PETER STÜRZEL

MODELLIERUNG UND AUSFÜHRUNG VON WORKFLOWS

unter Berücksichtigung
mobiler Kontextinformationen

KIT Scientific Publishing

Peter Stürzel

Modellierung und Ausführung von Workflows unter Berücksichtigung mobiler Kontextinformationen

Modellierung und Ausführung von Workflows unter Berücksichtigung mobiler Kontextinformationen

von
Peter Stürzel

KIT Scientific Publishing

Dissertation, Karlsruher Institut für Technologie
KIT-Fakultät für Wirtschaftswissenschaften

Tag der mündlichen Prüfung: 24. Mai 2017
Erster Gutachter: Prof. Dr. Andreas Oberweis
Zweiter Gutachter: Prof. Dr. J. Marius Zöllner

Impressum

Scientific Publishing

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

www.ksp.kit.edu

Print on Demand 2018 – Gedruckt auf FSC-zertifiziertem Papier

ISBN 978-3-7315-0754-3
DOI 10.5445/KSP/1000078961

Modellierung und Ausführung von Workflows unter Berücksichtigung mobiler Kontextinformationen

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften
(Dr.-Ing.)

von der Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Dipl.-Inform. Peter Stürzel

Karlsruhe

Tag der mündlichen Prüfung: 24.05.2017
Erster Gutachter: Prof. Dr. Andreas Oberweis
Zweiter Gutachter: Prof. Dr. J. Marius Zöllner

Vorwort

Die vorliegende Arbeit basiert u.a. auf vielen studentischen Arbeiten, die ich am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) am Lehrstuhl am Karlsruher Institut für Technologie (KIT) betreut habe. Weiterhin sind viele Ideen, wissenschaftliche Publikationen und praktische Evaluationen innerhalb der Projektarbeit des Verbundprojektes „Robot2Business“, das vom Bundesministerium für Wirtschaft und Technologie gefördert wurde, entstanden.

Danken möchte ich den vielen Meta-Unterstützer der vorliegenden Arbeit. Meinen größten Respekt möchte ich meinen Eltern aussprechen, die es trotz vier Kinder und schwieriger finanzieller Momente geschafft haben, einen Menschen aus mir zu machen. Danken möchte ich dem Rest der Familie, meinen Freunden und der ganzen spanischen Gruppe Karlsruhe, die mich stets mental unterstützt haben.

Zuletzt möchte ich es nicht verpassen den wichtigsten Bestandteilen meines Lebens zu danken. Danke Xiana, für deine Geduld, für deine Liebe, für deinen Lebenswillen, für deine Unerschrockenheit, für dein Mitgefühl und für deinen Glauben an mich. Danke Ana für deine Energie, die mir neue Kraft gegeben hat.

Der Wert der Leistung liegt im Geleisteten...

Albert Einstein

Peter Stürzel, Ludwigshafen am Rhein, Januar 2018

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

3GPP	3rd Generation Partnership Project
API	Application Programming Interface
App	Mobile Applikation
AR	Augmented Reality
BPMN	Business Process Modeling Notation
CAS	Context-Aware Service
CMS	Content Management Systems
COM	Component Object Model
CRM	Customer Relationship Management
CTT	Concur Task Tree
DBMS	Data Base Management System
ERP	Enterprise Resource Planning
FTP	File Transfer Protocol
GIS	Geoinformationssystem
GML	Geographic Markup Language
GSM	Global System for Global Communications
HCI	Human Computer Interaction
HMI	Human Machine Interface

HTTP	Hypertext Transfer Protocol
IKT	Informations- und Kommunikationstechnik
IM	Instant Messaging
IMAP	Internet Message Access Protocol
IS	Informationssystem
KPI	Key Performance Indicator
LBS	Location Based Services
LDAP	Lightweight Directory Access Protocol
LTE	3GPP Long Term Evolution
MAIS	Management Instance System
MEMIS	Member Management Instance System
MMS	Multi Messaging Service
MMSC	Multimedia Messaging Service Center
MPL	Mobile Process Landscaping
MSDE	Microsoft SQL Server Data Engine
mWfMS	Mobiles Workflow-Managementsystem
OCL	Object Constraint Language
OE	Ortseinschränkung
OMG	Object Management Group
OOP	Objekt-orientierte Programmierung

OSGi	Open Services Gateway initiative
P2P	Peer to Peer Konnektivität
POI	Point-of-Interest
POP3	Post Office Protocol, Version 3
R2B	Verbundprojekt Robot2Business
REST	Representational State Transfer
SCM	Supply Chain Management
SDK	Software Development Kit
SGML	Standard Markup Language
SMS	Short Message Service
SMSC	Short Message Service Center
SMTP	Simple Mail Transfer Protocol
SOAP	Netzwerkprotokoll auf IP-Basis
SQL	Structured Query Language
TAM	Technology Acceptance Model
TCP/IP	Transmission Control Protocol/ Internet Protocol
TRA	Theory of Reasoned Action
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications Systems

URI	Uniform Ressource Identifier
WAP	Wireless Application Protocol
WfM	Workflow-Management
WfMC	Workflow Management Coalition
WfMS	Workflow-Managementsystem
WLAN	Wireless Local Area Network
WML	Wireless Markup Language
WPAN	Wireless Personal Area Network
WS	Web Service
WSDL	Web Service Description Language
WSFL	Web Service Flow Language
XML	eXtensible Markup Language
XPDL	XML Process Definition Language

1 Einleitung

Die Wertschöpfung eines Unternehmens wird durch Geschäftsprozesse erbracht. Ein Geschäftsprozess besteht aus einer koordinierten Reihenfolge von Aktivitäten, die ausgeführt werden, um ein unternehmerisches Ziel zu erreichen. Das Management von Geschäftsprozessen ermöglicht die Kontrolle von Kosten, Zeit und Qualität bei der Leistungserbringung. Die Unterstützung der Geschäftsprozesse durch Informations- und Kommunikationstechnologie (IKT) ist ein maßgeblicher Erfolgsfaktor für Unternehmen. Geschäftsprozesse werden durch Informationssysteme bzw. Workflow-Managementsysteme (WfMS) automatisiert [Holl95] und sorgen auf der operativen Ebene für die informationstechnische Ausführung in Form von Workflows. Workflow-Management hat das Ziel, die Durchlaufzeiten von (Geschäfts-) Prozessen zu minimieren, Transparenz innerhalb der verschiedenen Unternehmensbereiche zu schaffen (z. B. durch Modellierung), Qualität zu erhöhen und durch besseren Ressourceneinsatz (z. B. durch Parallelisierung) Kosten zu reduzieren [Ober96, S. 25]. In Unternehmen werden Geschäftsprozessmodelle bzw. WfMS zur Überwindung des *Business/IT-Gap* eingesetzt [NeSt07].

Heutzutage wollen die Nutzer (Mitarbeiter und Kunden) der WfMS (Teile der) Geschäftsprozesse von ihrem mobilen Endgerät aus nutzen [TuPo04]. Die bestehenden stationären IKT müssen um die mobilen Aspekte erweitert werden, wie der weitere Verlauf der vorliegenden Arbeit zeigen wird. Beim Workflow-Management (WfM) werden mobile Systeme (mobile Geräte) integriert, um z. B. eine ortsabhängige Prozessausführung zu unterstützen. Bei der Integration der mobilen Aspekte unterliegen die WfMS bestimmten Restriktionen der mobilen Geräte (Bsp. geringe Displaygröße), die systemweit berücksichtigt werden müssen.

Die mobilen Geräte eröffnen dem WfM viele neuartige Möglichkeiten. Das mobile Gerät erfasst den Umgebungskontext mit Sensoren und kann diese Kontextinformationen verarbeiten und an das WfMS weiterreichen. Durch

die Eigenortung (siehe Kapitel 2.1.6) bzw. die Ortsinformationen lassen sich beispielsweise Ortsbezüge innerhalb von Anwendungen realisieren (z. B. Identifikation jeglicher Kunden/Mitarbeiter innerhalb einer bestimmten Umgebung). Solche Dienste auf den mobilen Geräten werden als *Context-Aware Services* (CAS) oder als *Location-Based Services* (LBS) bezeichnet [Kuep05, S .1-3, S. 247-269]. Das wirtschaftliche Interesse an den CAS ist groß und die Potentiale werden intensiv in der Forschung diskutiert [DLNW13].

1.1 Problemstellung

Um Geschäftsprozesse auf einem mobilen Gerät zu nutzen, müssen bestehende WfMS um die mobilen Aspekte mobiler Geräte erweitert werden. Die mobilen Geräte stellen bei der Integration neue Anforderungen an das WfMS [ChNB16]. Die Kontext- bzw. Ortsinformationen mobiler Geräte werden innerhalb des WfMS integriert, damit eine informationstechnische Anwendung (innerhalb des WfMS) die Daten auswerten bzw. verarbeiten kann. Im weiteren Verlauf der vorliegenden Arbeit wird auf die Unterstützung der sogenannten Primärkontexte von CAS fokussiert. Die vier Primärkontexte [Kuep05] bilden die Basis der Umgebungskontexte (siehe Kapitel 2.2):

- Zeit
- Ort
- Identität
- Aktivität

Identitäten werden meist mit Hilfe von Benutzerrollen innerhalb von Workflows bzw. Geschäftsprozessen abstrahiert (siehe Kapitel 2.4). Die Rolle definiert einen Archetyp (vgl. Pattern, Prototyp) mit bestimmten Benutzerrechten (innerhalb des Workflows), um sie mehreren Identitäten zuweisen zu können. In vielen Geschäftsprozess-modellierungssprachen und WfMS ist die Abstraktion über Rollen bereits integraler Bestandteil (siehe Kapitel 2.4).

Innerhalb der Geschäftsprozessmodelle werden Aktivitäten in einer logischen (abhängig von der Fachlichkeit) zeitlichen Reihenfolge abgebildet. Zeitliche Kontextinformationen werden bereits bei der Modellierung und Ausführung von Workflows berücksichtigt und unterstützt [Ober90, AdAH98]. Die drei Primärkontexte der CAS (Identitäten, Zeitbezüge und Aktivitäten) sind in Geschäftsprozess-modellierungssprachen integriert (siehe Kapitel 2.3-2.6). Es fehlt jedoch eine Integration im Kontext der mobilen Geräte (siehe Kapitel 7.2). In IS/WfMS müssen (je nach Anwendungsfall) die speziellen Bedürfnisse mobiler Geräte (geringe Batterielaufzeit, geringe Rechenkapazität, usw) bedacht werden.

Orte bzw. Ortsbezüge werden innerhalb der Geschäftsprozessmodellierung meist nicht berücksichtigt. Orte wurden bereits innerhalb der Workflow-Modellierung in einzelnen Fällen integriert [JHHS00, ChLe04, HeKi09], jedoch fehlt eine adäquate Modellierung der ortsbezogenen Informationen (siehe Kapitel 4). Bei der Modellierung ist eine klare und eindeutige Abbildung von ortsrelevanten Kontextinformationen (siehe Kapitel 2.2) durch eine Annotation notwendig, die innerhalb von geeigneten Geschäftsprozessmodellierungssprachen integriert werden muss (siehe Kapitel 5). Die Annotation muss eine Abbildung vom Modell hin zum ausführbaren Code gewährleisten, um den zu Beginn dieses Kapitels erwähnten *Business/IT-Gap* zu überwinden bzw. um einen möglichst hohen Grad an Automatisierung zu erzielen. Zusätzlich fehlt eine adäquate und systemunabhängige Architektur eines WfMS, das eine informationstechnische Unterstützung von Ortsbezügen gewährleistet (siehe Kapitel 6).

1.2 Beiträge der Arbeit

Eine effiziente Integration der Ortsbezüge in Workflows und die damit einhergehende Integration der mobilen Geräte ist ein kritischer Erfolgsfaktor für Unternehmen, die ihren Mitarbeitern und Kunden mobile Dienst anbieten. Als maßgeblicher Anteil wird die IT-gestützte Modellierung und Ausführung der Workflows innerhalb existierender WfMS betrachtet. Die Herausforderungen (siehe Kapitel 1.1) benötigen geeignete Lösungen. Abb. 1.1

veranschaulicht die Beziehung zwischen einem modellierten Geschäftsprozess bzw. einem Workflow und dem unterstützenden stationären bzw. mobilen WfMS, das mobile Kontextinformationen berücksichtigen soll.

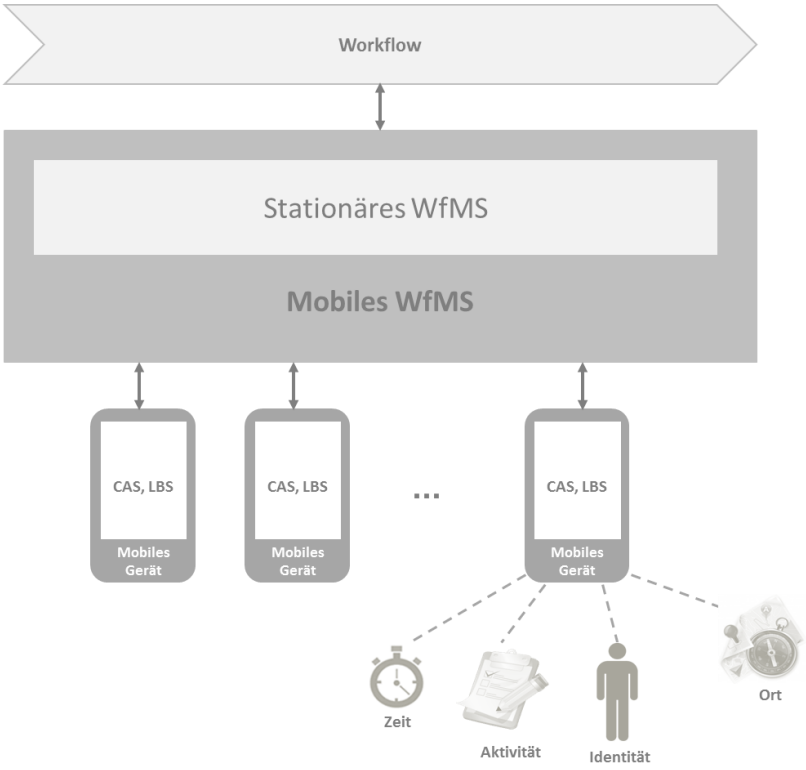


Abb. 1.1: Verknüpfung zwischen Workflow, WfMS, CAS und mobilen Primärkontexten

Ziele dieser Arbeit

Ziele dieser Arbeit sind, Workflow-Modelle um ortsbezogene Kontexte (vgl. Kapitel 4, Ortsbezüge) zu erweitern und Kontextinformationen von mobilen Systemen in bestehende stationäre WfMS zu integrieren (vgl. Abb. 1.1, mobile WfMS).

Um diese Ziele zu erreichen, basieren die Beiträge dieser Arbeit auf den relevanten Standards der *Workflow Management Coalition (WfMC).* Der Primärkontext Ort neben den weiteren Primärkontexten Zeit, Identität und Aktivität (siehe Abb. 1.1) bildet den Rahmen der vorliegenden Arbeit.

Für die Ausnutzung von Ortsbezügen bei der Workflowausführung werden ein Konzept und eine Modellierungsannotation benötigt. Die Modellierungsannotation soll eine informationstechnische Unterstützung erhalten, um die modellierten Workflows direkt innerhalb von WfMS ausführen zu können. Eine Software-Architektur soll die mobilen Aspekte berücksichtigen und eine Integration in bekannte Architekturen und Systeme ermöglichen (siehe Kapitel 2.5). Bestehende WfMS können auf Basis der vorgestellten Konzepte und Architekturbeschreibung zu mobilen WfMS erweitert werden.

Beitrag (1): Ortsbezüge für Workflows

In der vorliegenden Arbeit wird ein Konzept zur Berücksichtigung von Orten (bzw. Gebieten) in Workflows vorgestellt. Das Konzept wird mit Hilfe einer Annotation sprachunabhängig entwickelt. Die Annotation wird so entworfen, dass sie einen effizienten Umgang mit ortsbasierten Informationen gewährleistet. Ein Ortsmodell ist erforderlich, um das Konzept in möglichst viele Workflow-Modellierungssprachen integrieren zu können. Eine Modellierungsunterstützung auf Ebene des Workflowentwurfs gewährleistet eine frühe Berücksichtigung der Ortsinformationen innerhalb eines Softwareentwicklungsprozesses.

Die vorliegende Arbeit nutzt die Modellierung von Workflows als Instrument zum Entwurf des Informationssystems, in dem die Prozesse (Geschäftsprozesse, Systemintegrationsprozesse, Supportprozesse, usw.) in einem Modell beschrieben und anschließend automatisiert ausgeführt werden. Ein WfMS wird auf Basis der Referenzarchitektur der WfMC entworfen und innerhalb eines Architekturmodells vorgestellt (siehe Kapitel 6). Die zugehörige prototypische Implementierung (Tragfähigkeitsnachweis) wird in Kapitel 7.1 beschrieben.

Beitrag (2): Spracherweiterungen zur Integration von Ortsbezügen in Geschäftsmodellen.

Das Konzept der Ortsbezüge wird mit Hilfe von Spracherweiterungen für Petri-Netze (siehe Kapitel 2.6) und BPMN 2.0 (siehe Kapitel 2.7) vorgestellt, so dass es innerhalb weiterer Geschäftsprozessmodellierungssprachen integriert werden kann. Es wird gezeigt, dass Ortsbezüge als Spracherweiterung innerhalb von ortsabhängigen Anwendungsfällen nützlich anwendbar sind. Mit zusätzlichen Besonderheiten (siehe Kapitel 4.3.2-4.3.6 und Kapitel 5.3), wie z. B. Aggregationen oder hierarchischen Modellelementen wird die Anwendung der Ortsbezüge weiter verfeinert.

Beitrag (3): Evaluationen mobiler WfMS

Das mobile WfMS soll eine schnelle und flexible Modellierung und Ausführung der Workflows unterstützen. Der Softwareentwicklungsprozess wird anhand der Gegebenheiten des mobilen WfMS konkretisiert. Das mobile WfMS soll plattformunabhängig nach dem SOA-Paradigma entworfen und implementiert werden, so dass jedes Informationssystem mit dieser Erweiterung angereichert werden kann. Die serviceorientierte Architektur (SOA) dient als Hilfsmittel, um die Informationssysteme mit den Geschäftsprozessen zu verbinden [Josu08]. Ein Anwendungsfall aus dem landwirtschaftlichen Umfeld wird die Modellierung und Ausführung von Ortsbezügen zeigen. Der Anwendungsfall wurde innerhalb des Verbundforschungsprojektes Robot2Business[1] (R2B) entwickelt und erprobt.

Innerhalb eines mobilen Workflows werden viele Nutzeraufgaben mit Ortsbezügen modelliert. Diese Nutzeraufgaben werden auf einem mobilen Gerät bearbeitet. Es muss somit eine geeignete Abbildungsform innerhalb einer Anwendung für die Aufgaben (mobiler Nutzer) gefunden werden. Es werden verschiedene technische Lösungen zur Abbildung von Benutzeraufgaben vorgestellt. Zusätzlich wird eine mobile Applikation entwickelt, mit der z. B.

[1] www.r2b-online.de/

mehrere mobile Benutzeraufgaben bearbeitet werden können. Die Ergebnisse sollen Erkenntnisse für weitere Entwicklungsvorhaben zur Abbildung von Benutzeraufgaben in mobilen Applikationen liefern.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist wie folgt aufgebaut:

Das zweite Kapitel umfasst die Grundlagen dieser Arbeit. Zu Beginn wird der Begriff der Mobilität erläutert und in den Kontext der vorliegenden Arbeit gestellt. Besonders die mobilen Endgeräte und die forschungsrelevanten Entwicklungen im Umfeld mobiler IKT werden vorgestellt, um später deren Integration innerhalb eines Workflows einschätzen zu können. Das Themengebiet der Kontexte bzw. Kontextinformationen wird in Bezug auf einen mobilen Dienst (*Context-Aware Service* bzw. *Location-Based Service*) eingeführt. Definitionen und Begriffe des Geschäftsprozessmanagements werden erläutert und die Fokussierung auf Workflows wird begründet. WfMS zur Ausführung von Workflows werden mit allen zugehörigen Komponenten und Fähigkeiten vorgestellt, damit später eine Architektur eines WfMS (anforderungsbezogen) entwickelt werden kann. Petri-Netze und die BPMN 2.0 werden eingeführt (siehe Kapitel 2.6 und 2.7), um im weiteren Verlauf eine formale Definition einer Annotationsform von Ortsbezügen für Geschäftsprozessmodelle bereitzustellen (siehe Kapitel 5.1 und 5.2). In Kapitel 2.8 werden Endbenutzerklassifikationsmodelle vorgestellt, die als Basis für die Analyse von Endbenutzergruppen (siehe Anhang B) dienen. Eine Einführung in das Themengebiet der Softwarearchitekturentwicklung (siehe Kapitel 2.9) ist notwendig, damit eine Softwarearchitektur für das mWfMS entwickelt werden kann (siehe Kapitel 6).

In Kapitel 3 wird der Stand der Forschung im Bereich der mobilen WfMS zusammengefasst. Zu Beginn werden verschiedene Lebenszyklusmodelle von Workflows vorgestellt und die Auswirkungen auf die Softwareentwicklungsprozesse skizziert. Die Ergebnisse der Analyse existierender mobiler WfMS werden in verschiedene zeitliche Phasen untergliedert. Das Kapitel

wird mit der Definition mobiler WfMS im Kontext der vorliegenden Arbeit abgeschlossen.

Die ortsbezogene Annotationsform bzw. die Ortsbezüge zur Erweiterung von Geschäftsprozessmodellen werden in Kapitel 4 vorgestellt. In diesem Kapitel wird eine graphische und informationstechnische Unterstützung dieser Ortsbezüge zur Integration in Workflow-Modelle erörtert. Ein systemunabhängiges Ortsmodell wird entworfen, um die Ortsbezüge in mehrere Geschäftsprozessmodellierungssprachen integrieren zu können.

Eine Abbildung der graphischen Annotationsform in ausführbaren maschinenlesbaren Code ist notwendig, um eine möglichst große Produktivitätssteigerung bzw. eine automatisierte Ausführung zu erzielen. Die Ortsbezüge wurden innerhalb von Spracherweiterungen in zwei Geschäftsprozessmodellierungssprachen bzw. Workflow-Sprachen (Petri-Netze und BPMN 2.0) integriert. Beide Sprachen besitzen eine formale Beschreibung zur Gewährleistung einer eindeutigen Ausführungssemantik. Mehrere Beispiele zeigen die Modellierung von Ortsbezügen innerhalb von Geschäftsprozessmodellen auf (siehe Kapitel 5).

Eine Architektur für ein mobiles WfMS wird aufbauend auf der Referenzarchitektur der WfMC entwickelt und vorgestellt (siehe Kapitel 6). Die resultierende Architektur wird ausgehend von verschiedenen Betrachtungswinkeln modelliert. So lassen sich eine softwaretechnische Implementierung des ganzen Systems und die eingesetzte Entwicklungsmethode evaluieren, um entsprechende Verbesserungspotentiale im Entwicklungsprozess zu identifizieren.

In Kapitel 7 werden Anwendungsfälle und deren Realisierung mit mobilen WfMS beschrieben. Die in Kapitel 6 vorgestellte Architektur wird mit Hilfe eines mobilen Anwendungsfalls aus dem landwirtschaftlichen Bereich erprobt (siehe Kapitel 7.1). In Kapitel 7.2 wird die Automatisierung von Benutzeraufgaben mit mobilen Endgeräten untersucht. Verschiedene Anwendungsfälle im Umfeld von CAS/ LBS werden analysiert und innerhalb eines Datenmodells zusammengefasst, um darauf basierend eine mobile Applikation anbieten zu können. Das Datenmodell ist innerhalb einer Datenbank-

Applikation auf einem dezidierten Server implementiert worden. Mit Hilfe einer mobilen Applikation wurden diese LBS (bzw. CAS) softwaretechnisch unterstützt. In einer weiteren Untersuchung werden verschiedene mobile Technologien und Darstellungstechniken für sogenannte *Small Screens* vorgestellt, um Benutzeraufgaben auf Smartphones in Aufgabenlisten Nutzergerecht bereitzustellen (siehe Kapitel 7.3).

Mit der Zusammenfassung und dem Ausblick wird die Arbeit in Kapitel 8 beendet. Im Ausblick werden offen gebliebene Fragen und damit verbundene Aufgaben für künftige Forschungsarbeiten beschrieben.

2 Grundlagen

In diesem Kapitel werden die Grundlagen der vorliegenden Arbeit beschrieben. Beginnend mit der Definition des Begriffs „Mobilität“ (siehe Kapitel 2.1) werden verschiedene Sichtweisen dieses Themenkomplexes vorgestellt. Mobile Systeme (siehe Kapitel 2.1.1) und Mobile Technologien (siehe Kapitel 2.1.3) werden weiter konkretisiert. Eine Klassifikation mobiler Geräte und eine Beschreibung der geläufigsten Ortungsverfahren werden vorgestellt. Kontext bzw. Kontextinformationen werden in Kapitel 2.2 eingeführt. In Kapitel 2.3 werden Geschäftsprozesse definiert. Aufbauend darauf werden Workflows (siehe Kapitel 2.4) und Workflow-Managementsysteme (siehe Kapitel 2.5) vorgestellt. Eine kurze Einführung in Petri-Netze und die *Business Process Modeling and Notation* erfolgt in Kapitel 2.6 und 2.7. In Kapitel 2.8 werden Grundlagen zur Endbenutzerklassifikation eingeführt. Das Themengebiet der Softwarearchitekturentwicklung wird in Kapitel 2.9 vorgestellt.

2.1 Mobilität

Im Fokus dieser Arbeit wird Mobilität im Kontext von **mobiler IKT im Rahmen von Unternehmen und Organisationen** betrachtet. Eine Definition von Mobilität muss mehrere Aspekte berücksichtigen. Einerseits die Bewegungsfähigkeit und andererseits die Bewegung selbst. Die drei folgenden Mobilitätsformen, werden in vorliegender Arbeit (meist im Zusammenspiel) angewandt [Weile03]:

Die physische oder **räumliche Mobilität** beschreibt die Fähigkeit zum Ortswechsel. Im Zusammenhang dieser Arbeit ist die Tatsache wichtig, dass Mitarbeiter und Kunden eines Unternehmens (teilweise) physisch mobil sein müssen, um gewisse Aufgaben zu erledigen. Dabei kann die mobile Aufgabe des Mitarbeiters durch ein betriebliches Informationssystem (IS) unterstützt werden.

Die **elektronische Mobilität** wird maßgeblich durch die Datenübertragung charakterisiert. Die elektronische Mobilität umfasst viele Techniken im Umfeld des *Mobile Business* (z. B. WLAN, GSM). Die drahtlose (mobile) Datenübertragung wird meist mit Hilfe eines Mobiltelefons in einer gegebenen Netzinfrastruktur ermöglicht.

Die **virtuelle Mobilität** wird definiert als *„[…] eine Option des Menschen, sich mit Hilfe von Informations- und Kommunikationstechnik virtuell, d. h. der Möglichkeit nach, Mobilität zu erschließen, ohne hierfür selbst notwendigerweise (physisch) mobil d. h. beweglich zu sein"* [ZoKJ02]. Betriebliche Besprechungen finden z. B. virtuell über IKT als Telefonkonferenz statt. Heutzutage werden teilweise Besprechungen selbst in virtuellen Räume abgebildet (z. B. in *SecondLife*[1]).

Mobilität wird in einem bestimmten Kontext, also mit der Angabe eines zusätzlichen Merkmals, Zustandes oder einer bestimmten Aufgabe definiert.

2.1.1 Mobile Systeme

Ein **mobiles System** ist ein Informations- und Kommunikationstechnisches-System, das während einer physischen Raumüberwindung genutzt werden kann.

Um nun den Rahmen der mobilen IKT stärker einzugrenzen, werden verschiedene Klassifizierungen (innerhalb der elektronischen Mobilität) untersucht:

Eine Klassifizierung von [KrLj00] orientiert sich an den Endbenutzern selbst, indem zwischen *Traveling*, *Visiting* und *Wandering* unterschieden wird. *Traveling* bezeichnet eine physische Raumüberwindung allgemein. V*isiting* charakterisiert die Tätigkeiten bei einer (Dienst-)Reise. *Wandering* beschreibt ein „Umherstreifen", wie es in strukturierter Art und Weise bei betrieblichen Wartungs- und Reparaturarbeiten durchgeführt wird.

[1] http://secondlife.com/

Eine weiterführende Klassifizierung mobiler Systeme findet sich bei [Coop01, ChSL00, Roth05, KuXi16], welche zwischen Endbenutzermobilität, Endgerätemobilität und mobilen Anwendungen bzw. Diensten differenzieren:

Bei der **Endbenutzermobilität** kann der Nutzer unabhängig vom Ort IT-Systeme nutzen [Roth05] (Profile und aktuelle Zustände müssen vorgehalten werden). Im betrieblichen Informationssystem wird der Zustand jeder Aktion (zentral) abgebildet, um die Endbenutzermobilität zu unterstützen.

Unter dem Begriff der **Endgerätemobilität** wird die Eigenschaft der Ortsunabhängigkeit der mobilen Geräte, verbunden mit mobiler Konnektivität (den Mobilfunknetzen, z. B. GSM, UMTS) verstanden. [Roth05] definiert die Endgerätemobilität mit Hilfe der Eigenschaft, ein mobiles Endgerät ortsunabhängig nutzen zu können. Danach sollte das Endgerät über eine (vorübergehend unabhängige) autarke Stromversorgung (i. d. R. Akku) verfügen, um es eine gewisse Zeitspanne ohne Anschluss an eine Steckdose nutzen zu können. Die Endgerätemobilität setzt voraus, dass die gegebene Infrastruktur mit dem Endgerät harmoniert (z. B. das mobile Gerät macht einen GSM-Netzwechsel im Hintergrund; der Benutzer erfährt nichts davon).

Bei **mobilen Anwendungen und Dienste** wird der Zugriff als mobil bezeichnet, sofern er unabhängig vom Aufenthaltsort genutzt werden kann [Roth05, Coop01]. Voraussetzung dafür ist wieder, dass der Endbenutzer bzw. das Endgerät in eine Funknetzinfrastruktur (z. B. GSM) integriert ist.

2.1.2 Mobile IKT

Im Folgenden werden einige Definitionen im Kontext von mobiler IKT aufgeführt, die zeigen, wie weit der Begriff gefasst werden kann:

Das „*access anytime, anywhere*“-Paradigma wird von [POSB01] als eine Eigenschaft der Mobilität definiert. In der Praxis ist jedoch eine volle Funknetzabdeckung (jeder beliebige Ort) nicht möglich. Nach der Argumentation

von [POSB01] wird ein mobiles Endgerät zu einem Objekt, das unabhängig vom Ort benutzt werden kann. Der Endbenutzer ist mit dem mobilen Gerät „lokal“ (z. B. unter einer Telefonnummer) erreichbar, ohne jedoch „vor Ort“ (wie z. B. bei drahtgebundenen Anschlüssen) zu sein.

Kakihara und Sorensen [KaSo01, KaSo04] definieren drei Dimensionen von (menschlicher) Mobilität. Die räumliche, die zeitliche und die *kontextabhängige* Mobilität. Raum und Zeit stehen in direktem Bezug zueinander: Zu einem bestimmten Zeitpunkt befindet sich jeder Endbenutzer an einem bestimmten Ort. Die kontextabhängige Mobilität wird als die Fähigkeit definiert, auf bestimmte Gegebenheiten zu reagieren. Die kontextabhängige Mobilität ist abhängig von Ort und Zeit, jedoch auch von anderen Gegebenheiten (Kontexten, eine Definition des Begriffs wird in Kapitel 2.2 eingeführt).

In der Klassifizierung von [Anto05] wird Mobilität aus Sicht der Endbenutzerschnittstelle – also des User-Interfaces – betrachtet. In [Anto05] werden drei Arten von Mobilität identifiziert: die Mobilität der Infrastruktur, die Mobilität der Personen und die Mobilität der Informationen.

In betrieblicher Informations- und Kommunikationstechnik spielt Mobilität eine wesentliche Rolle [POSB01, KaSo01, KaSo04, Weile01, Weile04, Anto05, DLNW13, PZZZ13]. Abhängig vom Kontext des Autors (bzw. von seinem Untersuchungsgegenstand) erfolgt eine andere Klassifizierung der Mobilität. In der Logistik bzw. in der Verkehrs- und Transportmittelplanung existieren viele Arbeiten bezogen auf den physischen Ortswechsel, der durch Informationstechnik unterstützt wird. Eine Betrachtung der mobilen Nutzung aus Sicht des Endbenutzers von mobiler IKT beschreiben [Anto05, Coop01]. Nach [Roth05] muss mobile IKT von einem Endbenutzer genutzt werden und für diesen gebrauchstauglich sein, d. h. eine gute Bedienbarkeit besitzen. Erschwerend kommt bei mobiler IKT hinzu, dass einige Erkenntnisse die aus stationären IKT gewonnen wurden, bei der mobilen Benutzung nicht mehr gelten. Während ein Endbenutzer an einem stationären System die volle Aufmerksamkeit auf das System lenkt, ist dies in den unterschiedlichen mobilen betrieblichen Anwendungsfällen nicht gegeben. Andererseits haben mobile IKT gegenüber stationären Systemen erhebliche Vorteile, da

die entsprechenden Endgeräte durch den Benutzer mitgeführt werden können und dadurch häufig zur Verfügung stehen. Auch wenn einige Dienste nicht immer genutzt werden können (z. B. abhängig von der Verfügbarkeit des Internets). Weitere Einschränkungen (begrenzte Akkuleistung, begrenzte Rechenkapazität, usw.) sind jeweils abhängig vom mobilen Endgerät.

Aus Sicht des Software-Engineerings werden die Komponenten eines mobilen Systems beschrieben. Dabei wird auf die Eigenschaften mobiler Systeme eingegangen, die bei allen mobilen Systemen gemeinsam sind. Teilweise sind die Komponenten voneinander abhängig.

Tab. 2.1: Komponenten eines Mobilen Systems

Applikationen	Mobile Webseite, native App, hybride App
Betriebssystem	Symbian, iOS, Android, Maemo, WebOS, Windows Phone, Linux, …
Konnektivität	USB, WLAN, GSM, UMTS, Bluetooth, LTE, …
Hardware	Mp3-Player (mit/ohne Netzkonnektivität), Mobiltelefon, Laptop, Embedded-PC, Tablet, …

In einigen mobilen Systemen werden die Applikationen und Betriebssysteme gekoppelt bzw. vor dem Endbenutzer verborgen (z. B. bei Navigationsgeräten). Jedes mobile System besteht aus einer entsprechenden Hardwareumgebung, die sich je nach Sensorik und Funktionalität unterscheidet. Mobile Systeme besitzen eine Konnektivität, die mit verschiedenen mobilen Techniken (manchmal auch kombiniert) unterstützt wird (siehe Tab. 2.1). Als stationäre Systeme werden IKT-Systeme bezeichnet, die an einen bestimmten Ort (temporär) gebunden sind. Dies kann ein PC (Personal Computer), ein Faxgerät, ein Drucker oder sonst ein im Unternehmen eingesetztes informationsverarbeitendes Gerät sein. Im Vergleich zu mobilen Endgeräten ergeben sich einige Unterschiede. An dieser Stelle wird allerdings die Menge stationärer Geräte auf einen PC (Desktop Computer) eingeschränkt, um nicht zu viele Differenzierungen betrachten zu müssen.

Tab. 2.2: PC vs. mobiles Endgerät

Gerät	Stationärer PC	Mobiles Endgerät
Display	groß	klein
Ortsabhängig	ja	nein
Sensorik	nein	ja
Mensch-Maschine-Schnittstelle	Tastatur, Maus	Tastenfeld oder Touchscreen, Sprachbefehle, Gesten, …
Stromversorgung	Vom Stromnetz abhängig	Temporär autarker Betrieb von der Stromversorgung möglich

Die Unterschiede eines stationären PC gegenüber einem mobilen Endgerät sind in Tab. 2.2 aufgelistet. Ein mobiles Endgerät besitzt ein kleines Display (max. 4-5 Zoll, 9-11 cm), wohingegen PCs üblicherweise (ca. 30“, 76 cm) größere Displays einsetzen. Während PCs ortsabhängig sind, können mobile Varianten ortsunabhängig, beim Ortswechsel, benutzt werden. Mobile Geräte besitzen verschiedene Sensoren (z. B. Lagesensor, GPS, Mikrophon), während PCs allgemein über wenige Sensoren verfügen. Ein PC wird mittels Tastatur und Maus bedient, während ein mobiles Endgerät (früher oft mittels eines Eingabestiftes) heutzutage mit dem Tastenfeld, per Sprachbefehl oder mit einem Touchscreen bedient wird.

Im Vergleich der Interaktion eines Endbenutzers mit dem mobilen Gerät bzw. mit dem PC (siehe Tab. 2.2) lassen sich folgende Unterschiede charakterisieren: Die *Darstellung von Information* gestaltet sich bei mobilen Geräten auf Grund der geringeren Displaygröße unterschiedlich gegenüber stationären Systemen. Die Applikation muss beispielsweise so gestaltet werden, dass eine intelligente Benutzerführung eine ähnliche Übersicht über die Informationen erzielt, wie dies beispielsweise mit Hilfe eines größeren Bildschirms möglich ist. Die Benutzung eines *mobilen Systems* kann während des Ortswechsels durchgeführt werden.

Ein mobiles Gerät lässt sich innerhalb einer gewissen Zeitspanne autark von der Stromversorgung nutzen. Während stationäre Desktop-PCs jeweils immer eine Stromversorgung benötigen, können die mobilen Geräte (temporär) ortsunabhängig benutzt werden.

2.1.3 Mobile Technologien

Nachdem mobile Systeme vorgestellt worden sind, werden einige Begriffe aus der wissenschaftlichen Literatur im Umfeld mobiler Systeme voneinander abgegrenzt.

Wireless Computing wird oft synonym für den Begriff *Mobile Computing* verwendet [KayA72]. Mobile Computing beschreibt die Fähigkeit, sich ungebunden von einer physischen Schnittstelle im Raum bewegen zu können, während der Nutzung eines Endgerätes, ohne selbst einen Netzwechsel oder ähnliches vornehmen zu müssen [Roth05]. [DiHe95] definieren *Mobile Computing* als die Fähigkeit, das mobile Endgerät mit sich herumtragen zu können. Generell kann unter Verwendung von *Mobile Computing* einem mobilen Nutzer (Akteur, Identität) die Möglichkeit gegeben werden, mobile Information zu erstellen, darauf zuzugreifen, zu bearbeiten, zu speichern und darüber zu kommunizieren [Zimm99]. Da viele Eigenschaften von Geschäftsprozessen geteilt werden (siehe Kapitel 2.3), könnte bei einer Kombination beider Techniken die Flexibilität und die Effizienz bei der betrieblichen Arbeit erhöht werden.

Nomadic Computing wird wie folgt definiert: *„Nomadic Computing (NC) bezeichnet eine Nutzung von Informationstechnik (IT), die sich ergibt, wenn Nomaden einen bereitgestellten, umfassenden Satz an IT-Diensten (Anwendungs-, Kommunikations- und Transferdienste) in transparenter und integrierter Weise mittels mitgebrachter Systeme nutzen. Ein Nomade ist dabei eine Person, die an wechselnden Orten IT-Dienste zur Erfüllung ihrer Aufgaben und/oder zur Erreichung ihrer Ziele benötigt“* [Heil00]. *Nomadic Computing* soll einem mobilen Akteur ermöglichen, von überall auf seine Daten bzw. Anwendungen zugreifen zu können, wobei nicht an jedem Ort

bzw. zu jeder Zeit Netzkonnektivität vorhanden ist. Innerhalb der Applikation muss der Netzausfall berücksichtigt werden. In diesem Sinne hat auch Leonard Kleinrock *Nomadic Computing* definiert [Klei95].

Der Begriff des ***Pervasive Computing*** wurde von IBM geprägt und wird als die komplette Durchdringung des Alltags im (zukünftigen) Informationszeitalter beschrieben [IBM10]. Der Begriff der „Rechnerdurchdringung“ wird in diesem Zusammenhang oft synonym verwendet. Mobile Endgeräte des *Pervasive Computing* werden von IBM als *„smart devices“* umschrieben [IBM10].

Ubiquitous Computing kann analog zu *Pervasive Computing* verstanden werden, mit dem Unterschied, dass beim *Ubiquitous Computing* aus Nutzersicht unbegrenzte Ressourcen bereitstehen. *Pervasive Computing* ist die mobile Technik, die heutzutage durch aktuelle Endgeräte möglich geworden ist. Dagegen ist *Ubiquitous Computing* derzeit praktisch nicht umsetzbar. Mark Weiser beschrieb *Ubiquitous Computing* treffend mit dem viel zitierten Satz: *„Ubiquitous Computing is the method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user“* [Weis93].

Portable Computing beschreibt die Möglichkeit, mit einem Rechensystem einen Ortswechsel durchzuführen und jeweils vor Ort zu arbeiten. Dies entspricht der heutigen Arbeitsweise, die z. B. im Bereich der Unternehmensberatung gepflegt wird: Ein mobiler Arbeiter mit Kenntnissen in der Applikationssoftware bzgl. Konfiguration und Programmierung arbeitet vor Ort in einem fremden Unternehmen. Die mobilen Akteure verwenden z. B. ein Notebook für das *Portable Computing*.

Beim ***Wearable Computing*** werden Endgerät zu Kleidungstücken bzw. in diese integriert. Im Gegensatz zu *Mobile Computing* soll das Endgerät während der Benutzung am Endbenutzer selbst getragen bzw. befestigt werden (z. B. integriert in die Jacke oder Uhr). Dadurch soll (wie beim *Ubiquitous Computing*) auf die eigentliche Aufgabe fokussiert werden.

Die Aufgabe mobiler IKT liegt in der Kontextanalyse des Endbenutzers und deren Interpretation zur Unterstützung. Um eine Unterscheidung der mobilen Technologien zu gewährleisten, werden vier der insgesamt sechs vorgestellten Technologien nach [LyYo02] anhand der Dimensionen Mobilität und *Embeddedness* eingeordnet.

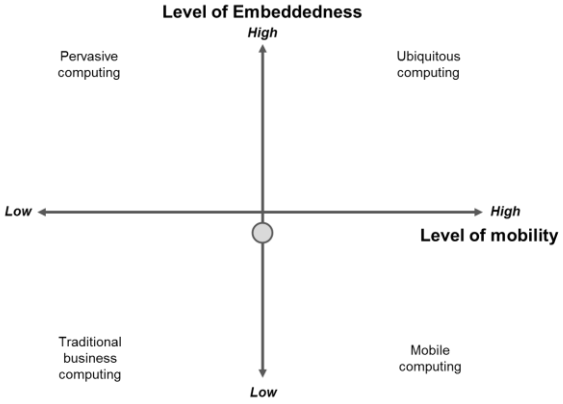


Abb. 2.1: Dimensionen des Ubiquitous Computing nach [LyYo02]

Mit *Level of Embeddedness* wird die Einbettung von (externen) Sensoren (z. B. RFID, NFC[2]) beschrieben [LyYo02]. Die Einbettung ermöglicht es dem mobilen System, mehr über den Kontext des Endbenutzers bzw. des Endgerätes in Erfahrung zu bringen (vgl. Kapitel 2.2) [KuXi16]. Dagegen wird mit dem Grad der Mobilität (*Level of Mobility*) ein physikalischer Ortswechsel beschrieben: Die Möglichkeit des Transports eines mobilen Endgeräts, auf dem Daten und Einstellungen des Endbenutzers bereitstehen.

Im weiteren Verlauf der Arbeit wird eine Dimension betrachtet, die sich an *Traditional Business Computing* (siehe Abb. 2.1) orientiert und über einen relativ hohen *Level of Embeddedness* sowie *Level of Mobility* verfügt. In

[2] Near Field Communication (NFC) ist ein Übertragungsstandard zum kontaktlosen Austausch von Daten über kurze Strecken.

Abb. 2.1 wurde bei der Einordnung der vorliegenden Arbeit ein mittlerer *Level of Embeddedness* und ein mittlerer *Level of Mobility* aus Sicht des Autors bestimmt: Das bedeutet, dass im Verlauf der vorliegenden Arbeit keine spezielle Hardware notwendig wird, sondern meist Standard-Geräte eingesetzt werden. Zudem ist eine hohe Rechenkapazität (wie z. B. beim *Ubiquitous Computing*) oder auch eine ganzheitliche Durchdringung des Alltags (vgl. *Pervasive Computing*) in Bezug auf den unterstützten Geschäftsprozess nur zum Teil erreicht worden.

2.1.4 Drahtlose Kommunikation

In vorliegender Arbeit sind die Kategorien der Mobilfunknetze (engl. *Wireless Wide Area Networks* – WWAN), der regionalen Netze (engl. *Wireless Metropolitan Network* – WMAN), der lokalen Netze (engl. *Wireless Local Area NetWork* – WLAN) und der Kurzstrecken-Funktechnik (engl. *Wireless Personal Area Network* – WPAN) relevant. In Abb. 2.2 wird die Klassifizierung mit einigen bekannten Standards dargestellt.

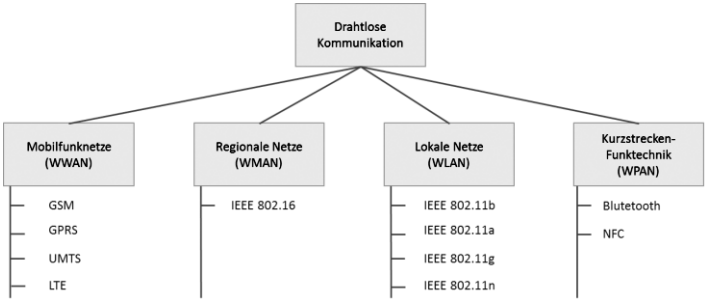


Abb. 2.2: Klassifizierung drahtloser Kommunikation (eigene Darstellung)

Die Klassifizierung aus Abb. 2.2 wird nachfolgend kurz erläutert [Roth05]:

Mobilfunknetze – sind durch den steigenden Bedarf an mobiler Sprachtelefonie entstanden. Mobilfunknetz sind zellular aufgebaut [Roth05, S. 45-48], wobei die Funkzellen untereinander durch ein drahtgebundenes Netzwerk

als Träger unterstützt werden. Diese Netze werden als Weitverkehrsfunknetze bezeichnet, da damit große Reichweiten abgedeckt werden.

Regionale Netze – decken einen kleineren regionalen Bereich ab (z. B. eine Stadt oder Region) im Gegensatz zu den Mobilfunknetzen. Es gibt zwei Arten von regionalen Netzen: Einerseits lassen sich diese Netze bereitstellen, indem mehrere Zugriffspunkte an verschiedenen Standorten, die im Verbund miteinander arbeiten, über WLAN zusammengeschaltet werden. Andererseits gibt es einen speziellen Standard (IEEE 802.16) für solche Netzverbünde.

Lokale Netze – gehören der IEEE 802.11-Familie an und werden als WLAN bezeichnet. Sie besitzen im Vergleich zu WPAN eine größere Sendeleistung und Reichweite. Lokale Netze haben innerhalb von Gebäuden eine geringere Reichweite als außerhalb (von ca. 200 m auf ca. 30 m).

Kurzstrecken-Funktechnik – wurden kreiert, um kurze Kabelverbindungen zu vermeiden. Die Reichweite liegt daher – je nach Standard – zwischen 0,2 und 50m. Es wird nur das unmittelbare „persönliche“ Umfeld (vgl. WPAN) mit angebunden.

In der vorliegenden Arbeit werden sowohl die Mobilfunktechniken GSM, GPRS und UMTS als auch WLANs eingesetzt. Eine Einführung in die Mobilfunktechniken geben z. B. [Roth05, S. 41-76, TuPo04, S. 36-48]. Grundlagen für WLANs können beispielsweise [Roth05, S. 79-102, Schi03, S. 248-285] entnommen werden.

Die drahtlose Kommunikation bzw. die Nutzung des mobilen Internets hat in den letzten Jahren ein enormes Wachstum erlebt [WiBP11]. Hohe Datenraten in der drahtlosen Kommunikation werden immer wichtiger, um Endbenutzer in Unternehmen und öffentlichen Netzwerken zu unterstützten. Die ersten Anwendungen von Mobilfunknetzen waren die Sprachtelefonie bzw. Dienstleistungen mit niedrigen Datenraten. In den letzten Jahren werden immer größere Bandbreiten geliefert, da sich die Anforderungen der Endbenutzer gewandelt haben. WLANs gewährleisten eine relativ hohe Bandbreite, stehen aber durch ihre hohe Absorptionsrate (je höher die Frequenz desto

höher die Absorption) nicht flächendeckend zur Verfügung. Mobilfunknetze werden ständig ausgebaut, um eine möglichst große Bandbreite (vgl. WLAN) auch außerhalb von Gebäuden zu gewährleisten. Die 3G-Netze (dritte Generation) haben sich in vielen Teilen der Welt etabliert, wobei derzeit der Ausbau der vierten Generation (4G) umgesetzt wird. Die Problematik von GSM- und 3G-Funksignalen ist, dass durch die Bauformen und Materialien von Gebäuden die Funksignale geschwächt – wenn nicht sogar blockiert – werden [Mone07]. Grundsätzlich gilt: Je höher die Frequenz, desto höher die Absorptionsrate. Dem möchte man mit der 4G-Technologie entgegenwirken, in dem höhere Signalqualitäten und niedrigere Frequenzen (siehe Abb. 2.3) eingesetzt werden.

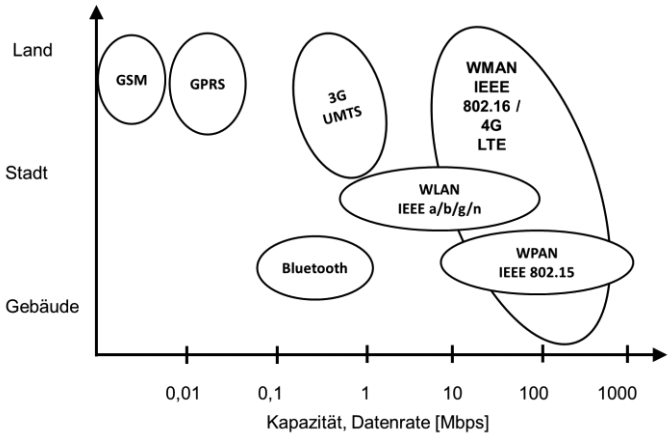


Abb. 2.3: Aktuelle drahtlose Kommunikationsformen (eigene Darstellung)

Der Zielkonflikt zwischen Netzabdeckung und Kapazität soll etwas näher betrachtet werden. Abb. 2.3 zeigt die Beziehung zwischen einigen Standards in Bezug auf die Abdeckung und die Kapazität. Die Abdeckung von Wireless Personal Area Network Zellen (WPAN) ist gering [Roth05, S. 70, vgl. Pikozelle], wohingegen bei der Datenübertragung ein Wert von zehn Mbps erreicht werden kann. GSM und 3G-Netze besitzen Zellen, die sich über mehrere Kilometer erstrecken und erreichen Datenraten unterhalb von zwei

Mbps. Neben der Kapazität muss auch die Netzabdeckung erhöht werden. Diese Konvergenz wird beispielsweise mit dem IEEE-Standard 802.16 (WiMAX) oder dem Mobilfunknetz der vierten Generation erzielt (siehe Abb. 2.3). Die Bandbreite der Frequenzen erhöht sich (WiMAX 2-11GHz), womit sich auch gleichzeitig die Netzabdeckung erhöht.

2.1.5 Klassifikation mobiler (End-)Geräte

Eine Klassifikation mobiler Endgeräte soll den Handlungsraum der vorliegenden Arbeit (vgl. mobile Technologien in Kapitel 2.1.3) weiter einschränken und dem Leser eine Vorstellung der untersuchten Endgeräte im Umfeld der vorliegenden Arbeit geben. Mobile Endgeräte unterscheiden sich nicht nur in ihrer Bedienung, Größe und Bauart, sondern auch in ihrer Funktionalität, der Leistungsfähigkeit, der Akkuleistung usw. Mobile Geräte können anhand ihrer *mobilen Eigenschaften* differenziert werden. Mobile Endgeräte haben gerade in den letzten Jahren eine große Dynamik erlebt: Es verschmelzen viele Gerätearten, z. B. sind in fast allen handelsüblichen Mobiletelefonen Kameras enthalten und müssten somit Kamera-Handy oder ähnlich genannt werden. Da Marketingabteilungen Geräte-Absatz-fördernde Bezeichnungen bevorzugen, wird beispielsweise die Kombination aus PDA und Mobiltelefon *Smartphone* genannt.

Eine Klassifikation nach dem Grad der Mobilität verschiedener mobiler Geräte wird in [GoMe03] vorgestellt (siehe Abb. 2.4).

Abb. 2.4: Grad der Mobilität nach [GoMe03]

Die elektronischen Endgeräte (siehe Abb. 2.4) werden in folgender Tab. 2.3 charakterisiert [GoMe03].

Tab. 2.3: Charakteristiken von mobilen Endgeräten nach [GoMe03]

Gerätetyp	**Formfaktor**	**Mobilitätsgrad**	**Interaktions-modus**	**Modularität**
Desktop	groß	fest	stationär	vollständig modulare Ein-/ Ausgänge
Laptop	mittel	transportabel	stationär	individuelle Einheit mit optionalen externen Mechanismen (audio)
Palmtop	klein	transportabel	stationär, mit wenigen Ausnahmen	individuelle Einheit mit optionalen externen Mechanismen (audio)
Handheld	klein bis mittel groß	mobil	mobile Interaktion ermöglichend	individuelle Einheit mit optionalen externen Ein-/ Ausgangsmechanismen
Wearable	klein	mobil	mobile Interaktion ermöglichend	vollständig modulare Ein-/ Ausgangs-mechanis-men

Die Charakteristiken in obiger Tabelle lassen der Klasse der mobilen Endgeräte Laptops, Palmtops, Handhelds und Wearables zuordnen. Während Laptops und Palmtops lediglich als transportabel bezeichnet werden, sind Handheld und Wearable auch während des Ortswechsels benutzbar. Eine

Klassifikation nach [ScDe08] klassifiziert mobile Endgeräte respektive Terminals (siehe Abb. 2.5).

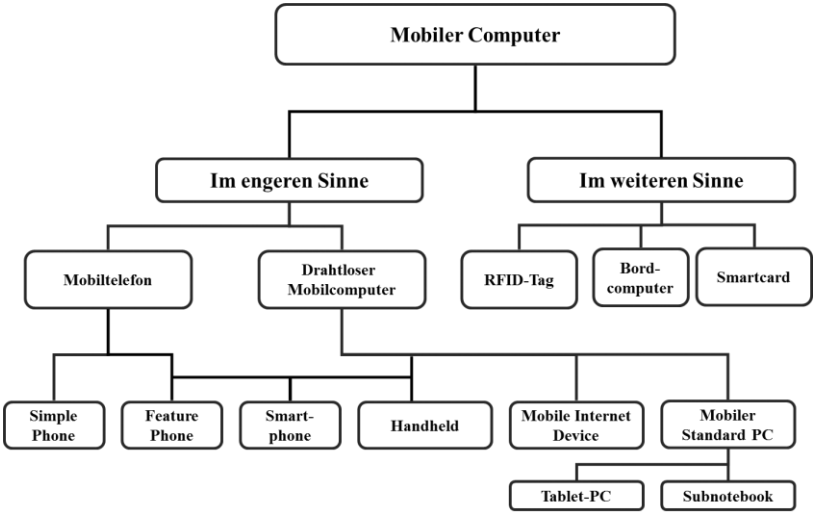


Abb. 2.5: Klassifikationsbaum mobiler Terminals nach [ScDe08]

Smartphones stellen die Verbindung zwischen den mobilen Computern und Mobiltelefonen dar (siehe Abb. 2.5). Die beiden Teilbereiche verschmelzen sozusagen miteinander, was anhand der Entwicklung der Tablets sichtbar wurde. Mobile Endgeräte haben eine rapide Entwicklung durchgemacht. Der Klassifikationsbaum [ScDe08] muss daher relativiert betrachtet werden. Die in den letzten Jahren entstandenen *Tablets* sind sicher keine mobilen Standard PCs mehr, sondern durch die weit verbreitete ARM-Architektur[3] eine Weiterentwicklung der Smartphones, was letztendlich auch an den ähnlichen/gleichen Betriebssystemen zu beobachten ist.

Im Folgenden werden die verschiedenen mobilen Endgerätetypen (vgl. Abb. 2.5) einzeln beschrieben:

[3] https://en.wikipedia.org/wiki/ARM_architecture

Mobile Standard-PCs

In der Klasse enthalten sind Laptops, Notebooks, Subnotebooks, Netbooks, Embedded-PCs. Diese Geräte können mit klassischen Betriebssystemen (Linux, Windows, MacOSX) betrieben werden. Ein Charakteristikum mobiler Standard-PC[4] sind z. B. der große Bildschirm (>10 Zoll), das Gewicht (> 1 kg) und die integrierte Peripherie (Kamera, Mikrophon), die in z. B. Desktop-PCs meist nicht enthalten sind.

