andere Add-Ins.

Beim Laden des Visio Add-Ins werden sukzessive folgende Schritte durchgeführt (vgl. Titel, 2011):

1. Beim Starten von Visio wird die Windows-Registrierung nach zugeordneten Add-Ins überprüft.
2. Wird das Visio Add-In gefunden, so wird die VSTO-Laufzeit gestartet.
3. Das aktuellste Anwendungs- und Bereitstellungsmanifest wird geladen.
4. Ist das Visio-Add-In vertrauenswürdig, sucht die VSTO-Laufzeit nach neuen Versionen der Assembly und lädt ggf. eine aktuelle Version in den ClickOnce-Cache.
5. Die VSTO-Laufzeit erzeugt eine neue Sandbox-Anwendungsdomäne, in die die Assembly anschließend geladen wird.
6. Die RequestComAddInAutomationService- und die RequestService-Methoden werden aufgerufen, falls diese überschrieben wurden.
7. Der Startup-Ereignishändler der Assembly wird aufgerufen.

Der beschriebene Vorgang und die Interaktion der Office-Anwendung mit dem Add-In über die primäre Interop-Assembly sind in Abbildung 56 dargestellt.

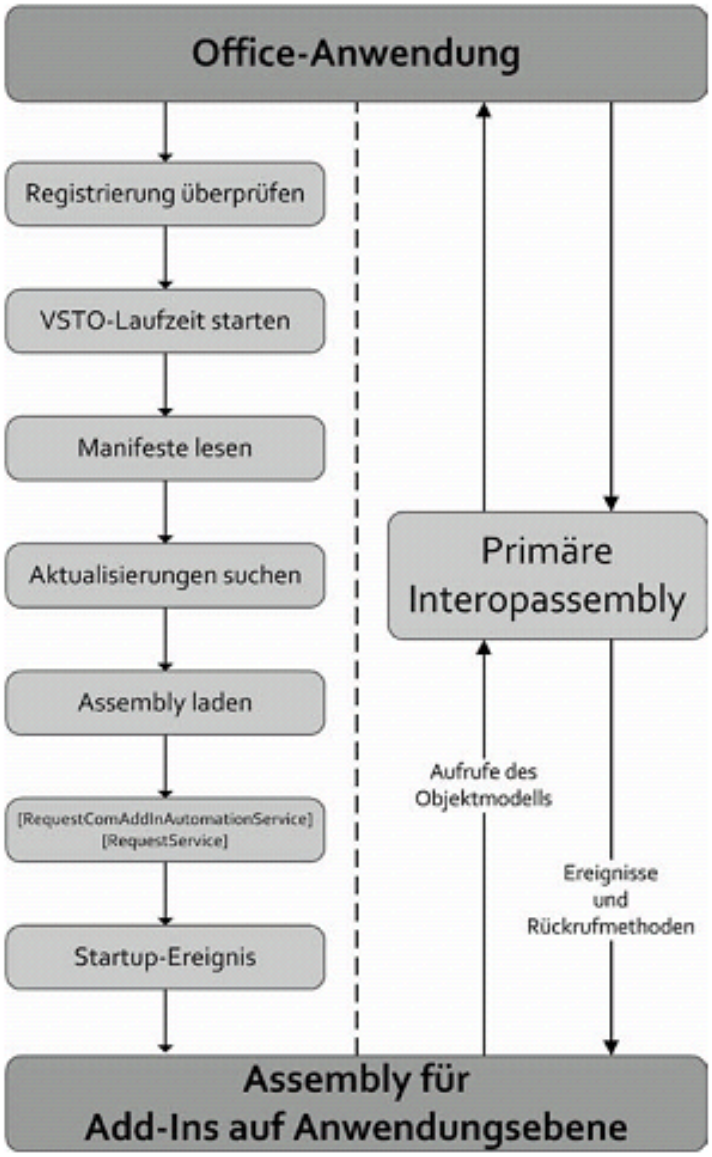


Abbildung 56 Laden und Ausführen des Visio Add-Ins (vgl. Titel, 2011)

C-Sharp Klassen und deren Beziehungen

Folgende Klassen der Export-Erweiterung, die kurz erläutert werden, waren für die Umsetzung dieser Arbeit zentral:

- **Serialization.cs**

  Die Klasse Serialization.cs beinhaltet die Kernfunktionalität der Erweiterung. Der Klassenkonstruktor wird von der Ribbon.cs Klasse aufgerufen.

  Die zentrale Methode serializeVisioDiagramm implementiert den eigentlichen Konvertieralgorithmus. Der Algorithmus, auf den noch gesondert eingegangen wird, beginnt mit der initialen Erstellung einer XML-

Gerüststruktur, die sukzessive mit XML-Elementen aus dem Diagramm gefüllt wird.

- **ConvertCoordinates.cs**

Das Koordinatensystem von Visio unterscheidet sich von dem der BPMN 2.0 Spezifikation. Die Klasse ConvertCoordinates.cs konvertiert das Visio-Koordinatensystem in das Koordinatensystem der BPMN 2.0 Spezifikation. Das Visio-Koordinatensystem beginnt oben links, während das BPMN-Koordinatensystem unten links beginnt. Übernimmt man die Angaben von Visio direkt in das XML-Dokument, werden die Zeichnungen nicht nur unterschiedlich aussehen, sondern es ergibt sich unter Umständen eine ganz andere Semantik, falls z.B. die Flusselemente nicht in den gleichen Schwimmbahnen liegen. Daher war es wichtig, einen Algorithmus zu implementieren, der die Umwandlung der Koordinaten des Koordinatensystem des Visio-Systems in das Koordinatensystem der BPMN 2.0 Spezifikation sicherstellt.

- **GenericInfo.cs**

Wie es der Name aussagt, liefert diese Klasse generische Informationen über die Shapes im Diagramm zurück. Die Klasse beinhaltet die drei Methoden *getShapeBounds*(), *getShapeGUID*() und *getConnectorPoints*(). Im Zuge des Diagrammdurchlaufs werden von jedem Shape Daten abgefragt. Einige Shapedaten sind generischer Art, sie müssen bei jedem Shape vorhanden sein. Daher wurden die Abfragen auf diese Daten als zentrale Funktionen in eine Extraklasse ausgelagert. Die Klasse kann zukünftig mit weiteren Abfragen dieser Art erweitert werden.

- **Ribbon.cs**

Diese Klasse beinhaltet die Ereignishandler für die Steuerelemente auf der Erweiterungsschaltfläche.

- **ThisAddIn.cs**

Auf die Funktionalitäten der ThisAddIn-Klasse wurde im vorhergehenden Abschnitt eingegangen.

- **BPMN20.designer.cs**

  Diese Klasse beinhaltet das komplette BPMN-Objektmodell, implementiert in C-Sharp. Sie dient als Container für die insgesamt über 150 C-Sharp-Klassen, die durch die Deserialisierung erstellt sind. Wie schon erwähnt, mussten bei einigen der Klassen manuellen Anpassungen vorgenommen werden, damit die spätere Serialisierung korrekt ablaufen kann.

4.2.2.3 Algorithmus für den Export

Der Algorithmus erstellt sukzessive eine hierarchische Objektstruktur, die aus mehreren Klasseninstanzen mit entsprechend gesetzten Eigenschaften besteht. Wenn alle Shapes durchlaufen sind, wird die so erstellte Objektstruktur in eine XML-Datei serialisiert. Da jede einzelne Klasse ein XML-Element repräsentiert, werden im Folgenden die Begriffe Klasse und XML-Element synonym verwendet. Begonnen wird mit der Initialisierung der obligatorischen Elementobjekte, die in jeder BPMN 2.0 XML-Struktur vorhanden sind, unabhängig davon, welche Symbole sich auf dem Diagramm befinden. Das ist z.B. das Root-Element <definitions>, das alle anderen Elemente umhüllt und im Objektmodell durch die C-Sharp-Klasse *tDefinitions* repräsentiert wird. Weitere Klassen, die initial erzeugt werden, sind *tProcess* und *BPMNDiagram*, sie repräsentieren jeweils den Semantik- und Präsentationsteil (<process> und <BPMNDiagram>) des BPMN-Modells.