Mobile Internet-Geräte (Tablets)

Tablets besitzen eine Bildschirmgröße zwischen 3,5 und 13 Zoll, verfügen über weniger Funktionalität und (normalerweise) kein mechanisches Laufwerk, sondern lediglich über ein großes ROM (Read Only Memory) in Kombination mit einem kleinen RAM (Read Access Memory). Meist sind spezielle Betriebssysteme im Einsatz (Echtzeit-Betriebssysteme), die sofort einsatzbereit sind. Vertreter dieser Klasse sind beispielsweise Navigationsgeräte oder E-Book-Reader. Heutzutage besitzen Tablets oft RAM (ohne ROM) und können, sofern es das Betriebssystem unterstützt, Applikationen (Apps) ausführen.

Handhelds

Eine synonyme Bezeichnung für Handhelds sind *Personal Digital Assistants* (PDA), die jeweils in einer Hand vom Endbenutzer gehalten werden können. Die Applikationen auf diesen Endgeräten wurden als *Personal Information Management* (PIM)-Applikationen bezeichnet. Die PIM-Apps sind z. B. Kalender, Adressbuch, Notizen oder Emailclients. Früher waren PIM meist mit viel ROM (im Vergleich zum RAM) ausgestattet, wohingegen modernere Varianten nur noch RAM beinhalten. Handhelds wurden meist mittels eines Eingabestiftes bedient. Manche Handhelds enthielten eine kleine Tastatur oder sogenannte Navigationsknöpfe. Handhelds besitzen nicht zwingend Konnektivität bzgl. der Mobilfunktechnologien (z. B. GSM, UMTS).

[4] https://de.wikipedia.org/wiki/Personal_Computer

Smartphones

„A [smartphone is a] cellular telephone with built-in applications and Internet access. Smartphones provide digital voice service as well as any combination of text messaging, e-mail, Web browsing, still camera, video camera, MP3 player, video player, television and organizer. In addition to their built-in functions, smartphones have become application delivery platforms, turning the once single-minded cellphone into a mobile computer“ [PCMa10]. Ein Smartphone ist im Grunde ein Handheld mit Mobilfunk und einem Touch-Screen, der mittels Finger (des Endbenutzers) bedient werden kann. Die kleine Displaygröße (<3-5 Zoll/<8-11cm) sorgt dafür, dass Smartphones auch als *Small-Screens* bezeichnet werden. Vorteil dieser Endgeräte ist, dass sie Verbindung zum Internet herstellen können.

Smartphones stehen nach dieser Beschreibung (vgl. [ScDe08]) zwischen tragbaren Computern und Mobiltelefonen. Sie besitzen eine vielfältige Ausstattung, sind mobil und haben Zugriff auf das Internet. Dies macht sie zum dauerhaften Begleiter, der aufgrund seiner Leistungsfähigkeit in der Lage ist, aufwendige Programme auszuführen. Moderne Apps setzen zwangsläufig auf neue, von traditionellen Mobiltelefonen her bisher unbekannte Funktionen. Die Smartphones werden in vorliegender Arbeit für experimentelle Versuche verwendet (siehe Kapitel 7.2).

Feature Phones

Feature Phones sind Mobiltelefone, die einen kleineren Bildschirm als Smartphones besitzen und keinen Touch-Screen enthalten. Sie verfügen über eine ähnliche Funktionalität wie Smartphones sind aber etwas eingeschränkter bzgl. der Eingabemöglichkeiten, da sie ein relativ kleines Tastenfeld besitzen. Wie bei Smartphones, kann auf diesen Endgeräten Drittanbieter-Software installiert werden, wobei sie jedoch meist proprietäre Betriebssysteme besitzen.

Mobiltelefone

Mobiltelefone sind mobilfunkfähige Endgeräte, über die meist nur Sprachkommunikation, SMS und wenige kleine Zusatzprogramme genutzt werden

können. Sie sind nicht in der Lage, IP-Verbindungen aufzubauen und haben im Vergleich zu Feature Phones einen noch kleineren Bildschirm. Sie besitzen immer proprietäre Betriebssysteme und haben wenig Speicher und Rechenleistung.

Mobile Endgeräte können nachfolgenden Eigenschaften klassifiziert werden [ScDe08]:

- Größe und Gewicht
- Ausgabemodi
- Leistung
- Benutzungsart
- Kommunikationsfähigkeit
- Art des Betriebssystems
- Erweiterbarkeit (per Software)

Eine Definition, die sich auf den Arbeitskontext des Nutzers konzentriert, definiert mobile Endgeräte als „*all diejenigen Endgeräte […], die für den mobilen Einsatz konzipiert sind*“ [TuPo04]. Beim Nutzen von mobilen Endgeräten ergibt sich immer eine 1:1 Zuordnung von Endgerät zum Endbenutzer (starke Kopplung zum Nutzer). Mobile Geräte entwickeln sich immer mehr in Richtung *Ubiquitous Computing* [TuPo04].

In der vorliegenden Arbeit werden Experimente mit Smartphones (siehe Kapitel 7.2) und Embedded-PCs (siehe Kapitel 7.1) durchgeführt. Es werden somit zwei Klassen von Endgeräten exemplarisch herausgegriffen, wobei viele Erkenntnisse der vorliegenden Arbeit auch auf andere Endgeräteklassen (siehe Abb. 2.5) angewandt werden können.

2.1.6 Ortungsverfahren

Ortung bzw. das Ortungsverfahren bezeichnet einen technischen Vorgang, bei dem die räumliche Position eines Zielobjektes ermittelt wird [Kuep05, S. 123]. Die Zielobjekte stellen mobile Geräte dar, deren Aufenthaltsort determiniert werden soll. Nach [BCKM04] lassen sich Ortungsverfahren in drei Kategorien aufteilen:

- **Indoor** – Ermittlung des Aufenthaltsortes eines Gerätes innerhalb eines Gebäudes.
- **Outdoor** – Ermittlung des Aufenthaltsortes eines Gerätes, das sich „im Freien“ befindet.
- **Hybrid** – Ein Gerät kann sowohl innerhalb, als auch außerhalb von Gebäuden geortet werden.

Eine weitere Kategorisierung erfolgt nach der Fähigkeit des mobilen Gerätes, sich selbst zu orten [Roth05]:

- **Selbstortungsverfahren** (engl. *positioning*) – Das mobile Gerät ist selbst in der Lage, seine Koordinaten zu ermitteln. Es wertet sogenannte *Beacons* (dt. Funkbake) aus, die von einem Sender ausgestrahlt werden. Die *Beacons* bestehen physisch aus Funk-, Infrarot- oder Ultraschallsignalen.
- **Fremdortungsverfahren** (engl. *tracking*) – Das mobile Gerät ist nicht selbst in der Lage, seine Koordinaten zu ermitteln. Die Koordinaten werden innerhalb des Netzwerkes ermittelt und an das mobile Endgerät gesendet. Das mobile Gerät ist dabei mit Hilfe eines *Tags* markiert und lässt sich eindeutig im Netzwerk identifizieren.

Die Ortungsverfahren selbst setzen zur Verifikation der Daten mehrere Methoden ein, um den Aufenthaltsort eines mobilen Gerätes präzise zu ermitteln. Die Basismethoden sind [Roth05]:

- **Cell of Origin** (COO) – Anhand der Kennung des Senders und dessen eingeschränkter Reichweite kann darüber der Aufenthaltsort des Empfängers ausgewertet werden. Diese Methode ist nur innerhalb von Netzstrukturen möglich, die mit sogenannten Zellstrukturen arbeiten.

- **Time of Arrival** (TOA) – Da sich elektromagnetische Signale stets mit der Geschwindigkeit von ≈300000 km/s bewegen, kann die Laufzeit eines Signals zwischen Sender und Empfänger gemessen werden. In GSM-Netzen wird das darauf basierende Verfahren *Observed Time Difference* angewandt.

- **Angle of Arrival** (AOA) – Im Falle einer auf bestimmte Kreis-Segmente beschränkten Sektor-Antenne kann ermittelt werden, aus welcher Richtung das Signal eingetroffen ist. Solche Antennen kommen in Mobilfunknetzen zum Einsatz. Die Information kann als zusätzliche Bereichseingrenzung z. B. innerhalb von Funkzellen genutzt werden.

- **Signalstärke** – Basierend auf der Messung der Signalstärke kann auf die Entfernung zum Sender geschlossen werden. Die Methode ist allerdings von der technischen Senderinfrastruktur, von Hindernissen (Häuser, Menschen), vom Wetter (Luftfeuchtigkeit) und vom mobilen Endgerät abhängig und führt zu ungenauen Ergebnissen.

- **Bildverarbeitung** – Auf Basis von aufgenommenen Bildern kann der aktuelle Aufenthaltsort identifiziert werden. Das Verfahren, sofern keine *Tags* eingesetzt werden können, benötigt jedoch eine große Menge an historischen Daten und viele aktuelle Bilder, um den Aufenthaltsort zu bestimmen.

Für die Ortsbestimmung existieren unterschiedliche technische Ansätze [Roth05, S. 267]:

1. **GPS**:
   Bei der Ortsbestimmung mittels Satellitensignal werden vom Endgerät aus mehrere Satelliten im Orbit identifiziert und es wird auf Basis der empfangenen Signale eine Position berechnet. Das bekannteste Satellitenortungsverfahren ist das **Global Positioning System** (GPS) [Mans09, S. 106]. Das GPS ist ein Satelliten-gestütztes System zur weltweiten Ortsbestimmung. Es ist in den 1970er Jahren vom Verteidigungsministerium der USA entwickelt worden. Seit 1995 ist das System vollständig in Betrieb. Bei GPS senden die Satelliten ständig ein kodiertes Signal (vgl. *Beacon*) mit ihrer aktuellen Position, dem aktuellen Zeitstempel und anderen Daten (z. B. Health-Status, Ephemeride, Almanach) aus. Der Empfänger ermittelt über das Signal seine eigene Position (Länge, Breite, Höhe) und die eigene Geschwindigkeit. Jedoch werden ständig 4-5 Satelliten in Sichtkontakt benötigt, die einen möglichst großen Winkelabstand (Elongation) untereinander aufweisen sollten. Über das TOA-Verfahren wird die Entfernung zum jeweiligen Satelliten (Radius) berechnet. Mit Hilfe der mindestens vier Radien wird die Schnittmenge gebildet, um den genauen Aufenthaltsort zu ermitteln. Die Koordinaten selbst lassen sich aus drei Satelliten berechnen. Auf Grund des fehlenden Zeitstempels wird ein vierter Satellit benötigt. Insgesamt werden 24 Satelliten im Orbit vorausgesetzt, um jeweils immer mindestens vier Satelliten pro Ort auf der Erde zur Verfügung zu stellen. Die Satelliten kreisen in einer Höhe von 20200 km. Da viele mobile Endgeräte nicht ständig Kontakt zu vier Satelliten aufrecht halten können, dauert die Ortsbestimmung initial relativ lange (bis zu 12 min). Daher wurde das A-GPS (Assisted-GPS) entwickelt. Es werden entsprechende Hilfsdaten (z. B. konkrete Angaben zu den verfügbaren Satelliten oder Korrekturwerte) über das Mobilfunknetz für den Empfänger zur Verfügung gestellt. Bei GPS wird ein bestimmtes Hardware-Modul, der GPS-Empfänger, benötigt. Außerdem funktioniert das Verfahren nur außerhalb von Gebäuden

[Roth05]. Ein weiteres derzeit viel diskutiertes Satellitenortungsverfahren ist das Galileo-System [Mans09, S. 241]. Es soll im Gegensatz zu GPS unter ziviler Führung und mit Verfälschungssicherung[5] betrieben werden.

2. **GSM-Ortung**:
   Die meisten mobilen Endgeräte (z. B. alle Handys und alle Smartphones) verfügen bereits über eine **Ortsbestimmung via GSM** (Global System for Mobile Communications). Das GSM-Netzwerk ist aus einzelnen Zellen aufgebaut. Bei der sogenannten Zellortung wird über die Mobilfunknetze eine Ortsbestimmung gewährleistet, indem die aktuelle Position auf Basis der Funkzelle ermittelt wird. Die entsprechenden GSM-Sender decken gewisse Zellbereiche innerhalb eines Gebietes ab. Das GSM-Netz speichert den aktuellen Aufenthaltsort eines Endgerätes jeweils im HLR (*Home Location Register*). Über das COO-Verfahren wird daraufhin die Position ermittelt. Die Positionen, die über GSM ermittelt werden sind in ländlichen Gebieten meist ungenau (ca. 30 km). Das TOA-Verfahren verbessert das Ergebnis (wird allerdings in der Praxis derzeit nicht angewandt) und resultiert bei einer Genauigkeit bei ca. 300 m, was für viele Anwendungen zu ungenau ist. Für eine genauere Ermittlung müsste das mobile Endgerät mit mehreren Basisstationen gleichzeitig verbunden sein, was durch das GSM-Protokoll nicht unterstützt wird [Roth05].

3. **WLAN-Ortung**:
   Die Ortsbestimmung per **WLAN** kann über viele unterschiedliche Verfahren erfolgen [AhIl10, S. 67-90, WaDi05]. Vom Grundprinzip her ist die Funktionsweise ähnlich dem GSM-Verfahren: WLAN-Access-Points werden innerhalb eines gewissen Umkreises bestimmt, der meist kleiner als 100 m ist. Die Positionen der verschiedenen WLAN-Access-Points werden mit Hilfe der MAC-Adresse

[5] GPS besitzt auch Mechanismen zur Integritätssicherung z. B. RAIM (Receiver Autonomous Integrity Monitoring) oder auch eine Verschlüsslung des Signals von den Satelliten ausgehend. Diese Verschlüsselung steht jedoch nur dem Militär zur Verfügung.

(eindeutige Kennung der AP-Hardware) innerhalb einer Lernphase in einer Datenbank abgespeichert. Die Datenbank wird von den Nutzern oder von einem entsprechenden Dienstanbieter gepflegt. Möchte ein mobiles Endgerät seine Position über die WLAN-Ortung ermitteln, wird das derzeit verfügbare Tupel an WLAN-Access-Points an den Dienstanbieter gesandt. Es wird ein Tupelvergleich mit den Einträgen in der Datenbank durchgeführt und ein wahrscheinlicher Bereich berechnet. Vorteil der WLAN-Ortung ist, dass sie innerhalb von Gebäuden bzw. innerstädtisch eingesetzt werden kann. Um das Verfahren zu verbessern könnte zusätzlich die Signalstärke gemessen werden, was allerdings an den unterschiedlichen Bauformen bzw. Hardwareausstattungen der WLAN-Access-Points oder den unterschiedlichen klimatischen Bedingungen scheitert. Die Genauigkeit des Verfahrens hängt von der Anzahl der registrierten WLAN-Access-Points ab. Durch die starke Verbreitung von WLAN-Access-Points kann innerhalb von Großstädten eine gute Genauigkeit erreicht werden [Zand09].

Weitere Ortungsverfahren existieren, um eine Indoor-Ortung (Ortung innerhalb von Gebäuden) zu ermöglichen: Infrarot-, Ultraschall und Funksignale dienen dabei als Hilfsmittel für die Ortsbestimmung [Roth05, S. 267].

2.1.7 Mobile Apps

Mobile Apps sind teilweise unabhängig vom mobilen Internet nutzbar und sind somit eine Ergänzung bestehender geschäftlicher Aktivitäten bzw. dienen meist der Kundenbindung und als Alleinstellungsmerkmal [GiKe03]. Viele Unternehmen (z. B. Webseitenbetreiber, Verlage, Spielehersteller, Standardsoftwarehersteller, etc) lassen neben nativen mobilen Apps auch HTML5-basierte mobile Webseiten programmieren, um möglichst viele mobile Endgeräte bedienen zu können [GSWM13]. Je nach Aufgabe bzw. Fokus der mobilen App kann ein dritter Ansatz gewählt werden, der die Vorteile beider vorherigen Ansätze vereint: eine hybride App. Im Folgenden werden die verschiedenen mobilen App-Arten kurz erörtert:

- **Die native mobile App** wird mit Hilfe von nativen Programmiersprachen wie *Objective-C* (für *iPhone*, *iPad*) oder Java (für *Android*) gebaut. Native Apps sind schnell, bieten eine gute *User Experience* und haben Zugriff auf alle Gerätefunktionen (Sensoren). Eine native App wird meist für eine spezifische Plattform eingesetzt (vgl. *iOS*, *Android*). Eine *Android*-App kann beispielsweise auf einem *iPhone* nicht ausgeführt werden [ChLe11].

- **Web Apps** sind Webseiten, die mit HTML5, CSS3 usw. gebaut werden. Der Zugriff auf die Apps erfolgt über den Webbrowser der mobilen Geräte. Web Apps können auf allen Plattformen und Geräten genutzt werden. Allerdings werden Web Apps nicht über die App-Stores vertrieben, so dass dieser wichtige Vertriebskanal für App-Entwickler nicht genutzt werden kann. Viele Gerätefunktionen können nicht genutzt werden, da nur rudimentäre Funktionen (Kontextinformationen) über Events vom Browser zur Verfügung gestellt werden [ChLe11].

- **Hybride Apps** sind mobile Apps die nativ entwickelt werden. Teile der Anwendung werden jedoch über Webtechnologien (nach-)geladen [DHTZ14]. Die nativ entwickelten Teile lassen die App wie eine native App aussehen. Die App-Stores können zum Vertrieb genutzt werden. Die Gerätefunktionen bzw. Kontextinformationen können über die jeweilige API genutzt werden. Eine hybride App ist eine App, die in Kombination mit HTML5 und nativen Programmiersprachen entwickelt wird. Die Entwickler können die Teile, die über Webtechnologien angebunden werden, in den verschiedenen Plattformen wiederverwenden [ChLe11].

Je nach Anwendungsfall, in dem die App eingesetzt werden soll, wird einer der drei Ansätze gewählt. In folgender Tab. 2.4 werden die Eigenschaften mobiler Apps voneinander abgegrenzt.

Tab. 2.4: Mobile Apps

Funktionen	**Native App**	**Hybride App**	**Web App**
Entwicklungssprache	Nativ	Nativ und/ oder Web	Web = HTML + Javascript
Code-Portierbarkeit	Nein	Ja	Ja
Geräte-Funktionen	Ja	Moderat	Nein
Zugriff auf APIs	Ja	Moderat	Nein
Aufwändige Graphik	Ja	Moderat	Moderat
User Interface	Ja	Moderat	Nein
Upgrades	Nur über App-Store	Über App-Store üblich	Einfach (Applikationsserver)
Aufwand Installation für Nutzer	Einfach (über App-Store)	Einfach (über App-Store)	Schwieriger (über Browser)

Eine handelsübliche App (auf einem Smartphone) erfasst heutzutage sensorische Kontexte (siehe Kapitel 2.2, vgl. Primärkontexte). Höhere Kontexte (z. B. historisches Wissen aus der Vergangenheit) sind der mobilen App meist vorenthalten. Eine Registrierung des vollständigen Endbenutzer-Kontextes ist möglich, jedoch nicht realistisch umsetzbar. Ziel des Entwicklers sollte es sein, die Benutzung einer App möglichst einfach zu gestalten. Viele Hersteller haben für mobile Plattformen sogenannte *Styleguides* veröffentlicht [Ostr02, Appl11], die dem Entwickler einer mobilen Applikation eine Anleitung geben. Sie helfen dem Entwickler, die Applikation möglichst Endbenutzergerecht zu implementieren. Die *Styleguides* fokussieren auf das Interaktionsdesign (s. u.) einer App und weniger auf das Interfacedesign. Das Interfacedesign wird für bestimmte Zielgruppen mobiler Endbenutzer definiert (siehe Anhang B) [Stapl07].

In [Appl11] werden zwei Arten von Apps unterschieden: Apps für einen Mehrwertdienst und Apps zur Unterhaltung. Die Apps der Mehrwertdienste

werden nochmals jeweils in zwei Kategorien untergliedert, nämlich in die Apps zur Informationsverarbeitung und Apps für einen bestimmten Nutzen:

- **Apps zur Informationsverarbeitung** – haben die Aufgabe, Informationen strukturiert darzustellen, um dem Endbenutzer diese leicht zugänglich zu machen. Der Endbenutzer soll eine bestimmte Aufgabe erfüllen können. Die App wird möglichst neutral (z. B. in der Farbgestaltung) gehalten, um nicht von der Aufgabe abzulenken. In [Appl11] werden als Beispiele Emailclient- und Kalender-App genannt.

- **Apps für einen bestimmten Nutzen** – sollen dem Endbenutzer ermöglichen, eine einfache Aufgabe zu erfüllen. Die Nutzer wollen in möglichst kurzer Zeit viele Informationen erhalten. Der Design-Aspekt ist für diese Kategorie von größerer Wichtigkeit. Graphische Metaphern eigenen sich besonders, um einen intuitiven Umgang mit der App zu ermöglichen. Die Wetter-App ist ein klassisches Beispiel für eine „App für einen bestimmten Nutzen“ [Appl11].

Die mobil-spezifischen Eigenschaften von Apps (auf Smartphones) beziehen sich meist auf das Display bzw. die Ein- und Ausgabefähigkeiten, die im Vergleich zu Desktop-Applikationen unterschiedlich sind [Appl11]:

- **Beschränkte Displaygröße:** Die Displays von Smartphones besitzen eine relativ hohe Auflösung. Allerdings sind die Darstellungsmöglichkeiten begrenzt. Entsprechend muss in den Apps eine Fokussierung auf ein klares und einfaches Interaktionsdesign bzw. auf die wesentlichen Elemente erfolgen. Die Anwender sollten nicht überfordert werden.

- **Keine Hilfestellungen:** Endbenutzer können Apps einfach aus den entsprechenden *Stores* herunterladen und starten. Die Endbenutzer möchten eine App intuitiv benutzen können. Es sollten möglichst Interaktionsdesigns verwendet werden, die der Endbenutzer bereits vom Betriebssystem oder anderen Apps kennengelernt hat.

- **Screen-Paradigma:** Ein Endbenutzer kann immer nur einen Screen gleichzeitig ansehen. Die Screens (einer App) können sequentiell durchlaufen werden, führen aber letztendlich dazu, dass der Endbenutzer sich App-Kontexte merken muss.

Das *Graphical User Interface* (GUI) ist die Schnittstelle, die zwischen dem Endbenutzer und dem (mobilen) Gerät fungiert [Wess02, S. 19]. Über diese Schnittstelle kommuniziert der Endbenutzer mit der App bzw. mit dem Endgerät. Die Kommunikation findet über graphische Symbole statt. Hinter der Symbolik werden die Funktionen innerhalb der App abstrahiert. Eine GUI bzw. die Mensch-Computer Interaktion (eng*l. Human Computer Interaction*, HCI) muss dabei so gestaltet werden, dass der Endbenutzer Ziele und Aufgaben in angemessenem Zeitaufwand bearbeiten kann. Gute Software zeichnet sich durch eine gute Funktionalität und durch ein gutes GUI-Design[6] aus, was zusätzlich aufeinander abgestimmt werden muss (siehe Kapitel 7.2.3).

Mit der graphischen Oberfläche wird die Außenwirkung gegenüber dem Nutzer erzielt [Dahm05, S. 16]. Das Design einer GUI wird in Interaktionsdesign, Interfacedesign und die Usability bzw. Software-Ergonomie verfeinert. Die drei Begriffe werden wie folgend voneinander abgegrenzt:

- Das Interaktionsdesign – bezeichnet den funktionalen Vorgang bei der Durchführung einer Aufgabe. Zentraler Bestandteil des Interaktionsdesigns ist die Anregung des Endbenutzers zum Agieren. Beispielsweise helfen in Webseiten Reiter bei der Auswahl von Kategorien.

- Das Interfacedesign – befasst sich mit dem eigentlichen Dialog (mit dem Endbenutzer). Das Interfacedesign bestimmt die graphische Gestaltung der Endbenutzerschnittstelle.

[6] Das GUI-Design kann mittels sogenannter Usability-Tests überprüft werden [Rask01].

- Usability/ Software-Ergonomie – bezeichnet die Gebrauchstauglichkeit einer Software. Gemessen wird die Usability daran, ob ein Endbenutzer mit einer Applikation effektiv, effizient und zufriedenstellend arbeiten kann und seine Ziele erreicht.

Beim Interfacedesign existieren zwei bekannte Leitsprüche [Hamm08, S. 28-30]:

- *form follows function*
- *form follows emotion*

Form follows function bedeutet, dass die Gestalt der GUI stets vom Interaktionsdesign des Programms abhängig ist. *Form follows emotion* bedeutet, dass die Gestalt der GUI von der Usability abhängig ist. Die GUI selbst ist somit vom entsprechenden Endbenutzertyp und von der eigentlichen Funktionalität der Applikation abhängig.

Für die Gestaltung von Interfaces existieren Normen, die anerkannte Regeln der Technik beinhalten. Die Anwendung bzw. Einhaltung dieser Normen führt zu Zeitersparnis und zu einer geringeren Fehleranzahl. Im Kontext der mobilen Endbenutzerschnittstelle existiert die ISO-Norm 9241 über „Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten“. Es werden in 17 Teilen verschiedene Anforderungen definiert. Normen werden jedoch auch kritisiert. [Dahm05, S. 132] bemängelt z. B. an Normen die Zerstörung des kreativen Prozesses.

Fazit:

Ein Interface bereitet gegenüber einem Benutzer eine bestimmte Aufgabe so auf, dass der Benutzer mit Hilfe dieses Werkzeugs die Aufgabe bearbeiten kann [Bons96]. Dem Benutzer wird ein eingeschränkter Raum an Handlungsalternativen zur Verfügung gestellt (vgl. Interfacedesign). Das Interface muss einfach und intuitiv gestaltet werden, damit eine möglichst große Benutzergruppe (siehe Anhang B) damit umgehen kann. Bei mobilen Apps ergibt sich ein Zusammenspiel von Nutzern, den (mobilen) Aufgaben und dem mobilen Endgerät (vgl. [Bons96]).

2.2 Kontext

Das menschliche Bewusstsein handelt stets innerhalb eines bestimmten Umfelds (Kontext) [Bier92]. Abhängig von diesem speziellen Kontext (z. B. innerhalb einer Familie) müssen die Bedürfnisse diesem Umfeld angepasst werden. Solche Kontexte bestehen auch innerhalb eines Unternehmens bzw. einer Organisation. Innerhalb einer Besprechung werden z. B. bestimmte Dinge und Zustände diskutiert, die als Kontext von allen Gesprächspartnern intuitiv oder bewusst aufgenommen werden. Kontext ist subjektiv, kann sich ständig ändern und kann wertvoll für weiterführende Aktionen oder Handlungen sein. Aktivitäten innerhalb eines Unternehmens stehen in einem bestimmten Kontext und werden davon abhängig ausgeführt. Kontext ist mehrdimensional und beinhaltet perzeptive Informationen (Informationen der menschlichen Wahrnehmung). Betriebliche (mehrdimensionale) Kontexte sind beispielsweise:

- Physikalische Kontexte (z. B. Ortsinformation)
- Umweltkontexte (z. B. Wetter)
- Soziale Kontexte (z. B. Team, Abteilung)
- Zeitkontexte (z. B. Arbeitszeiten)

Perzeptive Kontexte, also Kontexte die nicht informationstechnisch erfasst werden und die innerhalb eines Betriebs existieren sind beispielsweise:

- Erfahrungen aus der Vergangenheit
- Emotionale Zustände

Der Mensch verfügt über Kontextbewusstsein (engl. *context awareness)* [Dey01]. *Context awareness* wäre eine Bereicherung für jedes betriebliche Informationssystem, da sich die Informationstechnologie an den Menschen anpassen würde; nicht umgekehrt [Dey01].

Context aware computing [Weis91] hat die Zielsetzung, Computer so in den menschlichen Kontext (physische Umwelt) „einzuweben“, dass davon ausgehend Dienste oder Dienstleistungen angeboten, selektiert und konfiguriert

werden können. Computer sollen hinter den Kulissen des Alltags verschwinden und dem Menschen das Leben erleichtern [Weis91]. Diese Systeme müssen den Kontext über Sensorinformationen erfassen, den Kontext verstehen bzw. abstrahieren und auf Basis dieser Erkenntnisse weitere Aufgaben oder Aktionen auslösen. Die drei maßgeblichen Herausforderungen sind damit:

- Kontextaufnahme – erfolgt über Sensoren bzw. Sensorik.
- Kontextabstraktion – ist notwendig, um die Informationen weiter verarbeiten zu können.
- Kontextverarbeitung – stellt den abstrakten Kontext im System dar und beinhaltet dessen Management:
    - Algorithmen zur Verarbeitung
    - Speichern innerhalb einer Datenstruktur

Die Funktionalität wird innerhalb sogenannter *Context Aware Services* (CAW) unterstützt [Kuep05]. Das sind Dienste *„[…], die ihr Verhalten automatisch an einen oder verschiedene Parameter, die den Kontext des Zielobjektes ausdrücken, anpassen. Diese Parameter werden Kontextinformationen genannt“* [Kuep05]. Kontextinformationen sind wiederum heterogen und lassen sich in verschiedene Kategorien untergliedern: persönliche, technische, räumliche, soziale und physikalische Kontextinformationen. Es wird zwischen primären Kontextinformationen, die z. B. über die Sensorik eines mobilen Endgerätes erfasst werden, und zwischen sekundären Kontextinformationen, aggregierte oder zusammengefasste Kontextinformationen aus unterschiedlichen Quellen, unterschieden.

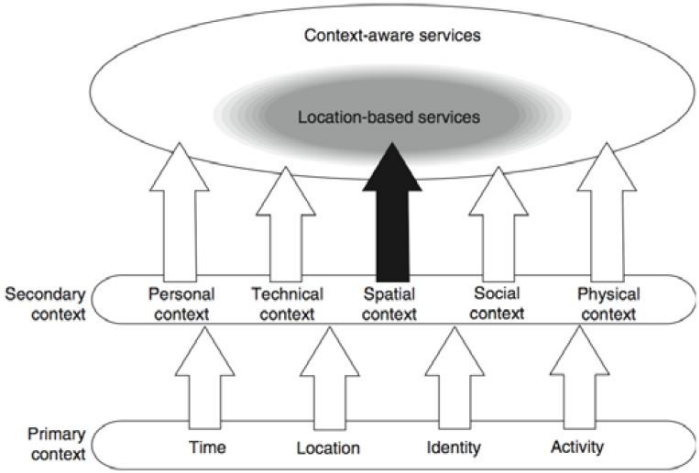


Abb. 2.6: Kontextinformationen nach [Kuep05]

Primäre Kontextinformationen sind Kontexte, die aus Sensordaten (im mobilen Endgerät) generiert werden (siehe Abb. 2.6): die Zeit, der Ort, die Identität und die Aktivität. Sekundäre Kontextinformationen sind darüber angeordnet (siehe Abb. 2.6) und umfassen die bereits erwähnten Kategorien.

Ein *Location-Based Service* (LBS) ist die Untermenge eines CAW, besitzt jedoch dieselben Primärkontexte (siehe Abb. 2.6). Der LBS besitzt eine stärkere Ausrichtung bzgl. räumlicher Kontexte (engl. *Spatial context*). LBS sind nach der Definition der *GSM Association*[7] *„[…] Dienste, die die Ortsinformationen eines Zielobjektes benutzen, um den erbrachten Dienst wertvoller zu machen, wobei das Zielobjekt eine Entität ist, die geortet werden muss und die nicht unbedingt der Benutzer des Dienstes ist“* (direkte Übersetzung aus [Kuep05]). Bei LBS werden gezielt Informationen nach bestimmten Kriterien gefiltert (vgl. Point-of-Interest, POI). Die Ortsinformation neben weiteren Kontextinformationen des mobilen Gerätes bzw. des Endbenutzers

[7] Die GSM Association ist ein weltweit agierendes Konsortium aus mehr als 600 GSM-Netzbetreibern und 130 Herstellern von Netzwerkinfrastruktur und Mobiltelefonen (http://www.gsmworld.com/).

werden aufbereitet: z. B. mit der Darstellung des aktuellen Aufenthaltsortes innerhalb einer Landkarte oder mit gezielter Abschaltung von Diensten innerhalb des Mobiltelefons an bestimmten Orten (ortsabhängig). Eine Trennung des mobilen Endgerätes vom eigentlichen Dienst (LBS) empfiehlt das *3rd Generation Partnership Project*[8] (3GPP), um eine weitere Differenzierung vornehmen zu können.

Die häufigsten LBS sind Daten- und Nachrichtenaustauschdienste, die jeweils auf WAP, GPRS, UMTS oder SMS basieren [Kuep05]. Heutzutage werden viele Cloud-Services aus dem *Cloud Computing*[9] innerhalb von LBS genutzt [RRLH14]. Klassische LBS waren z. B. Anrufweiterleitung, Roaming oder *Selective Routing*. Ein LBS braucht jeweils die Ortsinformation der Quell- oder der Zielentität. Die Ortsinformation wird mit Hilfe von Ortungsverfahren (siehe Kapitel 2.1.6) gewonnen. LBS existieren auch ohne Ortungsverfahren, allerdings muss die Ortsinformation durch den Endbenutzer in Form einer Adresse o. ä. (manuell) gepflegt werden, damit der LBS genutzt werden kann.

LBS werden in zwei verschiedene Kategorien, in reaktive und proaktive LBS untergliedert: Reaktive LBS besitzen einen sogenannten Pull-Mechanismus, bei der der Endbenutzer den Dienst aktiv (per Click) aktiviert (z. B. POI-Suche nach Tankstellen). Die Anfragen können an den LBS mehrmals hintereinander ausgeführt werden, so dass eine Endbenutzerinteraktion zwischen LBS und dem Nutzer gewährleistet wird. Proaktive LBS verwenden den sogenannten Push-Mechanismus. Der LBS wird dabei automatisch gestartet, sobald ein bestimmtes vorab definiertes Ereignis eintritt [ScBG99]. Die Endbenutzerinteraktion ist im Gegensatz zu reaktiven LBS asynchron, da eine Ortung ständig erfolgen muss. [ScVo04] haben dazu eine drei Schichten Architektur entworfen:

[8] Die 3GPP ist eine internationale Vereinigung verschiedener nationaler Standardisierungsbehörden zur einheitlichen Spezifikationen für GSM, UMTS und LTE zu erstellen.

[9] Cloud Computing abstrahiert IT-Infrastrukturen und passt dieses dynamisch an (skaliert). Die Verarbeitung der Daten wird nicht mehr innerhalb der Applikation vorgenommen, so dass oft von der „Daten-Verarbeitung innerhalb der Wolke“ gesprochen wird [RRLH14].

1. Die **Positionierungsschicht** bestimmt den aktuellen Ort des Zielobjektes mit Hilfe von Ortungsverfahren (hier: Hardware). Die Positionierungsschicht stellt diesen Ort in Relation mit einem GIS dar (Bsp.: zeigt Position in Landkarte).

2. Die **Applikationsschicht** stellt die Verbindung (Kommunikation) vom LBS zum Endbenutzer dar.

3. Die **Middleware-Schicht** koordiniert die beiden anderen Schichten, daher wird die Schicht in [ScVo04] als optional deklariert. Es ist zwischen Positionierungsschicht und Applikationsschicht eine direkte bidirektionale Kommunikation möglich. Außerdem kann von dieser dritten Schicht abstrahiert werden, so dass diese Schicht wiederum als Dienst im Sinne einer Middleware zur Verfügung gestellt wird [FlNa14].

Die rechtlichen Aspekte bei LBS (auf Seiten des Diensterbringers) sind komplex und müssen beachtet werden [RGKR07]: Es gibt soziale, ethische und weitere Dimensionen.

Bei der Kontextinformationsverarbeitung ist es erforderlich, bestimmte Ereignisse (z. B. Person befindet sich in Stadt Karlsruhe) zu definieren und diese innerhalb der Geschäftsprozesse auszuwerten [HHLL10, SSPM14], damit innerhalb der Prozesse auf solche Kontextinformationen bzw. Ereignisse reagiert werden kann. Letztendlich ist es abhängig vom Workflow bzw. Anwendungsfall selbst, welche Kontextinformationen relevant sind und welche nicht. Für die Entwicklung eines allgemeinen Lösungsansatzes werden nicht einzelne Workflows, sondern die Primärkontexte und ihre Integration innerhalb eines Workflow-Modells untersucht. Ein mobiles Gerät – beispielsweise ein Smartphone – kann mit einer Identität bzw. Person [PaHC06], einer bestimmten Zeit [Wein05], einer bestimmten Aktivität [Wein05] oder einem bestimmten Ort [DJJA04] innerhalb eines Informationssystems bzw. eines Workflow-Modells assoziiert werden.

2.2.1 Identität

Personen (Identitäten) werden innerhalb von Geschäftsprozessmodellen mit Hilfe von Benutzerrollen (siehe Kapitel 2.4) abstrahiert. Die Abbildung innerhalb eines Geschäftsprozessmodells wird durch das Prozess-Definitions-Metamodell der WfMC definiert (siehe Abb. 2.10). Oft werden die Rollen zusätzlich im Organisationsmodell innerhalb einer hierarchischen Baumstruktur eines Unternehmens hinterlegt bzw. damit verknüpft [SVOK11, S. 31]. Technisch werden dazu meist die LDAP[10]-Server/Verzeichnisdienste genutzt.

Innerhalb des Geschäftsprozessmanagements können Rollen (Identitäten) vielseitig genutzt werden. Über Rollen können beispielsweise Zugriffsberechtigungen gesteuert werden. Ortsabhängige Zugriffskontrolle ist in vorliegender Arbeit ein mit dem Kontext Ort verknüpftes Kontrollmodell, das über Benutzerrollen gesteuert wird (siehe Kapitel 7.2.2). Ein Experiment mit einem mobilen Anwendungsfall aus dem landwirtschaftlichen Bereich verdeutlicht den Praxisbezug (siehe Kapitel 7.1). Die experimentell nachgewiesene Funktionsweise des ortsabhängigen Zugriffskontroll-modells (siehe Kapitel 7.2) zeigt, dass der vorgestellte Ansatz innerhalb von betrieblichen IS angewandt werden kann.

Viele Workflows beinhalten Benutzeraufgaben, die über mobile Apps (teil-) automatisiert werden können. Über mobile Apps können beispielsweise verschiedene betriebliche Dienste (z. B. Kundenkommunikation, Shop) angeboten werden. Mobile Apps werden jedoch für eine bestimmte Endbenutzergruppe erstellt (siehe Anhang B).

2.2.2 Zeit

Der zeitliche Kontext innerhalb von Geschäftsprozessmodellen wird über den Begriff der Zeitstrukturen definiert. Hierfür wurden sogenannte Mengen

[10] Das Lightweight Directory Access Protocol (LDAP) ist ein Protokoll, das die Abfrage und die Modifikation von Informationen eines Verzeichnisdienstes (hier: Personenverzeichnis / Rollen) erlaubt. Spezifiziert: RFC 4510 / RFC 4511

von zeitlichen Bezugspunkten (Ereignisse) eingeführt [Ober96]. Die Ereignisse können mit definierten Funktionen und Relationen ergänzt werden. Darüber ist es möglich, kausale, relative und uhrbezogene Zeitstrukturen abzuleiten, die gerade bei der Anwendung innerhalb von betrieblichen Informationssystemen wichtig sind, um die Geschäftsprozesse zu steuern [Ramc74, BeDi91, Ober96, Wang98].

Die zeitlichen Kontexte bzw. die Zeitstrukturen sind weitestgehend erforscht, so dass im Rahmen der vorliegenden Arbeit keine weiteren Erkenntnisse gewonnen werden. Die Geschäftsprozessmodellierungssprache BPMN beinhaltet zeitlich motivierte Ereignisse innerhalb des Standards [FrRH10]. In Petri Netzen werden diese zeitlichen Dimensionen über Erweiterungen angeboten. Eine Erweiterung, die den formalen Rahmen der Netztheorie respektiert, d. h. die Schaltregeln nicht modifiziert, ist in [Ober96] veröffentlicht. Die Petri-Netze mit spezieller Interpretation, sogenannte Zeitsysteme, werden als temporale Semantik den Zeitstrukturen zugeordnet [Ober96, S.85ff].

2.2.3 Aktivität

Innerhalb eines Geschäftsprozessmodells werden Aktivitäten modelliert. Die Aktivitäten werden bei der Abbildung des Prozesses in einen Workflow (je nach Sprache) in Dienste abgebildet. Die Dienste, die innerhalb von Workflow-Modellen genutzt werden, sind vom Anwendungskontext der Prozesse abhängig. Dienste werden gewöhnlich Kontext-neutral gestalten, um sie in mehreren Anwendungskontexten (Prozessen) verwenden zu können. Bei Aktivitäten, die auf einem mobilen Endgerät ausgeführt werden, muss darauf geachtet werden, dass mobil-spezifische Aspekte berücksichtigt werden.

Bei dem Versuch in vorliegender Arbeit, eine App für mehrere Aktivitäten bereitzustellen (siehe Kapitel 7.2) wurde beispielsweise ein zentraler Dienst genutzt (Datenbank mit PostGIS[11]-Modul), um die ortsbasierten Aktivitäten auszuführen. Der Dienst wurde auf einem zentralen Server implementiert und als SOAP-basierter Webservice (siehe Kapitel 3.2.2) angeboten.

Häufig werden Aktivitäten für Nutzer (Benutzeraufgaben), die innerhalb eines WfMS genutzt werden, in Tasklisten verwaltet. Die Tasklisten eignen sich auch zur Darstellung auf einem mobilen Endgerät (siehe Kapitel 7.3).

Bei der Anbindung mobiler Aktivitäten (Dienste) muss stets ein Schnitt (mit Medienbruch) innerhalb des Workflows vollzogen werden (siehe Kapitel 3.2.3, *Mobile Process Landscaping*). Mit dieser Methode ist es möglich, die mobilen von den stationären Aktivitäten zu trennen und für die mobilen Aktivitäten in einem weiteren Schritt eine mobile Applikation (siehe Kapitel 2.1.7) zu implementieren.

2.2.4 Ort

Für die Berücksichtigung/Auswertung des aktuellen Ortes innerhalb von Workflows (resp. WfMS) werden einige Aspekte im Verlauf der Arbeit erörtert (siehe Kapitel 4):

- Ein theoretisches Konzept zur ortsabhängigen Unterstützung von Workflow-Modellen und -Instanzen wird in Kapitel 4.1 vorgestellt.
- Ein Ortsmodell auf Basis von OCLs[12] wird in Kapitel 4.2 erarbeitet
- Eine Annotationsform wird in Kapitel 4.3 informal eingeführt und erklärt.

[11] http://postgis.refractions.net/

[12] Die *Object Constraint Language* (OCL) ist eine formale Sprache, mit der u. a. UML-Modelle um spezifische Bedingungen in textueller Notation ergänzt werden kann. Weitere Hinweise zu OCL sind in [Balz09, S. 377] oder [Oest09, S. 147] zu finden. Eine Dokumentation bietet [WaKl03] oder die Spezifikation [ObMG10].

- Verschiedene Implikationen, die sich auf Basis der Notationsform innerhalb der Workflow-Modelle ergeben, werden in Kapitel 4.3.2, 4.3.3, 4.3.4 und 4.3.5 vorgestellt.

- Die Notationsform wird formal definiert (siehe Kapitel 5.1 und 5.2).

Ein praktischer Anwendungsfall der Notation von Ortsbezügen, der innerhalb des Verbundprojektes R2B entwickelt wurde, wird in Kapitel 7.1 vorgestellt.

2.3 Geschäftsprozesse

In einem Unternehmen entstehen Dienstleistungen bzw. Produkte durch das Zusammenspiel von Menschen, Maschinen, sonstigen Ressourcen (z. B. Bauteile, Verbrauchsstoffe) und Softwaresystemen. Bis zu einem fertigen Produkt sind Aktivitäten oder Aufgaben nach einer festgelegten Reihenfolge zu bearbeiten. Dabei ist die festgelegte Reihenfolge von verschiedenen Faktoren, wie z. B. Rohstoffe, Dokumente oder Betriebsmittel, abhängig (siehe Kapitel 6.1). Je nach Geschäftsprozess bzw. dessen Komplexität können Aktivitäten von einem oder mehreren Akteuren (menschlich und maschinell) in sequentieller, paralleler, iterativer Form oder bedingungsabhängig ausgeführt werden. Der abstrakte Begriff des Prozesses wird in DIN 66201 eingeführt. [Dave93, S.5] definiert den Begriff des Geschäftsprozesses als "[…] *structured, measured set of activities designed to produce a specific output for a particular customer to market."* [HaCh93] verstehen unter einem Geschäftsprozess: "*A collection of activities that takes one or more kinds of input and creates an output that is of value to the customer*". Die Definition eines Geschäftsprozesses der WfMC lautet [WfMC99, S. 10]: *„A set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships."* Der synonyme Begriff des betrieblichen Ablaufs ist von [Ober96, S.14], als "*eine Menge von manuellen, teil-automatisierten oder automatisierten Aktivitäten, die in*

einem Betrieb nach bestimmten Regeln auf ein bestimmtes Ziel hinausgeführt werden“ definiert worden.

Geschäftsprozesse lassen sich nach unterschiedlichen Arten (Typen) klassifizieren, die nach [HaSt04, S.25] folgender Maßen vorgeschlagen wird:

- Grad der Strukturiertheit
- Art und Häufigkeit des Auftretens
- Interne und externe Vorgänge

Der Grad der Strukturiertheit kann weiter unterteilt werden in vollständig strukturierte, teil- bzw. semi-strukturierte und unstrukturierte Geschäftsprozesse:

- **Strukturierte Prozesse** sind in ihrem Ablauf präzise vorbestimmt und werden entsprechend der ex ante definierten Regeln ausgeführt. Sie sind leicht wiederholbar und durch den definierten Ablauf gut automatisierbar.

- **Teil- bzw. semi-strukturierte Prozesse** beinhalten vorbestimmte Abläufe und Möglichkeiten der freien Entscheidung, an den beispielsweise eine menschliche Entscheidung den nächstenTeilablauf bestimmt. Solche Prozesse lassen sich automatisieren, indem z. eine Nutzereingabe in einer GUI als Ergebnis einer Regel verarbeitet wird.

- **Unstrukturierte Prozesse** folgen keinen festen Regeln. Sie enthalten viel „Freiraum“ für kreative Elemente. Unstrukturierte Prozesse sind jedoch nur selten wiederholbar bzw. automatisierbar.

Ein Geschäftsprozess kann von einem (einmalig) durchzuführenden Projekt abgegrenzt werden, da ein Geschäftsziel vorliegt, das es erlaubt oder erfordert, den Prozess zu wiederholen. Die Wiederholbarkeit mit den oben genannten formalen Definitionen (Ablauf, Schritte, Aktivitäten, …) wird in der Prozessdefinition festgelegt. Die Aktivitäten können weiter spezifiziert wer-

den, indem zwischen *automatisierter Aktivität* und *manueller Aktivität* unterschieden wird. Automatisierte Aktivitäten werden durch IT unterstützt ausgeführt, wohingegen manuelle Aktivitäten menschliche Interaktion benötigen und wenig formal spezifiziert werden (z. B. ein Beratungsgespräch).

Mittelpunkt des Geschäftsprozess-Managements sind die Geschäftsprozesse und deren Darstellung in Form von Geschäftsprozessmodellen. Die Begriffe Modell, Prozess und Prozessmodell spielen in vorliegender Arbeit eine zentrale Rolle und werden daher genauer betrachtet:

Herbert Stachowiak definiert ein Modell als ein beschränktes Abbild der Wirklichkeit mit drei Merkmalen [Stach73]:

- **Abbildung**: Ein Modell ist ein Abbild von einem (oder mehrerer) natürlichen oder künstlichem Original. Ein künstliches Original ist wiederrum ein Modell.

- **Verkürzung**: Ein Modell enthält nicht alle Attribute des Originals, sondern nur die Attribute des Originals, die für den Modellnutzer relevant erscheinen.

- **Pragmatismus**: Ein Modell ist dem Original nicht automatisch zugeordnet. Die Zuordnung wird durch die entsprechende Fokussierung (für Wen?, Warum?, Wozu?) relativiert. Es ist nur innerhalb einer Zeitspanne bzw. für einen bestimmten Zweck relevant. Ein Modell ist eine Interpretation bzw. eine Sicht auf das Original.

Eine weitere Definition stellt [Gier98, S. 12] vor:

„*Unter einem Modell wird* […] *die Abbildung eines realen Systems oder Systemausschnitts verstanden, welche besonders die in dem gegebenen Zusammenhang als wichtig erachteten Aspekte unter Vernachlässigung anderer, als weniger wichtig angesehener Gesichtspunkte darstellt.*“

Modelle beschreiben demnach in einem gewissen Zusammenhang (Kontext) die Realität unter Vernachlässigung aller irrelevanten Informationen [SRWN12]. Diese Eigenschaft lässt Modelle zum Hilfsmittel werden, um

komplexe Sachverhalte in einer überschaubaren Form (als Modell) darzustellen. Auf Basis eines entworfenen Modells können in einem weiteren Schritt Analysen erstellt oder Simulationen durchgeführt werden. Ein Modell hilft, die vielschichtige Realität beherrschbar zu machen, indem in einem gewissen Kontext (im Modell) wesentliche Aspekte eines Sachverhalts veranschaulicht werden.

Ein Prozess ist aus einer Menge von logischen zusammenhängenden Teilschritten aufgebaut und verfolgt ein oder mehrere Ziele.

Die komplexen Abläufe eines realen Prozesses werden durch die Modellierung auf Nutzerrelevante Sichten reduziert, da das entstehende Modell nur die wesentlichen Informationen wiedergibt.

„Ein Prozessmodell beschreibt die Struktur eines realen Prozesses. Es beschreibt alle möglichen Pfade entlang des Prozesses und bestimmt die Regeln für die Wahl der Pfade. Weiterhin bestimmt das Prozessmodell alle Aktivitäten, die ausgeführt werden müssen“ [RiSt04, S. 30].

Ein Geschäftsprozessmodell sollte für menschliche Nutzer graphisch visualisiert werden, damit das Modell bearbeitet und verstanden werden kann. Das Geschäftsprozessmodell bildet Sachverhalte (betriebliche Abläufe) aus der Domäne des Geschäftsprozess-Managements ab. Geschäftsprozessmodelle beinhalten diverse Ausführungsalternativen (Pfade) und berücksichtigen unter Umständen auch selten auftretende Ausnahmesituationen, die eine Relevanz für den abzubildenden Geschäftsprozess (den Ablauf) haben. Ein Geschäftsprozessmodell ist der Bauplan für alle möglichen Ausführungen eines Geschäftsprozesstyps. Die tatsächliche Durchführung eines Geschäftsprozesses wird als Instanz bezeichnet. (Geschäfts-)Prozessinstanzen werden aus dem Prozessmodell abgeleitet bzw. folgen einem im Modell enthaltenen Pfad. Der Pfad wird durch die Anwendung des Modells bzw. der darin definierten Entscheidungsregeln bestimmt. In Abb. 2.7 ist ein vereinfachtes Modell skizziert, um diesen Sachverhalt graphisch zu beschreiben.

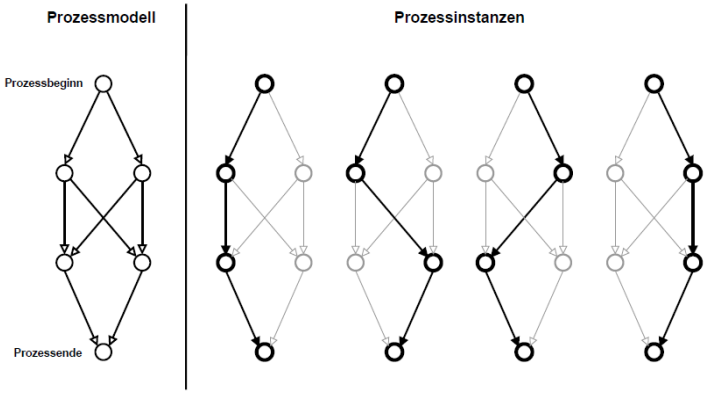


Abb. 2.7: Prozessmodell und Prozessinstanzen

In Abb. 2.7 (links) ist ein Prozessmodell mit sechs Elementen aufgebaut. Die Kreise entsprechen einzelnen Aktivitäten. Rechts sind jeweils mögliche Prozessinstanzen mittels tiefschwarzer Kanten als Pfade dargestellt. Pfade sind die Teile eines Prozesses, die während der Prozessausführung durchlaufen werden. An den Kreisen (den Elementen) wird eine Entscheidung gefordert, die zur Auswahl genau einer verfügbaren Alternative führt. Die Entscheidung entspricht einer exklusiven ODER-Verzweigung (XOR). Vier alternative Pfade ergeben sich, um das Prozessmodell zu durchlaufen. Alle zulässigen Instanzen des Prozessmodells werden dargestellt (siehe Abb. 2.7). Geschäftsprozesse besitzen in der Praxis einen komplexen Aufbau. Zyklen sind oft Teil der Struktur und es ist im Allgemeinen unmöglich, alle Verlaufsformen ex ante zu bestimmen. [AHKB03, AaHD05, RHAM06] beschreiben eine Vielzahl an unterschiedlichen Mustern, die für eine Prozessablauf-Abbildung zur Anwendung kommen.

Die Ausführung einer Prozessinstanz nimmt eine gewisse Zeit in Anspruch. In dieser Zeitspanne werden die einzelnen Elemente durchlaufen. Der Kontrollfluss definiert, in welcher Abfolge das Prozessmodell durchlaufen wird. Ein entsprechendes Element wird durch den Kontrollfluss aktiviert, wenn es erreicht wird. Da nicht alle Elemente eines Geschäftsprozessmodells (Berechnungs-)Zeit in Anspruch nehmen, kann von einem Zeitpunkt

anstatt eines Zeitintervalls ausgegangen werden ([ObMG10a], vgl. *Events*, BPMN 2.0). Beim Ablauf eines Geschäftsprozesses ist nicht gewährleistet, dass die Prozessinstanz stets sequentiell verläuft (siehe Abb. 2.7). Existieren nebenläufige Ausführungen, z. B. durch ein logisches UND- bzw. ODER-Verzweigung (kein XOR) ausgelöst, sind mehrere Kontrollflüsse zu einem Zeitpunkt vorhanden (siehe Abb. 2.8).

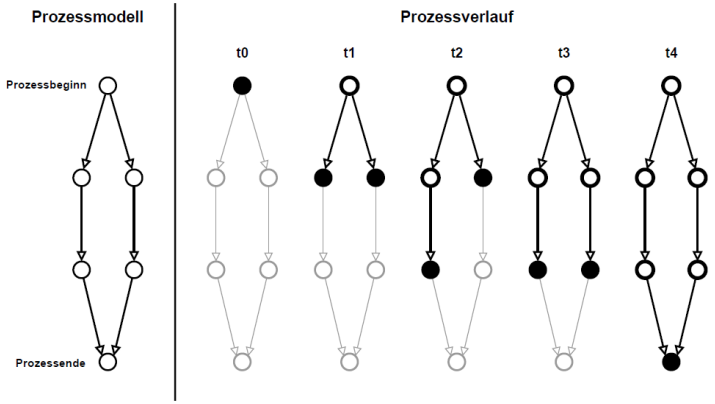


Abb. 2.8: Kontrollfluss

Ein einfaches Geschäftsprozessmodell (links) skizziert eine UND-Verzweigung und eine UND-Zusammenführung. Das Geschäftsprozessmodell besitzt nur einen möglichen Prozesspfad. Jeder einzelne Prozessschritt ist dargestellt (siehe Abb. 2.8). Die aktuelle Position bzw. das aktive Element des Kontrollflusses ist durch die schwarze Markierung am Knoten gekennzeichnet. Das Durchlaufen der Übergänge (der Pfeile) wurde in obiger Darstellung nicht abgebildet. Nach dem Start wird der Kontrollfluss in zwei unterschiedliche nebenläufige Segmente unterteilt. Dieses Teilen wird als nebenläufige Ausführung bezeichnet. Die beiden nebenläufigen Segmente sind unabhängig voneinander. In obiger Darstellung wurde eine zufällige Reihenfolge gewählt. Nach der vollständigen Ausführung beider sequentieller Pfade vereinigen sich die beiden Segmente wieder zu einem Kontrollfluss.

Geschäftsprozessmodelle werden mit Modellierungssprachen graphisch dargestellt. Die textuelle Form eignet sich für die rechnergestützte Verarbeitung: z. B. zur Persistenz oder zum Austausch von Geschäftsprozessmodellen (siehe Kapitel 2.4). Als textuelle Modellierungssprachen sind die *XML Process Definition Language* (XPDL) [WfMC08], die *Business Process Execution Language* (BPEL) [ACDG03, OASI07], Petri-Netze [Reis10a] oder die Business Process Model & Notation (BPMN ab Version 2.0) [ObMG10a] genannt, die mit verschiedenen Workflow-Engines ausgeführt werden. Eine graphische Darstellung eines Prozessmodells erleichtert dem menschlichen Betrachter die Interpretation eines Modells. Die graphische Darstellung eignet sich zur maschinen-orientierten Ausführung (also einer Automatisierung) weniger, sofern die Sprache keine Abbildung in eine explizite Text-orientierte Form besitzt. Vertreter graphischer Geschäftsprozess-modellierungssprachen sind Petri-Netze [Reis10a], die *Business Process Modeling Notation* (BPMN 2.0[13]) [FrRH10], Aktivitätsdiagramme [Kech09, S. 213] zugehörig zur *Unified Modeling Language* (UML[14]), ereignisgesteuerte Prozessketten (EPK) [KeNS92, ScTA05], sowie informale Flussdiagramme (siehe Abb. 2.7 und Abb. 2.8). Petri-Netze und die BPMN 2.0 werden in Kapitel 2.6 und 2.7 eingeführt.