Anschließend wird generell danach unterschieden, mit welcher Schablone das Diagramm erstellt wurde und ob es Swimlane-Shapes beinhaltet. In Abhängigkeit davon, welcher der Fälle vorliegt, werden unterschiedliche Wege beschritten:

- Falls das Diagramm mit der BPMN-Schablone erstellt wurde, findet eine weitere Unterscheidung statt, und zwar jene, ob das Diagramm Swimlanes beinhaltet oder die Prozesselemente (FlowNodes) Ereignisse, Aktivitäten, Gateways usw. frei im Diagramm positioniert sind.
  - Wenn das Diagramm Swimlane-Shapes beinhaltet, muss analysiert werden, welche Prozesselemente sich in den einzelnen Schwimmbahnen befinden, damit ein korrektes BPMN-XML-Modell

generiert werden kann. Die Swimlane-Shapes besitzen im Objektmodell von Visio die sogenannte Containereigenschaft. Solche Shapes können mehrere andere Shapes beinhalten, die logisch zusammengehören und auf die programmatisch zugegriffen werden kann. Im Fall eines BPMN-Diagramms ist das von Bedeutung, weil in der BPMN-XML-Struktur innerhalb des <lane>-Elements alle Identifizierer von Shapes aufgelistet werden müssen, die sich in der jeweiligen Schwimmbahn befinden.

  - Existieren im Diagramm keine Swimlane-Shapes, findet ein Prozess statt aber keine Konversation. Dementsprechend sind auch keine Zuordnungen von Aufgaben zu Organisationseinheiten oder Rollen möglich.

- Wurde das Diagramm mit der QUAM-Standardschablone erstellt, braucht es nicht auf die Existenz von Swimlanes und Pools untersucht werden; es findet aus BPMN-Perspektive keine Konversation statt. Es ist aus BPMN 2.0-Sicht ein Prozess vorhanden, der Prozesselemente, wie Aktivitäten, Gateways und Ereignisse, enthält.

Anschließend geht der Algorithmus eine Folge von Schritten durch, bei der alle Diagramm-Shapes durchlaufen werden. Jedes gefundene Shape wird auf bestimmte Eigenschaften untersucht, und es werden ggf. verschiedene Wege eingeschlagen. Zu jedem durchlaufenen Shape wird das entsprechende BPMN-Element erstellt und zur bestehenden Struktur addiert. Sind alle Diagrammshapes durchlaufen und entsprechende Objekte erstellt und zugeordnet, wird die Objektstruktur der XML-Serializer-Klasse als Parameter übergeben. Diese serialisiert dann über die Methode ***serialize()*** die Objektstruktur in eine BPMN-XML-Struktur.

Beim Durchlaufen des Prozessdiagramms werden viele Funktionen aus der Visio API aufgerufen, von denen nachfolgend die wichtigsten vorgestellt werden.

API Aufruf	Beschreibung
Globals.ThisAddIn.Application	Zugriff auf Visio.
Application.ActivePage.Shapes	Zugriff auf alle Shapes der aktuellen Seite.
Shape.get_Cells ("Prop.BpmnName")	Greift auf eine ShapeSheet-Zelle von Visio zu, darin sind alle Informationen des Shapes abgespeichert.
Shape.ConnectedShapes (…, "")	Alle mit dem aktuellen Shape via ausgehender Sequenzfluss verbunden Shapes ohne Filter („") werden zurückgeliefert.
Shape.GluedShapes(…,"");	Liefert alle Shapes zurück, die mit dem Shape direkt verbunden sind. Eine Filterung nach bestimmten Eigenschaften ist möglich.
containerProps= shape.ContainerProperties	Die Container-Eigenschaften eines Shapes werden überprüft.
shape.get_UniqueID	Liefert die innerhalb des Visio Dokuments eindeutige ID.

Tabelle 9 Wichtige Funktionen der Visio API

Abschließend wird als Test ein in QUAM modelliertes BPMN 2.0 Diagram exportiert und in das Modellierungswerkzeug Signavio importiert:

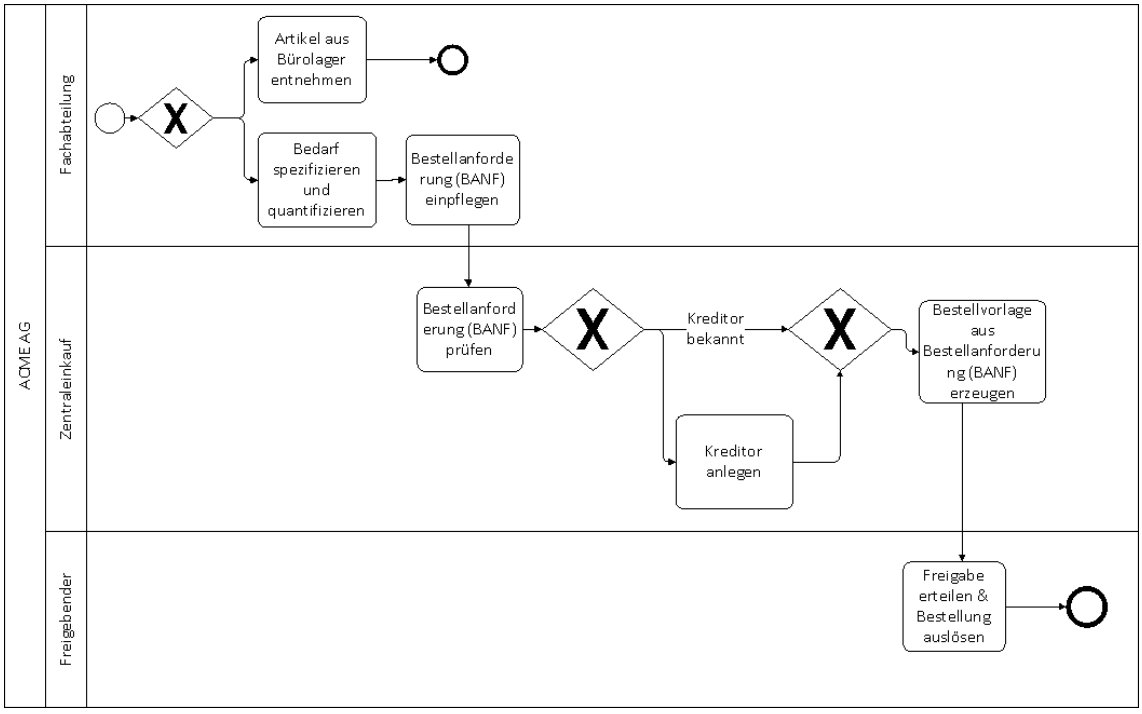


Abbildung 57 Testdiagramm, das von QUAM exportiert wurde

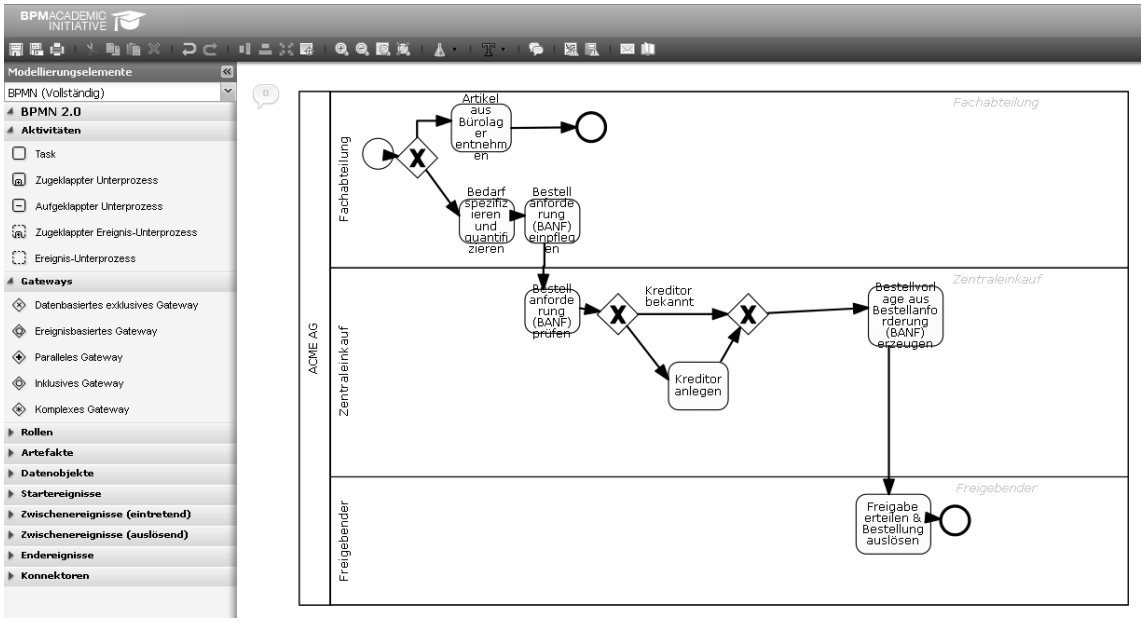


Abbildung 58 In das Modellierungswerkzeug Signavio importiertes Testdiagramm

Das Ergebnis zeigt deutlich, dass die Diagramme semantisch gleich sind und visuell, bis auf die Skalierung der Schriften, weitestgehend identisch aussehen.