2.4 Workflows

Geschäftsprozesse bzw. Teile eines Geschäftsprozesses werden anhand von unterschiedlichen Kriterien klassifiziert [RiSt04, S. 25]. Klassifizierungsmerkmale sind z. B. Art (Form) oder die Häufigkeit des Auftretens, interne und externe Vorgänge, sowie die Strukturiertheit des Ablaufs.

Unstrukturierte Geschäftsprozesse erfordern zur Ausführung Intelligenz und Problemlösungsfähigkeit eines (menschlichen) Nutzers. Vorgänge, die weder feste Strukturen besitzen noch ad-hoc ausgeführt werden, sind - aufgrund

[13] Ab der Version 2.0 besitzt die BPMN eine formale Beschreibung der Elemente (*formal execution semantics*), so dass die Sprache in einer Engine ausgeführt werden kann.

[14] Die Unified Modeling Language (UML) ist eine graphische Modellierungssprache zum Objektorientierten Softwareentwurf [Rump04].

der Flexibilität - menschlich zu unterstützen. Menschliche Endbenutzer bzw. Bearbeiter werden in Form von Rollen innerhalb von Geschäftsprozessmodellen abgebildet. Personen werden so nicht unmittelbar in Modelle eingebunden, da durch die Rolle eine abstrakte Form der Darstellung bzw. Dokumentation gefunden wurde. Steht eine Person etwa wegen Urlaub oder Krankheit vorübergehend nicht zur Verfügung, muss das betroffene Geschäftsprozessmodell nicht angepasst werden. Eine Rolle kann, so wie im vorherigen Beispiel dargestellt, von mehreren Endbenutzern wahrgenommen werden. In der Praxis sind oft viele Mitarbeiter für eine Rolle zuständig. In einem Call-Center kann z. B. die Rolle des Telefonagenten durch einen der gerade verfügbaren (bzw. freien) Mitarbeiter übernommen werden.

Einzelne Schritte eines Geschäftsprozesses oder ganze Teile (Segmente) werden automatisiert ausgeführt bzw. durch den Einsatz von Informationstechnik unterstützt. Es handelt sich dabei um strukturierte, wiederkehrende Geschäftsprozesse. Dieser rechnergestützte zusammenhängende Teil eines Geschäftsprozesses wird als Workflow bezeichnet [Holl95]. Die *Workflow Management Coalition*[15] definiert den Begriff wie folgt:

„*The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules*“ [WfMC99, S. 8].

Der Begriff des Workflows wird innerhalb des *Workflow-Managements* (WfM) und des *Business Process Management* (BPM) verwendet [GeHS95, S. 122]. In [Holl95, S. 6] wird der Begriff des Workflows wie folgt definiert:

„*The computerized facilitation or automation of a business process, in whole or part.“*

[15] Die Workflow Management Coalition (WfMC) ist eine Verbundorganisation im Bereich des Workflow Managements bestehend aus mehr als 300 Herstellern, Nutzern, Beratern und Wissenschaftlern (http://www.wfmc.org/).

Workflows werden nach [HaSt, S. 28 f] in zwei Kategorien untergliedert: Strukturiertheit und Fokus des Workflows. [AAAM97, S. 3, Aals98, S. 8] unterscheiden zwischen vier Typen von Workflows:

- *Administrative Workflows* sind stark strukturiert. Sie sind leicht wiederholbar und besitzen einen geringen Anteil an der Wertschöpfung für das Unternehmen.

- *Ad-hoc Workflows* sind selten auftretende Workflows, die wenig strukturiert sind.

- *Collaborative Workflows* zeichnen sich durch eine starke Rolleneinbindung innerhalb der Workflows aus. Solch ein Workflow ist nicht strukturiert, was z. B. [Aals98, S. 8] zu dem Urteil bringt, dass es sich hierbei gar nicht um Workflows handelt.

- *Production Workflows* sind eine Implementierung eines „kritischen" Geschäftsprozesses [AAAM97, S. 5]. Es sind wichtige Prozesse, die wesentlich an der Wertschöpfung des Unternehmens teilhaben und sich durch eine starke Strukturierung auszeichnen (z. B. Auftragsprozesse).

Aus den Prozessvariablen *Komplexität*, *Grad der Veränderlichkeit*, *Detaillierungsgrad*, *Grad der Arbeitsteilung* und *Interprozessverflechtung* werden die drei Prozesstypen *Routineprozess*, *Regelprozess* und *einmaliger Prozess* abgeleitet [PiRo95].

Van der Aalst [Aals98] definiert einen Workflow mittels unterschiedlicher „Dimensionen" allgemein (siehe Abb. 2.9).

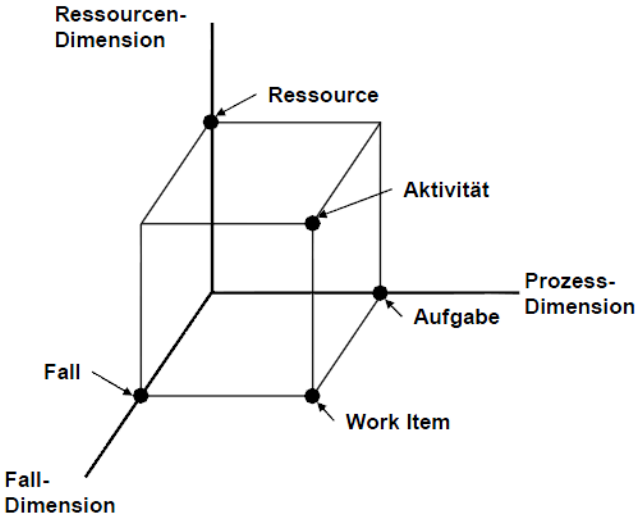


Abb. 2.9: Drei Dimensionen eines Workflows nach [Aals98]

Ein Workflow wird initiiert, wenn ein neuer „Fall“ (vgl. Kontext) auftritt bzw. zu bearbeiten ist (siehe Abb. 2.10). Für jeden „Fall“ wird ein Workflow ausgeführt. Ein Workflow wird aus einer Aneinanderreihung von Aufgaben definiert, die wiederum abhängig von Bedingungen (engl. *conditions*) sind [Aals98]. Es können Vor- und Nachbedingungen für Aufgaben (die in einem kausalen Zusammenhang stehen) spezifiziert werden [Aals98]. Der Workflow wird jeweils immer in einem bestimmten Kontext ausgeführt. Die Aufgabe steht in Relation zum „Fall“ und wird als *Work Item* bezeichnet. *Work Items* können durch eine Rolle bearbeitet werden. Wird eine Aufgabe in Bezug zum „Fall“ und der Ressource bestimmt, spricht [Aals98] von einer Aktivität. Die *Object Management Group*[16] (OMG) definiert (im Zusammenhang mit UML) den Begriff „Aktion“ für eine atomare Arbeitseinheit und „Aktivität“ für eine Folge von Aktionen [OMGS07]. Die WfMC kennt diese

[16] Die Object Management Group (OMG) ist ein Konsortium das 1989 gegründet wurde, um Standards zu entwickeln, die sich mit herstellerunabhängigen systemübergreifenden Objektorientierter Programmierung beschäftigen (http://www.omg.org/).

Unterscheidung nicht und verwendet den Begriff „Aktivität“ als durchgängigen Begriff [WfMC99]. Eine Aktivität wird immer in einem bestimmten Kontext („Fall“), abhängig von der Aufgabe und den Ressourcen (z. B. Identität, Ort) ausgeführt.

Die WfMC führt ein Metamodell für einen Workflow ein (siehe Abb. 2.10). Die Ressource Identität (*Role*) wird in diesem Metamodell explizit modelliert. Aktivitäten dienen als verbindendes Element zwischen den Dimensionen. Die Attribute zwischen den einzelnen Komponenten zeigen, welche Abhängigkeiten entstehen können (siehe Abb. 2.10).

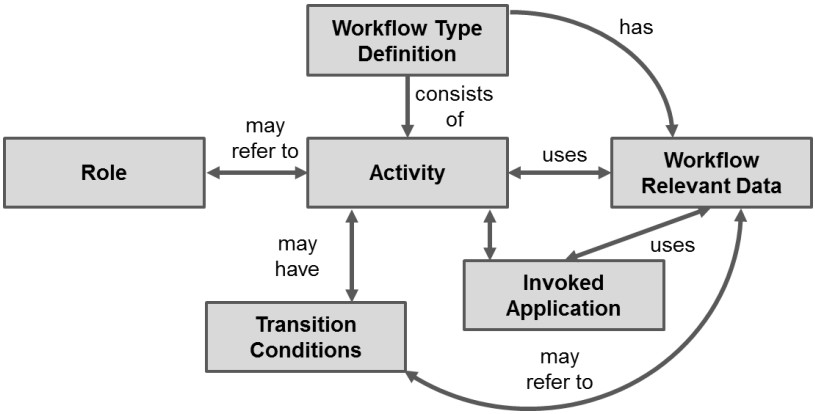


Abb. 2.10: WfMC Basic Process Definition Metamodel [Holl95]

Folgende Komponenteneigenschaften werden (basierend auf dem Metamodell) identifiziert [Müll05]:

Workflow-Typdefinition (Workflow Type Definition)

- Prozessname
- Versionsnummer
- Start- und Endbedingung des Prozesses
- Sicherheits-, Prüf- oder andere Kontrolldaten

Aktivität (Activity)

- Name der Aktivität
- Typ der Aktivität
- Bedingungen für die Pre- und Post-Aktivitäten
- Andere zeitliche Beschränkungen

Übergabebedingungen (Transition Conditions)

- Fluss- oder Ausführungsbedingungen

Workflow-bezogene Daten (Workflow Relevant Data)

- Dateiname und Pfad
- Datentyp

Rolle (Role)

- Name und organisatorische Verknüpfungen

Aufzurufende Applikation (Invoked Application)

- Typ und Name der Applikation
- Benötigte Parameter für die Ausführung der Applikation
- Pfad der Applikation

In dem Metamodell eines Workflows (siehe Abb. 2.10) werden alle Komponenten eines Workflows ersichtlich, die im späteren Verlauf der vorliegenden Arbeit Anwendung finden (siehe Kapitel 4).

2.5 Workflow-Managementsysteme (WfMS)

Ein Workflow-Managementsystem unterstützt das Management von automatisierten Geschäftsprozessen.

„*A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participant and, where required, invoke the use of IT tools and applications"* [WfMC99, S. 9].

Seit 1993 wird in der Workflow Management Coalition (WfMC) die Standardisierung von Workflows bzw. von *Business Process Management* (BPM) vorangetrieben. Um gemeinsame Charakteristiken und Komponententypen verschiedener Workflow-Systeme mit gleicher Terminologie zu beschreiben, wird das Referenz-Modell für Workflow-Managementsysteme definiert [Aals13].

Ein Workflow-Managementsystem (WfMS) stellt die Ausführung des Workflows sicher. Das Referenzmodell [Holl95] beschreibt einen generischen Aufbau eines Workflow-Managementsystems. Es setzt sich aus einzelnen Modulen zusammen, die über Schnittstellen miteinander interagieren. Jedes Modul beschreibt eine Klasse von Anwendungen, die spezifische Aufgaben innerhalb des Workflow-Managements übernehmen. Zu diesen Aufgaben zählen Modellierungswerkzeuge für Workflow-Modelle, Client-Anwendungen für Interaktionen mit Endbenutzern, Werkzeuge für die Administration bzw. Überwachung von Abläufen, externe Anwendungen, sowie ein zentrales Modul (bzw. zentrale Module), um die beschriebenen Bestandteile zu integrieren. Das zentrale Modul wird als Workflow-Engine bezeichnet. Es führt die definierten Abläufe aus und stellt eine Interaktion innerhalb der Engine bereit. Andere Engines können über die eigene Engine angebunden werden. Die modulare Organisation und die Beschaffenheit der Schnittstellen ermöglichen den Aufbau eines Gesamtsystems durch Softwareeinsatz unterschiedlicher Hersteller. Der schematische Aufbau eines WfMS nach WfMC wird in Abb. 2.11 dargestellt.

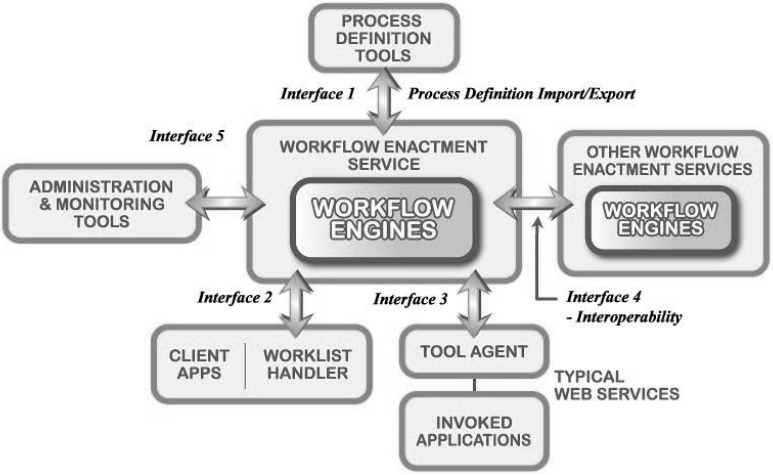


Abb. 2.11: WfMC-Referenzarchitektur

WfMS werden genutzt, um Prozessdefinitionen für einen entsprechenden Fall (Kontext) zu instanziieren und um Aktivitäten mit Ressourcen zu verknüpfen. Durch die WfMS wird der komplette Lebenszyklus eines Workflows von der Definition bis zur Ausführung unterstützt (siehe Kapitel 3.1).

Für eine Prozessdefinition (ein Prozessmodell) kann es innerhalb des WfMS mehrere Prozessinstanzen geben. Eine Prozessinstanz besteht wiederum aus Aktivitäten, die mit jeweils konkreten Ressourcen arbeiten. Die Basis, die Ableitung einer Prozessinstanz wird als Prozessschema bezeichnet (Bsp.: XML-Schema, xsd). Aktivitäten werden mit Applikationen verknüpft (siehe Abb. 2.10). Soll eine Aktivität automatisch abgearbeitet werden, so wird eine Applikation (Anwendung) aufgerufen. Wenn menschliche Interaktion notwendig ist, so wird eine Aufgabe für die entsprechende Rolle erstellt. Nach [Holl95] ist eine Aufgabe „[…] *the representation of the work to be processed (by a workflow participant) in the context of an activity within a process instance*". Eine Rolle nutzt ein IT-System, um Aufgaben abzuarbeiten bzw. zu erfüllen. Eine Aufgabe wird in vorliegender Arbeit (analog zu BPMN [Silv11]) als Benutzeraufgabe bezeichnet, wenn die Rolle zur Abarbeitung der Aufgabe ein IT-System nutzt (siehe Kapitel 7.2).

Die Aktivitäten werden in einem Workflow-Modell spezifiziert und durch eine Anwendung oder durch einen Endbenutzer als Benutzeraufgabe ausgeführt. Die von der WfMC ursprüngliche Bezeichnung eines *Work Item* soll ausdrücken, dass dies eine Benutzeraufgabe also eine von einer Rolle (mod. Bezeichnung *Workflow Participant* [OASI06]) ausgeführte Einheit ist. Heute wird meist der Begriff des *Human Task* verwendet, was gleichbedeutend mit einer Benutzeraufgabe ist.

In manchen WfMS wird zwischen Benutzeraufgaben und sogenannten Benachrichtigungen (engl. *Notifications*) unterschieden. Während Benutzeraufgaben synchron ausgeführt werden, sind *Notifications* asynchroner Natur.

WfMS sind Systeme, die die Definition, Verwaltung und Ausführung der Workflow-Modelle durch Software unterstützen. Die Ausführungsreihenfolge eines Prozessmodells wird mit einer Computer-Repräsentation der Workflow-Logik bestimmt (z. B. mit Hilfe eines XML-Dokuments).

In den folgenden Abschnitten werden die Komponenten eines WfMS einzeln vorgestellt:

2.5.1 Die Workflow-Interfaces

Ein WfMS besteht (siehe Abb. 2.11) aus mehreren Schnittstellen (engl. *interfaces*). Das WfMS ist modular aufgebaut, damit betriebliche Anwendungen einzelne Module des WfMS nutzen können, ohne die gesamte Middleware bereitstellen zu müssen.

Die Schnittstellen eines WfMS sind über eine Workflow-API (WAPI) standardisiert worden. In folgender Auflistung werden die Schnittstellen kurz erläutert. Eine ausführlichere Definition erfolgt in den Kapiteln 2.5.2 bis 2.5.6:

Interface 1: Das *Process Definition Tool* dient zur Erstellung bzw. Definition der Geschäftsprozesse. Es ist das Planungswerkzeug, das zum Entwurfszeitpunkt verwendet wird, um den Prozess zu modellieren.

Interface 2: Die *Client-Apps* oder *Worklist Handler* zeigen die Benutzerschnittstelle zum Endbenutzer, der die Aufgaben erledigen soll. Die Bearbeitung der Aufgabe kann mit einer geeigneten Applikation unterstützt werden oder mittels eines *Worklist Handler* weitergeleitet werden.

Interface 3: (Externe) Applikationen werden über Dienste aufgerufen oder sind direkt angebunden.

Interface 4: *Workflow Engines* sollen als Modul zur Verfügung stehen, um bei technischen Skalierungen schnell weitere Engine-Module hinzuzufügen. So können z. B. auch weitere Informationssysteme modular angebunden werden.

Interface 5: Hinter der fünften Schnittstelle verbirgt sich das *Monitoring* der ausgeführten Workflows. Die Instanz-Variablen (Daten) werden anhand einer bestimmten Merkmalsausprägung aggregiert und in graphischer Form aufbereitet, um Statistiken über die ausgeführten Instanzen zu erhalten. In einem Auftragsprozess wird z. B. für einen Kundenauftrag jeweils eine Instanz (eines Modells) geführt.

In der Architekturbeschreibung (siehe Abb. 2.12) eines WfMS sind keine konkreten Technologien oder Techniken standardisiert worden. Die Implementierungen der WfMS sind je nach Hersteller unterschiedlich.

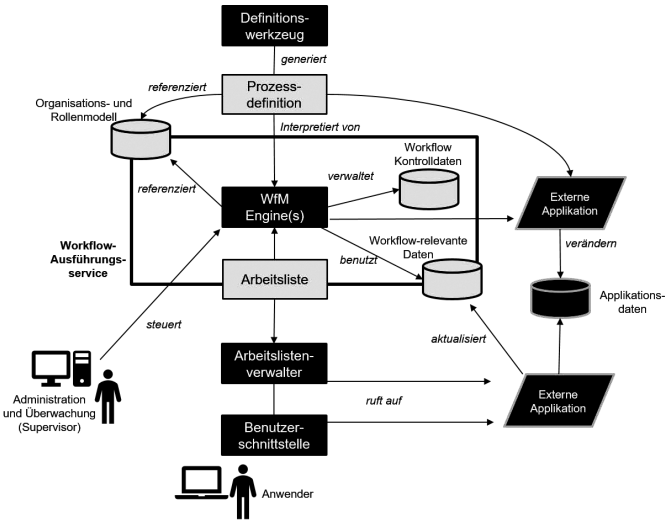


Abb. 2.12: WfMS nach [Holl95, S. 14]

2.5.2 Das Prozessdefinitionswerkzeug

Das Prozessdefinitionswerkzeug (siehe Abb. 2.12) – dient der graphischen Beschreibung eines Workflows (vgl. Kapitel 2.3). Workflows werden in einer Engine auf Basis der formalen Ausführungssemantik interpretiert. Das Modell benötigt somit eine Abbildung in eine formale Ausführungssprache.

Rollen werden bei der Modellierung der Workflows mit Aktivitäten verknüpft (assoziiert). Damit können die Aktivitäten über die entsprechenden Endbenutzer ausgeführt werden. Ein Akteur kann mehrere Rollen miteinschließen. Technisch werden Zugriffsberechtigungen in LDAP-Servern (siehe Kapitel 2.2.1) gepflegt.

Heutzutage gibt es eine Vielzahl an Modellierungswerkzeugen, je nach Anwendungsfall und verwendender Sprache. Ein Prozessdefinitionswerkzeug sollte eine Simulationskomponente enthalten, um eine „Machbarkeit“ der

Workflows so früh wie möglich abschätzen zu können. Die Simulationskomponente simuliert den Ablauf des modellierten Prozesses, um Schwachstellen aufzuzeigen und Verbesserungsvorschläge zu unterbreiten. Durch die Simulation kann festgestellt werden, welche Segmente im Workflow nicht durchlaufen wurden oder an welcher Stelle Verzögerungen auftraten, um den Workflow in einem nächsten Schritt umzuorganisieren (vgl. Kapitel 2.5.6). Natürlich kann eine Simulation nur erfolgreich sein, wenn die getroffenen Annahmen zu realistischen Daten führen. Letztendlich sollte stets eine Soll-Ist-Analyse nach der tatsächlichen Ausführung des Workflows stattfinden, um die Simulationskomponenten bzw. die Daten zu verbessern.

2.5.3 Die Benutzerschnittstelle

Die Benutzerschnittstelle ist die Komponente, auf die der Kunde bzw. der Mitarbeiter des Unternehmens (Workflow-Client) zugreift. Für Benutzeraufgaben kann z. B. die Darstellungsform einer Arbeitsliste (engl. *Tasklist*, siehe Abb. 2.12) gewählt werden. In der Form lässt sich die Taskliste nach verschiedenen Kriterien organisieren (z. B. Zeitliche-, Ortsgebundene-, Prioritäten-Abfolge). Die *Worklists* bzw. *Tasklists* (dt. Arbeitsliste oder Aufgabenliste) helfen den Rollen, die Aufgaben (*Tasks*) strukturiert abzuarbeiten. Die Arbeitslisten werden von der Workflow-Engine gefüllt. Aus Sicht der Benutzerschnittstelle ist es oft möglich, auf den „Pool“ an Arbeitsaufgaben zuzugreifen. Falls die Benutzerschnittstelle gegenüber Endbenutzern über die Unternehmensgrenzen hinweg angeboten werden, müssen die (mobilen) Aspekte der Benutzerschnittstelle berücksichtigt werden, damit die Qualität der Leistung garantiert werden kann.

2.5.4 Die externen Applikationen

Die Möglichkeit, externe Applikationen in ein WfMS zu integrieren, ist von besonderer Bedeutung für Unternehmen, in denen viele unterschiedliche Informationssysteme im Einsatz sind. Die Systeme sind integrativ in Verbindung mit anderen betrieblichen Informationssystemen verknüpft. Enterprise

Resource Planning (ERP)-Systeme oder Customer Relationship Management (CRM)-Systeme sind beispielsweise Vertreter solcher Applikationen bzw. IS. Die betrieblichen Informationssysteme integrieren Workflow-Funktionalität. Aus Sicht des WfMS werden andere Applikationen aufgerufen, um z. B. auf ein bestimmtes Ereignis innerhalb des Workflows zu reagieren.

2.5.5 Die WfM-Engine

Die Workflow-Engine bzw. die WfMS-Engine ist die zentrale Komponente eines WfMS. Die Engine führt die Workflows aus. Die definierten Prozesse werden in die Ausführungsumgebung *deployed* (dt. eingespielt). Andere Anwendungen (bzw. Dienste) werden aufgerufen (*invoked)*, die zur Bearbeitung von Aktivitäten notwendig sind. Die *Workflow Engine* hat u. U. Zugriff auf das Organisations- bzw. Rollenmodell einer Unternehmung, um Nutzern in Form einer Benutzeraufgabe zu unterstützen bzw. einer Rolle einer Aufgabe zuzuteilen.

In der vorliegenden Arbeit sollen *Administrative Workflows* und *Production Workflows* (siehe Abschnitt 2.4) unterstützt werden.

Ad-hoc Workflows und *Collaborative Workflows* werden nicht betrachtet. *Ad-hoc Workflows* würden beispielsweise Prozessmodell-Varianten erfordern, um möglichst flexibel auf Ad-hoc-Ereignisse reagieren zu können. *Collaborative Workflows* sind wenig strukturiert bzw. automatisierbar. Sie sind keine IT-abbildbare Prozesse [Aals98, S. 8].

Die Engines sind auf bestimmte Workflow-Sprachen spezialisiert (z. B. Petri-Netz, BPEL, BPMN). BPEL eignet sich beispielsweise speziell zum Entwurf einer SOA, da die Sprache auf verschiedenen Webservice-Standards basiert [LeLN11, S. 199ff.].

2.5.6 Die Administration und Überwachung

Beim „Monitoring“ (Administration, Überwachung) werden Workflow-instanz-bezogene Daten (der sogenannte Prozesskontext) protokolliert und innerhalb von Graphiken bzw. Diagrammen aufbereitet. Innerhalb dieser Überwachung werden Prozesskennzahlen aufbereitet. Soll-Ist-Analysen können während der Ausführung der Workflows über sogenannte Performance-Netze ermöglicht werden [Mevi06]. Die Daten einer bestimmten Prozessinstanz werden ausgewertet. Liegezeiten oder Bearbeitungszeiten können gemessen werden. Für die Nachvollziehbarkeit können weitere im Workflow-Modell enthaltene Geschäftsdaten analysiert werden. Viele Anforderungen für das Monitoring werden beispielsweise für die ISO 9000 Zertifizierung verlangt. Führungskräfte bzw. das Management eines Unternehmens erhalten durch diese Informationen eine Steuerungsmöglichkeit. Gleichzeitig kann das Management (schnell) reagieren, sofern die Prozesskennzahlen von ihrem Soll-Zustand abweichen. Auf Basis der vorgangsbezogenen Daten (Prozesskontext) kann die Effektivität des Workflows ermittelt werden. Flaschenhälse werden identifiziert, die zur Reorganisation des Workflow-Modells oder zu einer neuen Ressourcen-Aufteilung führen.

2.6 Einführung in Petri-Netze

Petri-Netze werden u. a. als graphische Modellierungssprache für Workflows benutzt. Der größte Vorteil von Petri-Netzen gegenüber anderen Modellierungssprachen ist ihre Eigenschaft, direkt nach der Modellierung ausgeführt und simuliert werden zu können. Workflows als Automatisierung von Geschäftsprozessen können mit Petri-Netzen umgesetzt werden [AAAM97, Aals98, Ober96, RiSt04].

Die Modellierungssprache bzw. Ausführungssprache ist ursprünglich von Carl Adam Petri in der Dissertation „Kommunikation mit Automaten“ vorgestellt worden [Reis10b]. Basierend auf diesem Ansatz sind seit 1962, der Veröffentlichung der Dissertation, viele wissenschaftliche Beiträge entstanden [Pete77]. Zahlreiche Erweiterungen, wie z. B. die Zeitstrukturen

[Ober90], wurden in die Petri-Netze integriert. Mit der Modellierung von Petri-Netzen kann das Systemverhalten, also auch das Verhalten betrieblicher Abläufe (Workflows), dargestellt werden. Aus dieser Motivation heraus beschreibt [Aals96] drei Gründe, weshalb Petri-Netze gerade in WfMS von Vorteil sind:

1. Besitzen eine formale Semantik und eigenen sich zur graphischen Darstellung
2. Sind Zustand- und Ereignisbasiert
3. Verfügbarkeit vieler *Analyseverfahren*

Die Grundlagen von Petri-Netzen behandeln sowohl die statische Struktur als auch das dynamische Verhalten. Die als einfaches Petri-Netz bezeichnete Modellierungssprache (Grundform) besteht aus einem einfachen Aufbau mit vier Sprachkonzepten, die in folgender Abb. 2.13 in graphischer Form dargestellt sind.

Abb. 2.13: Elemente eines Petri-Netzes

Eine Stelle, eine Transition, eine Kante und die Marke als Komponenten eines einfachen Petri-Netzes sind dargestellt (siehe Abb. 2.13). Netzelemente sind Stellen und Transitionen, die jeweils über Kanten miteinander verknüpft sind. Eine Marke wird als ausgefüllter Kreis graphisch dargestellt. In einfachen Petri-Netzen sind Marken nicht unterscheidbar. In höheren Petri-Netzen können Marken (farbig) unterschieden werden.

Die Elemente besitzen innerhalb des Netzes jeweils unterschiedliche Funktionen:

Eine Stelle soll eine Bedingung als statischer Bestandteil eines Petri-Netzes ausdrücken. Transitionen repräsentieren Aktivitäten und stellen den dynamischen Anteil eines Petri-Netzes dar. Der Kontrollfluss entsteht innerhalb eines Petri-Netzes, indem Stellen und Transitionen über Kanten jeweils abwechselnd verbunden werden und somit eine Reihenfolge innerhalb des Modells festlegen. Die Marken legen den Kontrollfluss fest, da sie entsprechende Zustände innerhalb der Stelle ausdrücken. Der einfache Aufbau eines Petri-Netzes unterscheidet die Notation von den bereits erwähnten BPMN oder UML-Aktivitätsdiagramme, die wesentlich mehr Sprachkonzepte besitzen. Die komplette Geschäftslogik eines Prozesses wird in Petri-Netz-Modellen mit Hilfe der gerade vorgestellten Elemente abgebildet. Die Modelle besitzen eine formale Syntax mit zugehöriger Semantik [Ober90, S. 98, RiSt04, S. 72]. Die formale Definition fasst die bisher textuell beschriebenen Elemente eines Petri-Netz-Modells zusammen [ReRo98]:

Definition 1: Petri-Netz

Ein Petri-Netz ist ein Tripel $N = (S, T, F)$ mit

- S (Stellen), T (Transitionen) sind endliche Mengen,
- $S \cap T = \emptyset$
- $S \cup T \neq \emptyset$
- $F \subseteq (S \times T) \cup (T \times S)$ ist eine binäre Relation über $S \cup T$

In der *Definition 1* wird ein bipartiter, gerichteter Graph beschrieben, der jedoch noch keine Aussage über das Verhalten innerhalb des Modells gibt. Diese, auch als Dynamik bezeichnete Eigenschaft eines Petri-Netzes, ist notwendig, um das Verhalten des Geschäftsprozesses bzw. Workflows abzubilden. Dazu werden Marken verwendet. Die Menge aller Marken im Petri-Netz wird als Markierung bezeichnet [ReRo98].

Definition 2: Markierung

Eine Markierung m eines Petri-Netzes $N = (S, T, F)$ ist eine Abbildung $m : S \rightarrow \mathbb{N}$.

- $n = |S|$ (Anzahl der Stellen)
- $m_i \in \mathbb{N} \; \forall \; i = 1, \ldots, n$ (Anzahl der Marken in Stelle i)

Die Markierung beschreibt für jede Stelle die Anzahl der darin enthaltenen Marken. Die Bedeutung einer Markierung ist wie folgt definiert [ReRo98]:

Definition 3: Markiertes Petri-Netz

Ein Petri-Netz $N = (S, T, F)$ mit der Markierung m wird als markiertes Petri-Netz (S, T, F, m) bezeichnet.

Beim Systemverhalten eines Modells müssen Übergänge von Zustand zu Zustand nachvollziehbar sein. Für diese Zustandsübergänge einfacher markierter Petri-Netze müssen Schaltregeln definiert werden. Vor- und Nachbereiche von Netzelementen müssen definiert werden. Der Vorbereich bezeichnet die Menge der Netzelemente, die eine ausgehende Verbindung zum aktuell betrachteten Netzelement hat. Der Nachbereich bezeichnet die Menge der Netzelemente, die vom aktuell betrachteten Netzelement ausgehen.

Die Definition des Vorbereichs $\bullet x$ eines Netzelements wird in folgendem Ausdruck formal eingeführt [Dese92, S.16]:

$$\bullet x = \{y \in S \cup T \mid (y, x) \in F\}, x \in S \cup T \qquad \text{(Vorbereich)}$$

Die Definition des Nachbereichs eines Netzelements wird in folgendem Ausdruck formal eingeführt [Dese92, S.16]:

$$x \bullet = \{y \in S \cup T \mid (x, y) \in F\}, x \in S \cup T \qquad \text{(Nachbereich)}$$

Die Schaltregeln eines Petri-Netzes werden mit Hilfe der Definition von Vor- und Nachbereich formal beschrieben [Dese92, S.17]:

Definition 4: Schaltregel

Eine Markierung m aktiviert eine Transition t, wenn $m(s) > 0$ für alle $s \in \bullet t$. Falls t unter m aktiviert ist, kann t schalten. Nach dem Schalten von t kommt es zu der Folgemarkierung m':

$$m'(s) = \begin{cases} m(s), & \text{wenn } s \notin \bullet t \wedge s \notin t\bullet \\ m(s)-1, & \text{wenn } s \in \bullet t \wedge s \notin t\bullet \\ m(s)+1, & \text{wenn } s \notin \bullet t \wedge s \in t\bullet \\ m(s), & \text{wenn } s \in \bullet t \wedge s \in t\bullet \end{cases}$$

Ein Schaltvorgang wird durch $m \rightarrow^t m'$ beschrieben.

Die obige Definition zeigt, dass Transitionen durch das Schalten die Anzahl der jeweils vorhandenen Marken in den Stellen beeinflussen, was mit den vier verschiedenen Fällen zum Ausdruck gebracht wird:

- Die Anzahl der Marken einer Stelle, die sich nicht im Vor- oder Nachbereich einer Transition befindet, wird nicht verändert.
- Ist die Stelle Teil des Vorbereichs einer Transition, wird die Anzahl der enthaltenen Marken um eins reduziert.
- Ist die Stelle Teil des Nachbereichs, wird eine Marke in die entsprechende Stelle hinzugefügt.
- Wenn sich vor Beginn des Schaltvorgangs im Vor- und Nachbereich eine Marke in der jeweiligen Stelle befindet, ändert sich die Anzahl Marken innerhalb der jeweiligen Stelle nicht.

Im weiteren Verlauf der vorliegenden Arbeit wird eine Transition jeweils mit einer Aktivität assoziiert. Die Ausführung der Aktivität transportiert eine oder mehrere Marken in die Folgemarkierung während sie eine bzw. mehrere Marken der Vorgängermarkierung konsumiert. Mit dieser Eigenschaft kann ein Kontrollfluss - wie in Kapitel 2.3 beschrieben - abgebildet werden.

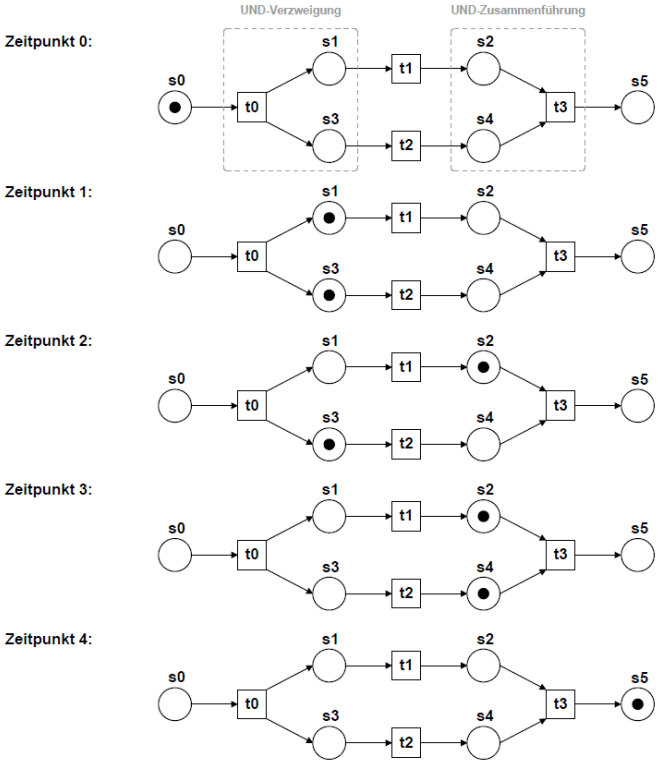


Abb. 2.14: Schaltvorgänge in Petri-Netzen

In Abb. 2.14 sind verschiedene Kontrollflüsse innerhalb eines Petri-Netzes dargestellt. Eine Besonderheit ist die dargestellte Nebenläufigkeit. Die beiden Ablaufstränge (t1, t2) sind voneinander unabhängig. In Abb. 2.14 wird jeder einzelne (Ablauf-)Schritt des Petri-Netzes nach den Zeitpunkten 0 bis 4, jeweils nach dem Schaltvorgang, dargestellt. Beim Zeitpunkt 0, also beim Start, besitzt das Petri-Netz die Anfangsmarkierung $m_0 = (1, 0, 0, 0, 0, 0)$. Nach *Definition 4* wird Transition t0 aktiviert und kann schalten. Ergebnis des Schaltens ($m_0 \rightarrow^{t0} m_1$) von t1 lässt sich zum Zeitpunkt 1 erkennen. Die Markierung $m_1 = (0, 1, 0, 1, 0, 0)$ zeigt an, dass die Transitionen t1 und t2

schalten können. Bei Zeitpunkt 2 schaltet die Transition t1. Dies ist allerdings nicht vorgegeben. Es könnte auch erst Transition t2 schalten. Die Markierung $m_2 = (0, 0, 1, 1, 0, 0)$ entsteht nach dem Schaltvorgang $m_1 \rightarrow^{t1} m_2$. Nur Transition t2 kann schalten, da t3 auf Grund des unvollständigen Vorbereichs (fehlende Marke) noch nicht aktiviert ist. Das Petri-Netz schaltet $m_2 \rightarrow^{t2} m_3$ und führt zu $m_3 = (0, 0, 1, 0, 1, 0)$. Zu diesem Zeitpunkt kann nur t3 schalten $m_3 \rightarrow^{t3} m_4$. Der Kontrollfluss endet mit der Markierung $m_3 = (0, 0, 0, 0, 0, 1)$.

2.7 Einführung in die Business Process Modeling and Notation

Das *Business Process Model & Notation* (BPMN) beinhaltet eine Flowchart-basierte Sprache, die 2004 von der (BPMI) verabschiedet wurde [Whit04]. Im Jahr 2006 wurde die Version 1.0 veröffentlicht. Seit 2011 besitzt die Sprache eine formale Beschreibung, so dass sie technisch ausgeführt werden kann [ObMG10a]. Die Sprache erhielt zudem eine Erweiterungsmöglichkeit und wurde letztendlich am 15.07.2013 zu einem internationalen Standard ISO/IEC 19510:2013. Während Petri-Netze ursprünglich nicht ausschließlich für das Geschäftsprozessmanagement konzipiert wurden, ist das BPMN konsequent an den Anforderungen dieser Disziplin ausgerichtet worden. Die Modellierung mit BPMN gewährleistet eine eindeutige und kompakte Darstellung von Abläufen. Neben einer einfach zu verwendeten Geschäftsprozessmodellierungssprache war das Ziel von BPMN, die Prozessautomatisierung zu unterstützen [WhMi08, S. 24ff]. Mit Hilfe der verschiedenen Sprachkonzepte von BPMN können sowohl fachliche als auch technische Modelle angefertigt werden. Die Sprache eignet sich insbesondere zur Überwindung des Business-IT-Gaps, um eine Brücke zwischen Fach- und IT-Abteilungen zu schlagen. Die Fachabteilungen modellieren die Geschäftsprozesse aus ihrer Sicht bzw. ihrem Blickwinkel, während die IT-Abteilungen die Prozesse in einem weiteren Schritt verfeinern bzw. konkretisieren, damit diese zur technischen Ausführung als Workflows verwendet werden können. Die Sprache beinhaltet zusätzlich eine Abbildungsvorschrift

in die ausführbare BPEL [ObMG09]. Jedoch ist solche eine Transformation seit der Version 2.0 nicht mehr notwendig, da BPMN 2.0 direkt ausgeführt werden kann [ObMG10a].

Die vielen Symbole und Dekorationen (der BPMN) mit ihren Varianten machen die Modellierung vergleichsweise komplex. Die BPMN 2.0-Modelle sind daher für das ungeübte Auge eines Neuanwenders schwer verständlich. Demgegenüber besitzen die Petri-Netze nur wenige Sprachelemente, wodurch eine eindeutige Beschreibung der möglichen Verläufe eines Geschäftsprozesses erlaubt wird. BPMN 2.0 verfolgt eine andere Strategie, indem eine Vielzahl an spezialisierten Sprachkonzepten geschaffen wurde. Auf Grund dieser Tatsache wird eine Einschränkung der Auswahl der BPMN-Elemente — je höher die Abstraktionsebene — empfohlen [FrRH10]. Es wird oft zwischen strategischer, operativer und technischer Ebene unterschieden [FrRH10]. Welche Vor- und Nachteile die Sprache liefert, wird in vorliegender Arbeit nicht erörtert.

BPMN-Elemente:

In BPMN 2.0 lassen sich fünf Kategorien von BPMN-Elementen unterscheiden [ObMG10a]:

- Flusselemente (engl. *Flow Objects*)
- Verbindungselemente (engl. *Connecting Objects*)
- Artefakte (engl. *Artifacts*)
- Datenobjekte (engl. *Data Objects*)
- *Swimlanes*

Flusselemente:

Betriebliche Abläufe werden in der BPMN 2.0 hauptsächlich mit Flusselementen modelliert. Die Flusselemente werden über sogenannte Sequenzflusselemente miteinander verknüpft, um einen oder mehrere Pfade durch das Geschäftsprozessmodell zu gewährleisten. Folgende Elemente stehen zur Verfügung:

- Aktivitäten (engl. *Activities*)
- Gateways
- Ereignisse (engl. *Events*)

Aktivitäten sind in BPMN 2.0 Tätigkeiten, die durchgeführt werden. Die Ausführung einer Aktivität beansprucht eine gewisse Zeitspanne. Wenn eine Aktivität atomar ist und nicht in weitere Einzelschritte zerlegt wird, kann sie auch als Aufgabe (engl. *Task*) bezeichnet werden. Zusammengesetzte Aktivitäten werden als Subprozesse (engl. *Subprocess*) benannt [FrRH10]. Der Subprozess kann in BPMN ein eigenständiges Prozessmodell sein [FrRH10]. Subprozesse können in BPMN 2.0 auf- und zugeklappt innerhalb von Modellen dargestellt werden. Zugeklappt werden die Subprozesse mit Hilfe eines Plus-Zeichens („+“) und einem quadratischen Rahmen als Dekoration der Aktivität dargestellt. In Abb. 2.15 werden beide Darstellungen demonstriert.

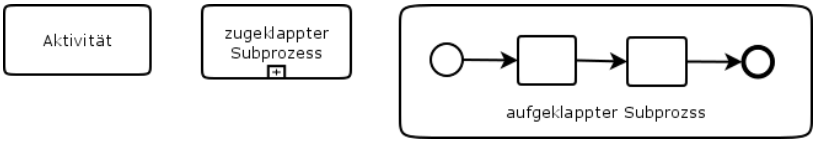


Abb. 2.15: Aktivitäten und Subprozesse in BPMN [ObMG10a].

Für das Symbol der Aktivität existieren weitere Dekorationen als Markierungen (z. B. eine Schleife) oder als Typisierungen (Tasktypen). Markierungen verleihen der Aktivität eine besondere Ablaufsteuerung. Typisierungen verleihen der Aktivität ein bestimmtes Verhalten (z. B. eine Manuelle Aktivität). Die Markierungen und Typisierungen sind generell miteinander kombinierbar. Weitere Spezifikationen, wie z. B. Transaktions- und Ereignisteilprozesse bzw. Aufrufaktivitäten können der BPMN-Spezifikation entnommen werden [ObMG10a].

Mit Gateways werden Verzweigungen sowie Zusammenführungen von Sequenzflüssen modelliert. An den ausgehenden Sequenzflüssen an einem

Gateway werden Entscheidungen bzw. Bedingungen getroffen. Es gibt datenbasierte und ereignisbasierte Gateways, die in Form einer Raute modelliert werden. Zur Differenzierung der verschiedenen Arten werden die Gateways mit Dekorationen versehen. In Abb. 2.16 werden drei verschiedene Gateway-Arten dargestellt.

Abb. 2.16: XOR-, UND-, und Inklusives Gateways in BPMN

Bei einem exklusiven ODER-Gateway (XOR) wird eine der ausgehenden Sequenzflusskanten aktiviert. Bei einer Zusammenführung eines XOR wird der Sequenzfluss bei der Aktivierung einer Verbindung weitergeleitet. Das parallele Gateway repräsentiert eine UND-Verknüpfung bzw. –Zusammenführung. Der Sequenzfluss wird an alle ausgehenden Sequenzflussverbindungen weitergeleitet. Bei einer Zusammenführung wird auf jede eingehende Sequenzflussverbindung gewartet. Ein inklusives ODER-Gateway leitet an mindestens eine ausgehende Verbindung weiter (m aus n). Die analoge Zusammenführung wartet auf mindestens einen aktivierten eingehenden Sequenzfluss (m aus n). Ein komplexes Gateway erlaubt mehrschichtige Entscheidungen. Darüber hinaus existieren Ereignisbasierte Gateways ausschließlich als Verzweigung. Eintretende Ereignisse (nach dem Gateway modelliert) entscheiden, welche Sequenzfluss-Verbindung aktiviert wird. Weitere Verzweigungen können über nicht-unterbrechende Ereignisse oder den sogenannten *Conditional Flow* erzielt werden [ObMG10a].

Ereignisse können in BPMN 2.0 an vielen Stellen im Kontrollfluss im Geschäftsprozessmodell platziert werden [ObMG10a]. Die Sprache unterscheidet zwischen Start-, Zwischen- und Endereignissen. Mit dem Eintreten eines Startereignisses wird automatisch eine Prozessinstanz initiiert. Zwischenereignisse werden einerseits innerhalb des Sequenzfluss als ein- oder auslösendes Ereignis modelliert. Andererseits besteht die Möglichkeit, Zwi-

schenereignisse an Aktivitäten als angeheftetes Zwischenereignis zu modellieren. Tritt ein Start- oder eintretendes Zwischenereignis ein, so wird der ausgehende Sequenzfluss aktiviert. Bei einem Endereignis wird der Sequenzfluss oder der ganze Prozess beendet. In Abb. 2.17 ist die Grundform des Start-, Zwischen- und Endereignisses dargestellt, die mit verschiedenen Dekorationen typisiert werden können [ObMG10a]. Ein Timer-Ereignis erlaubt z. B. die Definition eines Zeitpunktes oder Zeitintervalls. Start- und Zwischenereignisse können mit gestrichelten Linien gezeichnet werden. Dies bedeutet, dass der aktuelle Sequenzfluss nicht unterbrochen wird, jedoch ein zusätzlicher Sequenzfluss bei Eintreten des Ereignisses abgespalten wird.

Abb. 2.17: Start-, Zwischen-, und Endereignis in BPMN

Verbindungselemente:

Drei Arten von Verbindungselementen werden unterschieden:

- Sequenzflüsse (engl. *Sequence Flow*)
- Nachrichtenflüsse (engl. *Message Flows*)
- (Daten-)Assoziationen (engl. (*Data*) *Associations*)

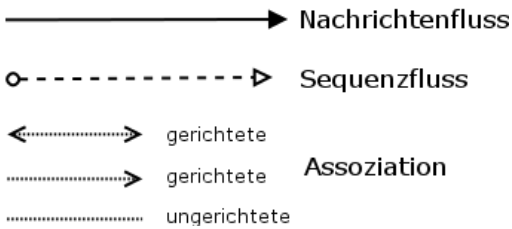


Abb. 2.18: Verbindungselemente in BPMN

Flusselemente werden mit Sequenzflüssen (siehe Abb. 2.18) miteinander verbunden, wodurch der Ablauf (Pfad) im Modell manifestiert wird. Nachrichtenflüsse werden benutzt, um Kommunikationsbeziehungen zwischen unterschiedlichen Systemen bzw. Beteiligten abzubilden. Assoziationen können wie Sequenz- oder Nachrichtenflüsse gerichtet sein. Mit Assoziationen werden Verknüpfungen zwischen Datenobjekten und Flusselementen dargestellt. Im BPMN 2.0 Metamodell sind Assoziationen als spezielle Artefakte eingeführt worden [ObMG10a].

Artefakte:

Die Kategorie der Artefakte umfasst Elemente, die den Sequenzfluss bzw. den Ablauf der Prozessinstanz nicht beeinflussen. Sie dienen als zusätzliche Informationen innerhalb eines BPMN-Diagramms. In der Spezifikation von BPMN [ObMG10a] werden die Artefakte als Erweiterungspunkte für domänenspezifische Objekte definiert. Folgende Standard-Artefakte werden in BPMN 2.0 definiert:

- Gruppierungen (engl. *Groups*)
- Anmerkungen (engl. *Text Annotations*)

Eine Gruppierung dient als Markierung von zusammengehörigen Elementen innerhalb des Diagramms. Anmerkungen sind meist nutzerspezifische Kommentare zum besseren Verständnis des Flusselementes.

Datenobjekte:

Datenobjekte sind elektronische Dokumente oder Datensätze einer Datenbank, die mit Daten-Assoziationen mit den Flussobjekten bzw. Sequenzflüssen verknüpft werden. Es gibt mehrere Arten von Datenobjekten:

- Datenobjekte (engl. *Data Objects*)
- Listen-Datenobjekte (engl. *List Data Objects*)
- Input-Datenobjekte (engl. *Data Inputs*)
- Output-Datenobjekte (engl. *Data Outputs*)
- Datenspeicher (engl. *Data Stores*)

Die Input-Datenobjekte werden von einer Quelle der Prozessinstanz zugeführt, während Output-Datenobjekte Ergebnis einer Instanz sind. Datenobjekte „fließen“ (mit dem Sequenzfluss) durch den Prozess und werden währenddessen gelesen und verändert. Diese auch als Prozesskontext bezeichneten Daten können mit Hilfe der Daten-Objekte in gesonderten Datenspeichern abgelegt werden, damit sie z. B. auch nach dem Beenden der Prozessinstanz verfügbar bleiben.

Swimlanes:

Die *Swimlanes* werden genutzt, um verschiedene Entitäten innerhalb von betrieblichen bzw. überbetrieblichen Abläufen voneinander abzugrenzen. Die Entitäten können verschachtelt werden, indem sogenannte Lanes innerhalb des Pools eingeführt werden. Mit Lanes können z. B. Abteilungen, verschiedene Auftraggeber oder Rollen abgebildet werden. Pools enthalten jeweils einen vollständigen Geschäftsprozess einer beteiligten Entität. Sequenzflüsse dürfen daher die Grenzen eines Pools nie überschreiten. Pools kontrollieren bzw. verwalten das Modell. Ein Pool kann in BPMN 2.0 zusammengeklappt (als *Black Box*) oder aufgeklappt mit einem zugehörigen Modell illustriert werden. Ein BPMN-Kollaborationsdiagramm ermöglicht die Visualisierung von Kommunikationsbeziehungen zwischen mehreren Entitäten. Die Prozesse der jeweiligen Pools sind unabhängig voneinander bzw. müssen wiederum als eigenes System betrachtet werden (z. B. eigenes Callback-Interface für asynchrone Kommunikation), die über Nachrichtenflüsse miteinander kommunizieren. Innerhalb von Pools dürfen keine Nachrichtenflüsse stattfinden.

BPMN-Diagrammarten:

In BPMN 2.0 sind unterschiedliche Diagrammarten spezifiziert:

- Prozessdiagramme
- Kollaborationsdiagramme
- Choreographiediagramme
- Konversationsdiagramme

Ein modelliertes Prozessdiagramm beschreibt die Ablaufpfade eines Geschäftsprozesses in Form eines Modells. Die bisher vorgestellten Sprachkonzepte werden innerhalb dieser Diagrammart angewandt. Innerhalb von Kollaborationsdiagrammen werden Nachrichtenflüsse zwischen verschiedenen Entitäten (Identitäten, Teilnehmern) graphisch modelliert. Die Prozessdetails werden vollständig verborgen, so dass nur die gemeinsame Interaktion sichtbar wird. Choreographie- und Konversationsdiagramme sollen eine noch übersichtlichere Darstellung der Entitäten Prozessübergreifender Abläufe liefern. Bei Choreographiediagrammen werden Aktivitäten modelliert, die den Informationsaustausch der Entitäten explizit darstellen. Innerhalb von Konversationsdiagrammen werden die Kommunikationsbeziehungen auf sogenannte Konversationen reduziert, indem logisch zusammengehörige Nachrichtenflüsse zusammengeführt werden. Die Konversationen können über Konversationslinks mit den kollaborierten Pools der Entitäten verknüpft werden. Im Rahmen der vorliegenden Arbeit werden Prozessdiagramme verwendet, wie beispielsweise in [Silv11] empfohlen wird.

Eine ausführliche Einführung in die Modellierungssprache BPMN 2.0 wird in [Allw09, FrRH10, Silv11] geliefert.

2.8 Modelle zur Endbenutzerklassifikation

Technische Applikationen bzw. mobile Apps werden für eine bestimmte Aufgabe und für eine bestimmte Nutzergruppe angefertigt [Appl15]. Für die Identifizierung potentieller Nutzer von mobilen Diensten (siehe Anhang B) werden verschiedene Begriffe voneinander abgegrenzt:

- Unter einer Klassifikation wird das Ergebnis der Klassifizierung (Kategorisierung) verstanden. Es bezeichnet das Verhalten bzw. die Eigenschaften einer bestimmten Klasse. Falls die Klassen vorab nicht bekannt sind, wird in vorliegender Arbeit von Segmentierung bzw. Segmenten gesprochen. Diese können z. B. Ergebnis einer Clusteranalyse [Lehn09, S. 265ff] sein und führen zu Gruppen mit neuen charakteristischen Eigenschaften.

- Bei einer Klassifizierung wird eine Nutzergruppe in kleinere Teilgruppen unterteilt, um dann für die kleineren Teilgruppen Lösungen anzubieten, die für die größere Gruppe nicht möglich sind. So wird der Markt potentieller Endbenutzer in verschiedene Klassen von Endbenutzern nach Zielen, Vorlieben, bestimmten Eigenschaften (Alter), Fähigkeiten, Wünschen oder anderen Charakteristika aufgespalten.

- Marketingmaßnahmen bzw. die Produkte werden nur genutzt, wenn entsprechende Endbenutzergruppen nach Anforderungen der Kunden berücksichtigt wurden [SeWC10, LöPe01, McDu07].

- Erst wenn die Benutzergruppe eindeutig klassifiziert ist, kann eine Aussage über das letztendliche Produkt (z. B. eine Applikation) getroffen werden [FiCh97]. Zentraler Schlüssel zum Erfolg einer App sind die Informationen über die Endbenutzer (hier: Eigenschaften, Präferenzen, Wünsche) [FeHS06].

- Eine Klassifikation besteht aus den Dimensionen: Demographie, Psychographie und Verhalten der Endbenutzer [GiKe03].

    - **Demographie** bezeichnet die quantitative Betrachtung der menschlichen Bevölkerung eines Landes (einer Gruppe). Dabei stellt die Demographie eine grundsätzliche Möglichkeit dar, bestimmte Gruppierungen nach Anzahl oder nach bestimmten Merkmalen zu erstellen. Die demographischen Unterdimensionen sind Alter, Geschlecht, Lohn, Religion, Familienstand oder Bildungsstand [GiKe03].

    - Die **Psychographie** beinhaltet Merkmale bzw. Einstellungen der Persönlichkeit der Endbenutzer (z. B. Interessen, Aktivitäten, Lebensstil oder bestimmte Werte und Meinungen).

- Am Produkt (App) kann über bestimmte Messungen des **Kauf- und Nutzungsverhalten** der Endbenutzer ein sogenanntes Verhaltensmuster festgestellt werden. Dabei werden Reaktionen auf Preise oder Werbemaßnahmen in bestimmten Kategorien erfasst.

- Psychographie und das Verhalten der Endbenutzer lassen sich nicht unabhängig voneinander betrachten, d.h. eine gegenseitige Beeinflussung kann nicht ausgeschlossen werden. Überschneidungen lassen sich z. B. in der Innovationsbereitschaft oder dem Übernahmeverhalten von Endbenutzern erkennen.

Die **geographischen Eigenschaften** werden als weitere Dimension bei der Klassifizierung von Endbenutzern berücksichtigt. Kulturelle Unterschiede führen zu einem anderen Nutzungsverhalten, das nicht vernachlässigt werden darf [CLKJ05, PaCh02].

Um eine Klassifikation (mobiler) Endbenutzer zu erhalten, werden Akzeptanzkriterien neuer Technologien bzw. mobiler Apps in der Gesellschaft analysiert. Bei der Einführung von (betrieblichen) Informationssystemen hat sich die Anwendung das *Technology Acceptance Model* (TAM) als Standard etabliert [KiHe06, DaBW89, GeCa02]. Ob das TAM auch für mWfMS geeignet ist, wird im Verlauf von Anhang B erörtert. Das TAM wurde aus dem *Theory of Reasoned Action* (TRA) [FiAj75] und der *Theory of Planned Behaviour* (TPB) [Ajze91] entwickelt [Davi86, Davi89].

Das TAM basiert auf den Begriffen „Perceived Usefulness“ und dem „Perceived Ease of Use“. Das „Perceived Usefulness“ verdeutlicht den Glauben der Nutzer, dass das Produkt ihre Tätigkeit (vgl. Benutzeraufgabe) unterstützt. „Perceived Ease of Use“ hingegen zeigt, ob der Nutzer die Technologie bzw. das Produkt einfach anwenden kann. Die verschiedenen Produkte werden in binären Variablen, den sogenannten „Design Features“ (siehe Abb. 2.19), abgebildet und verdeutlichen damit die Motivation der Endbenutzer [Davi86, S. 25-27].

Im TAM beeinflussen die zwei zentralen Elemente „Perceived Usefulness“ und „Perceived Ease of Use“ das Element „Attitude Toward Using“ (siehe

Abb. 2.19). Die „Attitude Toward Using“ mündet wiederum in die „Actual System Use“. Außerhalb der gestrichelten Linie (siehe Abb. 2.19) werden die gemessenen Daten analysiert, wohingegen die Daten der „User Motivation“ (siehe Abb. 2.19) über Endbenutzerbefragungen erhoben werden.

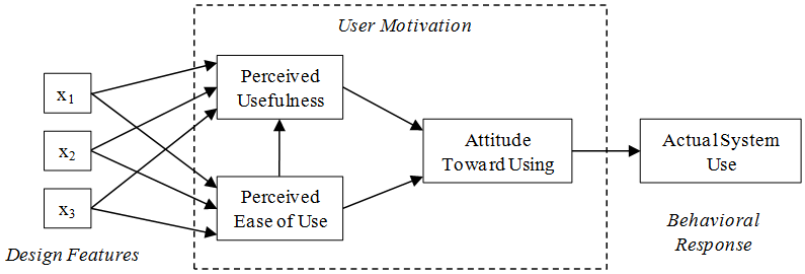


Abb. 2.19: TAM nach [Davi86, S. 24]

Das TAM dient als Basis für weitere Technologie-Akzeptanzmodelle [VMDD03, DaBW92, VeDa00]. Diese Weiterentwicklungen beziehen sich auf eine Erweiterung des *Perceived Enjoyment* [DaBW92, VeDa00]. Die Erweiterungen sind notwendig, da das TAM die Wirkungszusammenhänge nur grob darstellen lässt und das „Warum“ oft nicht explizit beantwortet werden kann [Bago07].

Das Diffusionsmodell misst die Innovationsbereitschaft bei der Nutzung von neuen Technologien [Roge03]. Die Diffusion von Innovationen wird als Prozess erfasst, der durch Verbreitung in der Gesellschaft stattfindet [Roge03]. Es werden fünf Nutzergruppen (sogenannte Adoptoren) untersucht [Roge03]:

- Innovatoren
- frühe Übernehmer
- frühe Mehrheit
- späte Mehrheit
- Nachzügler

Die Adoptoren lassen sich innerhalb eines Lebenszyklusmodells eines fiktiven Produktes nach zeitlichen Phasen voneinander abgrenzen (siehe Abb. 2.20).

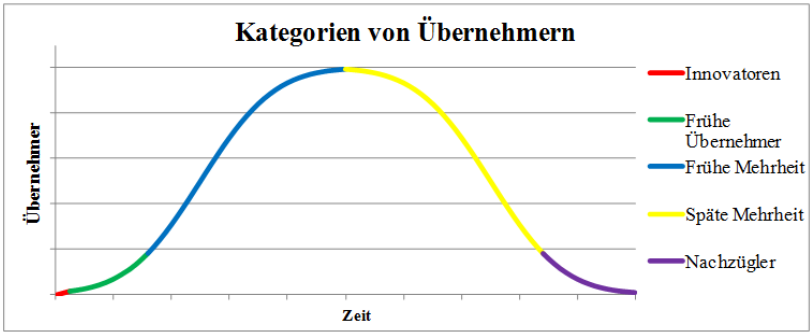


Abb. 2.20: Produktlebenszyklus mit Adoptoren, nach [Roge03, Fig. 7-3]

An der Gruppe der „Frühen Übernehmer“ (siehe Abb. 2.20) lässt sich erkennen, dass die Gruppe der „Frühen Mehrheit“ mit der „Späten Mehrheit“ die anderen Nutzergruppen überragen. Auch von der zeitlichen Dimension her behaupten sich diese Endbenutzergruppen (siehe Abb. 2.20). [Roge03, S. 282ff] bezeichnet den Punkt, an dem die „frühe Mehrheit“ von den „Frühen Übernehmern“ den Erwerb des Produktes übernehmen als „kritischen Punkt“ beim Vertrieb des Produktes. Ob das Produkt erfolgreich ist, wird an diesem „kritischen Punkt“ entschieden [Roge03, S. 282ff]. „Innovatoren“ besitzen ein überdurchschnittliches Grundeinkommen und haben Lust auf Innovation. Die „frühen Übernehmer“ unterscheiden sich durch das niedrigere Grundeinkommen (pro Kopf) gegenüber den „Innovatoren“. Die „Nachzügler“ sind meist ältere Menschen mit niedrigerem Grundeinkommen. Sie zeichnen sich durch Traditionsbewusstsein aus und distanzieren sich von Technik [Roge03, S. 282ff].

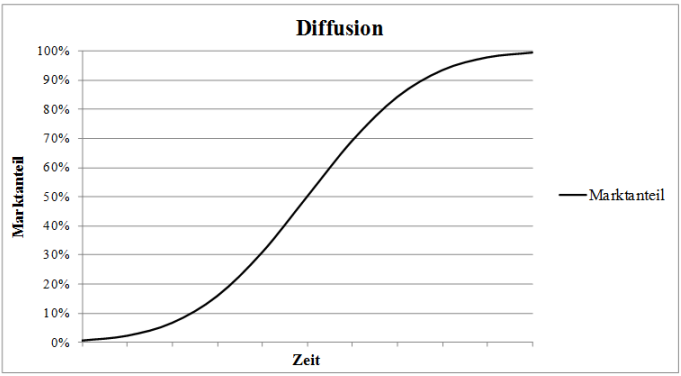


Abb. 2.21: Diffusionsmodell nach [Roge03]

Der Produktlebenszyklus (siehe Abb. 2.20) lässt sich zusätzlich kumuliert abbilden, um die Marktanteile genauer zu betrachten (siehe Abb. 2.21). [Roge03, S. 282ff] identifiziert folgende Eigenschaften der Adoptoren:

- **Innovatoren:** obere Gesellschaftsschicht, jung, weltoffen, gute Ausbildung, finanziell flexibel.
- **Frühe Übernehmer:** breite Gesellschaftsschicht, lokal orientiert, starke soziale Bindungen.
- **Frühe Mehrheit:** obere Mittelschicht.
- **Späte Mehrheit:** untere Gesellschaftsschicht, begrenzte finanzielle Mittel.
- **Nachzügler:** untere Gesellschaftsschicht, wenig finanzielle Mittel, älter, kein enges soziales Umfeld.

In [Roge03, S. 263-267] werden fünf verschiedene Faktoren identifiziert, die bei der Diffusion von Innovationen entscheidend sind:

1. relativer, subjektiver Vorteil gegenüber anderen Technologien („Perceived Usefulness“).

2. passend zu Erfahrungen und Wertschätzung des Endbenutzers

3. Komplexität des Produktes bei der Anwendung („Perceived Ease of Use“)

4. Möglichkeit des Ausprobierens

5. Wahrnehmbarkeit im sozialen Umfeld

TAM und Diffusionsmodell zur Messung der technologischen Innovationsbereitschaft wurden erfolgreich mit Technologieprodukten getestet. Modelle, die Nutzergruppen innerhalb von mobiler IKT untersuchen, werden in Anhang B betrachtet.

2.9 Entwurf einer Softwarearchitektur

Eine Softwarearchitektur bzw. Softwaressysteme werden aus Sicht der unterschiedlichen *Stakeholder* dargestellt [Kruch95]. Das *„4+1 view model“* beschreibt die verschiedenen Sichten (siehe Abb. 2.22).

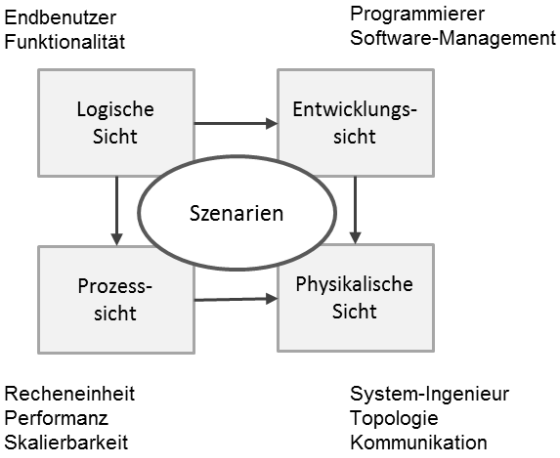


Abb. 2.22: Das 4+1-Architekturmodell nach [Kruch95]

Die Sichten werden jeweils abhängig vom Standpunkt der *Stakeholder* beschrieben. Das unterstützende Anwendungsfallbeispiel bzw. die Szenarien werden in Kapitel 6.1 eingeführt. Die einzelnen Komponenten (siehe Abb. 2.22) werden in folgender Auflistung erläutert [Kruch95]:

- ***Logische Sicht:*** In dieser Sicht werden die Funktionalität und die Endbenutzerunterstützung dargestellt. Dazu werden verschiedene Diagrammarten aus der UML genutzt: Klassendiagramme, Kommunikationsdiagramme, Verteilungsdiagramme und Sequenzdiagramme.

- ***Entwicklungssicht:*** Sicht des Softwareentwicklers auf eine Software-Architektur. Diese Sicht wird mit Hilfe von UML-Komponentendiagrammen erstellt.

- ***Prozesssicht:*** Die Prozesssicht behandelt den dynamischen Charakter eines Softwaresystems (Kommunikation und Systemverhalten). Solche Ablaufdiagramme werden mit UML-Aktivitätsdiagrammen abgebildet.

- ***Physikalische Sicht:*** Die physikalische Architekturansicht betrachtet das System aus Sicht eines Entwicklers, der das System auf entsprechender Hardware umsetzen soll. Daher spielen die physischen Leitungen bzw. Verbindungen eine besondere Rolle.

- ***Szenarien:*** Die Softwarearchitektur wird mit Hilfe von unterstützenden Anwendungsfällen erklärt. Diese als Szenarien bezeichneten Abläufe sind Anwendungsfallbeispiele der Architekturbeschreibung. Die Szenarien dienen als Startpunkt für eine prototypische Implementierung. Szenarien werden mit „Use Case“-Diagrammen beschrieben.

3 Stand der Forschung

Die Dienste und Produkte, die für den Kunden erbracht werden, stehen im Fokus beim Geschäftsprozessmanagement. Die Geschäftsprozesse werden analysiert, dokumentiert und diskutiert, damit die Wertschöpfung effizient durchgeführt wird. Für die Dokumentation der Prozesse werden Prozessmodelle verwendet.

Unternehmen, die keine WfM-basierte Lösung einsetzen, haben mit vielen Nachteilen zu kämpfen [Müll05]: Lange Durchlaufzeiten von Geschäftsprozessen kommen meist durch lange Wege- und Liegezeiten zustande. Oft werden Aufgaben bzw. Aktivitäten von Mitarbeitern vergessen oder Fristen verstreichen. Teilweise fehlen die notwendigen Informationen aus den verschiedenen Abteilungen, um einen Überblick überhaupt zu ermöglichen. Geschäftsprozesse ohne jegliche Abbildung erschweren die Identifizierung von Fehlern im Nachhinein (Nachvollziehbarkeit). Ineffizientes Arbeiten durch die Mitarbeiter kann nicht nachvollzogen werden, so dass ein Controlling nicht möglich ist. Kapazitätsengpässe können in anderen Bereichen entstehen (betriebliche Ressourcen), ohne dass zeitnah darauf reagiert wird. Für bestimmte Aufgaben kommt es zu langen Einarbeitungszeiten, da keine Dokumentationen vorliegen und die Kenntnisse nur mündlich und unvollständig weitergegeben werden.

Durch den Einsatz von WfMS im Unternehmen können die Durchlaufzeiten der Geschäftsprozesse beschleunigt werden. Durch die Abbildung innerhalb eines Prozessmodells werden die (internen) Vorgänge für alle Beteiligten transparenter. Mit der Messung der Durchlaufzeiten kann z. B. auf Ausnahmen (langlaufende Prozessinstanzen) besser reagiert werden. Die entsprechenden Aufgaben werden durch die WfMS ablaufbezogen unterstützt und eine Steuerung ist möglich, so dass z. B. entsprechende Akteure gezielt geführt werden können. Es wird ein dokumentenorientiertes Arbeiten (Dokumentenmanagement) möglich. Eine Soll-Ist-Analyse zeigt Schwachstellen auf, deren Überwindung zu Effizienzsteigerungen im Prozess führt.

Nachdem die Vor- und Nachteile von WfMS dargestellt wurden, stellt sich im Kontext der vorliegenden Arbeit die Frage, inwiefern die mobilen Kontextinformationen in das WfMS integriert werden können. In Kapitel 3.1 wird der Lebenszyklus eines Workflows betrachtet, um mit den Erkenntnissen aus Kapitel 2.1 zu analysieren, welche Arbeiten im Umfeld mobiler Kontextinformationen und den WfMS veröffentlicht wurden. Diese Arbeiten werden kritisch gewürdigt (siehe Kapitel 3.2), um den Handlungsrahmen der vorliegenden Arbeit zu definieren.

3.1 Workflow-Lebenszyklus

Workflows entstehen aus modellierten Geschäftsprozessen (siehe Kapitel 2.4). Nicht alle Geschäftsprozesse lassen sich jedoch innerhalb eines Workflows (bzw. durch ein WfMS) automatisieren. Zwischen der Modellierung und der Prozesssteuerung soll mit dem WfMS eine enge Verzahnung erreicht werden, um z. B. mit Hilfe einer Reorganisation des Modells eine Effizienzsteigerung in der Steuerung zu erzielen (siehe Abb. 3.1) [GrKe95, S. 211-245]. Dafür werden sogenannte Regelsteuerungen betrachtet [Müll05].

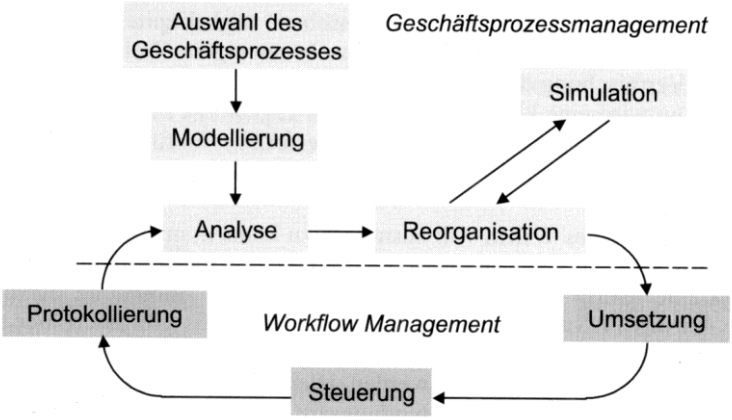


Abb. 3.1: Workflow-Regelkreis nach [Müll05]

Ziel des in Abb. 3.1 dargestellten Regelkreises ist die Messung von Laufzeitdaten im WfMS, um Schwachstellen zu identifizieren bzw. diese zu beseitigen und den Workflow dahingehend zu verbessern (siehe Abb. 3.1). Mit dieser Reorganisation werden die Workflow-Modelle verbessert [GrKe95, S. 211-245]. Nach der Reorganisation des Workflow-Modells wird dieser mittels Simulation validiert, bevor er implementiert wird.

Die Reorganisation erfolgt innerhalb der Modellierungsphase, indem das Modell umgestaltet wird. Der Regelkreis von [Müll05, S. 29] wurde erweitert, damit die Modellierung mit der Reorganisation vereint werden kann (siehe Abb. 3.2). Über die Auswahl eines Geschäftsprozesses beginnt der Regelkreis. Ein Modell kann z. B. vollständig neu entworfen werden. Durch eine Werkzeugunterstützung bei der Auswahl von Geschäftsprozessen kann ein Effizienzvorteil gegenüber einer manuellen Auswahl (bzw. einer Neuerstellung) erzielt werden [Kosc07]. Vorausgesetzt, es existieren zur Problemstellung bereits ähnliche Prozessmodelle. Die Wiederverwendung von Prozessmodellen ist ein weiterer Aspekt bei der Geschäftsprozessmodellierung (siehe Kapitel 7.1).

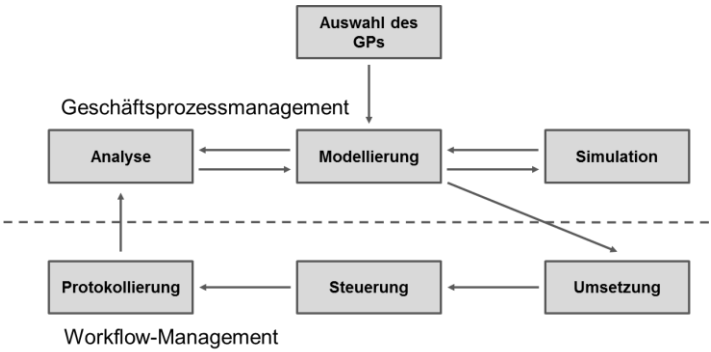


Abb. 3.2: Workflow-Regelkreis

Das Workflow-Management und das Geschäftsprozessmanagement (vgl. Abb. 3.1 und Abb. 3.2) werden voneinander getrennt betrachtet. Beide Disziplinen hängen voneinander ab.

In Abb. 3.3 ist das Workflow-Lebenszyklusmodell dargestellt [Roll98]. Das im Projekt MOVE (*Improvement of business processes with flexible workflow management systems*) entwickelte Lebenszyklusmodell wird in drei Segmente untergliedert: *Technology Design*, *Organizational Development* und *Employee Orientation*. Mit der Analyse und der Modellierung beginnt der Lebenszyklus. Die Implementierung folgt im nächsten Schritt, während mit *Application* die Nutzersicht auf das WfMS dargestellt wird. Mit der Evaluationsphase schließt der Kreis und führt zurück zur Modellierung. Der Lebenszyklus beginnt von vorne. Die drei Segmente in Abb. 3.3 demonstrieren die jeweilige Ausrichtung am Design, am Kunden bzw. an der Organisation. Im Lebenszyklusmodell (siehe Abb. 3.3) wird mit dem Begriff der Implementierung deutlich, dass eine explizite Implementierung (eine spezielle Applikation) durchgeführt wird.

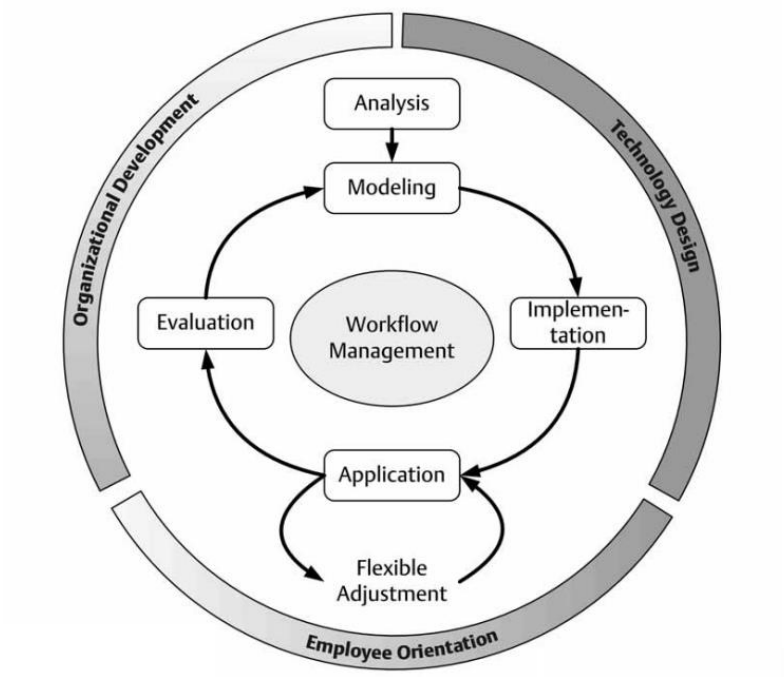


Abb. 3.3: Workflow-Lebenszyklusmodell [Roll98, S. 128]

Im Lebenszyklusmodell nach [Roll98] wird zwischen Geschäftsprozess und Workflow nicht unterschieden. Es wird von der Analyse direkt zur Modellierung und zur Implementierung übergeleitet, womit eine enge Verzahnung der Modellierung mit der Implementierung verlangt wird.

Im Lebenszyklusmodell (siehe Abb. 3.4) wird das Geschäftsprozessmanagement strikt vom Workflowmanagement in zwei verschiedene Zyklen getrennt [Gada01]. Es wird mit der Geschäftsprozessmodellierung begonnen, die zur Phase der Geschäftsprozessanalyse führt (vgl. Abb. 3.4). Die Geschäftsstrategie wird entwickelt und der Prozess wird ggfs. rekonstruiert, was in dem Fall einen Rücksprung hin zur Geschäftsprozessmodellierung bedeutet. Nach der Geschäftsprozessanalyse werden in einem weiteren Schritt die Workflows modelliert bzw. in eine textuelle Form (zur späteren Ausführung) übersetzt. In diesem kleinen Zyklus (siehe Abb. 3.4) wird der Workflow simuliert und anschließend „optimiert“ [Gada01]. Nach der Phase der Workflow-Modellierung werden die Workflows ausgeführt und in einem letzten Schritt überwacht mittels Monitoring. Danach beginnt der Lebenszyklus wieder von vorne, indem zur Geschäftsprozessmodellierung übergeleitet wird. Laut [Gada01] wird mit dem Geschäftsprozesszyklus die fachlich-konzeptionelle Gestaltung vorangetrieben, während der zweite Workflow-Zyklus die operative Ebene verfeinern soll. Dieser Lebenszyklus trennt somit strikt maschinenausführbare Workflows von Geschäftsprozessen aus konzeptioneller Sicht. Das Modell verlangt die Trennung zwischen fachlichen und technischen Modellen. Mit Hilfe von ausführbaren Prozessen kann das Problem der fachlichen und technischen Sicht gelöst werden. Die fachliche Sicht wird mittels Aktivitäten modelliert, die technisch verfeinert werden können. Die technischen Feinheiten werden in Folge in den jeweiligen Verfeinerungen ausmodelliert.

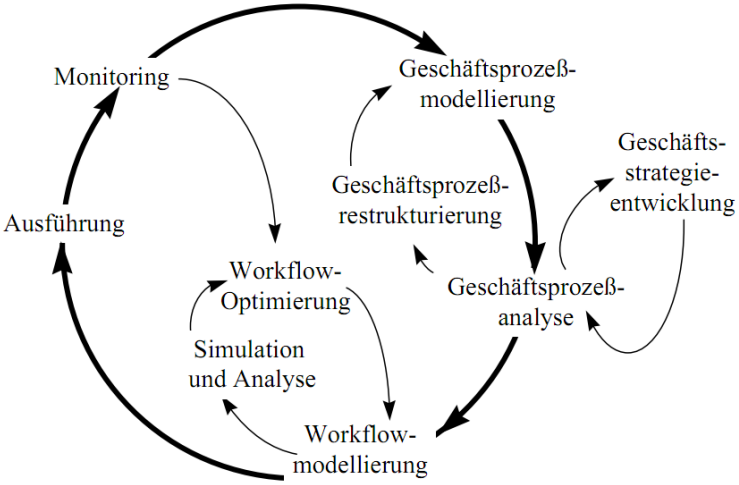


Abb. 3.4: Workflow-Lebenszyklusmodell nach [Gada01]

Zur Mühlen [Mueh02] definiert ein Workflow-Lebenszyklusmodell in Management-orientiertem Kontext (siehe Abb. 3.5). Der Lebenszyklus beginnt, wie in den bereits vorgestellten Modellen, mit der Analyse, die sich feingranular mit Projektzielen, mit der zukünftigen Workflow-Umgebung, der Organisation und den Regeln des Systems auseinandersetzt. Danach beginnt die Designphase. Es werden die Prozessstruktur, das Prozessmodell und die involvierten Ressourcen identifiziert, die während der Prozessausführung berücksichtigt werden. Die Prozessmodelle, wie in Abb. 3.5 an der Eingangskante zu *Processs Implementation* zu sehen, sind Eingangsvariablen für die Prozessimplementierung. In der *Process Enactment*-Phase werden verschiedene Prozessinstanzen auf Basis des Prozessmodells gestartet und durch die Prozessinfrastruktur koordiniert. Dies können z. B. *Notifications* an Endbenutzer sein oder Nachrichten über nicht verfügbare Ressourcen während der Prozessausführung. Daher soll das Prozess-Monitoring zur gleichen Zeit stattfinden (siehe Abb. 3.5). Mit der Prozessevaluation schließt der Lebenszyklus ab. In dieser Phase wird die ausgeführte Instanz mit der Simulation verglichen. Die Abweichungen werden für die Überarbeitung bzw. Verbesserungen der Prozessstruktur verwendet.

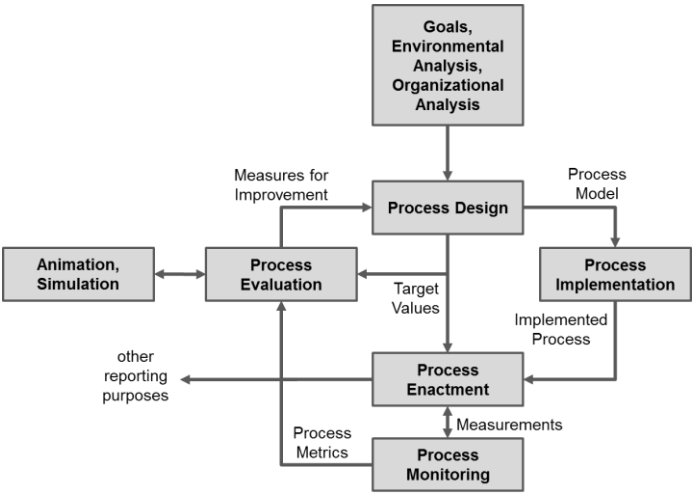


Abb. 3.5: Workflow Lebenszyklusmodell nach [Mueh02]

Fazit

In weiteren Publikationen über Lebenszyklusmodelle des Geschäftsprozessmanagements respektive Workflow-Managements [Heil94, Heil96, Gall97, GaSc95, StDe95, NePW02] konnten keine wesentlichen Unterschiede zu den bereits vorgestellten Modellen erkannt werden. Da eine weitere Analyse dieser Lebenszyklusmodelle den Rahmen der vorliegenden Arbeit sprengen würde, wurden die drei Modelle exemplarisch vorgestellt.

Die drei vorgestellten Lebenszyklusmodelle sind inhaltlich ähnlich [Gada01, Mueh02, Roll98], jedoch nicht gleich. Die Module innerhalb der Modelle sind untereinander vernetzt. Folgende einzelne Komponenten werden deutlich erkennbar: Modellierungskomponente (siehe Abb. 3.1) oder das Monitoring (in verschiedenen Komponenten, siehe Abb. 3.5). Für den weiteren Verlauf der Arbeit wird ein Lebenszyklusmodell festgelegt, das analog zum vorgestellten Modell von [Roll98] allerdings im serviceorientierten Kontext agieren soll (siehe Abb. 3.3). D.h. die Implementierung erfordert eine serviceorientierte Architektur.

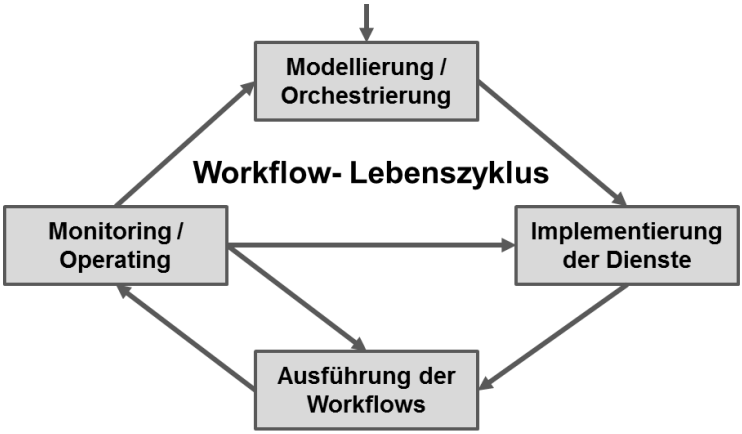


Abb. 3.6: Lebenszyklus eines Workflows

Im Lebenszyklusmodell (siehe Abb. 3.6) soll, analog zu [Mueh02], ein Monitoring und Operating durchgeführt werden (siehe Abb. 3.5). Das Operating überwacht die laufenden Prozessinstanzen ständig und kann aktiv die Prozesse (z. B. in neue Zustände oder Retry eines Service-Calls) überführen, während das Monitoring eher statistische Analysen bereitstellt und langfristige Änderungen am Prozessmodell beeinflusst. Je früher Fehler erkannt werden, desto geringer sind die Kosten für die Nacharbeit [BNST08]. Im Fokus der Betrachtung sind die Kosten bzw. ist der Handlungsspielraum des Systems, das den Workflow ausführt (vgl. WfMS). Ein WfMS kann mit einem relativ großen Handlungsspielraum bzgl. der Funktionalitäten entworfen werden. Sobald das Modell entworfen wurde und die technischen Details festgelegt wurden, ist der Handlungsraum der Funktionalitäten kleiner geworden [BNST08]. Gleichzeitig steigen die Kosten für das System an, wenn das System in Betrieb ist (in der Anwendung) werden Änderungen aufwändig. In vorliegender Arbeit wird der Workflow-Lebenszyklus mit der Modellierung bzw. der Orchestrierung der Dienste in einem Top-Down-Ansatz initiiert (siehe Abb. 3.6). Nach dem (Grob-)Entwurf müssen die darunterliegenden Dienste (z. B. Webservices) implementiert werden. Die Prozesse werden innerhalb einer Workflow-Engine ausgeführt und mit Hilfe des Operating bzw. Monitoring überwacht. Der Unterschied zwischen Operating und

Monitoring liegt in den unterschiedlichen Aufgabenbereichen. Während das Operating den Betrieb der Prozessinstanzen im Fokus hat, konzentriert sich das Monitoring auf die Auswertung und Darstellung der *Key Performance Indicator*[1].

Mobilität kann in jedem Teil eines WfMS vorhanden sein bzw. berücksichtigt werden. Zur weiteren Eingrenzung der vorliegenden Arbeit soll möglichst früh im Workflow-Lebenszyklus mit der Integration mobiler Kontextinformationen begonnen werden, um Mehrkosten durch Änderungen im Betrieb (siehe oben) entgegenzuwirken. Daher wird sich die vorliegende Arbeit mit dem Entwurf eines mobilen WfMS innerhalb einer SOA und der Modellierung/ Orchestrierung von Workflows beschäftigen. Weitere Vorteile der frühen Berücksichtigung der mobilen Aspekte sind: eine präzisere informationstechnische Integration und die Realisierung von Sicherheitsaspekten.

Es stellt sich im weiteren Verlauf der vorliegenden Arbeit folgende Herausforderung: Wie können mobile Kontextinformationen in Bezug auf die Analyse und Entwurf eines WfMS bzw. in der Modellierung und in Bezug auf die letztendliche Ausführung berücksichtigt und so früh wie möglich integriert werden, um einen relativ großen Handlungsraum zu erzielen und die Kosten niedrig zu halten?

3.2 State of the Art: Workflows unter Berücksichtigung mobiler Kontextinformationen

Die Literaturrecherche zum Thema der Workflows bzw. mobiler WfMS soll zeigen, inwiefern beim Entwurf und der Ausführung von Workflows *Context Awareness* (siehe Kapitel 2.2) berücksichtigt werden kann.

[1] Key Performance Indicator (KPI) sind betriebswirtschaftliche Kennzahlen, die aus Messungen der Prozessinstanzen (Daten aus dem Prozesskontext) gewonnen werden können.

In der wissenschaftlichen Literatur gibt es zahlreiche Veröffentlichungen im Umfeld von WfMS, die speziell dafür erstellt wurden, mobile Clients bzw. Mobilität zu unterstützen [AGKA95, AGKA96, Busb94, Buss95, Bolc00, CBLT06, DMPD99, DSBG06, GPWK07, HHGR06, JHHS00, JHSH99, KuZL06, KuZL07, MaMo04, MuWL00, PaCh07, PaPC97, PCDG02, RoHS06, SHHR07, StKn01, ZaBV09, Zuku97, Zuku99]. Allerdings konnte keine Veröffentlichung gefunden werden, die schon beim Prozessdesign mobile Kontextinformationen berücksichtigt und diese in irgendeiner Form in der verwendeten Modellierungssprache abbildet. Daher werden im Folgenden wissenschaftliche Arbeiten vorgestellt, die sich mit *Context Aware Services* (CAS) bzw. Mobilität im Rahmen von WfMS beschäftigt haben.

Die klassischen Systeme werden Publikationen aus den Jahren bis 2001 zugeordnet (siehe Kapitel 3.2.1). Moderneren WfMS (der zweiten Generation) werden dem Zeitraum von 2002 bis 2016 zugeschrieben (siehe Kapitel 3.2.2). In Kapitel 3.2.3 wird eine Methode vorgestellt, die mobilen Anteile innerhalb eines Geschäftsprozesses zu identifizieren, was als Vorarbeit innerhalb des WfM notwendig ist. Die Inhalte der beiden Kapitel 3.2.1 und 3.2.2 basieren teilweise auf [DKKO09].

3.2.1 Klassische mobile WfMS (erste Generation)

Die erste Generation mobiler WfMS werden wissenschaftlichen Publikationen bis 2001 zugeschrieben und als „klassische Systeme“ bezeichnet. Die mobil-spezifischen Aspekte der WfMS bzw. mobiler IKT – auch wenn heutzutage nativ durch das Betriebssystem bzw. das Kommunikationsprotokoll unterstützt – sind für den weiteren Verlauf der Arbeit interessant.

Exotica/FMDC ist ein Beispiel für ein WfMS, bei dem ein konventionelles WfMS (IBM Flowmark) um mobile Clients für die mobilen Nutzer erweitert wurde [AGKA96]. Änderungen an bestehenden Komponenten des Systems werden vermieden. Die Erweiterung fokussiert auf die Unterstützung geplanter Verbindungsabbrüche der mobilen Clients. Hierzu wird ein entsprechendes Protokoll beschrieben, welches es ermöglicht, dass Akteure Aktivitäten sperren und herunterladen können, bevor sie die drahtlose Verbindung

abbauen. Ungeplante Verbindungsabbrüche (Funklöcher) werden nicht berücksichtigt. In [AGKA95] wurden spezielle Begriffe wie "Mobiler Prozess", "Mobiler Workflow" und "Mobiles WfMS" definiert. Das WfMS „FlowMark" von IBM ist um „mobile Clients" erweitert worden. Das System wurde Client-seitig so gestaltet, dass der Client auch bei Verbindungsabbrüchen weiterarbeiten kann. Eine Sperrung von Aktivitäten ist notwendig, da im Falle einer noch nicht vollständig erfolgten Zuteilung von Aktivitäten (z. B. bei einem Verbindungsabbruch) der Zustand im Client nicht eindeutig ist und nicht in Erfahrung gebracht werden kann. Manchmal, so die Autoren, kann es vorkommen, dass eine Aktivität Anwendungen braucht, die nicht lokal zur Verfügung stehen. Ein spezieller Mechanismus schaltet zeitabhängig Sperren (engl. *locks*) ab, so dass das System bzw. die Aktivität weiterarbeiten kann.

Jing et al. beschreiben das **WHAM-System** [JHHS00]. Als mobil-spezifische Erweiterungen sieht dieses System insbesondere eine Worklist-Darstellung vor, bei der die zu erledigenden Aktivitäten in einer geographischen Karte eingetragen werden. So kann ein mobiler Akteur sich bei seiner Wahl für die nächste zu bearbeitende Aktivität danach richten, in welcher Entfernung die entsprechenden Orte liegen. WHAM unterscheidet zwei aufeinander aufbauende Formen der Zuordnung der anstehenden Aktivitäten zu den Akteuren: das „Global Resource Assignment" wird auf der zentralen Server-Infrastruktur durchgeführt und berücksichtigt auch den Aufenthaltsort des mobilen Akteurs. Die „Resource Assignment" wird auf dem mobilen Endgerät des Akteurs lokal ausgeführt: es priorisiert die in der Worklist aufgeführten Themen nach Ressourcen, z. B. die verfügbaren Ausrüstungsgegenstände, Verbrauchsmaterialien wie Ersatzteile oder persönlichen Präferenzen des Akteurs. WHAM unterscheidet zwei Arten der Ressourcen-Zuordnung: bei der optimistischen Zuordnung kann einem mobilen Akteur auch dann eine Aktivität zugeordnet werden, wenn er gerade nicht mit dem System verbunden ist, es also nicht bekannt ist, ob er sich tatsächlich in der Nähe des Ortes befindet, an der die entsprechende Aufgabe abzuarbeiten ist. Die optimistische Zuordnung wird lediglich für „steady" eingeordnete Akteure angewendet, also für Akteure, die sich innerhalb einer bestimmten Zeitspanne nicht über ein bestimmtes Gebiet hinausbewegt haben. Ist dies

nicht der Fall, wird der Akteur als „unsteady“ eingeordnet, so dass er nur dann Aktivitäten zugeordnet bekommt, wenn er gerade mit dem System verbunden ist. Sein tatsächlicher Aufenthaltsort ist bekannt. Wie ein Gebiet berechnet wird, anhand dessen entschieden werden kann – ob der Akteur „steady“ oder „unsteady“ ist – wird von den Autoren von WHAM nicht näher erörtert. WHAM kann Inhaltsdaten der Workflows (z. B. Bilder) in der Größe reduzieren (z. B. durch Kompressionsverfahren). Diese Funktion zielt vorrangig auf Bitmap-Daten wie eingescannte Dokumente oder Fotografien ab. In [JHSH99] wird erstmals eine *mobile Workflow Application* eingeführt und definiert. Die mobil ausgeführte Applikation greift auf lokale Ressourcen (Sensoren) zu. Gleichzeitig werden in [JHSH99] Prioritäten von Aufgaben (für das spätere WHAM-System) eingeführt und definiert.

Domingos et al. [DMPD99] stellen ein WfMS vor, bei dem die mobilen Endgeräte über Messaging-Kanäle mit der stationären Infrastruktur verbunden sind. Messaging-Kanäle bieten asynchrone Kommunikation, die sich besser als das herkömmliche „Request-Response“-Muster für mobile Anwendungsfälle eignen, da das Gesamtsystem so unempfindlicher in Bezug auf unvorhergesehene Kommunikationsabbrüche („Funklöcher“) wird. Das System unterstützt zwei grundlegende Arten der Zuordnung von Aktivitäten zu Akteuren, namentlich das „Offene Verfahren“ und das „Geschlossene Verfahren“. Ersteres ist für eine zuverlässige Kommunikationsverbindung zwischen mobilen Clients und Backend vorgesehen; dies ist z. B. dann gegeben, wenn der Akteur sich während der morgendlichen Einsatzbesprechung im Hauptquartier befindet. Der „Planer“ kann einzelne Aufgaben an mobile Akteure zuweisen. Die damit verbundene Sperre verfällt, wenn der mobile Client nicht in bestimmten Zeitabständen ein „Keep-Alive“-Signal übermittelt. Das „Offene Verfahren“ geht davon aus, dass eine neue Aktivität (z. B. ein dringender Reparatureinsatz) einem Akteur zuzuordnen ist, während dieser sich schon im Einsatz (unterwegs) befindet. Ein spezieller Messaging-Kanal steht bereit, über den der Planer und die Akteure aushandeln können, wem diese Aktivität zugeordnet wird. Dieses Verfahren sieht vor, dass die Aktivität an den Akteur vergeben wird, der zuerst eine Sperre anfordert. Das WfMS unterstützt Caching von Objekten auf dem mobilen

Endgerät. Dabei kann es zwei Versionen von Datenobjekten geben: die aktuelle Version und die zuletzt Bekannte (auf der zentralen Wf-Engine). Weiter existieren sogenannte *„revocable locks"*, die periodisch vom Team bestätigt werden müssen, damit sie nicht gelöscht werden. Wenn solche *locks* wieder freigegeben werden, muss ein *reassignment* die Aufgaben wieder neu zuordnen. In [DMPD99] wird in Ausnahmefällen bei schon zugeordneten Aufgaben ein *reassignment* ermöglicht.

Stormer & Knorr [StKn01] beschreiben ein WfMS namens „**AWA/PDA"**, das sich teilweise der Agenten-Technologie bedient, um den besonderen Anforderungen mobiler Anwendungsfälle gerecht zu werden. Unter Software-Agenten versteht man Software-Module [Huhn03], die bis zu einem gewissen Grad autonom zur Erreichung eines bestimmten Zieles handeln und hierfür untereinander Nachrichten austauschen. Bei mobilen Software-Agenten liegt insbesondere Code-Mobilität vor, d. h. eine Agenten-Instanz kann während der Ausführung auf verschiedenen physikalischen Rechnern verschoben werden. AWA/PDA basiert auf mehreren Typen solcher mobiler Agenten: eine laufende Workflow-Instanz wird durch einen Workflow-Agenten repräsentiert; für die Beschreibung des zugehörigen Workflow-Schemas ist der Prozess-Agent zuständig. Für jede durch einen Akteur abzuarbeitende Aktivität innerhalb eines Workflows gibt es einen Task-Agenten, der alle hierfür benötigten Daten (z. B. Formulare und Dokumente) kapselt. Die Zuordnung von Aktivitäten an einzelne Akteure wird durch den Worklist-Agenten vorgenommen. Der Task-Agent kann auf ein mobiles Endgerät migrieren, so dass die jeweilige Aktivität auf dem Endgerät autark ausgeführt wird. Auf dem Endgerät selbst läuft ein Personal-Agent, der allerdings kein mobiler Agent ist. Seine Aufgabe besteht in der Unterstützung des Nutzers: z. B. die Abfrage von Formulareingaben über die graphische Benutzeroberfläche. Dieses Verfahren zeigt den Unterschied der Agententechnologie gegenüber „echter" mobiler Systeme. Da bei den Agenten-basierten Systemen beim Systemverhalten mit Wahrscheinlichkeitsmodellen gearbeitet wird, kann das Systemverhalten nur abgeschätzt werden. Bei einem physischen mobilen System ist es möglich eine konkrete Aussage über die Lauffähigkeit des Systems zu treffen.

[Buss95] definiert „mobile Worklists“ und führt ausführlich in die mobilspezifischen Anforderungen der mobilen WfMS ein: Für mobile Endgeräte sollen immer ganze Arbeitspakete, also eine Gruppe von Aufgaben (nicht nur einzelne Tasks), zur Verfügung gestellt werden, damit eine autarke Arbeit möglich wird auch wenn sich der Endbenutzer in Gebieten bzw. Räumen ohne Netzempfang aufhält. Der Status *disconnected* muss im WfMS berücksichtigt werden, damit geplante und ungeplante Verbindungsabbrüche beachtet werden. Zur Berücksichtigung von *Deadlines* müssen die WfMS erweitert werden, damit z. B. Tasks an andere Mitarbeiter weitergegeben werden, bevor die Deadline verstrichen ist. Erinnerungen des Endbenutzers bzw. eine spezielle Aufbereitung nach einem entsprechenden Abarbeitungsschema der Aufgaben müssen integriert werden. Im Falle eines Verbindungsabbruchs sollen die Tasks, die der Client abarbeiten soll, automatisch bei den anderen mobilen Clients ausgeblendet werden, um eine doppelte Ausführung der Tasks zu verhindern. Eine weitere Anforderung in [Buss95] beschreibt einen *Installer*, der auf jedem mobilen Client für die Installation von Anwendungen vorhanden sein muss. Diese Anforderung ist in den meisten heutigen Systemen mit den App-Stores bereits erfüllt bzw. das ist eine etablierte Technik für den Nutzer, um die Auswahl und Installation zu erleichtern und die Bezahlung zu vereinfachen. Zusätzlich wird in [Buss95] ein *Minimal Data Set* gefordert. Die Datenmenge des *Minimal Data Set* muss berechnet werden, damit die Art der notwendigen Netzwerkverbindung überprüft werden kann und der Task an einen Endbenutzer mit genau diesen Voraussetzungen weitergeleitet wird. Daten die von vielen Endbenutzern gleichzeitig verwendet werden, müssen markiert werden, damit sie nicht heruntergeladen und für andere Endbenutzer gesperrt werden. Sonst würden sich die Endbenutzer gegenseitig blockieren. Der Verlauf bzw. die Spur der Daten (engl. *trace*) muss stets verfolgt bzw. dokumentiert werden, damit nachvollziehbar ist, an welchen Stellen z. B. Daten liegen geblieben sind. Falls es später zu Problemen im WfMS kommt, müssen die Ursachen gefunden und aufgelöst werden. Der Workflow-Designer (Modellierer) kann z. B. entscheiden, welche Schritte mobil ausgeführt werden und in welcher Form. Eine weitere Eigenschaft des „mobilen WfMS“ [Buss95] ist, dass ein Task von der Worklist im mobilen Client verborgen wird (Zustand unsichtbar), sofern er im „connected“-Status bearbeitet wird. Bei jedem Task muss somit

eindeutig hinterlegt werden, an welchem Ort er ausgeführt werden kann. Diese Information muss assoziiert mit dem Task und dem Nutzer (z. B. innerhalb einer Kartendarstellung) verfügbar gemacht werden. Rein technisch beinhaltet das WfMS Datencontainer, die alle Informationen beinhalten, die für die mobile Durchführung einer Aktivität notwendig sind. Die Datencontainer unterstützen die in [Buss95] definierten Anforderungen (z. B.: *App Description, Step Description, Deadline Description* und *Data Description*). Das in [Buss95] vorgestellte System wird „**MOBILE**" genannt. Allerdings ist dies irreführend, da damit nicht ein mobiles System beschrieben wird, sondern die Tatsache, dass eine im Prozess existierende Aktivität mobil ist. Formal wird eine Backus-Naur-Form-Grammatik (BNF) vorgestellt, mit der Workflows beschrieben werden. Bei sogenannten mobilen Workflows [Buss95] wird diese Grammatik um einige zusätzliche Angaben ergänzt, wie etwa das Min-Data-Set und „*Reconciliation"* (Move, Lock, Copy – Sync). Die Operationen (Move, Lock, Copy und Sync) werden verwendet, um eine Abstimmung der Tasks zu erreichen (z. B. kann mit Lock eine Sperre gesetzt werden, so dass kein anderer Kollege diesen Task gleichzeitig (redundant) bearbeiten kann).

Ein weiteres mobil-orientiertes WfMS wird in [Busb94] vorgestellt, bei dem einzelne Aufgaben an bestimmte Akteure (Endbenutzer) geroutet werden sollen. In dieser Arbeit wird der Aspekt der Kollaboration innerhalb der sogenannten „Informationsräume" vertieft. Beispielsweise sollen Angebote von mobilen Handelsvertretern unterwegs (mobil) erstellt werden. Es wird vorausgesetzt, dass Informationen über momentan verfügbare Produktionsdaten bzw. Verhandlungen mit Verkäufern über bestimmte Kontingente innerhalb des Systems bereitstehen. Eine mobil-spezifische Anforderung an das WfMS ist eine effektive Datenkompression (hoher Kompressionsgrad), um möglichst wenige Daten an das mobile Endgerät transferieren zu müssen. Es werden modularisierte Datenmengen (sogenannte „Datenuntermengen") zur logischen Organisation vorausgesetzt (dezentrales System). Eine Versionierung von Datenobjekten mit sogenannten „Softlocks" und die Durchführung von Datenabgleichen, die z. B. vor Verlassen des Büros angewendet werden sollen, sind weitere Eigenschaften des Systems. Vor diesem Datenabgleich soll ein Cache im mobilen Endgerät aufgebaut werden. Optional

werden in [Busb94] Updates auf lokalen Kopien angegeben, bei der Endbenutzer entscheiden können, ob das System sich automatisch oder manuell updaten soll. Ein Update soll nur dann erfolgen, wenn der stationäre Rechner z. B. gerade weniger CPU-Last (z. B. in der Mittagspause) hat. Die Kollaborationsunterstützung [Busb94] ermöglicht z. B., Einladungen zur Bearbeitung eines Tasks an verschiedene Endbenutzer zu schicken.

In dem WfMS **RainMan** [PaPC97] werden die Endbenutzer (Worker) vom Modellierer (Admin) mit sogenannten *Applets* (JavaME[2]) bedient. Weitere Eigenschaften von RainMan sind Workflow-Schemas, die angeblich zur Laufzeit geändert werden können [PaPC97]. Eine Worklist soll über externe Anwendungen ansprechbar bzw. beeinflussbar sein. Gleichzeitig muss der Client bei mehreren WF-Servern angemeldet sein, da zusätzlich verteilte Anwendungsfälle unterstützt werden. Die Server (bzw. Engines) sollen untereinander Teil-Workflows (Teilprozesse) sogenannte *Peer-to-Peer-Workflows* austauschen können [PaPC97]. In RainMan werden verschiedene Clients unterstützt und es gibt eine klare Trennung zwischen *Workflow Routing* (der Workflow-Engine) und der *Task Execution* (Client), damit „*plug-n-play*" möglich wird [PaPC97]. Clients können selber Aufgaben ablehnen, falls sie selbst nicht fähig sind, diese zu bearbeiten. Daher existiert auch ein Directory-Service, der solche „Capabilities" führt. Die Worklist fungiert als Mediator zwischen der Engine und dem Client. Für die Implementierung wird eine *publish/ subscribe*-Kommunikation [ARMC14] empfohlen. Aktivitäten sollen an mehrere Clients geschickt werden, um z. B. Angebote einzuholen bzw. Auswertungen zu machen, während diese bearbeitet werden. Es werden in [PaPC97] drei Sicherheitsmechanismen gefordert: Ein Authentifizierungsmechanismus zwischen Workflow-Client und Server soll die Identitäten validieren. Es soll ein Zugriffsmechanismus vorhanden sein, der Zugänge auf bestimmte Methoden innerhalb des Servers einschränkt. Und zuletzt sollte die Integrität und Geheimhaltung der Tasks sichergestellt werden.

[2] http://www.oracle.com/technetwork/java/javame/index.html

In [Zuku99, Zuku97] wird das WfMS **Waterloo-M** eingeführt, das auf Basis einer Datenbank operiert, wobei zwischen sogenannten „starken“ und „schwachen“ Aktivitäten (verschiedene Prioritäten) für die unterschiedlichen Konsistenzgarantien unterschieden wird. Gleichzeitig wird zwischen „starken“ und „schwachen“ Daten unterschieden. Starke Daten müssen global konsistent vorhanden sein, wohingegen schwache Daten nur lokal verfügbar sein dürfen. Zusätzlich wird in Waterloo-M eine graphische Darstellung von schwachen bzw. starken Daten und Aktivitäten vorgeschlagen. Die Nutzung der Datenbank-Techniken wird damit begründet, dass weniger auf dem Client selbst berechnet werden muss und gleichzeitig unnötige Datenbanktransaktionen vermieden werden. Deshalb werden drei unterschiedliche Phasen charakterisiert: In der Vorbereitungsphase werden vorhersehbare Verbindungsabbrüche durch lokale Replikationen und deren Folgeaktivitäten von Daten vorbereitet. In der Bearbeitungsphase werden sogenannte „schwache Aktivitäten“ ausgeführt, wenn keine Konnektivität besteht. In der Re-Integrationsphase werden Aktivitäten überprüft (Monitoring) und ausgeführt bzw. Daten werden synchronisiert. Schließlich wird in diesem mobilen WfMS eine Replikation von Daten bzw. eine Maskierung von unterschiedlichen Netzwerkverbindungen angeboten.

[MuWL00] zeigt, dass die Abarbeitung einer Taskliste oder einer einzelnen Aktivität ein spezieller Anwendungsfall eines WfMS ist. Mobile Endgeräte werden in diesem WfMS integriert. Ein mobiler Akteur (z. B. ein Heizungsinstallateur in [MuWL00]) wird bei der Arbeit durch eine Task-basierte Lösung unterstützt. Im WfMS selbst werden auf Basis einer speziellen Art eines höheren Petri-Netzes Modelle [MuWL00] entworfen bzw. modelliert. Die Ausführung der Prozesse erfolgt über das Petri-Netz, wobei die Marken eines Petri-Netzes spezielle Java-Objekte sind. Die eigentliche Funktion von [MuWL00] stellt eine abstrakte Schicht zwischen der Präsentationslogik und der Applikation bereit, um das mobile GUI schnell und effizient ändern zu können. So wird z. B. ermöglicht, normale Webseiten darzustellen bzw. diese mit Hilfe der Zwischenschicht auf die speziellen Bedürfnisse von mobilen Displays bzw. Endgeräten anzupassen.

Das **Magi**-System wird in [Bolc00] vorgestellt. Es handelt sich um eine Architekturbeschreibung für mobile und verteilte Workflows, die mit Hilfe eines Apache[3]-Servers bzw. des HTTP-Protokolls betrieben wird. Für die mobil-spezifischen Defizite im HTTP-Protokoll werden strukturelle Lösungen angeboten. Das Magi-System selbst besteht aus einem leichtgewichtigen Apache-Webserver und aus einer generischen Schnittstelle, die für die Anbindung von Diensten jeglicher Art eingeführt wird. Einige der Ansätze von [Bolc00] sind für die damalige Zeit innovativ, da der Endbenutzer mehr in die Betrachtung einbezogen wurde, z. B. soll eine einfache Möglichkeit geschaffen werden, um vom mobilen Endgerät aus Dateien speichern zu können. Es wird die Anbindung von WebDAV[4] empfohlen [Bolc00]. In der Magi-Architektur soll diese Komponente als Modul geladen werden, so dass möglichst dynamisch und fallbezogen ein System aufgebaut werden kann. Viele der damals eingeführten Schwächen von HTTP sind allerdings heutzutage nicht mehr vorhanden, da z. B. die neueste Version HTML 5 ein Caching-Verfahren[5] anbietet, was gerade für mobile Webseiten einen Vorteil darstellt. Die Webseiten können ohne Netzverbindung (offline) genutzt werden, sofern sie einmal geladen wurden.

In [PCDG02] wird ein Workflow-basiertes System innerhalb eines Krankenhaus-Anwendungsfalls vorgestellt, das es ermöglicht, mittels PDA Aufgaben und Informationen an andere Endbenutzer zu verteilen. Die GUI basiert auf mobilen Webseiten, die vom Endgerät aus (*Windows Mobile Pocket PC*) über WLAN abgerufen werden können. Das WfMS besteht aus zwei separaten Teilen: das Patientendaten-System und der Workflow-Server. Das Patientendaten-System ist mit Hilfe einer Client-Server-Architektur implementiert worden (Java und MySQL[6]-Datenbank). Der Workflow-Server basiert

[3] Apache ist ein HTTP-basierter Webserver, der von der *Apache Software Foundation* (einer der am häufigsten verwendeten Webserver) entwickelt wird (http://www.apache.org/).

[4] WebDAV (Web-based distributed authoring and versioning) ist ein Standard zur Bereitstellung von Dateien für das Internet. Endbenutzer können auf Daten wie auf einer eingebundenen Festplatte im Betriebssystem zugreifen.

[5] http://dev.w3.org/html5/spec/Overview.html#manifests

[6] Der MySQL-Server ist ein relationales Datenbanksystem.

auf dem *Component Objekt Model*[7]-Framework (COM) und *Microsoft SQL Server Data Engine*[8] (MSDE). Die komplette Kommunikation innerhalb des WfMS wird über das HTTP-Protokoll über Webseiten bzw. REST[9]-Services abgewickelt [PCDG02]. Das System basiert insgesamt auf den von der WfMC definierten Standards, was an den Modulen (z. B. am *Workflow Designer*) zu erkennen ist. Es wird vom Ort bzw. der Position in [PCDG02] nicht abstrahiert. Eine weiterführende Verwendung der Ortsinformationen ist nicht angedacht. Lediglich die ortsunabhängige Nutzung von mobilen Endgeräten wird motiviert, damit die Workflows überall im Krankenhaus nutzbar sind. Es werden jedoch mobil-spezifische Ansätze in [PCDG02] vorgestellt: die Displaygröße wird z. B. abhängig vom Endgerät angepasst, damit jede Information je nach Gerät mit einer guten Darstellung bereitgestellt werden kann.

3.2.2 Moderne mobile WfMS (zweite Generation)

Moderne mobile WfMS sind aus Publikationen ab 2002 hervorgegangen. Sie besitzen einen speziellen Entwurf (Berücksichtigung der Modellierung) und eine auf Standards aufbauende technische Implementierung. Weitere Charakteristika sind die XML-basierte Datenhaltung (auch verwendet als Austauschformat) und die Verfolgung des SOA-Paradigmas.

Sliver [HHGR06] ist ein repräsentatives Beispiel für ein mobiles WfMS der zweiten Generation. Hackmann et al. wollen mit Sliver ein Informationssystem auf Basis von BPEL entwickeln. Drei Anforderungen an das WfMS werden identifiziert [HHGR06]: Die Engine muss möglichst effizient implementiert sein und mit wenig Speicherplatz auskommen. Die Java ME API wird technisch vorausgesetzt und eine Vielzahl an Kommunikationsprotokollen

[7] Das Component Object Model ist (COM) ist eine Plattformtechnik, die von Microsoft entwickelt wurde, um unter Windows Interprozesskommunikation und dynamische Objekterzeugung bereitzustellen.

[8] Microsoft SQL Server Data Engine ist ein relationales Datenbankmanagementsystem von Microsoft.