5 Zusammenfassung und Ausblick

Nachdem in dieser Diplomarbeit zu Beginn die Darstellung der Grundlagen des Business Process Managements erfolgte, wurde anschließend ein Vergleich von derzeit weit verbreiteten Modellierungssprachen anhand bestimmter Kriterien vorgenommen. Ziel war es, eine geeignete Sprache als Erweiterung für die Prozessmodellierung in QUAM 2.0 zu implementieren. Beim Vergleich der Modellierungssprachen war es zum einen von Bedeutung, diese auf Kriterien, wie z.B. Verbreitungsgrad und unterstützte Phasen der Modellierung, zu untersuchen; zum anderen sollte es sich bei der auszuwählenden Modellierungssprache um einen Standard handeln. Es wurde festgestellt, dass die EPK und die BPMN derzeit als Prozessmodellierungssprachen eine große Bedeutung haben, wobei die BPMN seit der Verabschiedung ihrer letzten Version nicht mehr „nur eine graphische Notation“ darstellt, sondern ein von vielen Unternehmen anerkannter und implementierter Modellierungsstandard ist, der ein umfassendes Metamodell und eine ebensolche Ausführungssemantik besitzt. Aus diesen Gründen ist die Wahl einer in QUAM 2.0 zu implementierenden Modellierungssprache auf die BPMN 2.0 gefallen. In einem weiteren Kapitel wurden dementsprechend die technischen Grundlagen des Standards BPMN 2.0 vorgestellt; anschließend wurde detailliert auf das von der OMG verabschiedete Metamodell und seine Teilbereiche eingegangen. Im letzten Kapitel wurde die Erweiterung des QUAM 2.0 Systems um eine BPMN 2.0 Modellierungs- und Exportkomponente prototypisch entworfen und implementiert. Danach wurde das Exportergebnis in das Modellierungstool Signavio importiert und erfolgreich verifiziert. Somit wurde zum einen die Interoperabilität von in QUAM 2.0 modellierten Prozessmodellen erreicht; zum anderen wurde eine wichtige Vorbereitung in Richtung ausführbare Prozessmodelle geschaffen.

Der Markt für BPM-Systeme erfährt eine sehr dynamische Entwicklung; derzeit werden zunehmend Systeme angeboten, die auch die Ausführung von BPMN 2.0 Modellen unterstützen (Götz, 2011). Auch Microsoft bietet in Visio 2013 die Unterstützung der kompletten BPMN 2.0 Symbolpalette. Inwieweit sich BPMN 2.0 als Modellierungsstandard, Austauschformat und Ausführungssprache etablieren wird, bleibt abzuwarten.

6 Literaturverzeichnis

Allweyer, Thomas. 2012. *BPMN – eingesetzt zur Dokumentation, weniger zur Ausführung.* http://www.kurze-prozesse.de/2012/07/09/bpmn-eingesetzt-zur-dokumentation-weniger-zur-ausfuhrung/ : [09.10.2012]

—. 2009. *BPMN 2.0 - Business Process Model and Notation: Einführung in den Standard für die Geschäftsprozessmodellierung.* Norderstedt : Books on Demand

—. 2005. *Geschäftsprozessmanagement.* Bochum : Herdecke

Arndt, Hans-Knud. 2012. *Vorlesung Prozessmanagement.* http://bauhaus.cs.uni-magdeburg.de/cms/index : [01.09.2012]

Bartonitz, Martin. 2009. *BPM Round-Trip Engineering - Vision und Wirklichkeit.* http://www.bpm-netzwerk.de/articles/d915570d1fef5e23011ff065587e034c&source=4&query=bpmn. : [12.01.2012]

—. 2005. *BPMS, BPML, BPEL, WMS, XPDL,...* http://www.bpm-netzwerk.de/content/articles/viewArticle.do?id=17 : [01.10.2012]