[9] Representational State Transfer – Das Akronym REST bezeichnet ein Programmierparadigma für Webanwendungen, das auf gewöhnliche Webseiten zurückzuführen ist.

muss unterstützt werden. Herkömmliche Implementierungen einer BPEL-Engine beruhen auf einem Java-Applikationsserver (z. B. JBoss[10]) und sind für den Einsatz auf den mobilen Endgeräten nicht geeignet. Die mobilen Betriebssysteme [HHGR06] verwenden Java ME (Mobile and Embedded Java Platform). Allerdings unterscheidet sich das Java ME von Java SE (Java Standard Edition). Java SE Programme können nicht auf der Java ME Plattform ausgeführt werden. Die Autoren haben demzufolge eine neue Architektur entworfen und umgesetzt. Eine Transportschicht tauscht Nachrichtenobjekte in Form von XML-Dokumenten aus. Die Daten werden mit Hilfe der XML-Parser- und der SOAP-Schicht in Java-Objekte konvertiert. Der BPEL-Server verwendet diese Komponenten. Zum Beispiel benutzt der BPEL-Server einen XML-Parser, um BPEL in ausführbare Prozesse zu übersetzen. Sliver besitzt eine geringe Codegröße und unterstützt 16 der 20 wichtigsten Workflow-Patterns [HHGR06]. Die Autoren zeigen, dass rudimentäre XML-Parser (ohne Validierungsfunktionen) die Implementierung einer mobilen BPEL-Engine für mobile Geräte ermöglichen.

Chen et al. stellen mit dem System **FollowMe** eine Workflow-Engine mit Unterstützung von pvPDL (Pervasive Process Definition Language) vor [CBLT06], welche eine von den Autoren speziell für kontextsensitive Applikationen entworfene Beschreibungssprache ist. Es sollen in erster Linie kontextbezogene Workflows unterstützt werden. Die entwickelte Engine basiert auf dem OSGi-Rahmenwerk, das wiederum einen Applikationsserver (Classloader) voraussetzt. Der Nachweis, dass ein Applikationsserver auf einem mobilen Gerät lauffähig ist, wird jedoch nicht erbracht. Das Kontextmodell wird mit Hilfe von Ontologien beschrieben. Der Hauptfokus von [CBLT06] liegt einerseits bei der Unterstützung von kontextsensitiven Applikationen und andererseits bei der Beschleunigung der Softwareapplikationsentwicklung auf Basis von Workflows. In der Evaluation [CBLT06] wird eine Applikation mit und eine ohne Rahmenwerk implementiert. Bei der Workflow-Lösung ist der personelle und zeitliche Aufwand geringer.

[10] www.jboss.org

Ein mobiles Ad-hoc-Netzwerk (MANET) [Mack99] ermöglicht es, zur Laufzeit mobile Endgeräte zum Netzwerk hinzuzufügen, entsprechendes Routing zu implementieren und eine Kommunikation ohne spezielle Infrastruktur zur Verfügung zu stellen. Sen et al. stellen in [SHHR07] ein WfMS auf Basis eines MANETs vor. Es wird eine BPEL-Erweiterung präsentiert, die es ermöglicht, mobile Anwendungsfälle ad-hoc zu unterstützen. In [SHHR07] wird das WfMS stets als *Middleware* beschrieben. Damit wollen die Autoren viele unterschiedliche mobile Endgeräte und erstmals ein „*deployment on time"* unterstützen. Gleichzeitig soll von Endgerät zu Endgerät eine informationstechnische direkte Kommunikation ermöglicht werden. Als konkrete Anwendungsfälle werden in [SHHR07] mobile Auktionen direkt vor einem Werk des Künstlers während einer Ausstellung genannt. Andere ortsabhängige Anwendungsfälle, bei der ein Prüfer die Kontaminierung von Böden feststellt und diese Information an die entsprechenden Ingenieure weiterleitet, werden angeführt. Innerhalb eines Baustellenanwendungsfalls definiert ein Vorarbeiter eine Aufgabe, die auf Grund eines Defekts entstanden ist. Die Aufgabe wird mittels PDA geplant und an den entsprechenden Arbeiter weitergeleitet. Eine Gruppenfunktion, spezielle Nachrichten mit Einbeziehung dieser Gruppenfunktionalität, die zufällige Reihenfolge der Nachrichten und das *deployment* der Workflows zur Laufzeit sind Merkmale des Systems. Es wird eine semi-automatische Verteilung der Workflows auf die mobilen Geräte ermöglicht. Die Autoren berechnen beispielsweise die Einteilung der Aufgaben nach der Morgenbesprechung (Baubetrieb) mit Hilfe von Heuristiken, die verschiedene Ressourcen berücksichtigen. Das System ist auf Basis von CAST[11] [RoHS06] implementiert worden, um die Koordination zwischen den verschiedenen Teilnehmern bzw. Endgeräten bezüglich Einsatzort und -zeit in den Workflows zu verwalten. Mit fünf Netzwerktransaktions-Typen haben die Autoren gezeigt, dass ein WfMS (mit den entsprechenden Modulen des WfMC-Referenzmodells) möglich ist. Ergebnisse der Evaluation zeigen eine lange Laufzeit (mehrere Minuten) der heuristischen Algorithmen. Zur Beschleunigung des

[11] CAST ist ein Koordinations-Modell, um Interaktionen zwischen Agenten innerhalb von Ad-hoc-Netzwerken zu unterstützten [RoHS06].

Systems werden auch Lösungsansätze, wie beispielsweise die Begrenzungen des Entscheidungsraums, präsentiert.

Kunze et al. stellen für mobile Prozesse eine *Execution Engine* vor, welche auf der **DEMAC**-Middleware (*Distributed Environment for Mobility-Aware Computing*) aufbaut [KuZL07]. Als mobilen Anwendungsfall für Geschäftsprozesse beschreiben die Autoren eine Auto-Panne, bei deren Behebung ein mobiles Informationssystem die Pannenhelfer in Kooperation mit der Pannen-Meldezentrale unterstützt. [KuZL07] motivieren, dass die Kontextnutzung auf dem mobilen Endgerät einer der aktuellen Trends im *mobile computing* ist und gleichzeitig komplexe mobile Applikationen immer stärker nachgefragt werden. Heutzutage möchten die Nutzer einige Aufgaben immer und überall auf ihren mobilen Endgeräten erledigen können. Aus diesem Wunsch heraus argumentieren Kunze et al., warum sie versuchen, Teile der Prozesslogik auf dem mobilen Endgerät zu platzieren, um sogenannte *application-oriented processes* zu erhalten. In [KuZL07] wird eine Unterstützung für solche Applikationen vorgestellt. Es wird mit der DPDL (DEMAC Process Description Language) eine eigene Sprache mit entsprechendem Metamodell eingeführt. Kunze et al. motivieren dies mit einem Vergleich der Modellierungssprachen [KuZL07]. Darin wird gezeigt, welche mobil-spezifischen Eigenschaften durch die entsprechenden Workflow-Sprachen nicht unterstützt werden und wie dies mit Hilfe von DPDL kompensiert wird. Das modulare DEMAC-System wird mit dem sogenannten „Base-Modul“ vorgestellt, das analog zu einem WfMS eine Engine beinhaltet. In [KuZL06] wird die DEMAC-Architektur mit den Prinzipien des *late-binding*, der modulare Aufbau und die leichtgewichtige Ressourceneinbindung vorgestellt.

Gunarathne et al. beschreiben eine eigens implementierte BPEL-Engine **Mora**, welche einbettbar, skalierbar und erweiterbar ist [GPWK07]. Es wird BPEL als Ausführungssprache angewandt, da die Autoren XML-Web Services (WS) als den Standard ansehen, der sich in den nächsten Jahren durchsetzen wird. Kontextinformationen werden strukturiert im *Information Model* abgelegt. Das *Process Model* wird verwendet, um die eigentlichen Workflows abzuspeichern. Ein Kernel, der unabhängig von der Hardware ist, soll dafür sorgen, dass die Engine auch auf mehreren CPUs ausgeführt

werden kann. Die Implementierung erfolgt in Java-Threads. Allerdings zeigen die Autoren nicht, ob diese Engine, die auf Axis2[12] aufbaut, auch mit mobiler Hardware ausgeführt werden kann. Axis2 basiert auf Java 1.5, wobei auch eine reine C-Implementierung existiert, mit der eine Portierung auf ein mobiles Endgerät möglich wäre. In der Veröffentlichung werden hauptsächlich Details der Prozess-Engine diskutiert, wie z. B. ein Multi-Prozess-Scheduler oder das Laufzeitverhalten der Engine selbst. Die Mobilität selbst wird lediglich durch den leichtgewichtigen Ansatz der Mora-Engine angedeutet.

Pajunen und Chande beschreiben in [PaCh07] ein vollständiges WfMS, das Mobilität berücksichtigt. Das System ist eine Plattform zum Einspielen und Verwalten von Workflows, die auf Basis von Diensten bereitgestellt werden. Sie unterstützen einschlägige Standards (BPEL, XML, XPath). Zunächst werden die Interaktion und die Integration einer Workflow-Engine als Hauptindikator zur Unterstützung von Prozessen identifiziert und implementiert. Die Workflow-Engine ist an die Funktionen und Kapazitäten eines Smartphones (siehe Kapitel 2.1.4) angepasst worden. Die Unterbrechungsunterstützung wird als eines der Hauptkriterien angesehen, das in WfMS schon früh identifiziert wurde [AGKA96], aber bis heute noch nicht vollständig in eine Prozesssprache bzw. ein WfMS integriert ist. Mit der Forderung einer von herkömmlichen Systemen abweichenden Zugriffskontrolle wird der Ausblick dieses Artikels abgeschlossen [PaCh07]. Es wird aus [PaCh07] nicht ersichtlich, warum z. B. BPEL, WSDL, XML und XPath Voraussetzung für eine mobile-spezifische Unterstützung sind. Ein innovativer Ansatz der Publikation ist die Verwendung von Binär-Attachments, um die Nachrichten trotz des XML-*Overheads* möglichst schlank zu halten. Auch die ortsabhängige Ausführung bestimmter Aktivitäten wird in [PaCh07, S. 281] angedeutet. Darüber hinaus werden mobil-spezifische Protokolle wie SMS, MMS oder Bluetooth unterstützt. Eine Integration von Kalender und Adressbuch ist implementiert worden. Die Kartenanzeige wurde eng mit dem Betriebssystem verzahnt. Weitere Kontexte, die in der

[12] Axis 2 ist eine Webservice-Engine: http://axis.apache.org/axis2/java/core/

Publikation berücksichtigt werden [PaCh07]: die Ortszeit, die Netzabdeckung, den Ladestand der Batterie, die aktuelle geographische Position, die Bewegungsgeschwindigkeit und die aktuelle Temperatur.

Mit dem WfMS **Commune** [DSBG06] wird ein verteiltes WfMS vorgestellt. Die Instanzen eines Workflows werden über verschiedene Workflow-Engines ausgeführt. *Commune* definiert sogenannte *Mini-Workflows*: ein *Mini-Workflow* ist der Teil eines Workflows, der getrennt – also autark – auf einem mobilen Endgerät ohne Verbindung zum stationären WfMS ausgeführt werden kann. Der Teil, der wiederum auf dem stationären WfMS in der zugehörigen Instanz ausgeführt wird, soll dabei Außen-Workflow (engl. *outer workflow*) genannt werden. Ein weiterer mobiler Aspekt von *Commune* ist, dass kontextuelle Informationen, wie z. B. der Kalender des Endbenutzers oder die aktuelle Position (Ort) berücksichtigt werden kann, wenn diese „Dinge“ (Kontexte) den Endbenutzern respektive Rollen zugeordnet sind. In [DSBG06] werden rein technisch BPEL-basierte Workflows zerlegt und auf entsprechende Endgeräte verteilt. Auf den mobilen Endgeräten sind jeweils kleine BPEL-Engines lauffähig. Die *Mini-Workflows* werden nicht automatisch erstellt und verteilt. Es werden auch leistungsschwache Endgeräte bedacht, die keine BPEL-Engine verwenden können, indem diese ihre Workflows einfach an die nächsten Geräte weitergeben. Ein sogenannter *PeoplePicker* sucht entsprechende freie Personenressourcen (nach Fähigkeit und Verfügbarkeit) aus. Letztlich kommunizieren die *Mini-Workflows* mittels SOAP[13]-WS.

Eine spätere Publikation der Hamburger Gruppe, die mit dem DEMAC-System in diesem Kapitel bereits vorgestellt wurden, konzentriert sich hauptsächlich auf das Problem der Endgerätevielfalt [ZaBV09]. Dafür wird die Verwendung der abstrakten Sprache *Concur Task Tree* (CTT) zur Beschreibung von GUIs eingeführt. Im Fokus von [ZaBV09] stehen verteilte Geschäftsprozesse, die über einen *Mobile Interaction Service* genutzt werden (Erweiterung einer konventionellen Workflow-Engine). Dabei sollen die mobilen Endgeräte reibungslos in dieses System integriert werden können.

[13] Netzwerkprotokoll zum Austausch von Daten über *Remote Procedure Calls* (IP-basierte Netzwerke) [Beng14].

Einige Ideen von [ZaBV09] werden auch in vorliegender Arbeit aufgegriffen: Abhängig von den Kontextinformationen dürfen z. B. bestimmte Interaktionsformen nicht verfügbar sein. In einer Fertigungshalle (auf Grund der Lautstärke) kann das beispielsweise eine Sprachverbindung sein. Ortsbezüge werden in der Publikation am Rande erwähnt, allerdings nicht spezifiziert oder formalisiert. Die Autoren bemerken, dass die Orte kontinuierlich dokumentiert werden müssen, um festzustellen an welcher Stelle bzw. welchem Ort die Aktivität stattgefunden hat. Workflow-Beschreibungen sollen vom Endgerät aus heruntergeladen werden. Gleichzeitig soll die Workflow-Engine ohne ständige Konnektivität zum Client auskommen können, damit Kosten bzw. die mangelnde Verfügbarkeit des Netzes überbrückt werden können. Vor allem die Peripheriegeräte, die die Funktionalität des Endgerätes erweitern und den Endbenutzer unterstützen. Dies sind Speicherplatz, die Kamera oder der GPS-Empfänger [ZaBV09]. Größere Datenmengen sollen nur referenziert werden, so dass sie zu einem späteren Zeitpunkt bzw. während niedrigerer Transferlast bezogen werden können. Erstmals wurde eine Zuordnung von Aktivitäten zu Endbenutzern in Abhängigkeit des aktuellen Aufenthaltsortes veröffentlicht. Die prototypische Implementierung basierend auf Java ME wurde in [ZaBV09] beschrieben. Die Evaluation ist innerhalb einer Emulator-Umgebung durchgeführt worden.

Mnaouer et al. stellen in [MnSY04] die Entwicklung einer Workflow-Engine unter die Prämisse der schnellen Applikationsentwicklung. Dabei wollen die Autoren auch Standards unterstützen und mit ihrem Ansatz das „Grundgerüst des Modells“ bilden, um eine interoperable, flexible und einfach zu implementierende mobile Entwicklungsumgebung als Open Source zu veröffentlichen. Die unterstützten Standards sind: WSFL[14], XML, SOAP, UDDI[15]. Als Motivation ihres Ansatzes wird ein zukünftiges mobiles System mit einer dynamischen Workflow-Engine und vielen sogenannten *Core Services* im Backend vorgeschlagen. Die Implementierung der Wf-Engine bzw.

[14] *Web Service Flow Language* (Vorgänger von BPEL): http://www.service-architecture.com/articles/web-services/web_services_flow_language_wsfl.html

[15] *Universal Description, Discovery and Integration* – standardisierter Verzeichnisdienst einer SOA: http://uddi.xml.org/

die Unterstützung der genannten Standards wird mit Hilfe von kXML („kleiner“ XML-Parser) und kSOAP (SOAP-API für Java ME) unter Verwendung von Java ME implementiert. Die wichtigsten Designkriterien sind Sicherheit, ein Verbindungsmodell, einfache Entwicklung, Interoperabilität und Service-Orientierung.

In [MaMo04] wird Workflow-Management in Verbindung mit Mobilität vorgestellt. Die Autoren merken an, dass es gerade bei WLAN bzw. Bluetooth bei vielen teilnehmenden Knoten (Endgeräten) oft zu Engpässen kommt, da die verwendeten Funkressourcen meist von vielen Nutzern gleichzeitig verwendet werden. In [MaMo04] wird daher ein Workflow in viele autonome Workflows zerlegt und jeweils einem Endbenutzer bzw. einem Akteur zugeteilt. Dieses Prinzip soll dem Endbenutzer ein hohes Maß an Freiheit und Unabhängigkeit bescheren, indem die limitierten Ressourcen bzw. Endgeräte im Netz überbrückt werden. Die in der Publikation vorgestellte Delegationsmethode soll dabei die mobilspezifischen Eigenschaften (z. B. Verbindungsabbrüche) berücksichtigen. In [MaMo04] wird eine Implementierung auf Basis von BPEL vorgestellt. Die graphische Modellierungsnotation wird mit UML umgesetzt. In der Publikation wurde die Implementierung nur am Rande erwähnt, so dass keine konkrete Aussage über die Funktionsweise des WfMS gemacht werden kann.

Der in [CJKM08] vorgestellte Ansatz basiert auf Peer-to-Peer[16]-Datenaustausch (P2P) von Office-Dokumenten, die alle Meta- und Workflow-Schema und -Instanz-Daten selbst mitführen. Dieses System soll es ermöglichen, normale Büroarbeiten auf Reisen bzw. zwischendurch zu erledigen. Allerdings werden in [CJKM08] keine mobil-spezifischen Aufgaben beschrieben, sondern Aufgaben aus dem Büroalltag. Auf Grund der P2P-Fähigkeit müssen auf jedem Client Workflow-Engines zur Verfügung stehen. Daher wird das sogenannte *Device Discovery* verwendet, um den P2P-Ansatz zu unterstützen. Eine Besonderheit der Architektur ist, dass die *Workflow Execution*-Komponente (eine *Inbox* und *Outbox*) besitzt. Wegen dieser Anlehnung an

[16] Peer-to-Peer (P2P) Konnektivität bezeichnet die jeweils einzelne Verbindung eines Endgerätes zu einem anderen, ohne dabei jeweils über einen Verwaltungsserver registriert werden zu müssen.

die Funktionsweise eines lokalen eMail-Clients ist das System auch als Outlook-Plugin implementiert worden.

Die verteilte Ausführung eines Workflows mit Darstellung der einzelnen Aktivitäten einer Arbeitsliste in Form einer geographischen Karte wird in [LeAH08, BrPa05] aufgegriffen. Die geographische Karte ist in beiden Arbeiten allerdings nur ein Spezialfall der Visualisierung von Arbeitslisten. Es können beispielsweise abstrakte Distanzmaße verwendet werden, um die benötigte Expertise eines Akteurs für eine bestimmte Aufgabe darzustellen.

Fazit

Ein spezifisches Merkmal, das sowohl modernen als auch klassischen mobilen WfMS zugeschrieben werden kann, ist die Behandlung von geplanten und ungeplanten Unterbrechungen bzw. Verbindungsabbrüchen. Auch in der heutigen Zeit treten Verbindungsabbrüche beim Telefonieren bzw. bei der täglichen Arbeit auf. Problematisch dabei sind die Verbindungsabbrüche, die nicht vermieden werden können (z. B. Gebäude mit starker Abschirmung). Gleichzeitig existieren diverse unterschiedliche mobile Kommunikationskanäle (siehe Abb. 2.3). Diese werden jedoch zum heutigen Zeitpunkt nicht übergreifend genutzt. Die Nutzung verschiedener Kommunikationskanäle könnte diese Verbindungsabbruchproblematik abschwächen. In [SeAP06] wird zur Lösung dieser „*handover*-Problematik“ eine IP-basierte Lösung vorgestellt. Ein weiteres Merkmal aus den letzten beiden Kapiteln ist die Berücksichtigung von Kontextparametern z. B. bei der Aufgabenzuordnung (Tasklisten-Management). Dieser Aspekt der Kontextberücksichtigung wird im weiteren Verlauf der Arbeit in Kapitel 4 und 7.2 weiterführend untersucht. Die Reduzierung des zu übertragenden Datenvolumens ist in mehreren Arbeiten diskutiert worden und entsprechende Lösungen sind angeboten worden. Letztendlich ist dieser Aspekt – wie [JHHS00, Busb94] gezeigt haben – durch Datenkompressionsverfahren innerhalb des mWfMS einfach lösbar. Standardisierte Benutzungsschnittstellen – wie die beiden letzten Kapitel gezeigt haben –würden vielen Prozess-basierten mobilen Systemen erhebliche Akzeptanzvorteile geben. Diese Standardisierung kann allerdings nicht mittels eines Diktats in die Praxis umgesetzt werden. Die

Entwicklung in den weiteren Jahren wird zeigen, welche mobilen Betriebssysteme bzw. Plattformen sich durchsetzen werden. Unterschiedliche Techniken zur Darstellung einer Aufgabenliste werden beispielsweise in Kapitel 7.3 vorgestellt. Das letzte identifizierte Merkmal von mobilen WfMS ist die Integration verschiedenster Anwendungsprogramme auf dem mobilen Endgerät. Dies kann beispielsweise über die Verknüpfung von Kalender, Adressbuch oder einer Kartendarstellung (siehe z. B. Kapitel 7.3) erzielt werden. Im Verständnis der vorliegenden Arbeit verknüpft das mobile Prozess-basierte System verschiedene Anwendungsprogramme über den zu Grunde liegenden Prozess (siehe Kapitel 4).

In Tab. 3.1 werden die (mobilen) WfMS der beiden letzten Kapitel ausgewertet. Es wurde innerhalb der Publikationen auf Unterstützung (+) verschiedener – innerhalb der Publikationen identifizierter – Anforderungen geprüft:

- Berücksichtigung ungeplanter Verbindungsabbrüche
- Unterstützung der Kontextinformationsauswertung innerhalb des WfMS
- Tasklistenverwaltung des (mobilen) Clients; differenziert nach:
- der Basisfunktionalität (z. B. einfache Taskliste)
- der geographischen Unterstützung (z. B. Landkarte mit Aufgabenliste)
- weiteren Funktionalitäten (z. B. Sortierung nach Prioritäten)
- Integration verschiedener Anwendungsprogramme innerhalb des WfMS (z. B. Kalender, Email, Weiterleitung von Tasks)
- Mobiler Client
- Wf-Engine innerhalb des mobilen Endgeräts
- angewandte Wf-Sprache

Tab. 3.1: Auswertung mobiler WfMS

Quelle	Verbindungs-abbrüche (ungeplant)	Kontext-auswertung	Tasklisten			Integration versch. Anwendungs-programme	mobiler Client	mobiler Server	Wf-Sprache
			Basis	geographisch	weitere Funktionen				
[AGKA96]	-	-	+	-	+	-	+	-	-
[JHHS00]	-	+	+	+	+	-	+	-	-
[DMPD99]	+	-	+	-	+	-	+	-	-
[StKn01]	-	-	+	-	-	-	+	-	-
[Buss94]	+	+	+	-	-	+	+	-	-
[Buss95]	+	+	+	+	+	+	+	-	-
[PaPC97]	-	-	+	-	+	-	+	-	Petri-Netze
[Zuku99]	+	-	+	-	+	-	+	-	-
[MuWL00]	-	-	+	-	-	-	+	-	-
[Bolc00]	-	-	+	-	+	-	+	-	-
[PCDG02]	-	-	+	+	+	-	+	-	-
[HHGR06]	-	-	-	-	-	+	-	+	BPEL
[CBLT06]	-	+	-	-	-	+	-	+	PvPDL
[SHHR07]	-	-	+	+	+	+	-	+	BPEL erweit.
[KuZL07]	-	+	+	-	+	+	+	-	DPDL
[GPWK07]	-	+	-	-	-	-	-	+	BPEL
[PaCh07]	+	+	+	+	+	+	-	+	BPEL, XPath
[DSBG06]	-	+	+	-	+	-	+	+	BPEL
[ZaBV09]	-	+	+	+	+	-	-	+	CTT
[MnSY04]	-	-	-	-	-	-	+	-	WSFL
[MaMo04]	+	-	-	-	-	-	-	+	BPEL
[CJKM08]	-	-	-	-	-	-	-	+	Email-basiert

Die Vorteile von mobilen Technologien gegenüber statischen ortsgebundenen Systemen können über entsprechende Fallstudien mobiler Anwendungsfälle beschrieben werden [KhPW03, DuGa03, HeVa02]. Die Unterstützung von Workflows unter Berücksichtigung von Kontextinformationen gibt den Unternehmen Vorteile gegenüber denjenigen, die weiterhin auf rein stationäre IKT setzen. In folgendem Kapitel wird daher eine Methode vorgestellt, Workflows mit mobilen Anwendungsfällen innerhalb eines Unternehmens zu identifizieren.

3.2.3 Mobile Process Landscaping

Die Methode des *Mobile Process Landscaping* (MPL) soll Geschäftsprozesse respektive Workflows mit mobilen Anteilen basierend auf der Grundlage, dass der Benutzer seine Aufgaben mobil bearbeitet, identifizieren. Die MPL-Methode unterscheidet vier Detailebenen [KöGr04a]:

1. **Kern-Prozesse**: Beschreiben die grobe Unternehmensstruktur und die wesentlichen Elemente der Wertschöpfungskette.

2. **Sub-Prozesse**: Stellen die zweite Detailebene dar und beschreiben die Aufgaben und Funktionen der Core Prozesse.

3. **Aktivitäten**: Die Aktivitäten und Tätigkeiten der Sub-Prozesse werden in dieser Ebene detaillierter beschrieben.

4. **Informationsobjekte**: Aktivitäten und Tätigkeiten werden als Prozessabläufe von Informationsobjekten dargestellt. Es werden u. a. Datenflüsse bzw. Schnittstellen spezifiziert und die Einsatzmöglichkeit mobiler IKT untersucht.

Mit diesen vier Ebenen werden laut [KöGr04a] vier unterschiedliche Geschäftsprozesse identifiziert. Dabei wird jede höhere Ebene durch die darunterliegende Ebene verfeinert, d. h. die Prozesse werden detaillierter beschrieben, bis sie anschließend in Ebene vier ausgeführt (automatisiert) werden.

Der Analyseaufwand wird dabei laut [KöGr04a] durch den Top-Down-Ansatz minimal gehalten. Die wichtigsten Eigenschaften werden in folgender Auflistung beschrieben [KöGr04b]:

- Systematische Vorgehensweise bei der Entwicklung von Unterstützungsmöglichkeiten für mobile verteilte Geschäftsprozesse.

- Identifikation von zu unterstützenden Prozessen auf Basis einer detaillierten Kosten-Nutzen-Analyse. Der entstehende Mehrwert nach der Implementierung der Unterstützungslösung wird bereits vorab messbar gemacht.

- Die voraussichtliche Dauer, die anfallenden Kosten, sowie die benötigten Ressourcen für die Durchführung des *Mobile Process Landscaping*, lassen sich bereits von Beginn an für jeden durchzuführenden Schritt aufbauend auf der fachlichen Prozessmodellierung abschätzen.

- Das Mobile Process Landscaping ist an der Erzielung eines tatsächlichen, messbaren Mehrwerts orientiert. In jeder einzelnen Entwicklungsstufe wird die potenziell zu erreichende Verbesserung ermittelt. Sollte kein Verbesserungspotenzial identifiziert werden, kann das Projekt jederzeit mit einem definierten Ergebnis beendet werden.

- Das Mobile Process Landscaping ist eine konkrete operative Hilfestellung für die Einführung mobiler Technologien im Unternehmen.

Zusätzlich zu den bereits eingeführten Ebenen und den Eigenschaften des MPL wurde in [KöGr04b] ein detailliertes Vorgehensmodell vorgestellt (siehe Abb. 3.7).

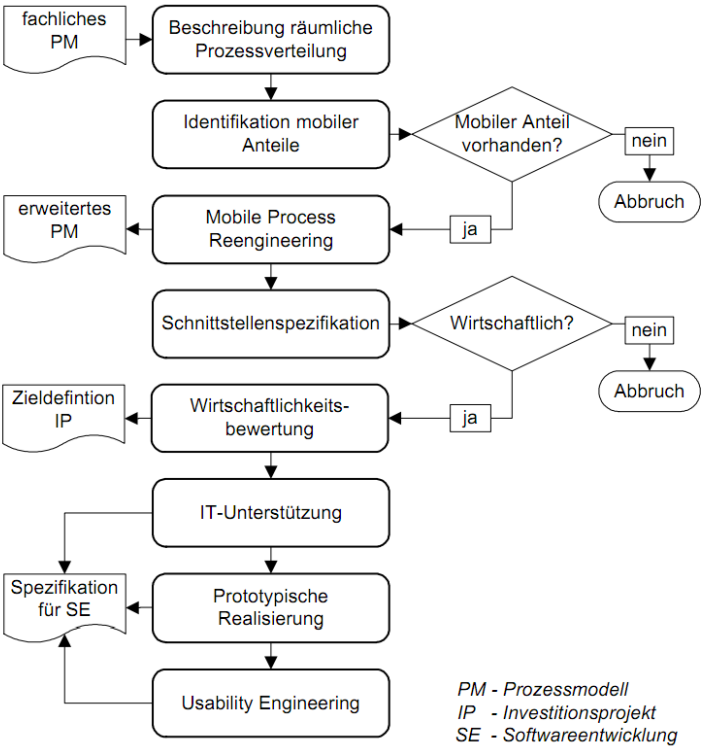


Abb. 3.7: Vorgehensmodell MPL nach [KöGr04b]

Grundlage des Vorgehensmodells (siehe Abb. 3.7) ist ein erstelltes Geschäftsprozessmodell bzw. ein Workflow-Modell, das aus fachlicher Sicht modelliert wurde. Aufbauend auf diesen fachlichen Workflows werden die Teile eines Workflows identifiziert, die ortsunabhängig mobil ausgeführt werden bzw. es werden diejenigen Workflow-Teile ausgewählt, die eine räumliche Beschreibung enthalten. Problematisch an diesem Verfahren ist allerdings die Tatsache, dass es bisweilen keine standardisierte geographische Annotationsform für Workflow-Modelle gibt. In Kapitel 4 wird eine entsprechende Annotation mit zugehörigem Schema eingeführt. Im nächsten

Schritt werden die Prozessanteile identifiziert, die auf einem mobilen Endgerät potentiell ausgeführt werden. Danach sollte der Workflow „*reengineered*“ werden. In diesem Prozessschritt wird das mobile Endgerät als Dienst interpretiert. Ein Dienst hat Eingangs- und Ausgangsparameter, die im Zuge der „Schnittstellenspezifikation“ definiert werden (siehe Abb. 3.7). Die Wirtschaftlichkeitsanalyse bzw. -bewertung im nächsten Schritt soll dabei helfen, ein positives Ergebnis mit z. B. einer zu entwickelnden mobilen Applikation im Unternehmen zu schaffen. Es muss dabei ermittelt werden, welche mobilen Technologien eingesetzt werden und wie diese den Workflow insgesamt verbessern, damit die Unternehmensziele erreicht werden bzw. die Vorgaben innerhalb des Unternehmens eingehalten werden. Das Ergebnis entscheidet darüber, ob das Projekt bzw. in welchem Umfang das Projekt durchgeführt werden kann. Falls die Umsetzung beschlossen wurde, sind die Anforderungen für die „IT-Unterstützung“ (siehe Abb. 3.7) festzulegen. In diesem Prozessschritt wird der Entwurf bzw. die Implementierung (der Softwareentwicklungsprozess) gewählt. Für einen Prototyp werden Usability-Tests durchgeführt, die eine Aussage über die letztendliche Gebrauchstauglichkeit der Applikation im Zusammenspiel der ganzen unternehmensweiten Software trifft. Ergebnis des MPL ist eine Spezifikation der IT-Unterstützung, die den Workflow innerhalb des Unternehmens unterstützt. Eine detaillierte Auflistung der einzelnen Prozessschritte kann aus [KöGr04b] entnommen werden (siehe Abb. 3.7).

Ein Beispiel eines fiktiven Unternehmens A aus der Stahlindustrie soll die Methode des MPL näher erläutern. Dabei werden zu Beginn – wie aus der Auflistung zu Beginn dieses Kapitels zu entnehmen ist – die Kern-Prozesse mit den voneinander räumlich getrennten Organisationseinheiten des Kunden und dem Unternehmen identifiziert (siehe Abb. 3.8).

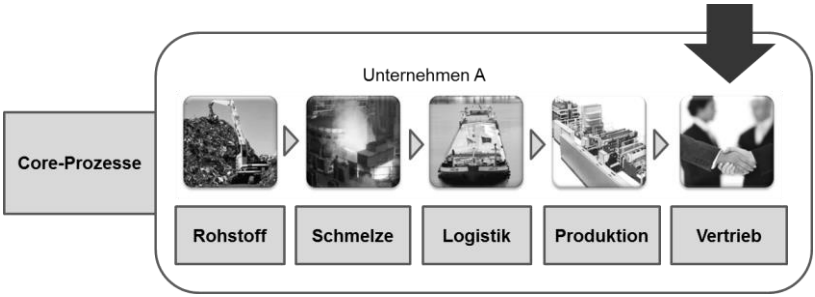


Abb. 3.8: Kern-Prozesse Unternehmen A

In Abb. 3.8 ist zu erkennen, dass gerade im Vertrieb eine nahe Verbindung zwischen Kunde und dem Unternehmen A vorhanden ist, somit also das Potential für mobile Applikationen hoch ist, da bei „Überschneidungen von Organisationseinheiten“ (z. B. Vertrieb verkauft Produkte an Kunden) mobiles Potential vorliegt [KöGr04b]. Die Sub-Prozesse des Vertriebs werden verfeinert (siehe Abb. 3.9). Sowohl innerhalb der Prozessschritte, aber auch dazwischen, sind mobile Potentiale zu finden. Gerade die Kommunikation vor einem Verkaufsgespräch könnte zusätzlich informationstechnisch unterstützt werden, was jedoch in der Methode des MPL nicht vorgesehen ist.

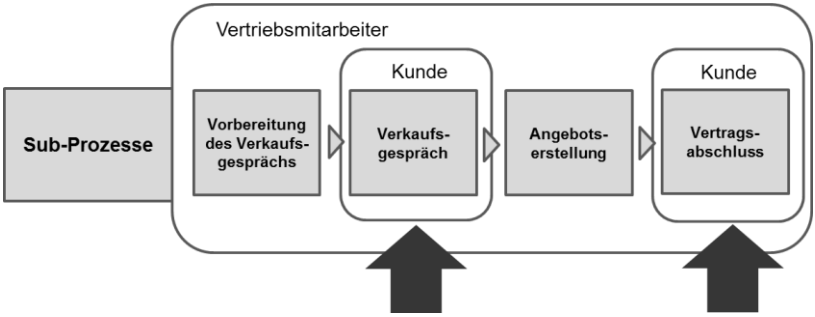


Abb. 3.9: Sub-Prozess Unternehmen A

In Abb. 3.10 werden die Aktivitäten der beispielhaften Angebotserstellung weiter verfeinert.

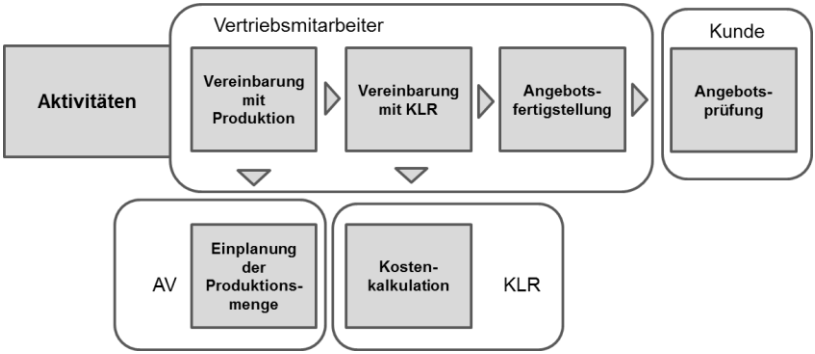


Abb. 3.10: Aktivitäten Unternehmen A

Die vollständige Angebotserstellung wird detailliert in der Ebene der *Information Objects* dargestellt (siehe Abb. 3.11). In dieser Ebene ergeben sich wieder Potentiale zwischen den Schnittstellen bzw. unabhängig von den Prozessschritten. Es könnten z. B. einzelne kritische Angebotsinhalte vorab mit Hilfe von Kurznachrichten während der Prozessausführung bestätigt werden.

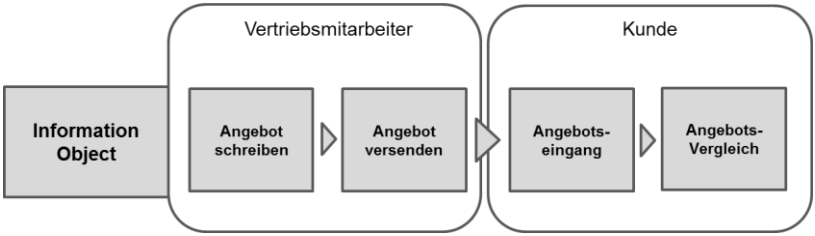


Abb. 3.11: Informationsobjekte Unternehmen A

Die Methode des MPL zeigt, dass es möglich ist, Workflows systematisch auf den sinnvollen Einsatz von mobilen Technologien hin zu analysieren. Allerdings wird lediglich der Ist-Workflow analysiert, der fehlerhaft bzw. ineffizient sein könnte. Grundsätzlich sollten Workflows innerhalb des Unternehmens unabhängig von einer technischen Ausrichtung entworfen bzw. modelliert werden, um möglichst viel Geschäftslogik bzw. zahlreiche Geschäftsziele zu integrieren. MPL stellt allerdings keine Modellierungsunterstützung für Workflows bzw. für die Berücksichtigung von Mobilität innerhalb von Workflows bereit vgl. [RDUD13]. MPL soll als Methode zur Prüfung von Workflows innerhalb eines Unternehmens Anwendung finden. Mit der Methode lassen sich stationäre und mobile Anteile innerhalb von Workflows trennen.

Wie schon eingangs der vorliegenden Arbeit erwähnt, soll nicht auf die Trennung von Geschäftsprozessen in stationäre und mobile Anteile fokussiert werden, sondern auf Workflows, die wiederum Kontextinformationen innerhalb der Ausführung berücksichtigen.

3.3 Mobile WfMS

Eine Definition eines mobilen WfMS [FWAC17] muss die Erkenntnisse bzw. die Definition mobiler IKT im Verständnis der vorliegenden Arbeit berücksichtigen (siehe Kapitel 2.1.1 und Kapitel 2.1.2).

Ein wichtiger Aspekt in der Definition muss die Ortsunabhängigkeit darstellen, die in bisherigen WfMS nicht beachtet worden ist. Einerseits kann ein WfMS durch die Mobilität ortsunabhängig sein, andererseits kann genau dieser Aspekt besondere Bedeutung im System erlangen, sofern z. B. abhängig vom Ort Aktivitäten, Prozesse oder Teilprozesse ausgeführt werden. Dieser Aspekt in Bezug auf die Kontextnutzung allgemein soll im weiteren Verlauf genauer untersucht werden: Inwiefern können Kontexte, die vom Endgerät bezogen werden können, innerhalb von Workflows beachtet werden? In mobilen Workflows – wie in Kapitel 3.2.1 und 3.2.2 vorgestellt – werden z. B. Ortsinformationen entweder statisch angenommen oder sind gar nicht von

Interesse. Im Verlauf der vorliegenden Arbeit wird dieser Aspekt weiterführend untersucht. Heutzutage werden immer mehr mobile IKT eingesetzt bzw. Workflows werden zunehmend mobil ausgeführt. Die mobile Technik wird immer günstiger und die laufenden Kosten können über sogenannte Flatrates reduziert werden. Gleichzeitig haben sich die Menschen daran gewöhnt, immer und überall auf Emails bzw. geschäftsrelevante Daten zuzugreifen. So können z. B. Wartezeiten bei Dienstreisen überbrückt werden, indem (Teile eines) Geschäftsprozess mobil bearbeitet werden. Weitere Vorteile der mobilen Bearbeitung von Geschäftsprozessen werden in [DSKO09] erörtert. In vorliegender Arbeit werden mobile (End-)Geräte in Form eines Smartphones, Notebooks, Embedded-PCs oder anderer ähnlicher mobiler Geräte betrachtet (siehe Kapitel 2.1.4). In Kapitel 7.2 wird auf die Geräteklasse der Smartphones fokussiert.

Der Aspekt der Kontextberücksichtigung innerhalb des WfM stellt zusätzliche Anforderungen (siehe Kapitel 2.2). Die Workflows werden mittels eines speziellen mobilen WfMS (mWfMS) unterstützt. Ein mWfMS ist gegeben, wenn *„[…] mindestens eine der im Referenzmodell beschriebenen Komponenten auf einem mobilen Gerät vollständig implementiert ist.“* Die Definition [DKKO09] definiert ein mWfMS analog zur Definition des WfMC-Referenzmodells eines WfMS (siehe Abb. 2.11). Die Sicht auf die Architektur erschließt die funktionellen Anforderungen an das WfMS. Gleichzeitig wird damit die Funktionalität des mWfMS ersichtlich. Hinter der Komponente des *Process Definition Tools* verbirgt sich die Modellierungsunterstützung des Workflows. *Interface 2* kennzeichnet die Endbenutzerschnittstelle (z. B. die mobile Taskliste; siehe auch Kapitel 7.3). Ein mobiles WfMS liegt vor – im Gegensatz zu einem normalen WfMS – wenn mindestens eine der Komponenten auf einem mobilen Gerät ausgeführt wird. In Abb. 3.12 wird die „logische Sicht“ (vgl. Kapitel 2.8) auf das Architekturmodell eines mWfMS analog zum Workflow-Referenzmodell der WfMC dargestellt.

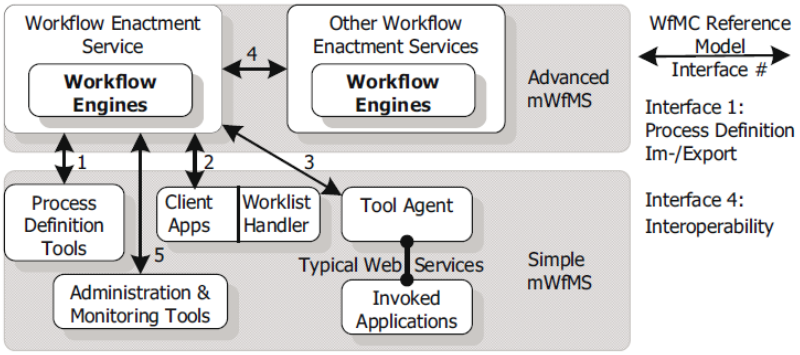


Abb. 3.12: Mobiles Workflow-Managementsystem nach [DKKO09]

In Abb. 3.12 ist eine weitere Unterscheidung von mWfMS zu erkennen: Wenn sich keine Workflow-Engine auf dem mobilen Endgerät befindet, so wird das mWfMS als „*einfaches mWfMS*“ bezeichnet. Dies kann z. B. eine einfache (mobile) Website sein, die speziell für mobile Geräte mit kleinen Displays erstellt wurde. Wenn sich allerdings eine lauffähige Workflow-Engine auf dem mobilen Endgerät befindet, wird das mWfMS als „*hochentwickeltes mWfMS*“ (*engl. advanced mWfMS*) bezeichnet. Bei einem hochentwickelten mWfMS können (Teil-)Prozesse lokal auf dem mobilen Gerät ausgeführt werden, was einen erheblichen Unterschied zum einfachen mWfMS darstellt.

Aus Abb. 3.12 lassen sich die wichtigsten Komponenten eines mWfMS ableiten. Zentrale Aspekte innerhalb eines WfMS sind die Modellierung und die Ausführung der Workflows. Die laufenden Workflows werden mit Hilfe des *Administration & Monitoring Tools* überwacht und kontrolliert. An dieser Stelle werden sogenannte *Key Performance Indicators* (KPI) ausgewertet und überprüft. Die Endbenutzerschnittstelle wird durch die *Client Apps* bzw. den *Worklist Handler* unterstützt. Andere wichtige Systeme bzw. Komponenten für den Workflow werden über den *Tool Agent* angebunden und sind durch die Komponente *Invoked Applications* im logischen Architekturmodell vertreten. Die mobile Erweiterung bzw. die Differenzierung des Referenzmodells der WfMC stellt eine hohe Abstraktion der Implementierung

dar. Letztendlich wurde dadurch gewährleistet, dass das Modell in der ursprünglichen Darstellungsform (siehe Abb. 2.11) seit ca. 20 Jahren bestand hat und nicht ergänzt oder erweitert werden musste. Auch im Kontext der vorliegenden Arbeit bzw. in Bezug auf die mobilen Kontextinformationen und der Verknüpfung mit dem WfMS eignet sich das Referenzmodell.

Im weiteren Verlauf der vorliegenden Arbeit wird stets allgemein von mWfMS gesprochen, während die Unterscheidung zwischen „hochentwickeltem mWfMS“ und „einfachem mWfMS“ einfach nachzuvollziehen ist. Implementierungen eines „einfachen mWfMS“ werden in Kapitel 7.2 vorgestellt. In Kapitel 6, Kapitel 7.1 und [SHHR07] wird z. B. ein hochentwickeltes mWfMS beschrieben.

4 Ortsbezüge in Workflows

Im vorliegenden Kapitel wird eine Annotation zur Modellierungsunterstützung von Workflows vorgestellt, um Ortsbezüge abzubilden. Workflows werden (teilweise) mit mobilen Geräten ausgeführt, die während der Zeitspanne der Anwendung mit Batterie/Akku autark betrieben werden können. Zur Bestimmung des aktuellen Aufenthaltsortes wird ein Eigen- oder Fremdortungsverfahren genutzt (siehe Kapitel 2.1.6). Die mobilen Geräte ermitteln ihren eigenen Aufenthaltsort über Ortungsverfahren und können die Informationen an das mWfMS weiterreichen. Im Kontext der vorliegenden Arbeit unterliegen Workflows abhängig vom Aufenthaltsort ortsabhängigen Einschränkungen (Ortsbezüge).

In Kapitel 4.1 werden alle unterschiedlichen Bestandteile des Konzepts von Ortsbezügen, der Ortsberücksichtigung innerhalb von Workflows, definiert und im weiteren Verlauf jeweils referenziert. In Kapitel 4.2 wird ein Ortsmodell zur Unterstützung des eingeführten Konzepts bereitgestellt. Eine graphische Annotation von Ortsbezügen (mit der Berücksichtigung verschiedener Sprachaspekte) wird in Kapitel 4.3 eingeführt. Die formale Definition zur Annotation von Petri-Netzen und der BPMN 2.0 wird in Kapitel 5 entwickelt. Die Modellierung von Ortsbezügen bzw. die sprachunabhängige Integration von Ortsbezügen wurde erstmals in [Voge10] vorgestellt. In vorliegender Arbeit werden die Erkenntnisse aus [Voge10] zusammengefasst, erweitert und mit einer Architektur eines WfMS praktisch evaluiert.

4.1 Das Konzept der Ortsbezüge

Das Konzept der Ortsbezüge sieht vor, dass einzelne Aktivitäten eines Workflows an festgelegte Orte gebunden werden bzw. deren Ausführung an diesem bestimmten Ort (oder Orten) verhindert wird. Im weiteren Verlauf der Arbeit wird daher von Ortseinschränkungen (OE) gesprochen. Ziel des Workflow-Entwurfs bzw. der Geschäftsprozessmodellierung ist es, OE zu

definieren, die für die Ausführungsorte von Aktivitäten gelten. Im Mittelpunkt dieses Kapitels, bedingt durch die Definitionsphase (siehe Kapitel 3.1), steht somit die Modellierung dieses Konzeptes [DSKO09]. Die technischen Maßnahmen, um die entsprechenden OE durchzusetzen, werden in Kapitel 6 demonstriert.

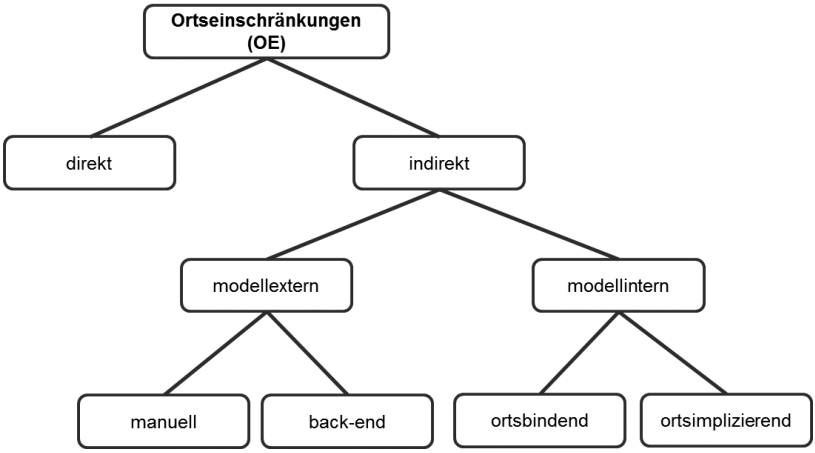


Abb. 4.1: Hierarchie von Ortseinschränkungen

Das Konzept der OE lässt sich in zwei Klassen einteilen (siehe Abb. 4.1). Einerseits in die Klasse der direkten OE und andererseits in die Klasse indirekter OE. Dabei sind die indirekten OE weiter aufteilbar, was im weiteren Verlauf dieses Kapitels noch erörtert wird. Alle OE teilen die Eigenschaft, ein Vorzeichen zu besitzen, das entweder *positiv* oder *negativ* ist. Das Vorzeichen entscheidet darüber, welche Auflage getroffen wird: Ist es *positiv*, wird eine Auflage erzeugt, die eine Aktivität an einem entsprechenden Ausführungsort zulässt. Aktivität „A“ **darf** z. B. nur an Ort „B“ ausgeführt werden. Ein *negatives* Vorzeichen hingegen erlaubt den Ausführungsort für die entsprechende Aktivität nicht. Die Aktivität „A“ **darf** z. B. an Ort „B“ **nicht** ausgeführt werden. Eine weitere zentrale Eigenschaft von OE sind die Basisdaten. Diese Daten sind notwendig, damit eine OE überhaupt ausgewertet werden kann. Sie tragen den Inhalt (*engl. Content*), ohne den eine OE nicht

interpretierbar ist. Basisdaten können in Form von Modellierungsinformationen während der Modellierung eingegeben werden. In diesem Fall werden die Daten als statische Basisdaten bezeichnet, da sie für die jeweils vom Modell abgeleiteten Workflow-Instanzen unveränderbar sind. Je nach Ausprägungsart der OE kann eine andere Erscheinungsform vorliegen. Basisdaten sind erst während der Ausführung des Workflows verfügbar und beeinflussen die Auflagen der OE in der jeweils abgeleiteten Instanz (individuell). Diese Art der Basisdaten wird als dynamische Basisdaten bezeichnet.

Bevor die weiteren Elemente (siehe Abb. 4.1) beschrieben werden, müssen vorab grundlegende Bausteine des Konzeptes eingeführt werden. Im nächsten Kapitel werden die ortsbeschreibenden Elemente eingeführt.

4.1.1 Die ortsbeschreibenden Elemente

Das Konzept der OE soll den Modellierer unterstützen den Ausführungsort von Aktivitäten zu beeinflussen. Die relevanten Ortsinformationen werden im Workflow-Modell abgebildet. Die ortsbeschreibenden Elemente haben die Aufgabe, die modellierten Informationen innerhalb des Workflows zu repräsentieren.

In vorliegender Arbeit werden drei Arten ortsbeschreibender Elemente verwendet:

- **Ort:** Ein Ort wird durch die definierte Grundfläche beschrieben.
- **Ortstyp:** Werden zwei oder mehr Orte zusammengefasst, wird in vorliegender Arbeit von einem Ortstyp gesprochen. Die Bildung der Ortstypen erfolgt meist unter logischen Gesichtspunkten, wie z. B. Staaten eines Landes oder die Länder innerhalb der Europäischen Union (EU).
- **Zuordnungsliste:** Eine Zuordnungsliste besteht aus einer Menge von binären Tupeln. Diese Tupel setzen jeweils immer zwei Orte miteinander in Beziehung, wobei der erste Eintrag den Quell- und der Zweite den Zieleintrag definiert (vgl. Abb. 4.2).

Die eingeführten Elemente müssen im Einklang mit einem geeigneten Ortsmodell definiert werden. Ein solches Ortsmodell wurde in [Deck11, S. 194-199] auf Basis einer speziellen Art von Petri-Netzen vorgestellt. Das Ortsmodell der vorliegenden Arbeit (siehe Kapitel 4.2) wird unabhängig von einer Geschäftsprozessmodellierungssprache definiert. Es setzt die ortsbeschreibenden Elemente in einen logischen Zusammenhang mit der Definition von Ortsbezügen.

4.1.2 Direkte Ortseinschränkungen

Die direkten Ortseinschränkungen (siehe Abb. 4.1) werden in der Modellierungsphase eines mobilen Workflows formuliert (siehe Kapitel 3.1). Im Gegensatz dazu können die indirekten Ortseinschränkungen erst zur Laufzeit des Workflows ausgewertet werden. Direkte Ortseinschränkungen werden zum Entwurfszeitpunkt bei der Entwicklung eines Workflow-Modells definiert und besitzen statische Basisdaten. Für alle abgeleiteten Prozessinstanzen sind die OE der Ausführungsorte identisch und werden eingehalten. Die Aktivitäten mit entsprechenden direkten OE werden als Zielobjekte innerhalb des Workflow-Modells bezeichnet. Zur Gewährleistung einer direkten Ausführung des Workflows wird jeder OE ein ortsbeschreibendes Element zugewiesen. In Tab. 4.1 wird dieser Zusammenhang verdeutlicht. Ein ortsbeschreibendes Element kann jeweils ein Ort, ein Ortstyp oder eine Zuordnungsliste sein (siehe Kapitel 4.1.1).

Tab. 4.1: Semantik direkter Ortseinschränkungen

	Ortsbeschreibendes Element
Positiv	Die Ausführung des Zielobjektes **muss** am entsprechenden ortsbeschreibenden Element stattfinden.
Negativ	Die Ausführung des Zielobjektes **darf nicht** am entsprechenden Ort ausgeführt werden. Allerdings an jedem anderen Ort.

4.1.3 Indirekte Ortseinschränkungen

Die indirekten OE werden (siehe Abb. 4.1) analog zu den direkten OE im Workflow-Modell abgebildet. Allerdings ist eine unmittelbare Ausführung nicht möglich. Die enthaltenen Basisdaten sind nicht automatisch vorhanden, sondern müssen aus externen Quellen bzw. von anderen Aktivitäten zur Laufzeit bestimmt werden. Diese Basisdaten werden als dynamische Basisdaten bezeichnet, die aus verschiedenen Quellen bezogen werden. Eine solche Quelle wird als Quellobjekt benannt. Ein Quellobjekt definiert den Zeitpunkt, an dem die dynamischen Basisdaten bereitstehen bzw. es für die jeweilige Workflow-Instanz möglich ist eine Ableitung der OE vorzunehmen und die entsprechende OE auszuführen. Das Quellobjekt wird auch als Trigger-Objekt bezeichnet (Trigger, engl. Auslöser), da es einen Schalter umlegt, um das Zielobjekt unter der entsprechenden OE auszuführen. Eine indirekte OE besitzt – wie auch die direkte OE – ein Zielobjekt, dessen Ausführungsort anhand der generierten Basisdaten und der damit erstellten Auflage erzeugt wird (siehe Abb. 4.2).

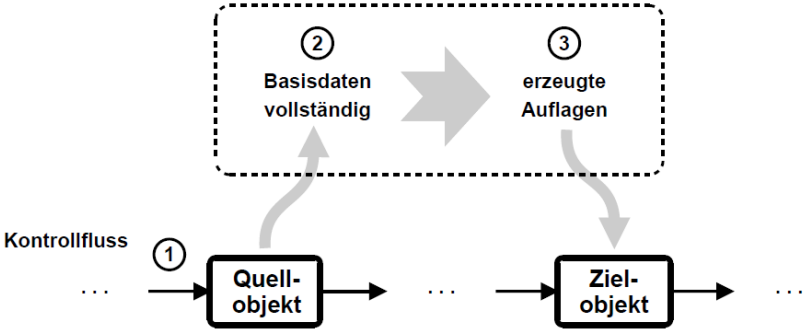


Abb. 4.2: Skizze einer indirekten Ortseinschränkung

Innerhalb eines fiktiven Workflows erreicht der Kontrollfluss eine Aktivität (siehe Abb. 4.2, Punkt 1): das Quellobjekt (die Quelltransition). Gleichzeitig wird vorausgesetzt, dass die dynamischen Basisdaten am Quellobjekt vorhanden sind (siehe Abb. 4.2, Punkt 2). Die OE können zur Bestimmung des

erlaubten Ausführungsortes des Zielobjektes auf Basis der dynamischen Basisdaten abgeleitet werden. In Punkt 3 beeinflusst die OE das Zielobjekt.

Zur weiteren Demonstration des Prinzips sei auf die *generische Programmierung* verwiesen [CEGV00]. Bei dieser Technik werden Objekte zur Laufzeit aus einer Menge von Daten, die zu Beginn eines Programmes noch nicht determiniert sind, identifiziert und je nach Programm zu einem neuen Objekt zusammengeführt. Es ist notwendig, dieses neu erzeugte Objekt vorab mittels eines Schemas beschrieben zu haben. Analog dazu ist die Funktionsweise der direkten und indirekten OE. Indirekte OE werden während der Laufzeit – sobald die dynamischen Basisdaten vorhanden sind – zu direkten OE, die die Basisdaten einer OE für das Zielobjekt bestimmen.

Die Generierung der OE für den Ausführungsort ist von den dynamischen Basisdaten abhängig. Dynamische Basisdaten unterscheiden sich in Art bzw. Herkunft und müssen differenziert behandelt werden. Zu diesem Zweck werden *modellexterne* und *modellinterne* OE eingeführt (siehe Abb. 4.1).

4.1.4 Modellexterne Ortseinschränkungen

Die Bezeichnung „modellextern“ lässt darauf schließen, dass die dynamischen Basisdaten nicht aus dem Workflow-Modell bezogen werden. Daher werden für diese Typen keine direkten ortsbeschreibenden Elemente assoziiert. Die OE für das Zielobjekt werden innerhalb der externen Datenquelle generiert. Eine zusätzliche Untergliederung der modellexternen OE in *manuell* und *back-end* (vgl. Abb. 4.1) soll zwei Fälle unterscheiden:

- Mit *manuellen* modellexternen OE werden OE bezeichnet, die von einem Endbenutzer (bzw. einer Rolle im WfMS) manuell bestimmt werden. Dieses manuelle Zuordnen kann z. B. über ein spezielles Frontend dem jeweiligen Anwender zur Verfügung gestellt werden. Es ist jedoch auch denkbar, dass der Modellierer diese Aufgabe während der Modellierung des Workflows durchführt, sofern eine entsprechende Werkzeugunterstützung vorhanden ist.

- Die modellexternen *back-end* OE hingegen bezeichnen die Basisdaten, die aus Workflow-Modell-externen IS (z. B. ERP-, Supply-Chain-Management (SCM) oder CRM-Systemen) stammen. Die meisten Unternehmen verfügen auf Grund von historischen IT-Entwicklungen über mehrere betriebliche IS. Diese IS dienen teilweise auch der Unterstützung von Workflows [Gada05, S. 257]. Werden die dynamischen Basisdaten von einem derartigen IS abgerufen, so wird von einer modellexternen back-end-OE gesprochen.

4.1.5 Modellinterne Ortseinschränkungen

Die modellinternen OE sind – im Gegensatz zu den modellexternen OE – auf keine Zulieferer bzw. externe IS angewiesen. Diese OE werden stets von dem Quellobjekt direkt beeinflusst, daher werden sie als Ortsregeln bezeichnet. Wird während eines Kontrollflusses das Quellobjekt erreicht, stehen die dynamischen Basisdaten für das Zielobjekt (automatisch) bereit. Die notwendigen dynamischen Basisdaten werden aus dem Workflow-Modell bzw. aus den einzelnen Workflow-Instanzen abgeleitet, während die statischen Basisdaten Teil des Workflow-Modells sind. Da die Instanz des Modells Basis für die Auflage(n) sind, werden diese OE als modellintern bezeichnet. Die modellinternen OE werden weiter untergliedert in ortsbindende und ortsimplizierende OE. Die Unterscheidung erfolgt anhand des Grades der Bindung: Einer ortsbindenden OE muss exakt ein ortsbeschreibendes Element in Form eines Ortes oder eines Ortstyps zugeordnet werden. Bei einer ortsimplizierenden OE muss hingegen die Zuordnungsliste das ortsbeschreibende Element sein.

Ausnahmen von modellinternen OE sind diejenigen, die nicht ausgeführt werden können. Direkte OE führen – unabhängig vom betrachteten Anwendungsfall – jeweils zu OE, die mit einem Zielobjekt assoziiert werden. Es sind alle statischen Basisdaten vorhanden. Modellexterne OE müssen die entsprechenden Auflagen für die OE selbst definieren. Bei modellinternen OE kann es vorkommen, dass das ortsbeschreibende Element kein Element im Quellobjekt enthält, das mit dem Ausführungsort des Zielobjekts über-

einstimmt. Da unter Umständen auf Grund der fehlenden dynamischen Basisdaten keine OE für das Zielobjekt erzeugt wird, entsteht ein Problem. Dieses ist im WfM bzw. im BPM auch bzgl. anderer fehlenden Daten bekannt. Die meisten WfMS haben keine Kontrollen über den Datenfluss innerhalb der Anwendungsfunktionen bzw. nutzen dieses Wissen nicht. „*Das heißt, das PMS*[1] *weiß nicht, welche Parameter beim Aufruf versorgt bzw. welche Eingabedaten vorhanden sein müssen und welche Rückgabewerte nach dem Aufruf zurückkommen*“ [DaRR11]. In Abb. 4.3 wird der zitierte Sachverhalt visualisiert. Eine weitere Folge dieses Problems ist, dass zum Entwurfszeitpunkt nicht überprüft werden kann, ob die Anwendungsfunktionen mit den Datenflüssen harmonieren.

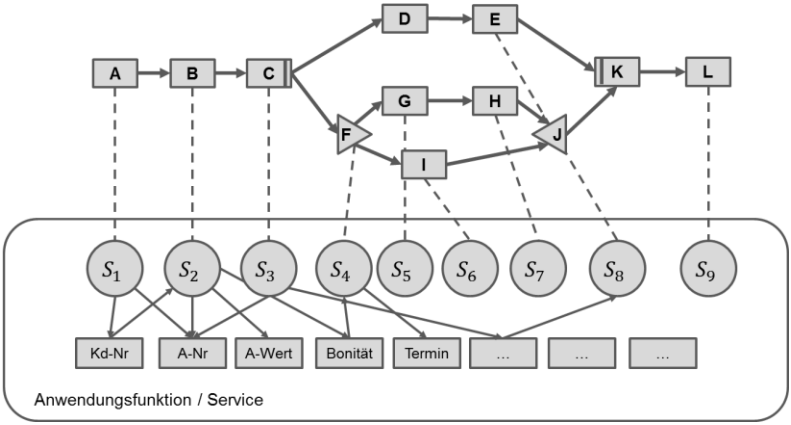


Abb. 4.3: Datenfluss im Workflow nach [DaRR11]

Dem Problem kann mit zwei Strategien entgegnet werden: Entweder berücksichtigt der Kontrollfluss solche fehlenden dynamischen Basisdaten überhaupt nicht, die OE wird somit ausgelassen, oder es muss mittels einer Datenvalidierung innerhalb des Workflows reagiert werden [SOSF04]. Das mWfMS muss somit sicherstellen, dass zu einem bestimmten Zeitpunkt bzw.

[1] Ein Prozess-Managementsystem (PMS) ist eine andere Bezeichnung für eine WfMS.

in einer bestimmten Aktivität innerhalb des Workflows ein bestimmtes Datum vorhanden ist. So kann garantiert werden, dass ein mögliches Quellobjekt einer ortsbindenden OE nur am verknüpften Ort stattfindet, der zum entsprechenden Ortstyp gehört. Für ortsimplizierende OE würde analog dazu die Ausführung des Quellobjektes entsprechend an den Quelleinträgen der Zuordnungsliste gebunden sein.

Im weiteren Verlauf werden verschiedene technische Lösungen vorgestellt. Generell kann die zweite Lösung immer erzwungen werden, indem der Rolle des Modellierers die Berechtigung gegeben wird, eine zusätzliche direkte OE für das Quellobjekt zu definieren, damit nur zulässige Lösungen bzw. Orte definiert werden. Mit diesem *Workaround* könnte weiterführend ermöglicht werden, dass eine Verknüpfung einer direkten OE mit einer Zuordnungsliste erlaubt wird. Vor diesem Hintergrund könnten auch einfache „*if… then… else*"-Regeln in das Modellierungstool integriert werden.

4.2 Das Ortsmodell

Das Konzept der OE sieht vor, dass Aktivitäten an Ausführungsorte gebunden werden. Es ist notwendig, die Orte in einer geeigneten Form zu modellieren, damit eine Abbildung in Datentypen möglich wird. In Kapitel 4.1.1 wurden die verwendeten ortsbeschreibenden Elemente eingeführt. Der Ort, der Ortstyp und die Zuordnungsliste bilden die Basis zur Darstellung von OE in Workflow-Modellen. Für die Modellierung dieser drei Elemente im Workflow-Modell wird weiterführend, um z. B. geographische Lage oder das Ausmaß bzw. die Zusammensetzung der Orte zu bestimmen, ein Ortsmodell eingeführt. Dies ist im Kontext der vorliegenden Arbeit Teil des Metamodells der entsprechenden Modellierungssprache, wird jedoch separat eingeführt, um das Konzept auf weitere Geschäftsprozessmodellierungssprachen anwenden zu können. In vorliegender Arbeit wird die Anwendbarkeit mit Petri-Netzen und BPMN als Modellierungssprache dargestellt (siehe Kapitel 5). Das angewandte Entwurfsprinzip des *separation of concerns* wird seit Dijkstra [Dijk82] in der Informatik beim Entwurf von Softwaresystemen gepflegt. In dieser Arbeit soll damit die Modellierung der Workflow-

Modelle und der entsprechenden Ortsmodellinstanzen getrennt werden. Das erfordert eine Definition einer geeigneten Schnittstelle, damit die Domänen (ortsbeschreibende Elemente und Ortsmodell) wieder verbunden werden können. Die drei eingeführten Elemente müssen mit dem Ortsmodell verknüpft werden. Auf diese Weise kann innerhalb der Workflow-Instanz darauf Bezug genommen werden. Ein weiterer Vorteil dieses Konzeptes ist, dass in dem entsprechenden Metamodell der Modellierungssprache (nur) diese drei Elemente ergänzt bzw. implementiert werden müssen. Die spätere technische Implementierung eines Ortsmodells kann unabhängig davon durchgeführt werden. Ein entsprechendes Element in der Workflow-Instanz benötigt ein Gegenstück im Ortsmodell. Es soll der Name des Objektes als Schlüssel für die Identifikation des Pendants (auf der anderen Seite) dienen. Die Bezeichner müssen eindeutig unterscheidbar sein. Die graphische Repräsentation innerhalb des Workflow-Modells mit Hilfe von Symbolen sind Referenzen auf die Pendants in der Ortsmodellinstanz. Damit wird eine mehrfache Verwendung der ortsbeschreibenden Elemente unterstützt. Die ortsbeschreibenden Elemente sind abstrakt gewählt worden, um keine weiteren (detaillierten) Vorgaben für das Ortsmodell treffen zu müssen. Somit sind verschiedene zwei- oder dreidimensionale Ortsmodelle möglich, die bzgl. ihrer geometrischen Definition variieren können (vgl. z. B. kartesisches Koordinatensystem oder Polarkoordinatensystem [Roth05, S. 274-275]).

Die Schnittstelle zwischen Orts- und Workflow-Modell wird mittels XML-Schema definiert [WWWC04] und ist in in Abb. 4.6 aufgelistet. Der strukturelle Aufbau der XML-Schema-Definition ist in Abb. 4.4 skizziert. In der XML-Datei sind die detailgetreuen Informationen aus der Ortsmodellinstanz enthalten. Sämtliche Informationen einer Ortsmodellinstanz können in der Schnittstelle nicht abgebildet werden. Im Kontext der vorliegenden Arbeit sind die ortsbeschreibenden Elemente abgebildet. Die oben erwähnten Schlüssel bzw. Schlüsselreferenzen und eine Definition von eindeutigen Werten sind in der XML-Schema-Datei dokumentiert. Die XML-Schema-Definition der Schnittstelle garantiert die Abbildungsregeln und die zulässigen Werte [Vlis02, S. 141].

Ein Beispiel für ein mögliches XML-Schema ist in Abb. 4.7 dargestellt. Das Anwendungsbeispiel besteht aus vier Orten mit jeweils zwei Ortstypen und zwei Zuordnungslisten. Die Orte werden den Ortstypen und Zuordnungslisten (nach logischen Zusammenhängen) zugewiesen.

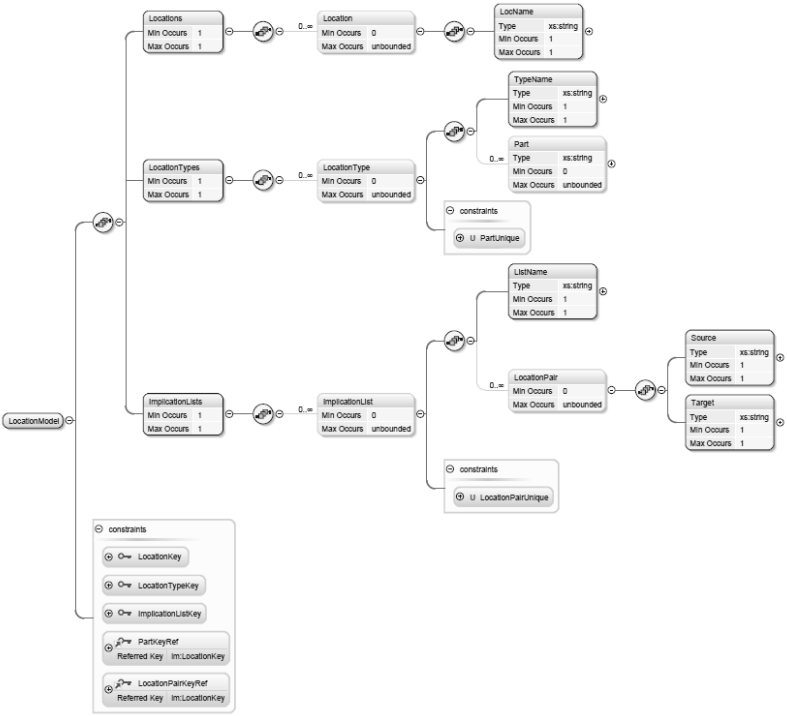


Abb. 4.4: Schnittstelle zwischen Orts- und Workflow-Modell

Die erwähnte Trennung von Orts- und Workflow-Modell bietet weitere Vorteile, da nur ein Metamodell für die Ortsmodellierung implementiert werden muss. So können beispielsweise mehrere Modellierungssprachen um das Prinzip der Ortsbezüge ergänzt werden und es müssen nicht stets Spezialimplementierungen für das Ortsmodell vollzogen werden. Ohne Implemen-

tierung eines Ortsmodells kann die Methode auch für Geschäftsprozesse eingesetzt werden. Bereits bestehende Ortsmodelle müssen lediglich an die drei unterschiedlichen fortzuschreibenden Elemente angepasst werden. Ein Austausch des Ortsmodells ist zu jedem gegebenen Zeitpunkt möglich, da nur die Schnittstelle angepasst werden muss.

Die Informationen aus einer Ortsmodellinstanz werden einem mWfMS bereitgestellt, was im weiteren Verlauf der Arbeit vorausgesetzt wird. Der Datenaustausch erfolgt über die vorgestellte Definition der Schnittstelle. Das vorgestellte XML-Schema kann Daten aus der Ortsmodellinstanz (XML-Dateien) an das mWfMS übergeben (siehe Abb. 4.6). Wie in Kapitel 2.5.3 beschrieben, ist eine der Aufgaben eines mWfMS die Zuweisung von Aktivitäten an Endbenutzer (Rollen). Dies ist eine der Automatisierungen, die ein mWfMS auszeichnet. Allerdings müssen Auflagen der Zielobjekte von OE berücksichtigt werden, wozu weitere Anforderungen bzw. eine weiterführende Funktionalität des mWfMS verlangt werden:

Da jeder **direkten OE** ein ortsbeschreibendes Element zugeordnet ist, muss das mWfMS entscheiden, ob der aktuelle Ort (des Endgerätes) der definierten Auflage des Ausführungsortes entspricht. Es soll im negativen Fall durch das mWfMS garantiert werden, dass die Ausführung (bei einer positiven OE) verhindert wird. Bei **modellinternen OE** ist bei der Betrachtung des Kontrollflusses nach der Ausführung eines Quellobjektes nur ein Ort (Koordinatenpunkt) bekannt. Das mWfMS muss daher stets in der Lage sein, eine Aussage bzgl. des „sich darin Befindens“ abhängig vom ortsbeschreibenden Element treffen zu können, da dies eindeutig an das Zielobjekt weitergeschrieben werden soll. Dynamische Basisdaten **modellexterner OE** müssen von der externen Quelle verarbeitet werden können. Bei **indirekten OE** muss vor der Ausführung des Zielobjektes geprüft werden, ob der aktuelle Ort der Auflage entspricht. Falls dies nicht der Fall ist, muss es vom mWfMS unterbunden werden. Es sollten nur eindeutige Aussagen bzgl. der Auflagen erzeugt werden. Das mWfMS muss im Falle eines Widerspruchs (z. B. Überschneidung zweier Orte eines Ortstyps) Funktionalität bereitstellen, um diese Widersprüche aufzulösen.

Auf Basis dieser Anforderungen wurde ein Ortsmodell als UML-Klassendiagramm entworfen (siehe Abb. 4.5). Das Ortsmodell wird mittels OCL-Ausdrücken formal spezifiziert, die im weiteren Verlauf dieses Kapitels vorgestellt werden.

Die Schnittstelle zwischen Orts- und Workflow-Modell wird mittels der drei ortsbeschreibenden Elemente definiert. Das Ortsmodell muss diese drei Elemente Ort, Ortstyp und Zuordnungsliste beinhalten. Innerhalb des Ortsmodells werden die drei Elemente mittels ihrer englischen Übersetzung (*Location*, *LocationType* und *ImplicationList*) dargestellt (siehe Abb. 4.5).

Ein kartesisches **Koordinatensystem** dient als Basis zur Erläuterung der drei ortsbeschreibenden Elemente. Die einzelnen Bestandteile *Point*, *PolygonPoint* und *Polygon* sind im Ortsmodell als Klassen dargestellt (siehe Abb. 4.5). Zentrales Element des geometrischen Ortsmodells ist der Koordinatenpunkt bzw. die Koordinate, die von der Klasse *Point* ableitbar ist. Eine Koordinate wird im kartesischen Koordinatensystem durch ein 2-Tupel in Form von Werten eindeutig spezifiziert. Ein Wert beschreibt den Abszissenabschnitt und der andere Wert den Ordinatenabschnitt eines Punktes. Sie werden durch die Attribute *xcord* und *ycord* beschrieben (siehe Abb. 4.5). Allerdings wird eine Ergänzung der Klasse *Point* z. B. bei der Anwendung des UTM-Koordinatensystems[2] benötigt, da eine Meridianzone determiniert werden muss [Kuep05, S. 31]. Die Klasse *Point* wird mittels Attribut eingeführt. Weitere Attribute sind erforderlich, wenn z. B. Gitterquadrate verwendet werden.

[2] Das *Universal Transverse Mercator* Koordinatensystem ist ein globales kartesisches Koordinatensystem, das die Erdoberfläche in 60 Meridianzonen einteilt [Roth05].

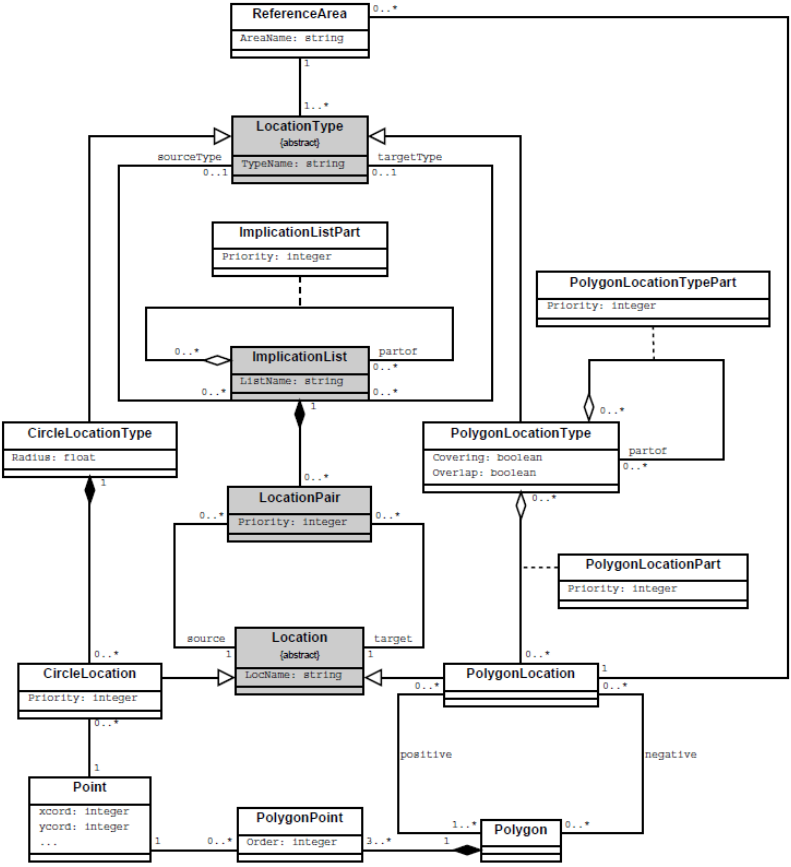


Abb. 4.5: Das Ortsmodell

Ein *Polygon*, als wichtiger Bestandteil des Ortsmodells, wird durch die Klasse *Polygon* beschrieben. Polygone werden durch mindestens drei Koordinatenpunkte determiniert. Die Punkte werden mittels eines Linienzugs miteinander verbunden, so dass letztendlich eine abgeschlossene Hülle (Punkte des Linienzugs) entsteht. Für vier oder mehr Koordinatenpunkte entstehen mehrere Möglichkeiten, das Polygon zu definieren. Es wird im Modell eine zusätzliche Angabe benötigt, um die Position des Punktes innerhalb

des Linienzugs zu determinieren. Die Klasse *PolygonPoint* sorgt dafür, dass ein Objekt vom Typ *PolygonPoint* mit genau einem Punktobjekt assoziiert wird. Die Komposition (vgl. objektorientierter Entwurf) sichert die drei Punkte, die für ein Polygon notwendig sind. Mit Hilfe des Attributs *Order* werden jedem Koordinatenpunkt positive ganzzahlige Werte zugeordnet, beginnend mit 0 aufsteigend um eins jeweils (n+1) bis zum letzten Punkt (N-1). Da die Werte eindeutig sein sollen, müssen die Ordnungswerte der Polygonpunkte paarweise verschieden sein. Diese Bedingung wird mit Hilfe einer OCL[3] beschrieben. Folgender OCL-Ausdruck stellt die richtige Reihenfolge und die Differenzierung der Punkte (paarweise verschieden) sicher:

```
-- Der Ordnungswert eines Polygonpunkts ist nicht negativ
context PolygonPoint
inv:
   self.Order >= 0

-- Die Ordnungswerte der Polygonpunkte eines Polygons sind kleiner, als die
-- Anzahl der Polygonpunkte des Polygons
context Polygon
inv:
   self.polygonPoint
      ->forAll(x | x.Order < self.polygonPoint->size())

-- Die Ordnungswerte der Polygonpunkte eines Polygons sind paarweise verschieden
context Polygon
inv:
   self.polygonPoint
      ->forAll(x1,x2 | x1<>x2 implies x1.Order<>x2.Order)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n + n^2 + n^2 = n + 2n^2$. Für alle n ≥ 2 ist beschränkt der Ausdruck $2n^2$·die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Obiger OCL-Ausdruck (bzw. siehe Ortsmodell in Abb. 4.5) verbietet eine Kreuzung der Linien in der Punktemenge. Da aber gerade dies manchmal erwünscht ist, muss zwischen zwei Arten von Polygonen unterschieden werden [Tous89]. Es wird ein Polygon, dessen Linien sich nicht überschneiden,

[3] Eine OCL kann mittels Sprachtransformation in andere Programmiersprachen übersetzt werden bzw. eine OCL ist interoperabel einsetzbar [KuGo12].

als einfaches Polygon bezeichnet. Die abgeschwächte Form (engl. *weakly-simple polygon*) erlaubt eine parallele Lage bis hin zur Berührung, jedoch keine Überkreuzung [Tous89].

Für die Abbildung von **Orten** ist die Klasse *Location* verantwortlich. Das dazugehörige Attribut *LocName* weist jedem Ortsobjekt einen eindeutigen Bezeichner zu. Es handelt sich um eine abstrakte Klasse, die die Spezialisierungen *CircleLocation* und *PolygonLocation* als Instanziierungen zulässt (siehe Abb. 4.5). Diese beiden Klassen wiederum lassen die Grundfläche eines Ortes bestimmen. Die geometrische Bestimmung der Orte ist je nach Klasse unterschiedlich (Kreis vs. Polygon).

Die Abbildung der **Ortstypen** werden durch die Klasse *LocationType* wiedergegeben. Ein Ortstyp kann mehrere Ortsobjekte zusammenfassen (siehe Kapitel 4.1.1). Auf Grundlage einer gemeinsamen Merkmalsausprägung werden die Ortstypen bestimmt. Diese Eigenschaft wird mit der *Geographic Markup Language*[4] (GML) geteilt, die als Ausprägung *feature* bzw. *feature type* definiert [OpGC07]. Die gebräuchlichen Geoinformationssysteme[5] (GIS) unterstützten *Layer*. *Layer* enthalten immer ein Objekt eines bestimmten Typs und können unabhängig von anderen Objekten ein- bzw. ausgeschalten werden. *Layers* werden übereinanderliegend angeordnet. Jedem Ortstyp ist ein bestimmter Referenzbereich zugeteilt, dessen Beziehung durch eine Assoziation der Klasse *ReferenceArea* geknüpft wird (siehe Abb. 4.5). Diese *ReferenceArea* ist optional und nicht Gegenstand der Definition der Schnittstelle. Der Nutzen aus dieser Klasse wird im weiteren Verlauf der Arbeit ersichtlich. Das Attribut *TypeName* der Klasse *LocationType* ist (wieder) für den eindeutigen Bezeichner des Ortstyps zuständig. Auch *LocationType* ist eine abstrakte Klasse, so dass nur über *CircleLocationType* und *PolygonLocationType* Instanzen erzeugt werden können, bei einer Instanziierung durch ein WfMS.

[4] Die *Geography Markup Language* ist eine XML-basierte Sprache u. a. zum Austausch von Ortsinformationen. http://www.opengeospatial.org/standards/gml

[5] Geoinformationssysteme (GIS) sind Informationssysteme zur Erfassung, Bearbeitung, Organisation, Analyse und Darstellung geographischer Daten. Dazugehörig ist, neben der Software, meist auch die Hardware http://www.freegis.org/.

Orte sind anhand von Polygonen eindeutig beschreibbar. In Abb. 4.5 auf der rechten Seite sind *PolygonLocationType*, *PolygonLocation*, *PolygonLocationPart* und *PolygonLocationTypePart* dargestellt. Im Ortsmodell steht für die Modellierung von Orten in Form von Polygonen die Klasse *PolygonLocation* bereit. Objekte dieser Klasse können einzelne oder mehrere Polygone enthalten. Es wird zwischen zwei Mengen, dem Positivbereich und dem Negativbereich, unterschieden, die über eine Assoziation der Klasse Polygon bestimmt werden. Der Positivbereich besteht aus mindestens einem Polygon, dessen Polygone (falls mehrere) nicht unbedingt zusammenhängend sein müssen. Jedoch ist es erforderlich, bestimmte Flächen (Polygone) wieder aus dem Positivbereich zu entfernen. Der Negativbereich darf im Gegensatz zum Positivbereich auch leer sein. Die Fläche des Positivbereichs wird durch den Negativbereich verkleinert. Dies erlaubt eine flexible Modellierung von polygonförmigen Orten, die u. a. auch Lücken aufweisen können. West-Berlin (innerhalb der Bundesrepublik Deutschland) war bis zur deutschen Einheit z. B. solch eine Ausnahme. Zusätzlich sind die modellierten Polygone wiederverwendbar. Im Ortsmodell wird die Zusammenfassung von Polygonorten zu Polygon-Ortstypen durch die Klasse *PolygonLocationType* unterstützt. Dessen Attribute *Covering* steuern die Überdeckung des Polygon-Ortstyps mit den entsprechenden Polygonorten. Im Falle von *true* muss eine vollständige Abdeckung vorhanden sein. Mit *false* gilt der umgekehrte Fall, d. h. Teile des Referenzbereichs dürfen ungenutzt bleiben. Mit dem Attribut *Overlap* (von *PolygonLocationType*) wird mit *true* ausgedrückt, dass sich die Flächen überlappen dürfen; mit *false* das Gegenteil.

Die eindeutige Ermittlung eines Ortes auf Basis der Angabe von Ortstyp und Koordinatenpunkt muss das Ortsmodell bzw. das mWfMS ermöglichen. Für die Bestimmung des Ausführungsortes einer Aktivität ist dies essentiell. Nicht in jedem Fall werden Polygon-Ortstypen überschneidungsfrei definiert. Es muss eine Lösung geschaffen werden, damit die Vorrangbeziehung eindeutig aufgelöst wird. Daher definiert das Ortsmodell eine einfache Rangfolge der Polygonorte, die durch die Aggregation zwischen den Klassen *PolygonLocationType* und *PolygonLocation* mit der Assoziationsklasse *PolygonLocationPart* ausgedrückt wird. Um einen ganzzahligen eindeutigen

Wert zuzuordnen ist das Attribut *Priority* enthalten. Im folgenden OCL-Ausdruck wird dieser Zusammenhang beschrieben:

```
-- Die einem Polygon-Ortstyp hinzugefügten Polygonorte unterscheiden sich
context PolygonLocationType
inv:
   self.polygonLocation
      ->forAll(x1,x2 | x1<>x2 implies x1.LocName <> x2.LocName)

-- Der Prioritätswert eines Polygonorts ist nicht negativ
context PolygonLocationPart
inv:
   self.Priority >= 0

-- Die Prioritätswerte der Polygonorte eines Polygon-Ortstyps sind kleiner als die
-- Anzahl der Polygonorte des Polygon-Ortstyps
context PolygonLocationType
inv:
   self.polygonLocationPart
      ->forAll(x | x.Priority < self.polygonLocation->size())

-- Die Prioritätswerte der Polygonorte eines Polygon-Ortstyps sind paarweise verschieden
context PolygonLocationType
inv:
   self.polygonLocationPart
      ->forAll(x1,x2 | x1<>x2 implies x1.Priority<>x2.Priority)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n^2 + n + n^2 + \ n^2 = n + 3n^2$. Für alle n ≥ 3 ist beschränkt der Ausdruck $3n^2 \cdot$die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Für den Fall, dass ein Polygon-Ortstyp nicht komplett neu erstellt wird, stellt das Ortsmodell zusätzlich einen Mechanismus bereit. Das Objekt vom Typ *PolygonLocationType* kann Objekte desselben Typs enthalten und wird mittels einer Aggregation modelliert. Die Bestimmung der Vorgängerbeziehung wird mit der Assoziationsklasse *PolygonLocationTypePart* bzw. dem Attribut *Priority* determiniert. Falls sich mehrere Orte überschneiden und aus unterschiedlichen Polygon-Ortstypen bestehen, hat der Ort Priorität, der den höheren Wert besitzt. Da die bereits definierten Vorgängerbeziehungen enthalten bleiben, können auch rekursive Strukturen aufgelöst werden (z. B. Polygone, die wiederrum Polygone beinhalten). Sind in einem zusammen-

gesetzten Polygon-Ortstyp zusätzlich Polygonorte direkt zugeordnet, besitzen diese immer höhere Priorität. Die gerade beschriebenen Eigenschaften sind in folgender OCL spezifiziert:

```
-- Die einem Polygon-Ortstyp hinzugefügten Polygon-Ortstypen unterscheiden sich
context PolygonLocationType
inv:
    self.partof
        ->forAll(x1,x2 | x1<>x2 implies x1.TypeName <> x2.TypeName)

-- Der Prioritätswert eines Polygon-Ortstyps ist nicht negativ
context PolygonLocationTypePart
inv:
    self.Priority >= 0

-- Die Prioritätswerte der enthaltenen Polygon-Ortstypen eines Polygon-Ortstyps sind kleiner als die
-- Anzahl der enthaltenen Polygon-Ortstypen des Polygon-Ortstyps
context PolygonLocationType
inv:
    self.polygonLocationTypePart[partof]
        ->forAll(x | x.Priority < self.polygonLocationTypePart[partof]->size())

-- Die einem Polygon-Ortstyp hinzugefügten Polygon-Ortstypen erhalten unterschiedliche Prioritätswerte
context PolygonLocationType
inv:
    self.polygonLocationTypePart[partof]
        ->forAll(x1,x2 | x1<>x2 implies x1.Priority<>x2.Priority)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n^2 + n + n^2 + n^2 = n + 3n^2$. Für alle n ≥ 3 ist beschränkt der Ausdruck $3n^2$·die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Die direkt zugeordneten - und die hinzugefügten - Polygon-Ortstypen werden mit den Attributen *Covering* und *Overlap* berücksichtigt. Bei kombinierten Polygon-Ortstypen muss generell ein großer Referenzbereich gewählt werden. Falls dies nicht eingehalten wird, werden die Orte, die außerhalb der Grenzen liegen, nicht definiert. Existieren beispielsweise für Bundesländer Polygon-Ortstypen, so wäre eine Kombination der vorhandenen Bundesländer einfacher, als eine neue Deklaration einzuführen.

Die meisten Orte lassen sich mittels Polygonen modellieren. Je nach Anwendungsfall kann es notwendig werden, einen runden Bereich um eine Koordinate zu deklarieren: Wenn der bestimmte Ausführungsort des Zielobjektes einer ortsbindenden OE von der Koordinate des Ausführungsortes des Quellobjektes einen minimalen bzw. maximalen Abstand aufweisen soll. Eine

Lösung mit Hilfe von Polygonorten ist aufwendig und daher wenig geeignet. Daher wird ein eigener Typ innerhalb des Ortsmodells eingeführt, um Kreise im Ortsmodell zu definieren. Ein runder Bereich um einen Koordinatenpunkt bietet zusätzlich die Möglichkeit, Ungenauigkeiten in der Ortung zu berücksichtigen. Die Klasse für Kreise ist *CircleLocation* (siehe Abb. 4.5), die aus *Location* abgeleitet wird. Die Größe des Kreises kann mittels zweier Werte, Koordinate und Radius, definiert werden (siehe Assoziation *CircleLocation* mit Klasse *Point*). Der Radius selbst ist nicht Teil der Klasse *CircleLocation*, sondern Bestandteil eines Kreis-Ortstyps, der eine endliche Menge von Kreisorten beschreibt. Diese Spezialisierung von *LocationType* (*CircleLocationType*) kann nicht ohne die Angabe des Radius existieren. Die Abhängigkeit zwischen *CircleLocationType* und *CircleLocation* wird über eine Komposition definiert (siehe Abb. 4.5). Der Wert des Radius muss größer Null sein, damit ein Kreis erzeugt werden kann. Der Ausdruck besitzt die Komplexität O(1):

```
1 -- Der Radius eines Kreis-Ortstyps ist größer 0
2 context CircleLocationType
3 inv:
4     self.Radius > 0
```

Der Referenzbereich eines Kreis-Ortstyps ist für die Eingrenzung der möglichen Koordinatenpunkte verantwortlich. Es können beliebig viele Punkte sein. Daher sollten in der praktischen Ausführung nur die tatsächlich relevanten Kreisorte dynamisch instanziiert werden. Prinzipiell kann jeder Koordinatenpunkt einen Kreis-Ortstyp bilden. Für ortsbindende OE können Kreis-Ortstypen nützlich sein, da der Ausführungsort des Zielobjekts auf Basis des Abstands zum Quellobjekt beeinflusst wird. Die Abbildung erfolgt mit der Klasse *CircleLocationType*. Soll beispielsweise in einem bestimmten Bereich um das Quellobjekt eine weitere Ausführung verhindert werden, kann eine negative ortsbindende OE, die mit dem Kreis-Ortstyp verknüpft ist, modelliert werden.

Während Kreis-Ortstypen zur Modellierung von ortsbindenden OE eingeführt wurden, können Kreisorte für direkte und modellexterne OE verwendet werden. Eine direkte OE kann mit einem Kreisort verbunden werden. Eine

Bindung der Ausführung des Zielobjekts an die Koordinate erfolgt mittels eines positiven Vorzeichens, wohingegen ein negatives Vorzeichen die Ausführung (im Bereich) untersagt. Auch die externe Datenquelle könnte einen Kreisort liefern. Letztendlich entsprechen die Auflagen für den Zielort den vorab in diesem Kapitel beschriebenen Bedingungen. Eine überschneidungsfreie eindeutige Interpretation muss für Instanzen eines Kreisortes, sofern nicht dynamisch instanziiert, gewährleistet werden. Das Attribut *Priority* gewährleistet dieses Verhalten:

```
-- Der Prioritätswert eines Kreisorts ist nicht negativ
context CircleLocation
inv:
   self.Priority >= 0

-- Die Prioritätswerte der Kreisorte eines Kreis-Ortstyps sind kleiner als die
-- Anzahl der Kreisorte des Kreis-Ortstyps
context CircleLocationType
inv:
   self.circelLocation
      ->forAll(x | x.Priority < self.circleLocation->size())

-- Die Prioritätswerte der Kreisorte eines Kreis-Ortstyps sind paarweise verschieden
context CircleLocationType
inv:
   self.circelLocation
      ->forAll(x1,x2 | x1<>x2 implies x1.Priority<>x2.Priority)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n + n^2 + n^2 = n + 2n^2$. Für alle n $\geq$ 2 ist beschränkt der Ausdruck $2n^2$·die Funktion T(n). Damit gilt für die Funktion T(n) $\in O(n^2)$.

Die **Zuordnungsliste**, repräsentiert durch *ImplicationList*, wird analog zu den Orten und Ortstypen mittels eindeutiger Bezeichner identifiziert (siehe Abb. 4.5, Attribut *ListName*). Die notwendigen Komponenten für die Modellierung der Zuordnungslisten sind in Abb. 4.5 dargestellt (siehe Klassen *LocationPair* und *ImplicationListPair*). Die binären Relationen der Zuordnungslisten, sind mittels der Komposition zwischen *ImplicationList* und *LocationPair* definiert. Die Klasse *LocationPair* wird verwendet, um Ortspaare zu modellieren. Ortspaare bestehen aus einem Quell- und einem Zie-

leintrag, die jeweils über Assoziationen der Klasse *LocationPair* und *Location* (siehe Abb. 4.5) verknüpft sind. Quell- bzw. Zieleinträge dürfen vom Typ *CircleLocation* und auch *PolygonLocation* sein. Zur Einschränkung können die Einträge auf bestimmte Ortstypen begrenzt werden. Dazu sind die beiden Assoziationen in *ImplicationList* und *LocationType* vorhanden. Bei einer Eingrenzung der Quell- und Zieleinträge auf bestimmte Ortstypen, müssen die folgenden Bedingungen eingehalten werden:

```
-- Wenn ein Quell-Ortstyp festgelegt wurde, müssen alle Quelleinträge der
-- Ortspaare diesem Ortstyp angehören
context ImplicationList
inv:
   if(self.sourceType->notEmpty())
   then
      if(self.sourceType.oclIsTypeOf(PolygonLocationType))
      then
         self.locationPair.source
            ->forAll( x | x.oclIsInstanceOf(PolygonLocation)
               and x.polygonLocationType
                  ->exists( y | y.TypeName = self.sourceType.TypeName))
      else
         self.locationPair.source
            ->forAll( x | x.oclIsInstanceOf(CircleLocation)
               and x.circleLocationType.TypeName = self.sourceType.TypeName)
      endif
   endif

-- Wenn ein Ziel-Ortstyp festgelegt wurde, müssen alle Zieleinträge der
-- Ortspaare diesem Ortstyp angehören
context ImplicationList
inv:
   if(self.targetType->notEmpty())
   then
      if(self.targetType.oclIsTypeOf(PolygonLocationType))
      then
         self.locationPair.target
            ->forAll( x | x.oclIsInstanceOf(PolygonLocation)
               and x.polygonLocationType
                  ->exists( y | y.TypeName = self.targetType.TypeName))
      else
         self.locationPair.target
            ->forAll( x | x.oclIsInstanceOf(CircleLocation)
               and x.circleLocationType.TypeName = self.targetType.TypeName)
      endif
   endif
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n + n = 2n$. Für alle n ≥ 2 ist beschränkt der Ausdruck $2n$ die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n)$.

Mehrere Tupel können das Ergebnis einer Auswertung des Koordinatenpunktes sein, wenn z. B. nach einer Ausführung des Quellobjektes eine ortsimplizierende OE vorhanden ist. Der Grund dafür liegt in der Definition einer Zuordnungsliste, da sie mehrere Tupel mit dem gleichen Quelleintrag enthalten darf. Die Auflagen von OE basieren dann auf allen Tupeln (siehe Kapitel 4.1). Positive Vorzeichen beschränken die Zielobjektausführung auf einen beliebigen Ort (als Zieleintrag eines Tupels), während negative Vorzeichen die Ausführung (genau dort) untersagen. Überschneidungen der durch Quelleinträge referenzierten Orte sind möglich. Es können mehrere unterschiedliche Tupel Bedeutung haben. Wenn beispielsweise die Koordinate im Überschneidungsbereich liegt, kann der Ausführungsort des Quelleintrags nicht eindeutig ermittelt werden. Eine ähnliche Problematik ist zu Beginn dieses Kapitels beschrieben worden. Analog dazu werden hier Prioritätswerte für das Ortsmodell eingeführt. Die Vorgängerbeziehungen der Tupel werden auf Basis von Prioritäten geregelt. Höhere Werte werden zuerst berücksichtigt und durch einen ganzzahligen Wert des Attributs *Priority* in der Klasse *LocationPair* ausgedrückt. Diese Werte müssen stets für alle Ortspaar-Objekte eindeutig sein:

```
-- Die einer Zuordnungsliste hinzugefügten Ortspaare unterscheiden sich
context ImplicationList
inv:
   self.locationPair
      ->forAll(x1,x2 | x1<>x2 implies
         x1.source <> x2.source or x1.target <> x2.target)

-- Der Prioritätswert eines Ortspaars ist nicht negativ
context LocationPair
inv:
   self.Priority >= 0

-- Die Prioritätswerte der Ortspaare einer Zuordnungsliste sind kleiner als die
-- Anzahl der Ortspaare der Zuordnungsliste
context ImplicationList
inv:
   self.locationPair
      ->forAll(x | x.Priority < self.locationPair->size())

-- Die Prioritätswerte der Ortspaare einer Zuordnungsliste sind paarweise verschieden
context ImplicationList
inv:
   self.locationPair
      ->forAll(x1,x2 | x1<>x2 implies x1.Priority<>x2.Priority)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n^2 + n + n^2 + \ n^2 = n + 3n^2$. Für alle n ≥ 3 ist beschränkt der Ausdruck $3n^2 \cdot$die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Die obigen vier Ausdrücke gewährleisten eine korrekte Reihenfolge. Die Komplexität wird mit O(n*n) abgeschätzt. Die Tupel mit demselben referenzierten Ort im Quelleintrag (wie das ermittelte Tupel) sind jedoch ebenfalls für die Erzeugung der Auflage, unabhängig von den Prioritätswerten, relevant. Zuordnungslisten sollen – wie die Ortstypen – zu Einheiten aggregiert werden können. Die Klasse *ImplicationList* ist als Aggregation mit sich selbst verknüpft (siehe Abb. 4.5). Es beinhaltet eine zusätzliche Assoziationsklasse mit dem Namen *ImplicationListPart*. Mit dem Attribut *Priority* soll die eindeutige Vorgängerbeziehung zwischen den Tupeln geregelt werden:

```
1  -- Die einer Zuordnungsliste hinzugefügten Zuordnungslisten unterscheiden sich
2  context ImplicationList
3  inv:
4     self.partof
5        ->forAll(x1,x2 | x1<>x2 implies x1.ListName <> x2.ListName)
6
7  -- Der Prioritätswert einer Zuordnungsliste ist nicht negativ
8  context ImplicationListPart
9  inv:
10    self.Priority >= 0
11
12 -- Die Prioritätswerte der enthaltenen Zuordnungslisten einer Zuordnungsliste sind
13 -- kleiner als die Anzahl der enthaltenen Zuordnungslisten der Zuordnungsliste
14 context ImplicationList
15 inv:
16    self.implicationListPart[partof]
17       ->forAll(x | x.Priority < self.implicationListPart[partof]->size())
18
19 -- Die einer Zuordnungsliste hinzugefügten Zuordnungslisten erhalten
20 -- unterschiedliche Prioritätswerte
21 context ImplicationList
22 inv:
23    self.implicationListPart[partof]
24       ->forAll(x1,x2 | x1<>x2 implies x1.Priority<>x2.Priority)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n^2 + n + n^2 + n^2 = n + 3n^2$. Für alle n ≥ 3 ist beschränkt der Ausdruck $3n^2$·die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Bei einer zusammengesetzten Zuordnungsliste (aus bereits vorhandenen Definitionen) bleibt die (eventuell vorhandene) Beschränkung der Quell- und Zieleinträge erhalten. Restriktionen wirken sich somit nur auf die enthaltenen Ortspaare aus. Zur Vermeidung einer unendlichen Rekursion (Schleife) ist für Zuordnungslisten ein spezieller OCL-Ausdruck definiert worden:

```
-- Ermittlung der transitiven Hülle einer Zuordnungsliste
context ImplicationList
dev:
   allparts() : Set(ImplicationList)
      = self.partof->union(self.partof.allparts())

-- Eine Zuordnungsliste darf sich nicht selbst enthalten
context ImplicationList
inv:
   allparts()->excludes(self)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n + n = 2n$. Für alle n ≥ 2 ist beschränkt der Ausdruck $2n$ die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n)$.

Um ortsbeschreibende Elemente im Workflow-Modell den Objekten des Ortsmodells zuzuordnen - wie anfangs des Kapitels erwähnt - müssen die Bezeichner eindeutig sein. Die Attributwerte *LocName*, *TypeName* und *ListName* (im Ortsmodell) müssen eindeutig sein und innerhalb der Klasse verschieden gewählt werden. Zwei Bezeichner unterschiedlicher Art mit gleichem Namen werden nicht empfohlen, da dies innerhalb einer Implementierung eines mWfMS zu Problemen führen könnte. Die eindeutigen Bezeichner für Orte, Ortstypen und Zuordnungslisten werden durch folgenden OCL-Ausdruck spezifiziert:

```
-- Der Name eines Ortes ist eindeutig
context Location
inv:
   self.allInstances
      ->forAll(x1,x2 | x1<>x2 implies x1.LocName<>x2.LocName)

-- Der Name eines Ortstyps ist eindeutig
context LocationType
inv:
   self.allInstances
      ->forAll(x1,x2 | x1<>x2 implies x1.TypeName<>x2.TypeName)

-- Der Name einer Zuordnungsliste ist eindeutig
context ImplicationList
inv:
   self.allInstances
      ->forAll(x1,x2 | x1<>x2 implies x1.ListName<>x2.ListName)
```

Wenn der OCL-Ausdruck innerhalb einer Zeitfunktion T(n) abgebildet wird, so erhält man: $T(n) = n^2 + n^2 + n^2 = 3n^2$. Für alle n ≥ 3 ist beschränkt der Ausdruck $3n^2$·die Funktion T(n). Damit gilt für die Funktion T(n) ∈ $O(n^2)$.

Für die Abschätzung der Komplexität des ganzen Ortsmodells können die einzelnen Komplexitätsabschätzungen der OCLs addiert bzw. vereinfacht werden:

$$O(n^2) + O(n) + O(n^2) + O(n^2) + O(n) + O(n^2) + O(1) + O(n^2) + O(n^2) + O(n^2)$$

$$= 7 * O(n^2) + 2 * O(n) + O(1) = O(n^2)$$

Mit einer effizienten Implementierung bzw. mit jeweils effizienten Algorithmen kann die Komplexität auf O(n log n) reduziert werden [SaSa10, S. 115ff].

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:lm="http://mynamespace.mydomain.org"
targetNamespace="http://mynamespace.mydomain.org"
elementFormDefault="qualified">
  <xs:element name="LocationModel">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Locations">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Location" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="LocName" type="xs:string" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="LocationTypes">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="LocationType" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="TypeName" type="xs:string" />
                    <xs:element name="Part" type="xs:string" minOccurs="0"
                                                          maxOccurs="unbounded" />
                  </xs:sequence>
                </xs:complexType>
                <xs:unique name="PartUnique">
                  <xs:selector xpath="lm:Part" />
                  <xs:field xpath="." />
                </xs:unique>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="ImplicationLists">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ImplicationList" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="ListName" type="xs:string" />
                    <xs:element name="LocationPair" minOccurs="0" maxOccurs="unbounded">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="Source" type="xs:string" />
                          <xs:element name="Target" type="xs:string" />
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
                <xs:unique name="LocationPairUnique">
                  <xs:selector xpath="lm:LocationPair" />
                  <xs:field xpath="lm:Source" />
                  <xs:field xpath="lm:Target" />
                </xs:unique>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
```

(Fortsetzung nächste Seite)

```
69      <xs:key name="LocationKey">
70        <xs:selector xpath="lm:Locations/lm:Location" />
71        <xs:field xpath="lm:LocName"></xs:field>
72      </xs:key>
73      <xs:key name="LocationTypeKey">
74        <xs:selector xpath="lm:LocationTypes/lm:LocationType" />
75        <xs:field xpath="lm:TypeName"></xs:field>
76      </xs:key>
77      <xs:key name="ImplicationListKey">
78        <xs:selector xpath="lm:ImplicationLists/lm:ImplicationList" />
79        <xs:field xpath="lm:ListName"></xs:field>
80      </xs:key>
81      <xs:keyref name="PartKeyRef" refer="lm:LocationKey">
82        <xs:selector xpath="lm:LocationTypes/lm:LocationType/lm:Part" />
83        <xs:field xpath="." />
84      </xs:keyref>
85      <xs:keyref name="LocationPairKeyRef" refer="lm:LocationKey">
86        <xs:selector xpath="lm:ImplicationLists/lm:ImplicationList/lm:LocationPair/lm:*" />
87        <xs:field xpath="." />
88      </xs:keyref>
89    </xs:element>
90  </xs:schema>
```

Abb. 4.6: XML-Schema Workflow Ortsmodell