—. 2010. *Fallstricke beim BPM Round-Trip Engineering – ein typisches Beispiel im Kontext der BPMN.* http://www.saperionblog.com/lang/de/fallstricke-beim-bpm-round-trip-engineering-ein-typisches-beispiel/2364/ : [01.10.2012]

Boddenberg, Ulrich. 2011. *Microsoft SharePoint Server 2010 und SharePoint Foundation 2010.* Bonn : Galileo Computing

Brodsky, Stephen A. 2002. *Mastering XMI: Java Programming with XMI, XML, and UML.* New York : Wiley&Sons

Computerwoche. 2012. *Computerwoche.* http://www.computerwoche.de/software/soa-bpm/2350328/ : [25.05.2012]

Drawehn, Jens. 2010. *Business Process Modeling 2010: Modellierung von ausführbaren Geschäftsprozessen mit der Business Process Modeling Notation.* Stuttgart : Fraunhofer-Institut für Arbeitswirtschaft und Organisation

Dumke, Reiner. 2003. *Software Engineering.* Magdeburg : Vieweg, 2003

EVIATEC. *Microsoft SharePoint Funktionsübersicht.* http://www.eviatec.de/LP_SP_Funktionsuebersicht.html : [14.06.2012]

Fischer, Peter. 2007. *Lexikon der Informatik.* Berlin Heidelberg : Springer

Freund, Jakob und Götzer, Klaus. 2008. *Vom Geschäftsprozess zum Workflow. Ein Leitfaden für die Praxis.* München : Hanser

Freund, Jakob und Rücker, Bernd. 2010. *Praxisbuch BPMN 2.0.* München, Wien : Hanser

Gadatsch, Andreas. 2005. *Grundkurs Geschäftsprozessmanagement.* Wiesbaden : Vieweg

Gadatsch, Andreas und Daryoush, Daniel Vaziri. 2012. *Ergebnisse der Kurzumfrage zum Stand von BPMN im deutschsprachigen Raum.* Hochschule Bonn-Rhein-Sieg : Schriftenreihe des FB Wirtschaftswissenschaft Sankt Augustin

Götz, Manuel. 2011. *BPM-Systeme im Vergleich.* [12.09.2012] : http://www.itransparent.de/sites/default/files/Vergleich_BPM_Tools_ActiveVOS_Activiti_Intalio_BizAgi.pdf

Jeckle, Mario. 2004. *Scriptum zur Vorlesung XML.* http://www.jeckle.de/vorlesung/xml/script.html#StrukturelleGrundkonzepte : [01.05.2012]

Josef, Staud. 2006. *Geschäftsprozessanalyse: Ereignisgesteuerte Prozessketten und objektorientierte Geschäftsprozessmodellierung für Betriebswirtschaftliche Standardsoftware.* Berlin : Springer

Koplowitz, Rob. 2010. *Best Practices in SharePoint 2010 Adoption and Migration.* s.l. : Forrester

Krause, Jörg. 2010. SharePoint für Entwickler. *IT-Republik*

Kruczynski. 2008. *Prozessmodellierung im Wettbewerb: EPK vs. BP.* s.l. : is report

Lelic, Senaj. 2004. *Visio Shape Programmierung.* httü://msdn.microsoft.com/de-de/library/bb979238.aspx : [02.05.2012]

LINTRA. *Prozessmodellierung und Management auf Basis von Web 2.0.* http://www.lintra.net/deutsch/news/prozessmodellierung-und-management-auf-basis-von-web-2.0.html : [01.09.2012]

—. 2010. *QUAM Benutzerhandbuch.* Magdeburg : s.n.

Martin, Rene. 2011. *Microsoft Visio 2010-Programmierung.* s.l. : Microsoft

Microsoft. 2005. *Einführung in die XML-Serialisierung.* http://msdn.microsoft.com/de-de/library/182eeyhh(v=vs.80).aspx : [10.10.2012]

—. 2005. *Einführung in die XML-Serialisierung.* http://msdn.microsoft.com/de-de/library/182eeyhh(v=vs.80).aspx : [10.10.2012]

—. 2012. *LINQ (Language-Integrated Query, sprachintegrierte Abfrage).* http://msdn.microsoft.com/de-de/library/bb397926.aspx : [10.10.2012]

—. 2012. *LINQ to XML.* htt://msdn.microsoft.com/de-de/library/bb397926.aspx : [10.10.2012]