```xml
<?xml version="1.0" encoding="utf-8"?>
<LocationModel xmlns="http://mynamespace.mydomain.org"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://mynamespace.mydomain.org lm.xsd">
  <Locations>
    <Location>
      <LocName>Ort 01</LocName>
    </Location>
    <Location>
      <LocName>Ort 02</LocName>
    </Location>
    <Location>
      <LocName>Ort 03</LocName>
    </Location>
    <Location>
      <LocName>Ort 04</LocName>
    </Location>
  </Locations>
  <LocationTypes>
    <LocationType>
      <TypeName>Ortstyp T1</TypeName>
      <Part>Ort 01</Part>
      <Part>Ort 02</Part>
    </LocationType>
    <LocationType>
      <TypeName>Ortstyp T2</TypeName>
      <Part>Ort 02</Part>
      <Part>Ort 03</Part>
    </LocationType>
  </LocationTypes>
  <ImplicationLists>
    <ImplicationList>
      <ListName>Zuordnungsliste Z1</ListName>
      <LocationPair>
        <Source>Ort 01</Source>
        <Target>Ort 03</Target>
      </LocationPair>
      <LocationPair>
        <Source>Ort 03</Source>
        <Target>Ort 02</Target>
      </LocationPair>
    </ImplicationList>
    <ImplicationList>
      <ListName>Zuordnungsliste Z2</ListName>
      <LocationPair>
        <Source>Ort 01</Source>
        <Target>Ort 03</Target>
      </LocationPair>
    </ImplicationList>
  </ImplicationLists>
</LocationModel>
```

Abb. 4.7: Beispiel-Ortsmodell in XML-Darstellung

4.3 Die Notation von Ortseinschränkungen

Eine ähnliche Erweiterung für UML-Aktivitätsdiagramme [Deck11, S. 187-265] und von Pr/T-Netzen (eine höhere Form von Petri-Netzen mit individuellen Marken) [Deck11, S. 279-301] existiert bereits. Eine Integration in (einfache) Petri-Netze und BPMN 2.0 erfolgt in Kapitel 5. Zu Beginn des vorliegenden Kapitels 4.3.1 werden die graphischen Elemente zur Definition von Ortseinschränkungen eingeführt. In Kapitel 4.3.2 werden Aggregationen zur Darstellung von Ortsbezügen vorgestellt. Hierarchische Modellelemente und das Verhalten der Annotation mit Verfeinerungen werden in Kapitel 4.3.2 betrachtet. Eine einfache modellübergreifende Verknüpfung für OE wird in Kapitel 4.3.4 eingeführt. Inaktive OE bzw. eine Behandlung von OE, deren Basisdaten zur Laufzeit nicht vorhanden sind, werden in Kapitel 4.3.5 diskutiert.

4.3.1 Die graphischen Elemente

In [Ober96, S. 31] werden Anforderungen an Modellierungssprachen beschrieben. Weitere Anforderungen bzw. Voraussetzungen der vorliegenden Arbeit zur Entwicklung einer graphischen Notation sind:

Die eingeführten Elemente zur Definition von Ortsbezügen müssen eindeutig innerhalb der graphischen Notation abgebildet werden. Da die Notation in mehreren Modellierungssprachen angewandt werden soll, müssen die graphischen Symbole so ausgewählt werden, dass sie möglichst mit keiner der verwendeten Sprachen im Konflikt stehen (in vorliegender Arbeit BPMN 2.0 und Petri-Netze). Die Notation sollte wenige Symbole enthalten, damit sie übersichtlich und leicht erlernbar ist. Eine „Überfrachtung“ des Workflow-Modells muss vermieden werden.

Es werden zwei Arten von graphischen Elementen unterschieden: Die Gruppe der Symbole bildet die verschiedenen Ortsbezüge ab. Die Kanten dienen als Verknüpfungselemente. Die Ortsbezüge lassen sich in einer Baumdarstellung strukturieren (siehe Abb. 4.8).

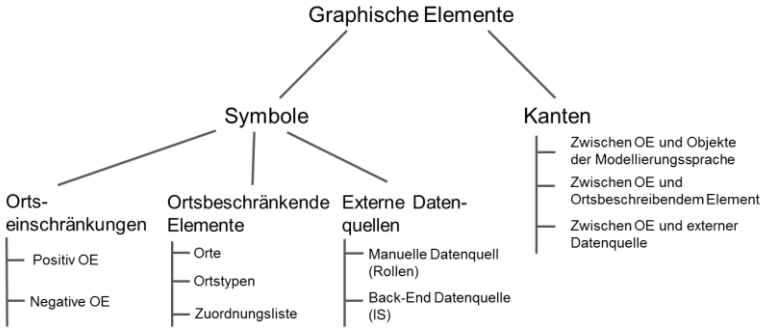


Abb. 4.8: Graphische Elemente von Ortseinschränkungen

Zur Modellierung eines Ortsbezugs soll ein neues Symbol eingeführt werden. Alle direkten und indirekten OE werden mit Hilfe von weiteren Annotationen ersichtlich. Um kein graphisches Objekt einer bekannten Modellierungssprache zu verwenden (siehe Kapitel 2.3), wurde eine gestrichelte Linie gewählt (siehe Abb. 4.9). Durch diese Darstellung kann das Symbol gut von den existierenden Kontroll- oder Nachrichtenflüssen abgegrenzt werden.

Die Form in Abb. 4.9 wird im weiteren Verlauf als generische Grundform der OE bezeichnet und ist als Quadrat dargestellt, da noch keine Informationen über die Ausprägung der OE enthalten sind. Jede OE muss ein zugeordnetes Vorzeichen besitzen (siehe Kapitel 4.1), um zwischen positiver und negativer OE differenzieren zu können. Die Form wird mit einer zusätzlichen Dekoration versehen. Die Dekoration („+“) ist für eine positive OE gewählt worden. Analog dazu gilt für eine negative OE die Beschriftung („-“) (siehe Abb. 4.9).

Abb. 4.9: Form einer OE (links), Positive OE (Mitte), Negative OE (rechts)

Die Vorzeichen gelten für alle unterschiedlichen Arten von OE (siehe Abb. 4.1). Sie sind jedoch nicht in der Lage, weitere Ausprägungsarten, wie z. B. indirekte OE zu unterscheiden. Es wird eine weitere Differenzierung der vorgestellten generischen Form entworfen, die über entsprechende Kanten zwischen den Objekten realisiert wird.

Die **ortsbeschreibenden Elemente** sind ein wesentliches Merkmal der graphischen Repräsentation von OE (siehe Kapitel 4.1.1). Die Trennung von Orts- und Workflow-Modell verdeutlicht, dass sowohl Lage und Ausmaß bzw. die Zusammensetzung von ortsbeschreibenden Elementen Bestandteil einer Ortsmodellinstanz sind (siehe Abb. 4.5). Die Symbole sind damit lediglich Referenzen auf Objekte einer Ortsmodellinstanz und können mehrfach innerhalb des Workflow-Modells angewandt werden, indem die Bindung an das entsprechende Objekt über den eindeutig zugewiesenen Namen erfolgt.

Analog zur graphischen Notation von OE wird für die ortsbeschreibenden Elemente ein eigenes Symbol mit gestrichelter Linie eingeführt (siehe Abb. 4.10).

Abb. 4.10: Form eines generischen ortsbeschreibenden Elements

Zur Unterscheidung der drei verschiedenen ortsbeschreibenden Elemente wird die generische Form um eine zusätzliche Dekoration ergänzt. Ein Präfix vor dem eindeutigen Namen des ortsbeschreibenden Elements unterstützt die eindeutige Interpretation im mWfMS und lässt sich zusätzlich leicht durch eine spezielle Symbolik ersetzen. In Abb. 4.11 werden die drei verschiedenen graphischen Repräsentationen der ortsbeschreibenden Elemente dargestellt.

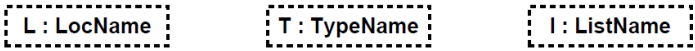


Abb. 4.11: Die drei ortsbeschreibenden Elemente

In Abb. 4.11 ist mit dem „L“ (engl. *location*) das Präfix für Orte dargestellt, gefolgt von dem eigentlichen Namen des Ortes. Das Präfix „T“ (engl. *type*) bezeichnet den Ortstyp, gefolgt von dem Namen des Ortstyps. Analog dazu wird durch den Buchstaben „I“ (engl. implication) die Zuordnungsliste mit dem zugehörigen eindeutigen Bezeichner deklariert. Auf diese Weise lassen sich alle drei ortsbeschreibenden Elemente voneinander abgrenzen.

Direkte Ortseinschränkungen beeinflussen die Ausführungsorte von Aktivitäten basierend auf Informationen, die schon im Workflow-Modell hinterlegt sind. Dabei bestehen die – als direkte Basisdaten eingeführten – Informationen jeweils aus drei unterschiedlichen Bestandteilen. Mit dem Vorzeichen, einem Zielobjekt und der Angabe des ortsbeschreibenden Elements lässt sich eine vollständige Definition einer direkten OE vornehmen. Die Definition des Vorzeichens ist in Abb. 4.9 eingeführt worden. Das Zielobjekt, dessen Ausführungsort den OE entsprechen muss – als zweiter Bestandteil der direkten Basisdaten – muss nun in Relation zu den direkten OE gesetzt werden. Dies erfordert eine Kante zwischen dem Objekt der OE hin zum Zielobjekt. Auf Grund des Nachrichtenflusses wird eine gerichtete Kante in Form einer gestrichelten Linie modelliert. Weiterer Bestandteil der statischen Basisdaten ist eines der ortsbeschreibenden Elemente (siehe Abb. 4.11). Zwischen ortsbeschreibendem Element und Symbol der OE muss eine weitere Beziehung in Form einer gerichteten gestrichelten Kante – vom ortsbeschreibenden Element ausgehend – eingefügt werden. Es können insgesamt sechs unterschiedliche Kombinationen (zwei unterschiedliche Vorzeichen *mal* drei unterschiedliche ortsbeschreibende Elemente) von direkten OE modelliert werden (siehe Abb. 4.12).

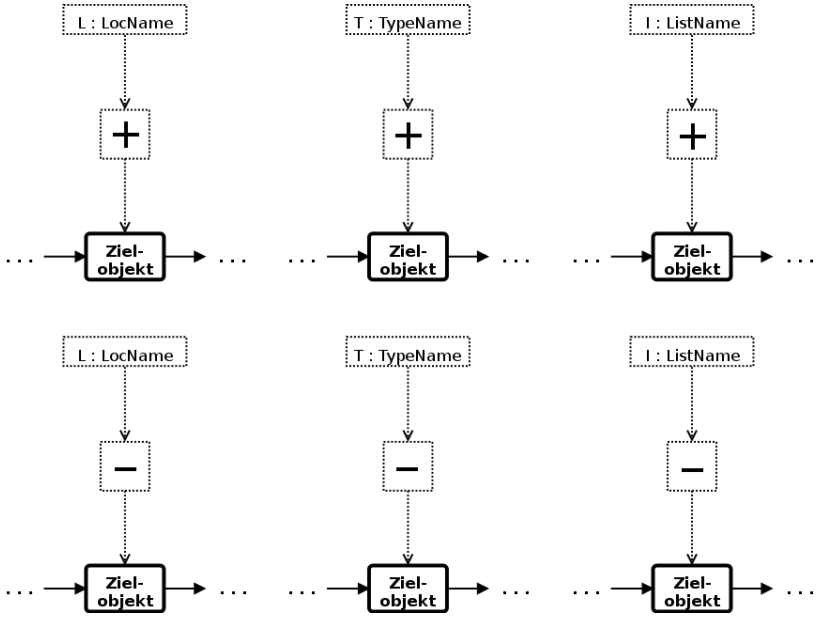


Abb. 4.12: Graphische Darstellung direkter OE

Da die Kombination von OE mit mehreren ortsbeschreibenden Elementen mehrere Interpretationen des Modells zulassen würden, ist jeweils einer OE nur ein ortsbeschreibendes Element zugeordnet. Falls eine Mehrfachinterpretation ermöglicht werden soll, müsste ein weiteres Notationselement eingeführt werden. Das ist jedoch nicht erstrebenswert, da hierdurch die Komplexität der Annotation zunehmen würde. Die Kombination von ortsbeschreibenden Elementen ist nicht notwendig, da die gewünschten Kombinationen innerhalb des Ortsmodells in Form eines neuen Ortstyps abgebildet und als ortsbeschreibendes Elemente im Workflow-Modell verwendet werden. Auf diese Weise sind die modellierten OE zusätzlich intuitiver interpretierbar, da das Zielobjekt von z. B einer positiven OE an einem beliebigen Ort des Ortstyps stattfindet. Im negativen Fall darf es natürlich an keinem Ort des Ortstyps ausgeführt werden. Anwendungsbeispiele direkter OE werden in Abb. 4.13 dargestellt.

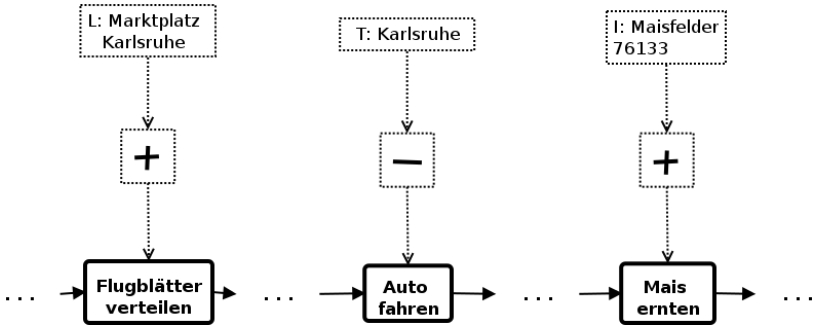


Abb. 4.13: Anwendungsbeispiele direkter OE

Von links nach rechts in Abb. 4.13 wird eine direkte OE modelliert, die es nur am Marktplatz Karlsruhe erlaubt Flugblätter zu verteilen. Die nächste OE untersagt das Auto fahren im Stadtgebiet von Karlsruhe (z. B. auf Grund der Umweltzone). Die letzte OE erlaubt die Ernte auf allen Maisfelder im Polygon der Postleitzahl 76133.

Indirekte Ortseinschränkungen bestehen aus dynamischen Basisdaten, d. h. die Daten stehen erst während der Ausführung der Workflow-Instanz zur Verfügung. Da die Modellierung der Workflows zum Zeitpunkt der Designphase stattfindet (siehe Kapitel 3.1), kann die Modellierung dieser dynamischen Anteile anhand der Typen der ortsbeschreibenden Elemente durchgeführt werden. Die statischen Basisdaten müssen jedoch explizit modelliert werden. Die indirekten OE bestehen somit aus Vorzeichen, dem Quellobjekt und einem Zielobjekt. Das Vorzeichen und das Quellobjekt werden analog zu der beschriebenen graphischen Präsentation (s. o.) definiert. Als neuer Bestandteil wird das Quellobjekt mit einer gerichteten Kante an die indirekte OE gebunden.

Es werden zwei Arten von indirekten OE (die modellexternen und die modellinternen OE) unterschieden (siehe Kapitel 4.1.4 und Kapitel 4.1.5). **Modellexterne indirekte OE** beziehen im Verlauf der Ausführung einer Workflow-Instanz Daten aus externen Datenquellen. Dabei können für jede

abgeleitete Workflow-Instanz jeweils individuell Daten bereitgestellt werden. Zum Entwurfszeitpunkt sind diese Daten noch nicht verfügbar. Es wird bei modellexternen indirekten OE zwischen zwei Arten externer Datenquellen unterschieden: Die Datenquellen in Form von Rollen, die die Datenquellen mit dynamischen Basisdaten versehen können und die Datenquellen aus externen IS, die sogenannte Backend-OE (System-OE) enthalten (siehe Abb. 4.14).

Abb. 4.14: Graphische Darstellung externer Datenquellen

Die (universelle) Form für die Beschreibung von ortsbeschreibenden Elementen in Kombination mit Präfix und einem Bezeichner ist ausschlaggebend für die Art der externen Datenquelle. Das Präfix „R“ (engl. *role*) steht für eine manuelle Datenquelle die von einer Rolle ausgefüllt wird. Das Präfix „S“ (engl. *system*) verdeutlicht, dass die notwendigen Informationen aus einem IS stammen (siehe Abb. 4.15). Praktische Anwendungsbeispiele sind in Abb. 4.16 dargestellt.

Im Beispiel links (siehe Abb. 4.16) soll eine Stromtrasse bzw. Stromleitung auf Grund eines Defekts repariert werden. Um zu gewährleisten, dass die Nachkontrolle der Reparatur auch am gleichen Ort wie die Reparatur stattfindet, wird vom Ort der Reparatur eine positive indirekte OE für den Prüfer (Kontrolleur) bzw. seine Kontrolle der Stromleitung definiert. Im zweiten Beispiel links wird eine Kundenadresse in einem Kundencenter in das CRM einer Firma aufgenommen. Die Firma hat den Kunden jeweils ein spezifisches Kundencenter zugewiesen und möchte, dass die Kunden nur ihr zugewiesenes Kundencenter nutzen. Mit der OE wird sichergestellt, dass eine Aktualisierung der Stammdaten nur in diesem Kundencenter vorgenommen wird.

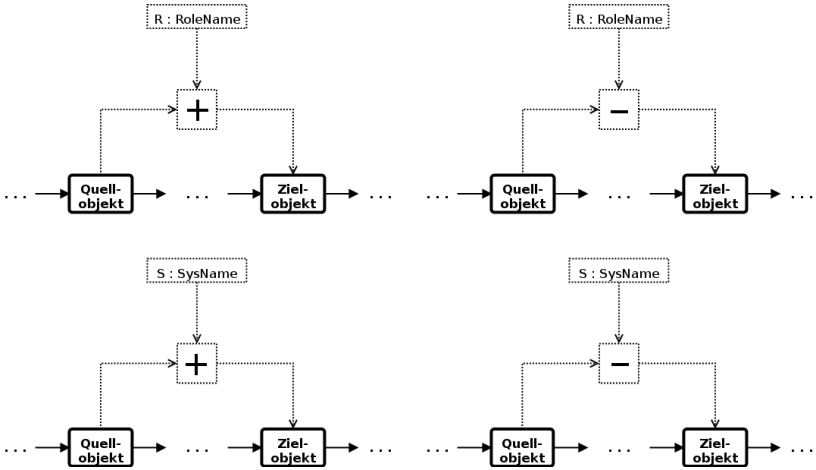


Abb. 4.15: Graphische Darstellung modellexterner indirekter OE

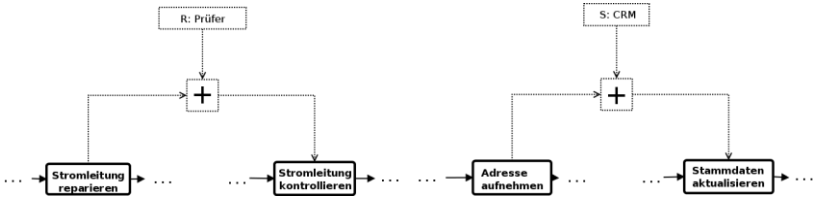


Abb. 4.16: Anwendungsbeispiele indirekter OE

OE sind während des Kontrollflusses erzeugbar, sobald das Quellobjekt erreicht wird. Jedoch kann bei mobilen WfMS nicht zu jedem Zeitpunkt garantiert werden, dass die Basisdaten vorliegen. Ein temporärer Verbindungsabbruch zum Netzprovider könnte beispielsweise den Bezug der Basisdaten verhindern. Das mobile WfMS muss auf solche Verbindungsabbrüche vorbereitet sein und ggfs. ein nochmaliges Beziehen der Basisdaten eigenständig durchführen, sobald die Konnektivität wiederhergestellt ist.

Bei einem längeren Ausfall der Verbindung können jedoch die Basisdaten nicht bezogen werden. Es entsteht somit das Problem, dass die Ausführung der OE nicht garantiert werden kann. Dies muss bei der Modellierung (mit einer speziellen Symbolik oder einer farblichen Markierung) berücksichtigt werden. Um dieser mobil-spezifischen Eigenschaft gerecht zu werden, sollte die Möglichkeit bestehen, Quellobjekte von modellexternen OE per Konfiguration auszulassen. Das Symbol dieser OE ist nicht mit einem Quellobjekt assoziiert (siehe Abb. 4.17).

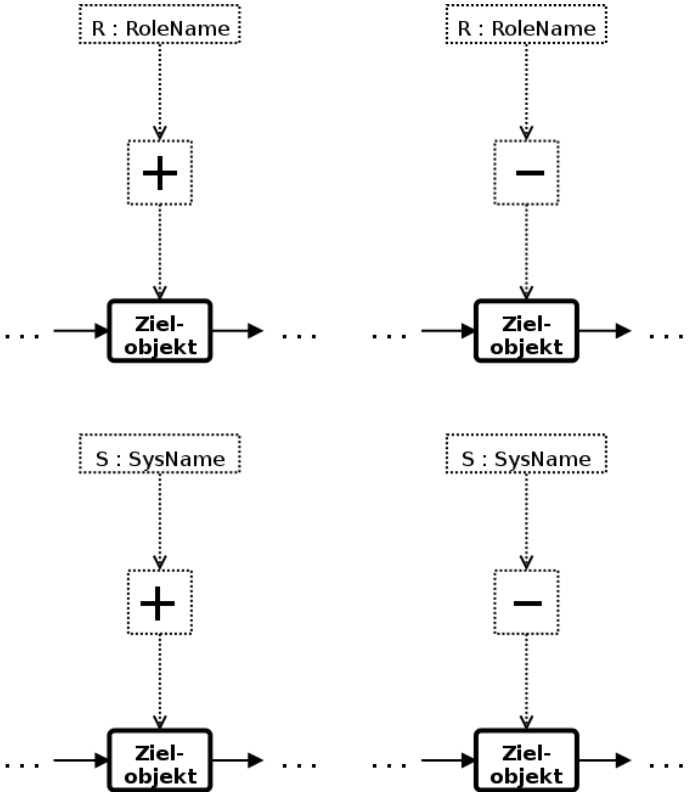


Abb. 4.17: Graphische Darstellung modellexterner OE ohne Quellobjekte

Indirekte OE können jedoch erst dann vollständig interpretiert werden, wenn die notwendigen Basisdaten vorhanden sind (siehe Kapitel 4.1.3). Die Basisdaten müssen vor der Ausführung des Zielobjektes verfügbar sein. Dies gilt für alle modellexternen OE. Falls keine dynamischen Basisdaten für die entsprechenden OE vorhanden sind, wird die OE als „inaktiv“ markiert.

Modellinterne indirekte OE besitzen analog zu den modellexternen OE ein Vorzeichen, ein Quell- und ein Zielobjekt. Zusätzlich sind die ortsbeschreibenden Elemente Teil der modellinternen OE. Eine gerichtete Kante an die OE wird direkt angebunden, wobei jeweils das ortsbeschreibende Element den Ausgangspunkt darstellt (siehe Abb. 4.18). Ortsbindende OE beinhalten die Verknüpfung mittels eines Orts oder eines Ortstyps (vgl. Kapitel 4.1.5). Modellinterne indirekte OE (ortsimplizierende OE) besitzen die gleiche Funktionalität wie ortsbindende OE (siehe Abb. 4.19).

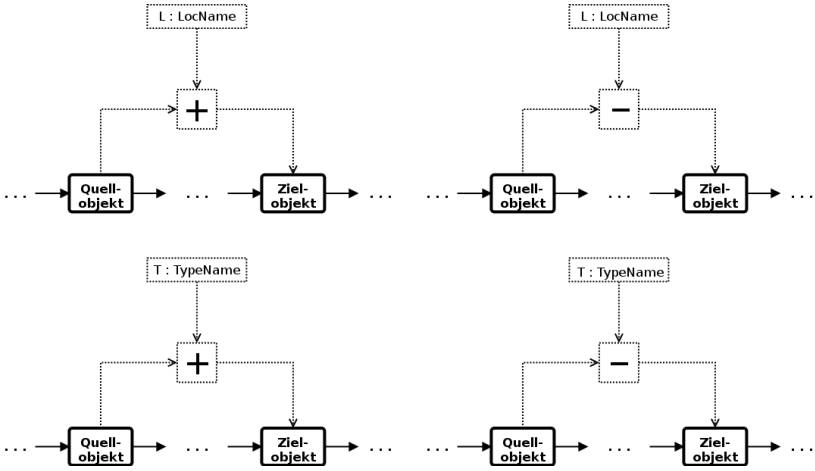


Abb. 4.18: Graphische Darstellung ortsbindender OE

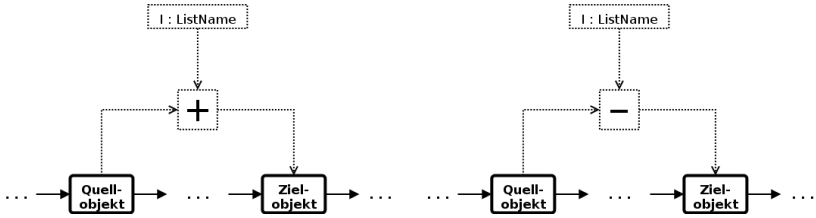


Abb. 4.19: Graphische Darstellung ortsimplizierender OE

4.3.2 Aggregationen

Es sind für alle Ausprägungen von OE eindeutige graphische Darstellungen eingeführt worden. Die graphischen Repräsentationen von OE können durch Aggregationen vereinfacht werden, so dass das Workflow-Modell übersichtlicher dargestellt wird. Es soll – abhängig von der Art der OE – ermöglicht werden, Aggregationen zu modellieren.

Die Wiederverwendung von **Konfigurationselementen** soll den Aufwand bei der Modellierung verringern und die Übersichtlichkeit verbessern. Konfigurationselemente sind Referenzen auf Objekte der Ortsmodellinstanz und können durch Rollen oder durch IS repräsentiert werden (modellextern). Konfigurationselemente werden im Workflow-Modell, abhängig von der Anordnung der Elemente, wiederverwendet.

Im Beispiel aus Abb. 4.20 sind zwei positive modellinterne OE modelliert. Das Zielobjekt wird zum Quellobjekt für die nächste OE. Die Auflage der OE wird an das nächste Objekt weitergegeben, das am gleichen Ort ausgeführt werden soll. Die indirekte OE kann nur ausgewertet werden, wenn das Quellobjekt an einem Ort innerhalb des im Konfigurationselement referenzierten Bereichs stattfindet. Um dies zu gewährleisten, werden die Ausführungsorte durch die direkte OE mit einem einzigen Konfigurationselement verbunden (siehe Abb. 4.21).

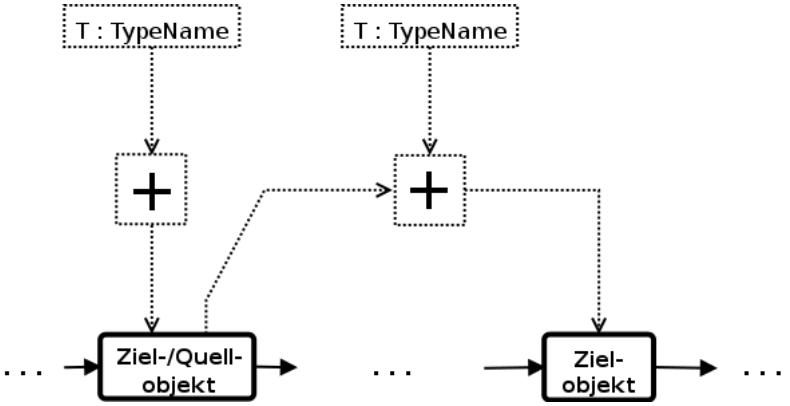


Abb. 4.20: Graphische Darstellung Konfigurationselemente

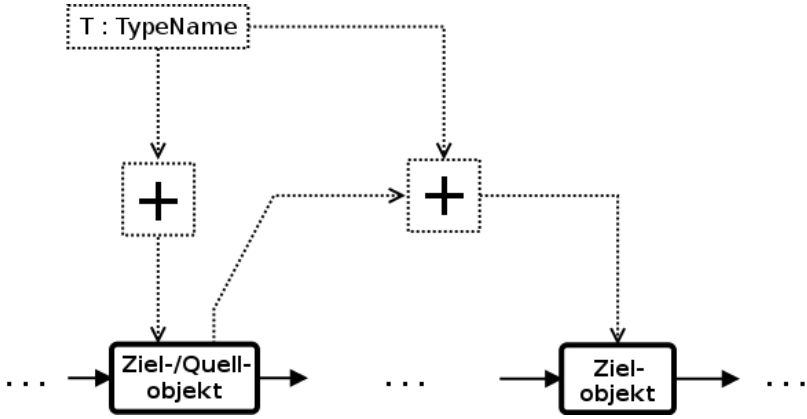


Abb. 4.21: Graphische Darstellung aggregierte Konfigurationselemente

Die Aggregation von **direkten Ortseinschränkungen** setzt voraus, dass die gleichen OE für Ausführungsorte im Workflow-Modell eingehalten werden. Es muss nicht jedes Objekt mit einer eigenen Instanz verbunden werden, sofern jeweils eine einzelne OE erzeugt wird. In Abb. 4.22 werden einzelne OE ohne Aggregation dargestellt.

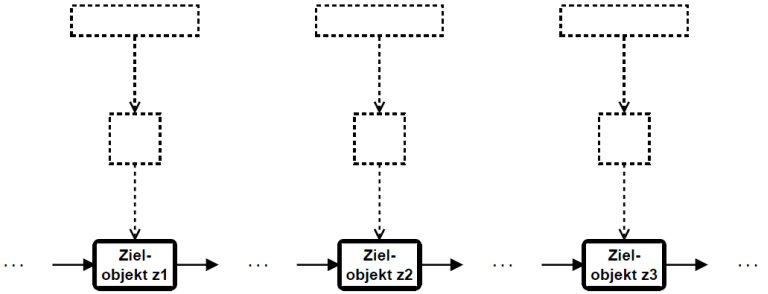


Abb. 4.22: Beispiel einzelner OE ohne Aggregation

Eine Aggregation kann mit Hilfe einer einzelnen OE modelliert werden, die zu jedem Zielobjekt eine Verbindung hält (siehe Abb. 4.23).

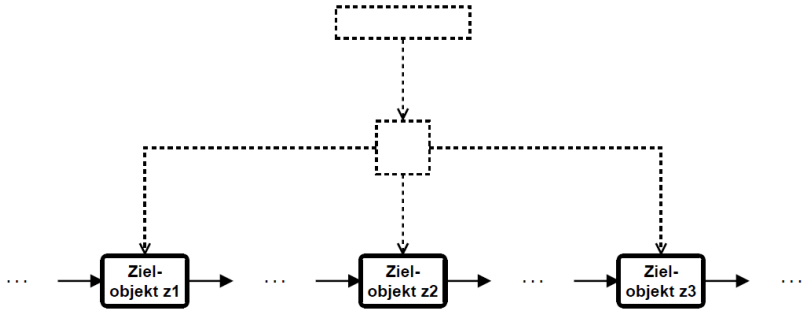


Abb. 4.23: Beispiel einzelner OE mit Aggregation

Die OE (siehe Abb. 4.23) muss für alle Zielobjekte das gleiche Vorzeichen enthalten. Zusätzlich müssen die Konfigurationselemente denselben Ort, Ortstyp oder dieselbe Zuordnungsliste besitzen. Diese Voraussetzungen gewährleisten eine eindeutige Interpretation der Aggregation.

Die aggregierte OE wirkt einzeln auf die Zielobjekte. Wird ein Ortstyp beispielsweise mit einer positiven direkten OE verknüpft, werden die Zielobjekte nicht zwingend am selben Ort des Ortstyps ausgeführt. Die Ursprungssemantik bleibt – unabhängig von der Aggregation – erhalten.

Aggregationen von **indirekten Ortseinschränkungen** sind komplexer als direkte OE, da neben dem Zielobjekt auch ein Quellobjekt berücksichtigt wird. Modellexterne OE ohne Quellobjekt (als Spezialfall) können jedoch (analog zu direkten OE) aggregiert werden. Eine Aggregation ist dann zulässig, wenn Vorzeichen und externe Datenquelle identisch sind. Bei indirekten OE müssen Vorzeichen und die Konfigurationselemente gleich sein, während sich Quell- und Zielobjekte in ihrer Ausprägung und Anzahl unterscheiden können. Es können beispielsweise OE mit unterschiedlichen Quell- und Zielobjekten dargestellt werden (siehe Abb. 4.24).

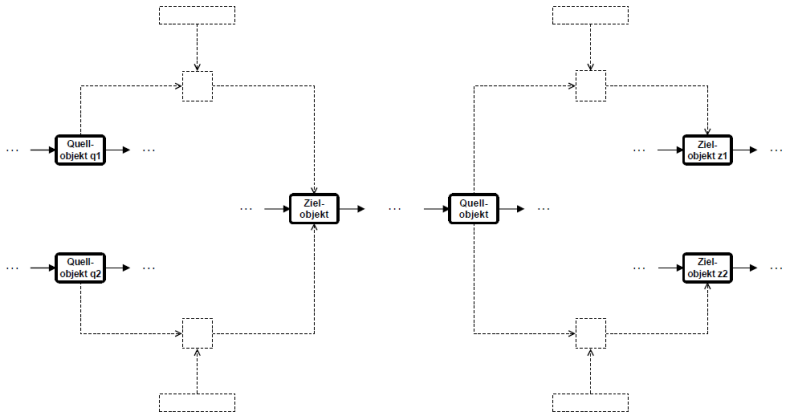


Abb. 4.24: Aggregation indirekter OE - Beispiel

In Abb. 4.24 (links) sind zwei OE abgebildet, deren Vorzeichen für das Zielobjekt und das verknüpfte Konfigurationselement identisch sind. Unterschieden werden die OE anhand der Quellobjekte. Die OE sind jeweils gleich in ihrer Ausprägung. In dem Modellierungsbeispiel (rechts) ist die

Differenzierung anhand der unterschiedlichen Zielobjekte möglich. Die aggregierte Darstellungsform der beiden beschriebenen Anwendungsbeispiele zeigt, dass eine Aggregation vorteilhaft angewendet werden kann (siehe Abb. 4.25). Das Modell enthält, durch die Aggregation redundanter Informationen, weniger Elemente und kann somit besser vom Betrachter erfasst werden. Die Funktionsweise der OE wird durch die Aggregation nicht geändert.

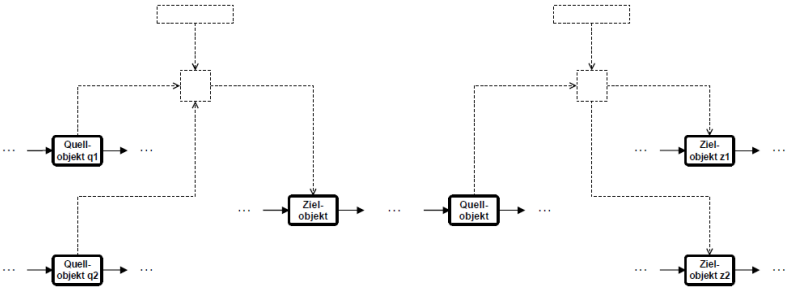


Abb. 4.25: Aggregation indirekter OE (Bsp. 1)

Bei einer weiterführenden Aggregation der dargestellten Anwendungsbeispiele (siehe Abb. 4.24), wird die OE der Quellobjekte mit den Zielobjekten verbunden (siehe Abb. 4.26).

Das Anwendungsbeispiel aggregiert vier unterschiedliche OE mit je einem Quell- und Zielobjekt, sofern die gleichen Konfigurationen vorhanden sind (siehe Abb. 4.26). Auch für indirekte OE gilt die individuelle Wirkung auf das Zielobjekt. Das Anwendungsbeispiel (siehe Abb. 4.27) einer Autovermietung veranschaulicht nochmals diese individuelle Wirkung. Mit dieser aggregierten indirekten OE wird sichergestellt, dass die Fahrtenbuchpflege bzw. die Abgabe des KFZ an dem Ort stattfinden, an dem vorher das KFZ gemietet und nachkontrolliert wurde.

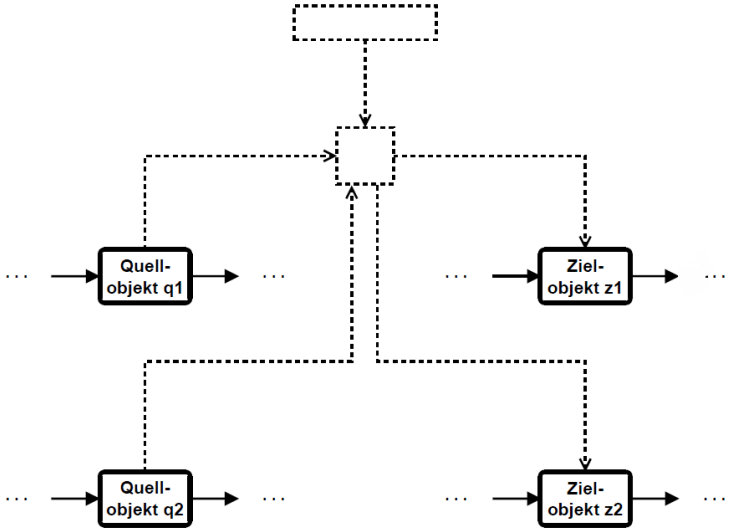


Abb. 4.26: Aggregation indirekter OE (Bsp. 2)

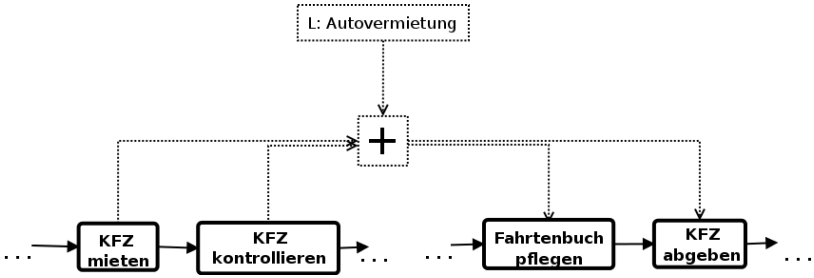


Abb. 4.27: Anwendungsbeispiel aggregierter indirekter OE

Wenn Quell- und Zielobjekte nach Verzweigungen (innerhalb des Modells) modelliert sind, die sich jeweils per ODER-Verzweigung (XOR) gegenseitig ausschließen, kann mit Hilfe einer Aggregation eine übersichtliche Darstellung geschaffen werden. Die Aggregation liefert eine kompakte Darstellung

und kann für alle Objekte eine OE generieren, da jeweils ein Quell- und Zielobjekt durchlaufen wird. Eine nicht-aggregierte Darstellung kann schnell zu Überfrachtungen im Workflow-Modell führen.

4.3.3 Hierarchische Modellelemente

Viele graphischen Workflow-Modellierungssprachen verfügen über Mechanismen zur Strukturierung von Workflow-Modellen in hierarchischer Form. Oft wird dieser Mechanismus mit Unterprozessen (Subprozessen) modelliert, die auf einer höheren Ebene in einer Hierarchie aggregiert dargestellt werden. Dieser Mechanismus wird in UML-Aktivitätsdiagrammen z. B. mit Partitionen umgesetzt. In BPMN-Modellen hilft die Annotation einer Aktivität (kleines Pluszeichen), die dadurch einen verknüpften Subprozess kenntlich macht [FrRH10]. In Petri-Netzen können durch Transitionen als Ergebnis einer Vergröberung Teilnetze repräsentiert werden [Ober90, S. 26]. Diese Mechanismen sollen einer Überfrachtung des graphischen Modells entgegenwirken bzw. mehr Übersicht - durch die hierarchische Anordnung - gewährleisten. Während des Entwurfs eines Modells können zusätzlich Top-Down und Bottom-Up Vorgehensweisen (Strategien) umgesetzt werden [SpSc08]. Unwichtige Zusammenhänge (je nach Betrachtungsebene) werden ausgeblendet und wiederkehrende Workflow-Objekte mit Hilfe des Mechanismus als zusammengefasstes Element dargestellt. Die hierarchischen Modellelemente haben Einfluss auf die verschiedenen Ausprägungen von OE. OE müssen eindeutig interpretiert und innerhalb eines graphischen Modells abgebildet werden. Wenn Quell- und Zielobjekte nicht atomar sind, kann es beispielsweise zu Fehlinterpretationen kommen (siehe Abb. 4.28). Die OE (rechts) kann als direkte oder indirekte OE interpretiert werden.

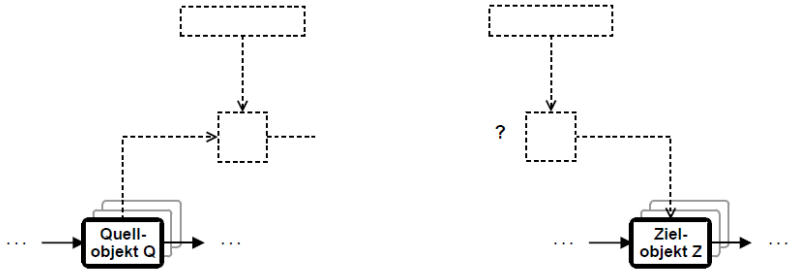


Abb. 4.28: Hierarchische Quell- und Zielobjekte

Bei **hierarchischen Zielobjekten** (siehe Abb. 4.28) ist das hierarchische Element des Workflow-Modells das Zielobjekt. Die OE müssen somit auf alle Objekte (Aktivitäten) der Zielhierarchie wirken. Die Wirkung auf weitere OE innerhalb des hierarchischen Zielobjekts ist durch diese Definition nicht determiniert. Es könnten z. B. bei einer unbedachten Modellierung schnell alle OE, die jeweils in derselben Hierarchie angeordnet sind, mit dieser OE, die eine Wirkung auf den ganzen Teilprozess hat, überschrieben werden. OE müssen die Objekte einer hierarchischen Struktur gemeinsam beeinflussen. Vorausgesetzt es gibt jeweils eine OE, die auf den ganzen Teilprozess wirkt, muss gewährleistet werden, dass keine OE überschrieben wird. In Abb. 4.29 ist die Definition mit und ohne Zusatz modelliert.

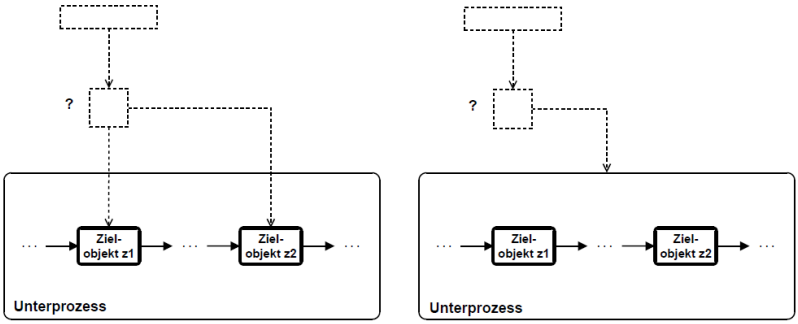


Abb. 4.29: Definition (mit Zusatz; rechts) hierarchischer Zielobjekte

Wird z. B. eine positive direkte OE mit einem Ortstyp verknüpft, fordert die Definition ohne Zusatz (links) die Ausführung der einzelnen Bestandteile des hierarchischen Zielobjekts an einem beliebigen Ort des Ortstyps. Die Ausführungsorte müssen nicht gleich sein, sondern lediglich dem Ortstyp entsprechen. Der Zusatz zur Definition bindet die Ausführung der Elemente ganzheitlich an einen beliebigen Ort des Ortstyps innerhalb der Hierarchie. Dieser Ort muss jedoch gleichbleiben. Bei negativen OE innerhalb einer Hierarchie tritt dieser Fall nicht auf, da nur Aussagen über verbotene Ausführungs-orte getroffen werden. Für positive OE ist die Differenzierung ebenfalls nicht notwendig, da einzelne elementare Orte definiert sind.

Bei **hierarchischen Quellobjekten** modellinterner OE entsprechen die dynamischen Basisdaten des Ausführungsortes dem Quellobjekt. Eine eindeutige Interpretation ist nur dann möglich, wenn atomare Orte als dynamische Basisdaten vorliegen (siehe Kapitel 4.1.5). Unterschiedliche Elemente innerhalb der Hierarchie des Quellobjekts können an verschiedenen Ausführungsorten stattfinden. Eine eindeutige Ermittlung der dynamischen Basisdaten ist daher ggfs. nicht möglich. Eine implizite Regel kann helfen, alle Elemente einer Hierarchie des Quellobjekts am selben Ausführungsort stattfinden zu lassen. Jedoch kann es gleichwohl vorkommen, dass keine OE der Elemente innerhalb des hierarchischen Quellobjektes wirken. Falls mehrere Alternativen vorhanden sind, kann die OE in diesem Fall ausgelassen werden. Es wird keine OE erzeugt. Diese Methode besitzt Vorteile, da nicht auflösbare modellinterne OE durch die Modellierung einer zusätzlichen direkten OE Fehlinterpretationen verhindert. Die direkte OE erzeugt die Auflage für das hierarchische Quellobjekt der modellinternen OE. Falls eine explizite Festlegung auf einen einzelnen Ort (innerhalb des Ortstyps) gewünscht ist, muss die OE mit demselben Ortstyp wie die ortsbindende OE verknüpft werden. Bei ortsimplizierenden OE muss analog verfahren werden. Für die Modellierung hierarchischer Modellelemente ist es notwendig, dass Alternativen für die entsprechenden Wirkungszusammenhänge der hierarchischen Zielobjekte zur Verfügung gestellt werden (siehe Abb. 4.30).

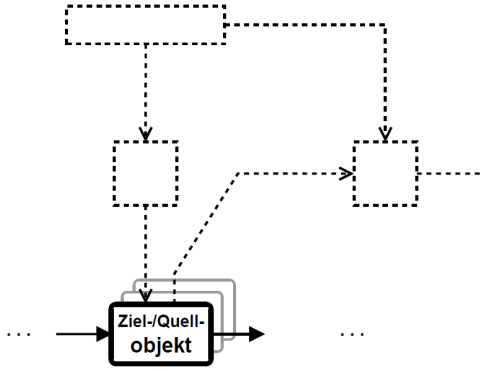


Abb. 4.30: das Quellobjekt ist Bestandteil eines Teilprozesses

Die graphische Darstellung der Quell- und Zielobjekte (z. B. beim Ausblenden während der Modellierung) ist ausschlaggebend für die Interpretation der OE. Ein Endpunkt einer OE kann in einer hierarchischen Struktur z. B. nicht mehr sichtbar sein. Die Verbindungslinien (an die Quell- bzw. Zielobjekte) können nicht mehr angelegt werden. Die direkten OE bzw. modellexternen OE sollten deshalb einfach (mit den Subprozessen) ausgeblendet werden. Jedoch könnte es (je nach Anwendungsfall) sinnvoll sein, gerade diese OE anzuzeigen. Das Symbol des zusammengeklappten hierarchisch strukturierten Objekts könnte mit einer geeigneten Dekoration versehen werden, um diese Details kenntlich zu machen.

Für indirekte OE (mit Quellobjekt) sind die Verschachtelungen innerhalb hierarchischer Modellobjekte problematischer, da die Endpunkte in unterschiedlichen hierarchischen Strukturen verdeutlicht werden müssen. Die Darstellung der OE kann wieder eingeschränkt werden. Wenn z. B. Quell- und Zielobjekte nicht direkt angezeigt werden, sollten die OE ausgeblendet werden. Gleichzeitig kann jedoch die explizite Darstellung der OE für ein besseres Verständnis des Workflow-Modells bei den Prozessbeteiligten sorgen (siehe Abb. 4.31).

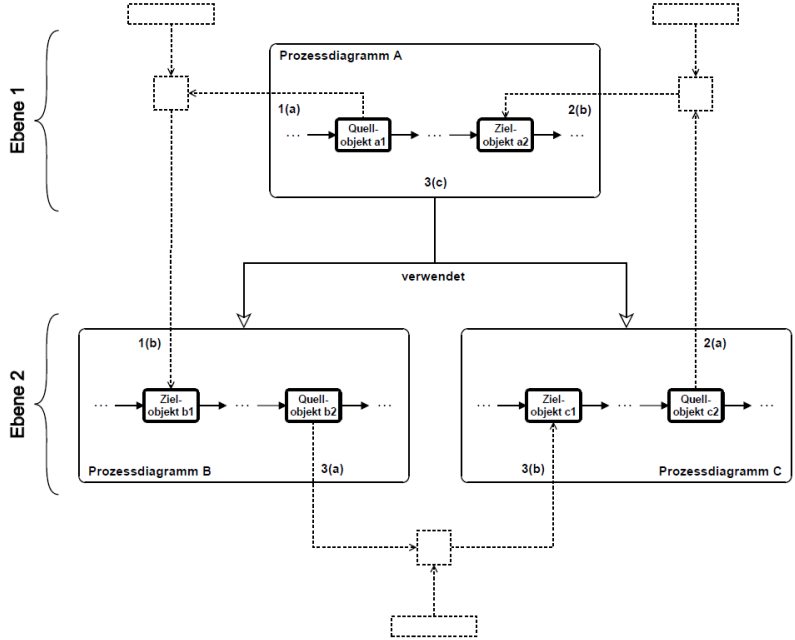


Abb. 4.31: Ortseinschränkungen in Workflow-Hierarchien (schematische Darstelllung)

In Abb. 4.31 werden abstrakte Beispiele dargestellt, die in Kombination mit hierarchisch strukturierten Objekten vorkommen. Im Beispiel werden zwei Ebenen einer Hierarchie betrachtet. In Diagramm A wird ein Workflow-Modell dargestellt, das aus zwei hierarchischen Objekten besteht. Jedes einzelne Objekt besteht aus einem Teilprozess, der im Modell B und C dargestellt ist. Die Beispiele werden in folgende Fälle unterschieden (siehe Abb. 4.31):

(Die Ziffern in der Auflistung entsprechen den Ziffern in der Abbildung)

1. Quellobjekt eine Ebene höher als Zielobjekt:

    a. Diagramm zeigt Quellobjekt
    b. Diagramm zeigt Zielobjekt

2. Quellobjekt eine Ebene tiefer als Zielobjekt:

    c. Diagramm zeigt Quellobjekt
    d. Diagramm zeigt Zielobjekt

3. Quell- und Zielobjekt auf unterschiedlichen Ebenen (ohne direkten Pfad):

    e. Diagramm zeigt Quellobjekt
    f. Diagramm zeigt Zielobjekt
    g. Diagramm zeigt Vorgänger (3.a und 3.b)

Diese Fallunterscheidung kann analog auf Hierarchien mit mehreren Ebenen angewandt werden. Bei hierarchischen Modellelementen muss die Perspektive auf die Workflow-Hierarchie in Kombination mit der OE berücksichtigt werden (z. B. im Anwendungsfall 1.a, 2.b und 3.c, wenn sich Quell- und Zielobjekt in einem zusammengeklappten Teilprozess befinden). Die graphische Darstellung kann nur ermöglicht werden, wenn die Perspektive (auf die Workflow-Hierarchie) einen geeigneten Ankerpunkt für die Verbindung bereitstellt. Wird nur eine Verbindung zum Objekt ersetzt, so werden die Informationen verfälscht. Es würde der Eindruck entstehen, dass nicht das verborgene Objekt, sondern das hierarchische Objekt den Endpunkt zeigt. Diese Darstellung muss vermieden werden. Fehlinterpretationen entgegenzuwirken erfordert eine Anpassung der graphischen Darstellung der entsprechenden OE. Ist die Verbindung zum Quell- oder Zielobjekt durch ein hierarchisches Objekt ersetzt worden, muss dieser Zusammenhang im Modell dargestellt werden. Das Symbol der OE kann mit einer weiteren Dekoration versehen werden. Es kann somit dargestellt werden, welche OE nicht direkt mit dem Ziel- oder Quellobjekt verbunden ist. Bei Aggregationen kann dies jedoch zu weiteren Problemen in der Darstellung führen, falls beispielsweise mehrere eingehende und ausgehende Kanten vorhanden sind.

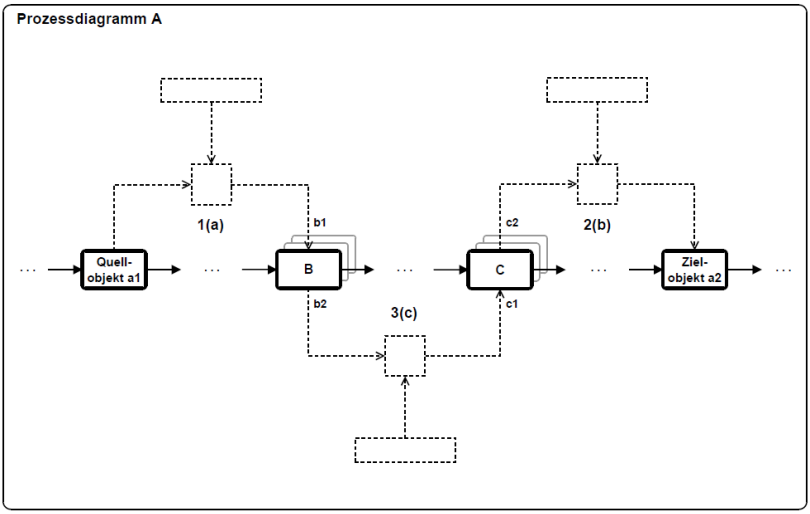


Abb. 4.32: Ortseinschränkungen mit dekorierten Ankerpunkten (schematische Darstellung)

Alternativ kann eine Anpassung von Verbindungslinien durchgeführt werden. Falls die Verbindungslinien an ein zusammengefasstes hierarchisches Objekt angelegt werden (mit enthaltenem Quell- bzw. Zielobjekt), muss dies durch einen zusätzlichen Ankerpunkt angezeigt werden. Mit dieser Darstellung können falsche Interpretationen vermieden werden (siehe Abb. 4.32). Die Verbindungslinien sind mit den Namen der Endpunkte beschriftet und verdeutlichen, dass das entsprechende Objekt (nicht der gesamte Teilprozess) mit der OE berücksichtigt wird. Es kann erforderlich sein, z. B. wenn mehrere Quell- bzw. Zielobjekte innerhalb desselben hierarchischen Workflow-Modells enthalten sind, dass mehrfache Beschriftungen eingeführt werden müssen. Ein separater Eintrag am ausgehenden bzw. eingehenden Ende der Verbindung verdeutlicht den Zusammenhang. Für die Fälle 1.b, 2.a, 3.a und 3.b fehlt solch ein Ankerpunkt. Falls z. B. ein Objekt auf eine OE zeigt, dessen Gegenstück nicht unterhalb der aktuellen Hierarchieebene liegt, kann dies mit Hilfe einer Verknüpfung gelöst werden (siehe Kapitel 4.3.4).

Ein Anwendungsbeispiel der vorgestellten hierarchischen Modellelemente beschreibt der Kundenprozess einer Autovermietung, der das ausgeliehene Auto an dem Ort zurückgeben soll, an dem er die Fahrzeugkontrolle (vor Fahrtantritt) durchgeführt hat (siehe Abb. 4.33).

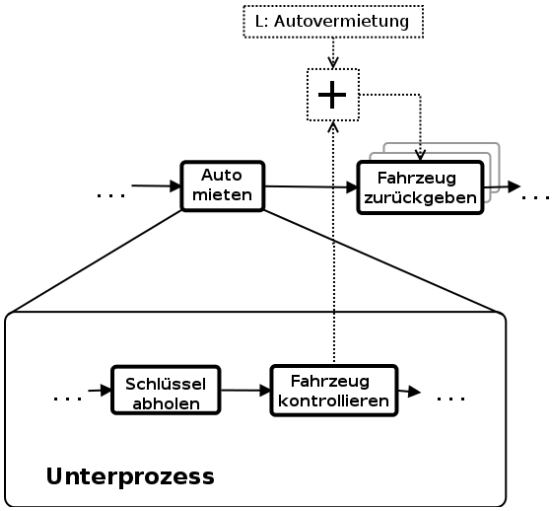


Abb. 4.33: Anwendungsbeispiel hierarchischer Modellelemente

4.3.4 Verknüpfungen als Modell-übergreifende Elemente

Bei bestimmten Geschäftsprozessen ist eine übergreifende Informationsvermittlung, z. B. über verschiedene Unternehmensbereiche und Prozesse hinweg, notwendig. OE können prozessübergreifend wirken. Bei der Modellierung von OE werden Prozessgrenzen überschritten. Bei indirekten OE, deren Quellobjekt nicht im selben Workflow-Modell wie das Zielobjekt liegt, kann dies zu Problemen führen. Falls im Modell das Quell- und Zielobjekt der indirekten OE und die Rollen dem gleichen Workflow-Modell zugehörig sind, werden mit der bisher eingeführten Notation keine Probleme entstehen. Ist diese Voraussetzung jedoch nicht gegeben, muss ein Mechanismus

geschaffen werden, der zusammengehörige Komponenten identifiziert. Es werden Verknüpfungssymbole, die für die Verbindung zweier Endpunkte zuständig sind, eingeführt.

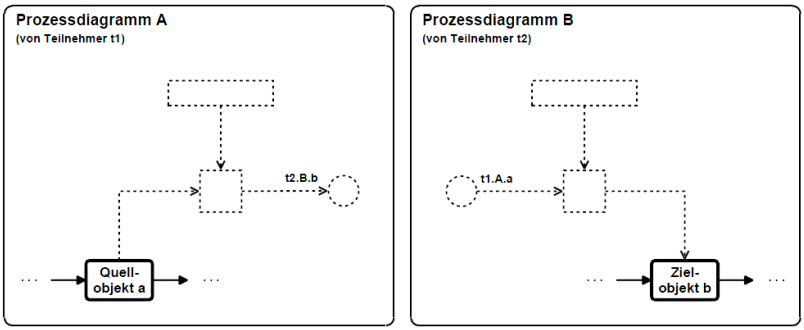


Abb. 4.34: Verknüpfungspunkte von OE

In Abb. 4.34 wird eine Workflow-übergreifende indirekte OE dargestellt. In Prozessdiagramm A befindet sich das Quellobjekt „a“, das vom Teilnehmer „t1“ bearbeitet wird (siehe Abb. 4.34). In Prozessdiagramm B (von Teilnehmer „t2“) wird das Zielobjekt „b“ dargestellt. Für die Identifizierung der Endpunkte müssen eindeutige Bezeichner eingeführt werden. Die Bezeichner können aus Teilnehmername, Diagrammname, der Bezeichnung von Quell- und Zielobjekt und dem Verknüpfungssymbol generiert werden. Die OE sollte in beiden Modellen dargestellt werden, um Verwechslungen zu vermeiden. Eine aggregierte Darstellungsform von OE kann dazu führen, dass mehrere Verknüpfungssymbole innerhalb eines Workflow-Modells notwendig sind. In Modellierungswerkzeugen ist die Navigation meist durch Öffnung eines Diagramms im Verzeichnis möglich. Bei einem überfrachteten Workflow-Modell kann eine Verknüpfung die Übersicht verbessern, da weit entfernte Modellelemente überbrückt werden können.

4.3.5 Inaktive Ortseinschränkungen

Innerhalb eines Workflow-Modells kann es Pfade geben, die durch den Kontrollfluss ein Zielobjekt einer indirekten OE erreichen, ohne vorher das Quellobjekt ausgeführt zu haben. In diesem Fall fehlen die dynamischen Basisdaten und die OE für das Zielobjekt kann nicht erzeugt werden. Solche OE werden als *inaktive OE* bezeichnet. Zielobjekte inaktiver OE besitzen keine Basisdaten und werden nicht ausgeführt bzw. berücksichtigt.

Die inaktiven OE können als zusätzliche Funktionalität genutzt werden. Wird z. B. der Ausführungsort eines Zielobjektes nur unter gewissen Bedingungen geplant, kann eine Entscheidung im Verlauf des Workflows den Kontrollfluss zunächst zum Quellobjekt der OE lenken, bevor das Zielobjekt erreicht wird. Dadurch determiniert der konkrete Workflow die Erzeugung der OE.

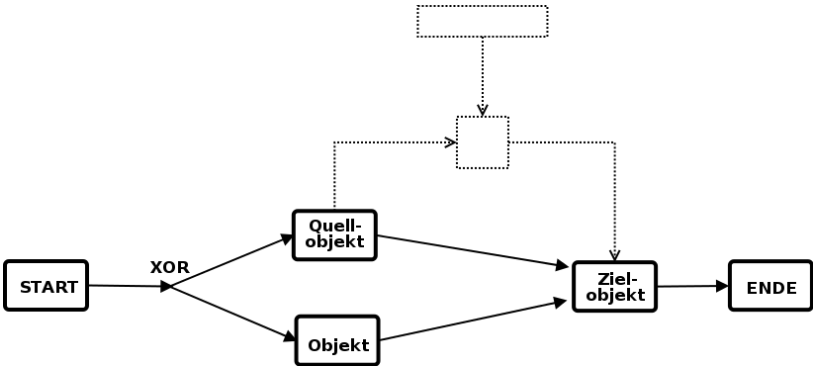


Abb. 4.35: Inaktive OE in einem Workflow-Modell

In Abb. 4.35 wird ein informelles Workflow-Modell mit einer indirekten OE dargestellt. Zu Beginn des Kontrollflusses (nach Ausführung des „START“) wird eine ODER-Verzweigung (XOR) realisiert. Falls der Kontrollfluss über „Objekt“ läuft, kann im weiteren Verlauf „Quellobjekt“ nicht mehr erreicht werden. Wenn der Kontrollfluss „Objekt“ ausführt, fehlen die notwendigen

dynamischen Basisdaten für das „Zielobjekt“, welches dadurch keine OE für einen Ausführungsort besitzt (siehe Abb. 4.35). Es kann somit nur eine OE abgeleitet werden, wenn der obere Pfad ausgeführt wird.

Bei modellexternen OE können Situationen entstehen, bei denen bewusst OE ausgelassen werden sollen. Das mWfMS (bzw. der Prozessbeteiligte) sollte die Entscheidung treffen. Es kann beispielsweise zu einem technischen Ausfall kommen, in dessen Folge eine OE ausgelassen wird. Eine nicht auflösbare modellinterne OE (aus Kapitel 4.1.5) kann ebenfalls auftreten: Falls der Ausführungsort des Quellobjekts einer modellinternen OE nicht geeignet ist, um basierend darauf eine OE zu erzeugen. Die in Kapitel 4.3.2 beschriebenen hierarchischen Strukturen, deren Ausführungsorte nicht eindeutig sein können, verursachen das gleiche Problem.

Weiterführende Aspekte der OE-Notation wurden in [Deck11] untersucht. Es werden z. B. logische Widersprüche, die durch das Ortsmodell auftreten können, betrachtet [ChDe10, Deck11, S. 223-238]. Weiterführende graphische Probleme werden in [Deck11, S. 207-221] diskutiert.

4.3.6 Anomalien

Anomalien von Ortsbezügen können entstehen, wenn mehrere Auflagen eines Ausführungsortes in Konkurrenz zueinanderstehen. Ursache und Wirkung von Anomalien sind in [ChDe10] beschrieben.

Anomalien können z. B. durch folgende Ereignisse ausgelöst werden:

1. Zwei oder mehr OE erzeugen Auflage für gleiches Zielobjekt (siehe Abb. 4.36, links)

2. Zielobjekt einer OE geht in einem hierarchisch strukturierten Zielobjekt einer anderen OE auf (siehe Abb. 4.36, rechts)

3. redundante Basisdaten (vgl. Kapitel 5.3.1)

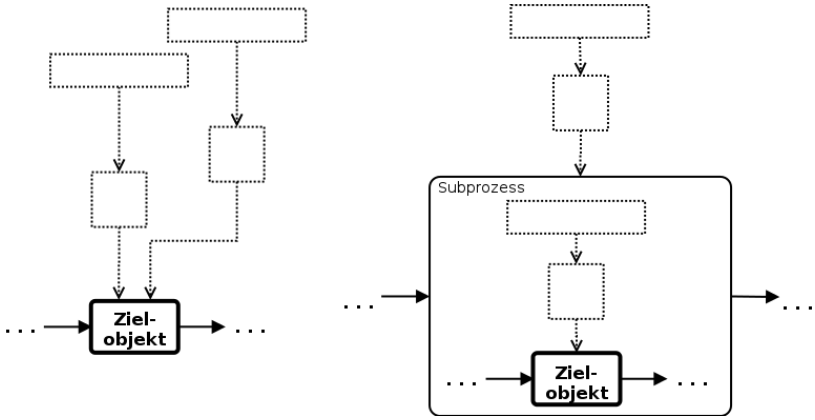


Abb. 4.36: Anomalien und deren Ursachen

Mehrfache Auflagen an Ausführungsorte führen zu Anomalien, wenn keine eindeutigen Aussagen über den jeweiligen Ausführungsort getroffen werden können. Anomalien müssen bzgl. ihrer Wirkung auf das mWfMS bzw. den Anwendungsfall untersucht werden. Anomalien von Ortsbezügen können beispielsweise Redundanzen oder Widersprüche sein.

- **Redundanzen:** Wenn Auflagen für Elemente teilweise bzw. vollständig gleiche Forderungen haben.
- **Widersprüche:** Wenn die Auflagen für ein Element im Konflikt zueinanderstehen.

Bei direkten OE können beide Arten von Anomalien ex ante analysiert und z. B. durch das mWfMS aufgelöst werden, da bei direkten OE die Basisdaten vor der Prozessausführung vorliegen. Redundanzen führen zu längeren Laufzeiten des mWfMS, jedoch nicht zu strukturellen Fehlern innerhalb der Prozesse. Sie können z. B. systemisch abgemildert werden, indem die Orte als eigener Ortstyp im mWfMS aufgelöst werden. Bei Widersprüchen können einfache pragmatische Ansätze, wie z. B. nur die aktuellste Auflage einer OE zu berücksichtigen, helfen. Bei Anwendungsfällen, in denen solch

einfache Algorithmen keine Anwendung finden, müssen die Auflagen stets kontrolliert bzw. vorab identifiziert werden.

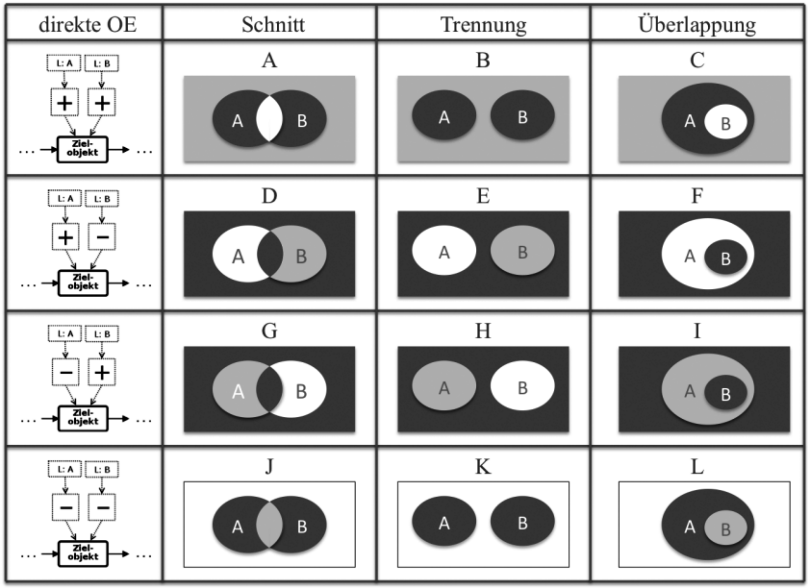
Bei indirekten OE können die Anomalien nur zur Laufzeit entdeckt werden. Anomalien sind bei indirekten OE fast nicht vorab prognostizierbar. Simulationen können helfen Probleme bzw. Problemstellen im Prozess zu identifizieren.

Nur wenn die Gesamtheit aller Auflagen geprüft wird, kann eine Aussage über Anomalie-Freiheit getroffen werden. Da allerdings, wie in Kapitel 4.3.5 gezeigt wurde, teilweise gar keine Auflagen erzeugt werden, weil sie inaktiv werden oder aber das Entfernen einer aktiven Auflage aus der Auflagenliste diese wieder deaktiviert, müssen nicht zwangsweise Widerspruch-Anomalien auftreten.

Sichere Anomalien lassen sich von **potentiellen Anomalien** unterscheiden. Sichere Anomalien können z. B. auf Grundlage statischer Basisdaten bzw. des modellierten Kontrollflusses ermittelt werden. Für potentielle Anomalien können Wahrscheinlichkeitsangaben helfen, um das Auftreten vorab systemisch zu analysieren.

In Tab. 4.2 wird ein generisches Beispiel von je zwei Auflagen gezeigt, die sich in einer Reihe von Fällen nach Überschneidung, Trennung und Überdeckung und dem Vorzeichen unterscheiden. Die Gleichheit wird nicht betrachtet (siehe Tab. 4.2). Es werden alle Konstellationen dargestellt. Zulässige Bereiche werden über Auflagen definiert. Die erste Auflage bezieht sich auf Ort A; die Zweite auf Ort B. Es entstehen jeweils drei unterschiedliche Bereichstypen (siehe Tab. 4.2, *weiss*, *grau* und *schwarz*). Weiße Flächen genügen beiden Auflagen, wohingegen schwarz eingefärbte Flächen einen Widerspruch zwischen den Auflagen aufzeigen. Die übrigen grauen Flächen lassen erkennen, dass die Auflagen einheitliche Bereiche beschreiben. Während redundante ortsbeschreibende Auflagen bei grauen und weißen Flächen vorhanden sind, zeigen die grauen Flächen einen Widerspruch an.

Tab. 4.2: Anomalien von Ortsbezügen, Fallbetrachtung



Soll eine Ausführung einer Auflage innerhalb von Ort A und Ort B stattfinden, liegen disjunkte Mengen vor (vollständiger Widerspruch, da keine Menge enthalten ist, vgl. Fall „B“). Die gleiche Anomalie lässt sich im Fall „I“ finden. In den Fällen „B“, „E“, „H“ und „K“ enthalten beide Auflagen jeweils die Forderungen der anderen oder präzisiert diese. Bei Fall „C“ und „L“ ist Ort B Teil von Ort A. Gleichzeitig wird der zulässige Bereich des anderen eingeschnitten. Im Fall „C“ ist die erste Auflage nicht notwendig, da die zweite Auflage diese weiter präzisiert. Bei Fall „L“ wird die zweite Auflage durch die erste garantiert. In Fall „K“ liegen keine redundanten Auflagen vor.

Abschließend sollte angemerkt werden, dass die obigen Beispiele keine komplexen Strukturen, also mehr als zwei Auflagen, beinhalten. Je mehr Auflagen, desto komplexer, da jede Auflage aus mehreren Orten (vgl. Ortstypen) oder einer ganzen Menge (vgl. Zuordnungsliste) aufgebaut sein kann.

5 Integration von Ortsbezügen in Geschäftsprozess-modellierungssprachen

Das Konzept der Ortsbezüge (siehe Kapitel 4) wird in Petri-Netze (siehe Kapitel 5.1) und BPMN 2.0 (siehe Kapitel 5.2) als jeweilige Spracherweiterung der beiden Geschäftsprozessmodellierungssprachen integriert. In Kapitel 5.1.2 folgen exemplarische Petri-Netz- und BPMN-Modellbeispiele mit Ortsbezügen zur Verdeutlichung der praktischen Anwendung.

5.1 Integration in Petri-Netze

5.1.1 Definition Petri-Netze

Petri-Netze eignen sich für die Darstellung und Ausführung von Workflows (siehe Kapitel 2.6). Die OE werden als Annotation für Petri-Netze eingeführt (siehe Kapitel 4.3). Die Funktionen der Spracherweiterung werden in Abb. 4.8 dargestellt. Die graphische Notation der Spracherweiterung bedarf einer Semantik innerhalb der Petri-Netze (siehe Kapitel 4.3.1).

Ortsbezüge bzw. OE sind im bisherigen Verlauf der vorliegenden Arbeit unabhängig von der Zielsprache eingeführt worden. Der Kontrollfluss ändert sich durch die Annotation von OE nicht (siehe Kapitel 2.3). Eine OE wird als zusätzliche Kontrollstruktur eingeführt. Auf diese Weise ist es möglich, dass z. B. alle OE aus dem Modell entfernt werden, ohne dass sich die Syntax des Modells ändert. Beim Entfernen der OE bzw. der Kontrollstruktur werden lediglich die Auflagen der Ausführungsorte für „Aktivitäten“ gelöscht. Die Annotation gewährleistet innerhalb eines Modellierungswerkzeugs für Workflows dessen einfache Ausblendung der Kontrollstruktur. Die Vorgehensweise sieht vor, OE jederzeit aus dem Modell entfernen zu können, so dass der Kontrollfluss der Workflows stets gleichbleibt. Auf diese

Weise ist es beispielsweise möglich, weiterhin bekannte Analyseverfahren für Petri-Netze anzuwenden [GiVa03].

Das Prinzip des *separation of concerns* soll auf den Entwurf der Annotation (innerhalb der Petri-Netze) angewandt werden, so dass kein Werkzeug oder Austauschformat für Petri-Netze eine andere Semantik erhält. Mit dieser Vorgehensweise kann die Annotation eingeführt werden, ohne dass eine Überarbeitung der Sprachwerkzeuge notwendig ist. Die graphischen Symbole der OE (siehe Kapitel 4.3.1) dürfen die Elemente der Petri-Netze in ihrer Semantik nicht beeinflussen. Es muss daher ein neuer Verbindungstyp eingeführt werden, der die Verknüpfung der OE (ortsbeschreibende Elemente, externe Datenquelle) mit den Objekten der Petri-Netze (Quell- und Zielobjekte) verbindet.

Die graphische Darstellung des Vorzeichens muss innerhalb der Petri-Netze mit Hilfe einer Mengenbeschreibung ergänzt werden (siehe Kapitel 4.3.1). Die Menge der positiven OE und die Menge der negativen OE werden durch folgende Bezeichner beschrieben:

- C^+ endliche Menge der positiven OE
- C^- endliche Menge der negativen OE

Weitere Mengendefinitionen sind notwendig, um die drei ortsbeschreibenden Elemente, die Orte, Ortstypen und die Zuordnungslisten in Petri-Netzen einzuführen. Die ortsbeschreibenden Elemente werden als Schnittstelle zum entsprechenden Ortsmodell angewandt. Für die Elemente der Ortsmodellinstanz muss ein entsprechendes Gegenstück zu den ortsbeschreibenden Elementen vorhanden sein (siehe Kapitel 4.2). Es werden folgende Bezeichner für die Mengen (analog zu den Vorzeichen) eingeführt:

- $\mathcal{L}^{L}$ endliche Mengen der Orte
- $\mathcal{L}^{Y}$ endliche Mengen der Ortstypen
- $\mathcal{L}^{I}$ endliche Mengen der Zuordnungslisten

Für modellexterne Datenquellen einer OE müssen zusätzlich Bezeichner eingeführt werden, da bei modellexternen OE dies die Bezeichner der Herkunft dynamischer Basisdaten sind. Es werden zwei Mengen für die Bezeichner definiert:

- $\mathcal{R}^{\mathrm{M}}$ endliche Mengen der manuellen Datenquellen (Rollen)
- $\mathcal{R}^{\mathrm{B}}$ endliche Mengen der IS-/ Backend-Datenquellen

Die dargestellten (atomaren) Mengen werden in übergreifende Mengen zusammengefasst, damit die weiteren Definitionen möglichst kompakt (ohne Redundanzen) eingeführt werden können:

- $\mathrm{C} = \mathrm{C}^{+} \cup \mathrm{C}^{-}$ (1)
- $\mathcal{L} = \mathcal{L}^{\mathcal{L}} \cup \mathcal{L}^{\mathrm{Y}} \cup \mathcal{L}^{\mathrm{I}}$ (2)
- $\mathcal{R} = \mathcal{R}^{\mathrm{M}} \cup \mathcal{R}^{\mathrm{B}}$ (3)
- $\mathrm{X} = \mathcal{L} \cup \mathcal{R}$ (4)

Positive und negative Vorzeichen werden in Formel (1) zur Menge der OE C aggregiert. In Formel (2) wird die Menge der Orte, der Ortstypen und der Zuordnungslisten als Gesamtmenge $\mathcal{L}$ definiert. Die Menge aller externen Datenquellen $\mathcal{R}$ wird in Formel (3) gebildet. In Formel (4) werden die Mengen der ortsbeschreibenden Elemente und die Mengen der Datenquellen zu der Menge der Konfigurationselemente X zusammengesetzt.

Für Petri-Netze müssen neue Verbindungstypen eingeführt werden, damit die OE in die Sprache integriert werden können. In Petri-Netzen existiert nur ein Verbindungstyp (siehe Kapitel 2.6, *Definition Petri-Netze*): Die Relation F definiert einen Verbindungstyp und verbindet jeweils zwei unterschiedliche Elemente miteinander. Die Tupel dieser Relation sind für den Kontrollfluss verantwortlich. Die Verwendung des Verbindungstyps für OE würde dem Prinzip des *separation of concerns* widersprechen. Die zusätzliche Kontrollstruktur soll unabhängig von der eigentlichen Sprache als Erweiterung eingeführt werden. Es wird ein weiterer Verbindungstyp eingeführt, der analog zu den Assoziationen von BPMN oder der UML als „Assoziation“ mit dem Bezeichner A (siehe Formel (5)) zu sehen ist. Die Assoziationen sind unabhängig vom Kontrollfluss bzw. beeinflussen die Workflow-Instanz

nicht implizit. Es sei denn, der Modellierer möchte bezwecken, dass die OE den Kontrollfluss des Workflows explizit beeinflusst. Die Assoziationen sollen eine Verbindungsmöglichkeit für die zugehörigen Elemente der OE in Petri-Netz-Modelle bereitstellen. Quellobjekte in Form von Transitionen werden mit indirekten OE verknüpft. Eine weitere Assoziation verbindet OE mit Zielobjekten (in Form von Transitionen). Ortsbeschreibende Elemente müssen mit direkten oder modellinternen OE verbunden werden. Externe Datenquellen werden mit den modellexternen OE verknüpft. Die genannten Verbindungen (Assoziationen) werden in Definition (5) mit Hilfe eines kartesischen Produkts definiert:

- $\mathrm{A} \subseteq (T \times \mathrm{C}) \cup (\mathrm{C} \times T) \cup (\mathrm{X} \times \mathrm{C})$ (5)

Eine weitere Definition des Vor- und Nachbereichs von OE ist notwendig, um die Schaltregeln mit der Annotation der OE zu spezifizieren. Die Transitionen im Vor- und Nachbereich ($\bullet\, c^T$ und $c^T \,\bullet$) und die ortsbeschreibenden Elemente bzw. externen Datenquellen im Vorbereich ($\bullet\, c^{\mathrm{X}}$) müssen betrachtet werden:

- $\bullet\, c^T = \{t \in T \mid (t,c) \in A\},\ c \in C$ (6)
- $c^T \,\bullet = \{t \in T \mid (c,t) \in A\},\ c \in C$ (7)
- $\bullet\, c^{\mathrm{X}} = \{x \in \mathrm{X} \mid (x,c) \in A\},\ c \in C$ (8)

Die Vor- und Nachbereiche und die verschiedenen Mengen, die zur Definition von OE in Petri-Netzen benötigt werden, sind eingeführt worden. Damit ist es möglich, eine Definition für Ortseinschränkungen einzuführen:

Definition Ortseinschränkungen für Petri-Netze:

Ein Petri-Netz $N = (S,T,F)$ mit Ortseinschränkungen ist definiert als $(N, \mathrm{C}, \mathrm{X}, \mathrm{A})$ mit

1. C und X sind endliche Mengen
2. $\mathrm{C} \cap \mathrm{X} = \emptyset$

3. $\mathrm{C} \cap \mathrm{S} = \emptyset$
4. $\mathrm{C} \cap T = \emptyset$
5. $\mathrm{X} \cap S = \emptyset$
6. $\mathrm{X} \cap T = \emptyset$
7. $\mathrm{A} \subseteq (T \times \mathrm{C}) \cup (\mathrm{C} \times T) \cup (\mathrm{X} \times \mathrm{C})$ ist eine binäre Relation über $T \cup \mathrm{C} \cup \mathrm{X}$
8. $\forall\, \mathrm{c} \in \mathrm{C}$ gilt:
   a. $|\bullet\, c^T\,| \leq 1$ (maximal eine Transition im Vorbereich einer OE)
   b. $|\, c^T \bullet\,| = 1$ (genau eine Transition im Nachbereich einer OE)
   c. $|\bullet\, c^{\mathrm{X}}\,| = 1$ (ein Konfigurationselement im Vorbereich einer OE)

OE werden stets mit den eingeführten Assoziationen an die Transitionen verknüpft. Indirekte OE dürfen mit dem Quellobjekt (Transition) assoziiert werden (vgl. 8.a). OE besitzen immer exakt ein Zielobjekt (vgl. 8.b). Es darf immer nur eine Assoziation zwischen OE und Konfigurationselement existieren (vgl. 8.c).

Die Schaltregel eines Petri-Netzes (siehe Kapitel 2.6) wird durch die Erweiterung um Ortsbezüge nicht geändert. Der Vorbereich eines Netzelements (bzw. einer Transition) wird jedoch neu definiert, damit die Transition erst aktiviert wird, sofern eine Auflage einer OE vorliegt:

- $x = \{(y \in S \cup T) \cap (z \in \mathrm{C} \cup T)\,|\,(y,x) \in F, (z,x) \in \mathrm{A}\},$
  $x \in S \cup T \cup \mathrm{A}$ (Vorbereich)

Für die in Kapitel 4.3.1 eingeführten aggregierten Darstellungsformen von OE muss die obige Definition in Punkt 8 folgendermaßen angepasst werden:

$\forall c \in C$ gilt:

a. $|\bullet c^T| \geq 1$ (mindestens eine Transition im Vorbereich einer OE)

b. $|c^T \bullet| \geq 1$ (mindestens eine Transition in Nachbereich einer OE)

c. $|\bullet c^X| = 1$ (ein Konfigurationselement im Vorbereich einer OE)

Je nach OE sind die Merkmale der Ausprägung in Tab. 5.1 aufgelistet. Die Werte in den Klammern sind für Workflow-Modelle in aggregierter Form relevant.

Tab. 5.1: Differenzierung OE von Petri-Netz-Modellen

	$\lvert\bullet c^T\rvert$	$\lvert c^T\bullet\rvert$	$\lvert\bullet c^L\rvert$	$\lvert\bullet c^Y\rvert$	$\lvert\bullet c^I\rvert$	$\lvert\bullet c^M\rvert$	$\lvert\bullet c^B\rvert$
direkt	0	1 (≥ 1)	1	0	0	0	0
	0	1 (≥ 1)	0	1	0	0	0
	0	1 (≥ 1)	0	0	1	0	0
manuell	≤ 1 (≥ 0)	1 (≥ 1)	0	0	0	1	0
Back-End	≤ 1 (≥ 0)	1 (≥ 1)	0	0	0	0	1
ortsbindend	1 (≥ 1)	1 (≥ 1)	1	0	0	0	0
	1 (≥ 1)	1 (≥ 1)	0	1	0	0	0
ortsimplizierend	1 (≥ 1)	1 (≥ 1)	0	0	1	0	0

Die Vorbereiche (für die Markierung) von OE von Orten, Ortstypen und Zuordnungslisten bzw. manueller und Backend-Systeme sind in folgenden Formeln (9) – (13) aufgelistet.

- $\bullet c^L = \{l \in \mathcal{L}^L \mid (l,c) \in A\},\ c \in C$ (9)
- $\bullet c^Y = \{y \in \mathcal{L}^Y \mid (y,c) \in A\},\ c \in C$ (10)
- $\bullet c^I = \{i \in \mathcal{L}^I \mid (i,c) \in A\},\ c \in C$ (11)
- $\bullet c^M = \{l \in \mathcal{R}^M \mid (m,c) \in A\},\ c \in C$ (12)
- $\bullet c^B = \{b \in \mathcal{R}^B \mid (b,c) \in A\},\ c \in C$ (13)

5.1.2 Modellbeispiele mit Ortsbezügen

In den folgenden Anwendungsbeispielen werden Geschäftsprozessmodelle mit Ortsbezügen modelliert. Die Beispiele sind zwar rein fiktiv, jedoch ist die Umsetzung mit heutiger Technologie realisierbar.

Mitarbeiter beanspruchen (z. B. für eine Dienstreise) spezifische Leistungen, die über ein betriebliches Informationssystem automatisiert wurden. Die Geschäftsprozesse sind innerhalb von Workflow-Modellen abgebildet worden, die nun ausgeführt werden können.

Das fiktive Unternehmen hat zwei Filialen innerhalb der Stadt (siehe Abb. 5.1). Die Stadt besteht aus verschiedenen Stadtteilen (Innenstadt, Südstadt, Oststadt, ...) und wird in drei Bezirke (Nord-, Süd- und Ostbezirk) aufgeteilt. Die Innenstadt ist nicht Teil dieser Bezirke. In der Stadt sind zwei Mietwagenfirmen ansässig. Zur Vereinfachung wird angenommen, dass die Mietwagenfirmen auch eine Werkstatt für Fahrzeugreparaturen anbieten.

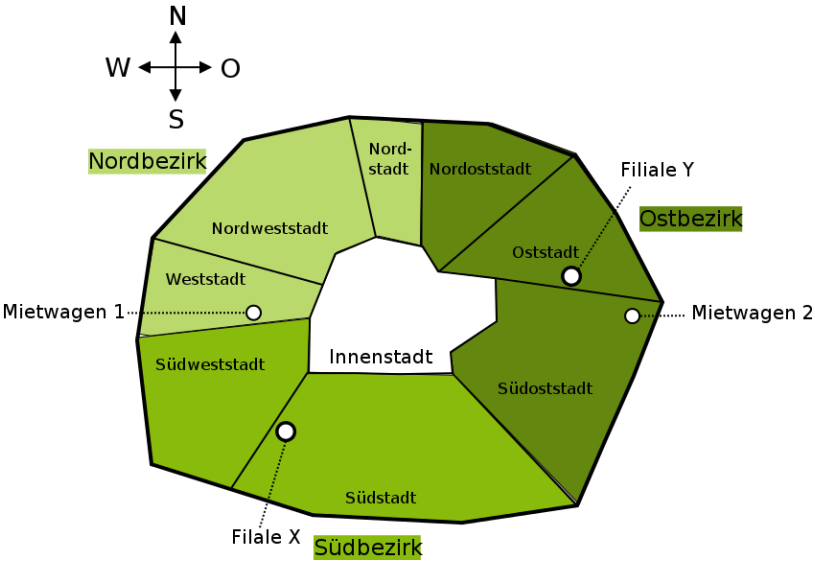


Abb. 5.1: Kartendarstellung der Anwendungsbeispiele

Ortsmodell und Datenquellen

Tab. 5.2: Orte und Ortstypen in den Anwendungsbeispielen

Bezeichnung	Typ	Orte
Stadt	Polygone	Nordbezirk, Ostbezirk, Südbezirk, Innenstadt
Nordbezirk	Polygone	Weststadt, Nordweststadt, Nordstadt
Ostbezirk	Polygone	Nordoststadt, Oststadt, Südoststadt
Südbezirk	Polygone	Südweststadt, Südstadt
Filialen	Polygon	Filiale X, Filiale Y
Aktueller Stellplatz	Kreis (r=10m)	Dynamische Instanziierung
Kundenadresse	Kreis (r=20m)	Kunde 1, Kunde 2, ..., Kunde n
Aktuelle Filiale	Kreis (r=1km)	Dynamische Instanziierung
Umkreis 5km	Kreis (r=5km)	Dynamische Instanziierung
Radius 50m	Kreis (r=50m)	Dynamische Instanziierung

Die einzelnen Stadtteile sind in drei Bezirke aufgeteilt worden. Das Ortsmodell (siehe Kapitel 4.2) erlaubt das Hinzufügen von Ortstypen und Orten zu Ortstypen. Dadurch wurde z. B. der Ortstyp „Stadt“ aus den drei Bezirken und der „Innenstadt" aufgebaut (siehe Tab. 5.2). Bei einigen Orten wird empfohlen, sogenannte Kreis-Ortstypen zu verwenden, wie die nachfolgenden Beispiele zeigen werden. Um die Ortsmodellinstanz zu komplettieren, müssten Lage und Ausmaß der einzelnen Bestandteile angegeben werden. Da dies jedoch der vorliegenden Arbeit keine weiteren Erkenntnisse liefert, wurde darauf verzichtet.

Zur Veranschaulichung der Verwendung von Zuordnungslisten, sind folgende Mietwagenbezirke (siehe Tab. 5.3 bis Tab. 5.6) als bereits vorhandene Ortsmodellinstanzen definiert.

Tab. 5.3: Mietwagen Nordbezirk

Quelle	Ziel
Nordstadt	Mietwagen 2
Nordweststadt	Mietwagen 1
Weststadt	Mietwagen 1

Tab. 5.4: Mietwagen Südbezirk

Quelle	Ziel
Südweststadt	Mietwagen 1
Südstadt	Mietwagen 1

Tab. 5.5: Mietwagen Ostbezirk

Quelle	Ziel
Nordoststadt	Mietwagen 1
Oststadt	Mietwagen 1
Südoststadt	Mietwagen 1

Tab. 5.6: Mietwagen Innenstadt

Quelle	Ziel
Innenstadt	Mietwagen 1
Innenstadt	Mietwagen 2

Für die Anwendungsbeispiele ist eine Zuordnungsliste definiert worden, die für jede Filiale den richtigen Mietwagenverleih wählt (siehe Tab. 5.7).

Tab. 5.7: Mietwagen Stadt

Zuordnungsliste:	
Mietwagen Südbezirk	
Mietwagen Nordbezirk	
Mietwagen Ostbezirk	
Mietwagen Innenstadt	
Quelle	Ziel
Innenstadt	Mietwagen 2

Innerhalb der Anwendungsbeispiele wird eine externe Datenquelle (vgl. modellexterne OE) verwendet. Die externe Datenquelle wird als *Identity Management System* (IMS) definiert, das jeweils mitarbeiterbezogene Daten verwaltet.

Neben dem WfMS der Firma (vgl. Intranet) existiert ein WfMS der Mietwagenfirma/Werkstatt. Die Systeme unterstützen unterschiedliche Aufgaben und tauschen dafür Informationen aus. Eine detaillierte Definition der Schnittstellen ist zum Verständnis der Funktionsweise nicht notwendig.

5.1.3 Subprozess Vor-Ort-Service

In diesem Anwendungsfall sollen besonders wichtige Mitarbeiter (z. B. Manager, Fachpersonal) eines Unternehmens einen sogenannten „Vor-Ort-Service“ erhalten. Diese Mitarbeiter können ein Mietwagen an ihrem Wohnort

anfordern und genau dort auch wieder zurückgeben. Der Prozess des Anwendungsbeispiels wird von der Mietwagenfirma oder von der Firma des Mitarbeiters gesteuert. Beides ist möglich, dass auf eine Differenzierung in der Modellierung (des Prozessbesitzers) verzichtet wurde. Der Mitarbeiter der Firma bzw. der Kunde der Mietwagenfirma wird mit der Adresse angefahren, die das System aus dem *Identity Management System* (IMS) bezogen hat. „Kunde anfahren“ ist das Zielobjekt einer positiven modellexternen OE. Das IMS liefert die Adresse des Mitarbeiters bzw. des Kunden.

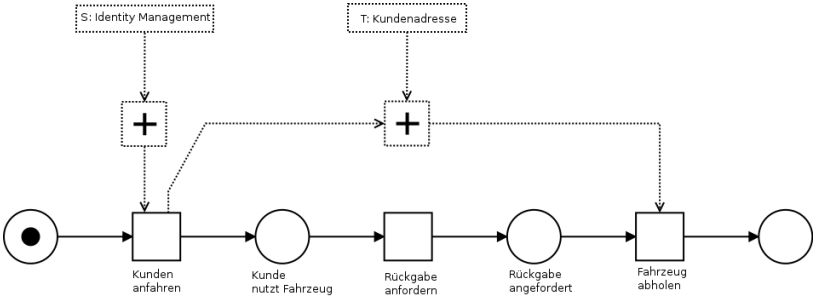


Abb. 5.2: Innerbetriebliches Fahrzeugmanagement: Subprozess Vor-Ort-Service

Mit der positiven ortsbindenden OE mit Quellobjekt „Kunden anfahren“ und dem Zielobjekt „Fahrzeug abholen“ wird die Abholung des Fahrzeugs genau an dem Ort festgelegt, an dem es auch vorab übergeben wurde. In zweiter OE wird der Ortstyp „Kundenadresse“ als ortsbeschreibendes Element verwendet.

5.1.4 Subprozess Fahrzeugauslastungsmanagement

Das Prozessmodell des Fahrzeugauslastungsmanagements ist in jeder der Filialen der Firma auf einem WfMS eingespielt worden. Dieser Prozess gewährleistet, dass zu jedem Zeitpunkt genügend interne Firmenwagen vorhanden sind. Sobald an einem Standort festgestellt wird, dass sich mehr als 10 (oberer Pfad) oder weniger als 2 Fahrzeuge (unterer Pfad) vor Ort befinden, wird der Prozess instanziiert (siehe Abb. 5.3). Im oberen Pfad wird eine

Überbelegung festgestellt bzw. direkt eine Fahrzeugüberführung veranlasst. „Fahrzeug überführen“ ist Quellobjekt einer negativen ortsbindenden OE, damit das Fahrzeug nicht wieder an gleichem Standort (mit Überbelegung) abgegeben wird. Um die Rückgabe des Fahrzeugs an einer unterbelegten Filiale zur gewährleisten, wurde eine positive direkte OE mit aufgenommen. Die positive OE ist in diesem Fall notwendig, da die negative OE lediglich einen Ort verbietet.

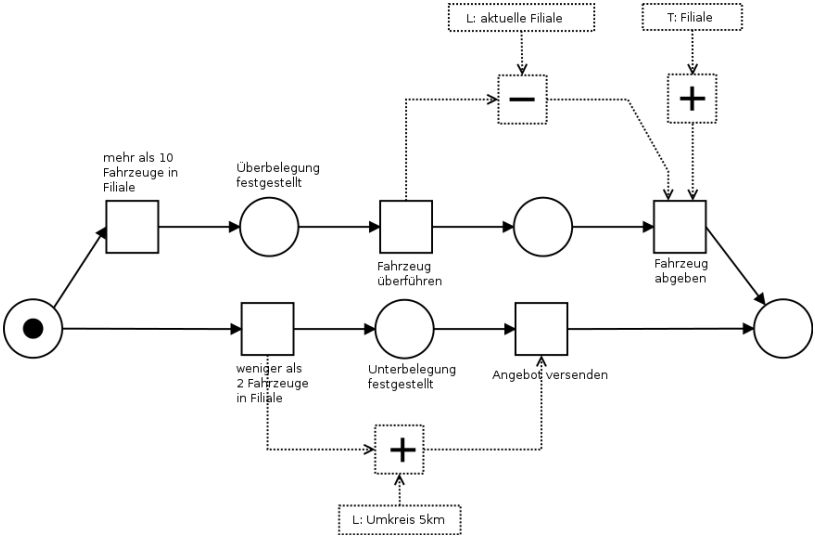


Abb. 5.3: Innerbetriebliches Fahrzeugmanagement: Subprozess Fahrzeugauslastungsmanagement

Die Firma hatte festgestellt, dass viele Mitarbeiter Firmenwagen ausgeliehen haben, ohne sie direkt zu nutzen. Im unteren Pfad (siehe Abb. 5.3) wird daher an alle firmeninternen Fahrzeuge im Umkreis von 5km zur unterbelegten Filiale ein spezielles Mitarbeiterangebot versendet (z. B. Email in das Postfach des Mitarbeiters, der das entsprechende Fahrzeug an diesem Ort vorhält), um die Belegungen auszugleichen.

5.2 Integration in BPMN 2.0

5.2.1 Definition BPMN 2.0

Die BPMN 2.0 eignet sich für die Darstellung und Ausführung von Workflows (siehe Kapitel 2.7). Die OE werden – analog zu den Petri-Netzen – als Annotation für BPMN-Modellelemente eingeführt (siehe Kapitel 4.3). Die Funktionen der Spracherweiterung werden in Abb. 4.8 dargestellt. Die graphische Notation der Spracherweiterung bedarf einer Semantik innerhalb der BPMN 2.0 (siehe Kapitel 4.3.1).

Analog zu den Petri-Netzen werden OE bzw. Ortsbezüge unabhängig vom Kontrollfluss (hier: Sequenz- und Nachrichtenfluss) eingeführt, damit die Ortsbezüge jederzeit ausgeblendet bzw. entfernt werden können.

Innerhalb der BPMN dürfen somit Sequenz- und Nachrichtenflüsse nicht von der Annotation in ihrer Funktion und Bedeutung geändert werden. Daher scheiden beide Flusselemente als Ansatzpunkte für die Modellerweiterung aus. Die BPMN besitzt passende Mechanismen, um die Integration des Konzepts der OE zu gewährleisten. In vorliegender Arbeit werden die zwei Sprachkonzepte **Artefakte** und **Assoziationen** (siehe Kapitel 2.7) verwendet.

Artefakte integrieren spezifische Informationen innerhalb der BPMN-Diagramme (vgl. Kapitel 2.7), welche unabhängig vom modellierten Sequenz- und Nachrichtenfluss sind. Damit ändern Artefakte das Verhalten eines Modells nicht. Da es sich bei den OE um ein spezifisches Domänenkonzept aus dem Umfeld des Geschäftsprozessmanagements handelt und die BPMN-Spezifikation solche Artefakte als Erweiterungsmöglichkeit explizit vorsieht [ObMG10a, S. 57,61], werden in vorliegender Arbeit diese Ansatzpunkte zur Integration verwendet. Domänenspezifische Informationen dürfen den Modellablauf nicht beeinflussen bzw. können daher nicht mit Sequenz- oder Nachrichtenflusselementen verknüpft werden. Artefakte dürfen nicht Quell- oder Zielobjekt eines Sequenz- oder Nachrichtenflusses sein [ObMG10a, S.66ff]. Um die Artefakte mit den Flussobjekten von BPMN zu verbinden,

wird ein neuer Verbindungstyp benötigt. Für solche Verbindungen existieren Assoziationen, die selbst als Artefakt innerhalb des BPMN-Metamodells vorhanden sind.

Die BPMN-Spezifikation definiert die Struktur der Sprache mit UML-Klassendiagrammen und XML-Schema-Definitionen. In Abb. 5.4 wird ein Ausschnitt des Metamodells dargestellt, indem die relevanten Strukturen graphisch in einem UML-Klassendiagramm abgebildet werden. Gleichzeitig existiert eine XML-Schema-Definition (siehe Abb. 5.5), die den korrespondierenden Ausschnitt formal abbildet.

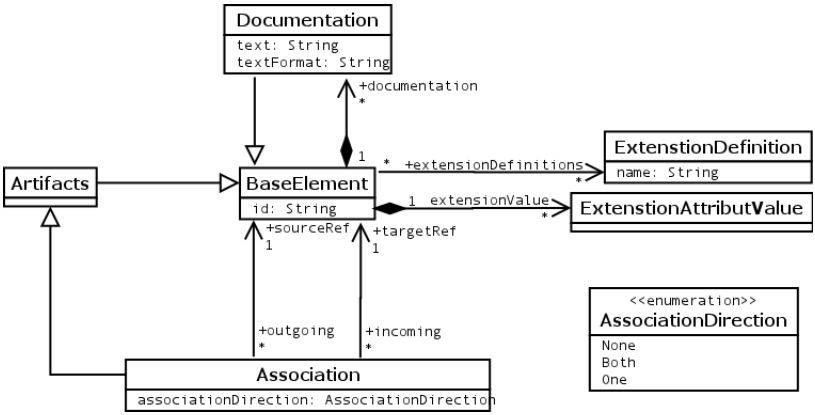


Abb. 5.4: Ausschnitt des BPMN-Metamodells (UML-Klassendiagramm) [ObMG10a]