—. 2012. *Office Developer Center.* http://msdn.microsoft.com/de-de/office/hh133430.aspx : [01.10.2012]

—. 2010. *Server- und Websitearchitektur: Übersicht über das Objektmodell.* http://msdn.microsoft.com/de-de/library/ms473633.aspx : [10.02.2012]

—. 2010. *SharePoint 2010 Content: ECM-Funktionen für alle Benutzer.* http://sharepoint.microsoft.com/de-de/product/capabilities/content/Seiten/default.aspx : [01.09.2012]

Miers, Derek. 2010. *SHAREPOINT AND BPM — FINDING THE SWEET SPOT.* s.l. : Forrester Research

Miles, Doug. 2011. *State of the ECM Industry 2011.* s.l. : AIIM

Morelli, Frank. 2010. *Geschäftsprozessmodellierung ist tot – lang lebe die Geschäftsprozessmodellierung.* Pforzheim : Hochschule Pforzheim

Object Management Group. 2007. *Unified Modeling Language (OMG UML), Infrastructure V2.1.2.* Chicago : s.n.

OMG. 2006. *BPMI.* http://www.bpmi.org/ : [12.08.2012]

—. 2012. *BPMN Implementers.* http://www.bpmn.org/#tabs-implementers : [01.09.2012]

—. 2010. *Business Process Model and Notation.* http://www.omg.org/spec/BPMN/2.0/PDF : [12.10.2012]

Osterloh, Margit und Frost, Jenna. 2006. *Prozessmanagement als Kernkompetenz.* Wiesbaden : Gabler

Parker, David J. 2010. *Microsoft Visio 2010 Business Process Diagramming and Validation.* s.l. : Microsoft

Pettey, Christy und Goasduff, Laurence . 2010. *Gartner Reveals Five Business Process Management Predictions for 2010 and Beyond.* http://www.gartner.com/it/page.jsp?id=1278415 : [10.10.2012]

Rademakers, Tijs. 2012. *Activity in Action.* s.l. : Manning

Rosemann, Michael, Kugeler, Martin und Becker, Jörg. 2008. *Prozessmanagement.* Berlin, Heidelberg : Springer

Rowley, Michael. 2009. *Thinking about BPM? What you should REALLY ask your BPMS vendor.* http://www.activevos.com/blog/bpel/thinking-about-bpm-what-you-really-should-ask-your-bmps-vendor/2009/05/08/ : [25.06.2012]

Schmelzer, Herman und Sesselmann, Wolfgang. 2010. *Geschäftsprozessmanagement in der Praxis.* s.l. : Hanser

Schmidt, Goetz. 2011. *Business Process Management Common Body of Knowledge - BPM CBOK.* s.l. : European Association of Business Process Management

Silver, Bruce. 2011. *BPMN Method and Style.* s.l. : Cody-Cassidy

—. 2012. *BPMS Watch.* http://www.brsilver.com/ : [01.09.2012]

Staud, Josef. 2006. *Geschäftsprozessanalyse: Ereignisgesteuerte Prozessketten und objektorientierte Geschäftsprozessmodellierung für Betriebswirtschaftliche Standardsoftware.* Berlin : Springer

Steudel, Isabelle und Städtler, Nico. 2008. *Prozessmodellierung in BPMN und EPK.* Leipzig : s.n.

Titel, Jan. 2011. *OFFICE 2010 Programmierung mit VSTO und .NET 4.0.* s.l. : Hanser

Vonhoegen, Helmut. 2007. *Einstieg in XML: Grundlagen, Praxis, Referenz .* s.l. : Galileo Computing

W3C. 2008. *Extensible Markup Language (XML) 1.0 (Fifth Edition).* http://www.w3.org/TR/REC-xml/ : [12.10.2012]

—. 2004. *XML Information Set.* http://www.w3.org/TR/xml-infoset/ : [12.10.2012]

Wikipedia. 2012. *Unified Modeling Language.* http://de.wikipedia.org/wiki/Unified_Modeling_Language : [01.11.2012]

Wittges. 2005. *Verbindung von Geschäftsprozessmodellierung und Workflow-Implementierung.* Hohenheim : Gabler

Abschließende Erklärung

Ich versichere hiermit, dass ich die vorliegende Diplomarbeit selbständig, ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Magdeburg, den 06. Februar 2013

Tihomir Todorov