```
<?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="artifact" type="tArtifact" />
  <xsd:complexType name="tArtifact" abstract="true">
    <xsd:complexContent>
      <xsd:extension base="tBaseElement" />
    </xsd:complexContent>
  </xsd:complexType>
  ...

  <xsd:element name="association" type="tAssociation" substitutionGroup="artifact" />
  <xsd:complexType name="tAssociation">
    <xsd:complexContent>
      <xsd:extension base="tArtifact">
        <xsd:attribute name="sourceRef" type="xsd:QName" use="required" />
        <xsd:attribute name="targetRef" type="xsd:QName" use="required" />
        <xsd:attribute name="associationDirection" type="tAssociationDirection"
                       default="None" />
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:simpleType name="tAssociationDirection">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="None" />
      <xsd:enumeration value="One" />
      <xsd:enumeration value="Both" />
    </xsd:restriction>
  </xsd:simpleType>
  ...

  <xsd:element name="baseElement" type="tBaseElement" />
  <xsd:complexType name="tBaseElement" abstract="true">
    <xsd:sequence>
      <xsd:element ref="documentation" minOccurs="0" maxOccurs="unbounded" />
      <xsd:element ref="extensionElements" minOccurs="0" maxOccurs="1" />
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:ID" use="optional" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
  </xsd:complexType>
  ...

  <xsd:element name="documentation" type="tDocumentation" />
  <xsd:complexType name="tDocumentation" mixed="true">
    <xsd:sequence>
      <xsd:any namespace="##any" processContents="lax" minOccurs="0" />
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:ID" use="optional" />
    <xsd:attribute name="textFormat" type="xsd:string" default="text/plain" />
  </xsd:complexType>
  ...

</xsd:schema>
```

Abb. 5.5: Ausschnitt aus BPMN 2.0 Metamodell (XML-Schema-Definition) [ObMG10a]

Die Klasse *BaseElement* ist abstrakte Superklasse für jegliche Bestandteile des BPMN-Metamodells. Eine Erweiterung dieser Klasse bedingt, dass alle

wesentlichen Klassen die definierten UML-Assoziationen und –Attribute des Metamodells erben. Die Spezialisierungen von *BaseElement* besitzen aus diesem Grund eine automatisch generierte eindeutige ID, die als Wert des geerbten Attributs „id“ abgebildet wird. BPMN-Elemente vom Typ *Documentation* können für die textuelle Beschreibung der abgeleiteten Klassen verwendet werden. Die speziellen Klassen zur Erweiterung von BPMN-Elementen (*ExtensionDefinition* und *ExtensionAttributValue*) werden in vorliegender Arbeit nicht verwendet, da keine bestehenden Elemente erweitert, sondern neue Elemente integriert werden.

Die Klasse *Artifact* des Klassendiagramms bzw. der Typ *tArtifact* (vom XML-Schema) sind die Basis der Erweiterung, da die Integration der OE anhand von Artefakten innerhalb der BPMN-Diagramme abgebildet werden soll. Es wird für jedes neue Konzept eine Klasse von *Artifact* abgeleitet. Die entsprechenden Klassen sind in der Lage, in Kombination mit den bereits vorhandenen Assoziationen, die Informationen einer OE aufzunehmen und graphisch abzubilden. In Abb. 5.6 wird das Artefakt *LocationConstraint* innerhalb des Klassendiagramms beschrieben. Die analoge Erweiterung des XML-Schemas ist in Abb. 5.7 dargestellt.

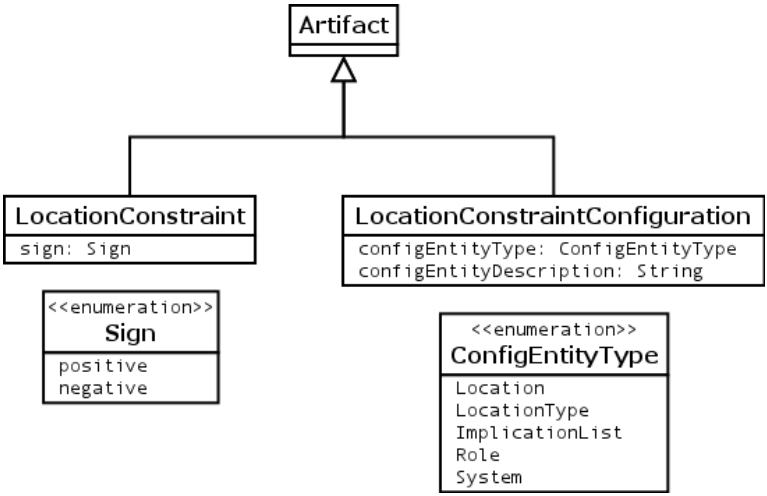


Abb. 5.6: OE-Erweiterung BPMN Metamodell (UML-Klassendiagramm)

```
<xsd:element name="locationConstraint" type="tLocationConstraint"
             substitutionGroup="artifact" />
<xsd:complexType name="tLocationConstraint">
  <xsd:complexContent>
    <xsd:extension base="tArtifact">

      <xsd:attribute name="sign" type="tSign" use="required" />

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:simpleType name="tSign">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="positive" />
    <xsd:enumeration value="negative" />
  </xsd:restriction>
</xsd:simpleType>
```

Abb. 5.7: OE-Erweiterung BPMN (Teil1) Metamodell (XML-Schema-Definition)

Die Artefakte *locationConstraint* bzw. *tLocationConstraint* beschreiben die notwendigen Informationen der in Kapitel 4.3 definierten Darstellung. Das Konzept enthält zusätzlich zu den geerbten Elementen das Attribut *Sign* (*tSign*), um die Art des zugeordneten Vorzeichens aufzunehmen. Die Konfigurationselemente werden über das Artefakt *LocationConstraintConfiguration* beschrieben, das zwei Attribute besitzt: *configEntityType* bestimmt den Typ und *configEntityDescription* die Bezeichner von Ort, Ortstyp oder Zuordnungsliste. Die folgende Auflistung zeigt alle möglichen Attributwerte mit Bedeutungen auf.

- *Location*: der Ort als ortsbeschreibendes Element
- *LocationType*: der Ortstyp als ortsbeschreibendes Element
- *ImplicationList*: die Zuordnungsliste als ortsbeschreibendes Element
- *Role*: eine Identität als Datenquelle
- *System*: ein IS als externe Datenquelle

Die Spezifikation des Konfigurationselements im XML-Schema wird in Abb. 5.8 gezeigt.

```
<xsd:element name="locationConstraintConfiguration"
             type="tLocationConstraintConfiguration" substitutionGroup="artifact" />
<xsd:complexType name="tLocationConstraintConfiguration">
  <xsd:complexContent>
    <xsd:extension base="tArtifact">
      <xsd:sequence>

        <xsd:element name="configEntityDescription" type="xsd:string" />

      </xsd:sequence>

      <xsd:attribute name="configEntityType" type="tConfigEntityType" use="required" />

    </xsd:extension>
  </xsd:complexContent>

</xsd:complexType>

<xsd:simpleType name="tConfigEntityType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Location" />
    <xsd:enumeration value="LocationType" />
    <xsd:enumeration value="ImplicationList" />
    <xsd:enumeration value="Role" />
    <xsd:enumeration value="System" />
  </xsd:restriction>
</xsd:simpleType>
```

Abb. 5.8: OE-Erweiterung (Teil2) BPMN Metamodell (XML Schema Definition)

Die neu definierten Artefakte werden über gerichtete Assoziationen miteinander und mit den Flusselementen verknüpft. Die Assoziationen von OE mit Konfigurationselementen und Flusselementen benötigen jedoch weitere Restriktionen. In Abb. 5.9 sind die zulässigen Kombinationen definiert.

```
-- Eine OE hat mindestens ein Zielobjekt
context LocationConstraint
inv:
   self.outgoing->notEmpty()

-- Die ausgehenden Assoziationen einer OE sind gerichtet
context LocationConstraint
inv:
   self.outgoing
      ->forAll(x | x.associationDirection = AssociationDirection::One)

-- Die Zielobjekte einer OE sind Aktivitäten
context LocationConstraint
inv:
   self.outgoing.taregtRef
      ->forAll(x | x.oclIsTypeOf(Activity))

-- Eine OE hat genau ein Konfigurationselement
context LocationConstraint
inv:
   self.incomming
      ->select(oclIsTypeOf(LocationConstraintConfiguration))
         ->size() = 1

-- Die Quellobjekte einer OE sind Aktivitäten
context LocationConstraint
inv:
   self.incoming.sourceRef
      ->reject(oclIsTypeOf(LocationConstraintConfiguration))
         ->forAll(x | x.oclIsTypeOf(Activity))

-- Die eingehenden Assoziationen einer OE sind gerichtet
context LocationConstraint
inv:
   self.incoming
      ->forAll(x | x.associationDirection = AssociationDirection::One)

-- Ein Konfigurationselement besitzt keine eingehenden Assoziationen
context LocationConstraintConfiguration
inv:
   self.incoming->isEmpty()

-- Ein Konfigurationselement ist nur mit OE assoziiert
context LocationConstraintConfiguration
inv:
   self.outgoing.taregtRef
      ->forAll(x | x.oclIsTypeOf(LocationConstraint))
```

Abb. 5.9: Assoziationsregeln OE-Erweiterung (OCL)

Im obigen OCL-Ausdruck sind Aggregation (vgl. Kapitel 4.3.2) bereits integriert. Ortseinschränkungen wirken in BPMN (nur) auf Aktivitäten. Eine Erweiterung der zulässigen Modellelemente (z. B. Verzweigungen oder Kanten) ist möglich.

Das Konzept der Ortsbezüge (siehe Kapitel 4) ist somit in BPMN 2.0-Diagrammen abbildbar. Weitere Beispiele der OE-Modellierung folgen in Kapitel 5.2.2 und Kapitel 5.2.3 zur Verdeutlichung der Praxistauglichkeit der eingeführten Modellerweiterung.

Analog zu der im vorliegenden Kapitel vorgestellten Integration kann eine Integration in weitere Workflow-Sprachen (z. B. UML-Aktivitätsdiagramme, EPKs) erfolgen. Das Metamodell von UML-Aktivitätsdiagrammen ist ähnlich zum BPMN-Metamodell als XML-Ausdruck spezifiziert [Kech09]. Es wird folglich auf eine widerholende Demonstration in vorliegender Arbeit verzichtet.

Eine praxistaugliche Integration von OE in die BPMN muss jeweils in der Workflow-Engine erfolgen. Die OCL-Spezifikation (siehe oben) müssen dafür in den jeweiligen ausführenden Code der Engine übersetzt und integriert werden.

Ein direkter Vergleich zwischen der Integration in die BPMN und die Petri-Netze wird in vorliegender Arbeit nicht gezogen: Letztendlich werden beim Bau von informationstechnischen Systemen viele Sprachen eingesetzt. Oft sind die eingesetzten Sprachen den Vorlieben der (Produkt-)Entwickler geschuldet. Folglich wird der Einsatz einer Workflow-Sprache empfohlen, die sich innerhalb der (Produkt-)Entwicklung etabliert hat.

5.2.2 Subprozess Mietwagenbestellung

Ein Mitarbeiter möchte eine Dienstreise zu einem Kunden planen. Er konnte über das Intranet kein firmeninternes Fahrzeug buchen, da auf Grund der hohen Auslastung keine mehr bereitstehen. Über den Prozess bzw. das Intranet wird er zum Mietwagenbestellservice weitergeleitet, um sich dort ein

Fahrzeug zu reservieren. Der Mitarbeiter (siehe Abb. 5.10) bestellt das Fahrzeug und vereinbart (nach Antwort einer Autovermietung) einen Termin. Der Mietwagen wird rechtzeitig vor dem Termin intern beauftragt (siehe Abb. 5.10, „Termin“ am Tag der Auslieferung), damit ein Auslieferer der Autovermietung den Wagen zu der Filiale bringt, von dem der Kunde den Auftrag versendet hatte. Hinter der „Fahrzeug ausliefern“-Aktivität befindet sich ein weiterer Subprozess. Sobald der Kunde das Fahrzeug übernommen hat, startet mit dem „Auto übernehmen“-Prozess der Vorgang, der unterstützend vom internen System bereitgestellt wird (siehe Abb. 5.10). Nach der Dienstreise möchte der Mitarbeiter das Fahrzeug zurückgeben und schickt z. B. über eine mobile Applikation den Auftrag „Fahrzeug abgeben“. Der Autovermietungsprozess erhält eine Nachricht mit dem Auftrag und dem Ort, von dem der Auftrag abgesendet wurde. Zum erfolgreichen Abschließen des Prozesses wird noch eine Abrechnung erstellt, die mit dem postalischen Verschicken (siehe Abb. 5.10, Rechnung verschicken) beendet wird. Im Anwendungsbeispiel werden zwei positive OE modelliert. Die OE bestimmen jeweils die Filiale, an dem sich der Mitarbeiter gerade befindet.

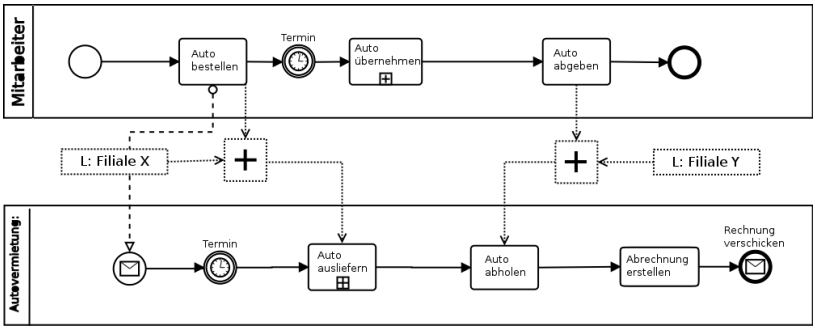


Abb. 5.10: Innerbetriebliches Fahrzeugmanagement: Subprozess Mietwagen-bestellung

Das vorliegende Anwendungsbeispiel könnte auch mit zwei OE, die sich über eine dynamische Instanziierung mit Ortsinformationen versorgen, modelliert werden. Dazu würde beispielsweise ein „Aktueller Stellplatz“ (siehe Tab. 5.2) mit einer maximalen Genauigkeit verwendet werden. Sollte der

Mitarbeiter sich jedoch beim Beauftragen der Autovermietung nicht direkt an diesem Ort befinden, so würden die Ortsangaben nicht präzise genug sein, um den Auftrag (manuelles Eingreifen) auszuführen.

Nachdem sich dieser Service innerhalb der Firma fiktiv bewährt hat, beschließt die Firma, diesen Service auch Kunden im Stadtgebiet über eine öffentlich zugängliche mobile Applikation kostenpflichtig anzubieten. Da nun von jedem Ort im Stadtgebiet ein Mietwagen bestellt werden kann, sind sogenannte Zuordnungslisten eingeführt worden (siehe Abb. 5.11 bzw. Tab. 5.2 bis Tab. 5.5). Das Anwendungsbeispiel (siehe Abb. 5.10) wurde zusätzlich erweitert, damit dem externen Kunden eine Rechnung gestellt werden kann.

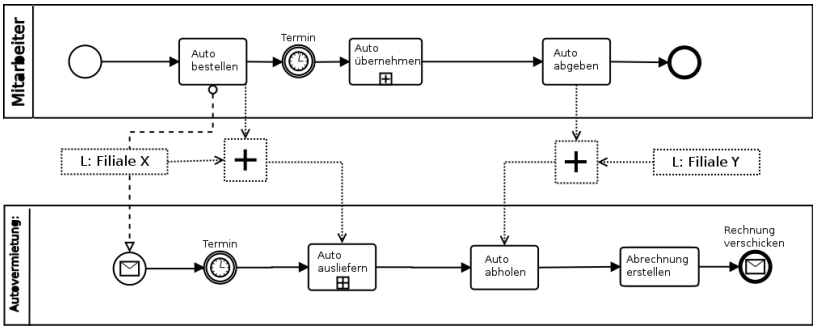


Abb. 5.11: Fahrzeugmanagement: Mietwagenbestellung

Die Zuordnungsliste „Mietwagen Stadt“ enthält mehrere Listen. Die Liste „Innenstadt“ (siehe Abb. 5.11) definiert zwei Tupel mit dem gleichen Quelleintrag (Innenstadt). Die OE bindet somit „Mietwagen 1“ oder „Mietwagen 2“ an das Zielobjekt, wenn das Quellobjekt in der Innenstadt lag. Wenn nicht zusätzlich in „Mietwagen Stadt“ für „Innenstadt“ das Zielobjekt „Mietwagen 2“ als sogenannte Übersteuerung definiert wäre (siehe Tab. 5.7), würde das mobile WfMS selbst entscheiden müssen, welche Mietwagenfirma beauftragt werden soll.

5.2.3 Subprozess Reparaturservice

Mitarbeiter mit Firmenwagen haben die Möglichkeit, jederzeit einen Reparaturservice für ihr Fahrzeug anzufordern. Der Service gewährleistet, dass das Fahrzeug an einer der Filialen abgeholt, repariert und im Anschluss wieder zur Filiale zurückgebracht wird. Falls die Reparatur des Fahrzeugs jedoch länger als einen Arbeitstag dauert, wird ein Leihwagen an die entsprechende Filiale geordert, damit der Mitarbeiter nach der Arbeit nach Hause fahren kann. Der zugehörige Prozess ist in Abb. 5.12 dargestellt.

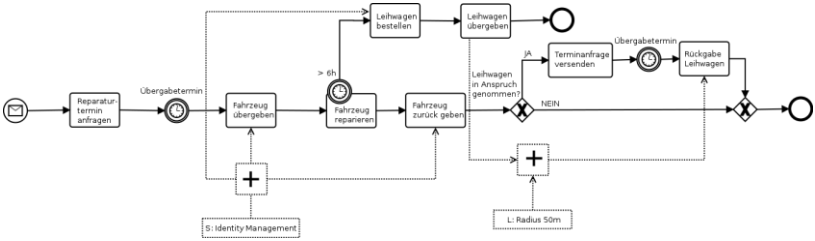


Abb. 5.12: Innerbetriebliches Fahrzeugmanagement: Subprozess Reparaturservice

Der Geschäftsprozess beginnt mit der Aufforderung des Mitarbeiters, eine Reparatur am Fahrzeug durchzuführen. Heutzutage zeigen die Fahrzeuge bei Fahrtantritt dem Fahrer an, welche Wartungs- oder Reparaturarbeiten am Fahrzeug durchgeführt werden müssen. Der Mitarbeiter würde nach der Wahrnehmung solch einer Meldung beispielsweise im Intranet diesen vorliegenden Prozess initiieren. Der Prozess (siehe Abb. 5.12) beginnt mit einer Terminanfrage, die der Mitarbeiter absetzt. Zum (vereinbarten) Übergabetermin wird das Fahrzeug an einer der beiden Filialen der Firma übergeben. Wird während der Ausführung von „Fahrzeug reparieren“ ein Überschreiten der zulässigen Reparaturzeit (<6h) festgestellt, so wird automatisch ein Leihwagen zur entsprechenden Filiale geordert. Sobald die Reparatur abgeschlossen ist, wird das Fahrzeug dem Mitarbeiter an derselben Filiale zurückgegeben. Sofern ein Mietwagen in Anspruch genommen wurde, muss dieser in Folge auch wieder zurückgegeben werden. Daher versendet der

Prozess eine Terminanfrage an den Mitarbeiter und determiniert einen Rückgabetermin mit der Mietwagenfirma. An die Aktivitäten „Fahrzeug übergeben“, „Leihwagen bereitstellen“ und „Fahrzeug zurück geben“ sind positive modellexterne OE gebunden, die im IMS der Firma die Heimatfiliale des Mitarbeiters ermittelt. Im fiktiven Szenario hat sich bei der Übergabe vom Mietwagen bewährt, den gleichen Ort, wie auch bei der Übernahme, auszuwählen (siehe Abb. 5.12, positive ortsbindende OE mit Radius 50 m).

In den vorliegenden fiktiven Anwendungsbeispielen wurde die Praxisrelevanz der OE demonstriert. Es wurde beispielhaft gezeigt, wie aus einem Geschäftsprozess für interne Zwecke ein neues Geschäftsmodell erschlossen werden kann, sofern eine Automatisierung innerhalb eines Informationssystems vorab erfolgte.

5.3 Ergänzende Spracherweiterungen

Bei der Modellierung von Ortsbezügen innerhalb von Workflow-Modellen, können weiterführende Sprachergänzungen helfen, den Umgang zu vereinfachen.

5.3.1 Algorithmen der Auflagenliste

Durch Schleifen innerhalb von Workflow-Modellen kann es (vgl. Abb. 4.35) zu inaktiven OE kommen. Es können Pfade definiert werden, die einzelne Elemente des Modells mehrfach enthalten. Solche Rücksprünge können innerhalb der Petri-Netze und BPMN 2.0 definiert werden und führen dazu, dass Quellobjekte indirekter OE mehrfach ausgeführt werden. Dadurch wiederum entstehen redundante dynamische Basisdaten an den betroffenen OE. Im Falle unterschiedlicher dynamischer Basisdaten entstehen mehrere alternative Auflagen, die in einer sogenannten **Auflagenliste** gesammelt werden. Die Auflagenliste ist zu Beginn der Prozessinstanz leer und wird erst im Laufe des Kontrollflusses gefüllt. Es stellt sich die Frage, welche Auflage innerhalb der OE nun berücksichtigt werden soll.

Eine Alternative wäre, die dynamischen Basisdaten grundsätzlich nicht zu berücksichtigen, die aus zurückliegenden Ausführungen der Quellobjekte stammen. Somit würden ausschließlich die direkt vorangegangenen Auflagen für das Zielobjekt relevant werden. In Anbetracht der Auflagenliste wird das letzte Element der Liste (LAST) angewandt. Analog zu diesem Verfahren kann nur das erste Element der Liste ausgewählt werden (FIRST). FIRST identifiziert die Auflage, die in der Workflow-Instanz zuerst generiert und der Auflagenliste hinzugefügt wurde. Alternativ können alle Elemente der Auflagenliste (ALL-Verfahren) berücksichtigt werden. Jedoch können mehrere konkurrierende Auflagen entstehen, die dadurch zu einer Anomalie (siehe Kapitel 4.3.6) führen, falls sich darunter Auflagen gegenseitig widersprechen. Eine weitere Alternative wählt eine zufällige Auflage der Liste aus (ANY). In Tab. 5.8 werden die einfachen Algorithmen der Auflagenliste aufgelistet erläutert.

Tab. 5.8: Auswahlverfahren dynamischer Basisdaten

Algorithmus	**Beschreibung**
LAST	letzter Eintrag der Auflagenliste wird berücksichtigt
FIRST	erster Eintrag der Auflagenliste wird berücksichtigt
ALL	alle Auflagen der Auflagenliste werden berücksichtigt
ANY	eine beliebige Auflage der Auflagenliste wird berücksichtigt

Die Auflagenliste selbst ändert sich durch die Anwendung eines Algorithmus und durch die gleichzeitige Ausführung des Zielobjektes nicht. Für eine erneute Ausführung des Zielobjektes (in beispielsweise einer Schleife) erfolgt die Auswahl auf Basis desselben Verfahrens mit erweiterter Auflagenliste.

Falls eine Änderung der Auflagenliste angefordert ist, z. B. wenn ein Quellobjekt eine Auflage für das Zielobjekt erzeugt und nach dessen Ausführung die Auflage sofort gestrichen werden soll, muss ein weiterer Mechanismus

implementiert werden. Die Algorithmen (siehe Tab. 5.8) werden mit zusätzlichen Optionen ausgestattet. Das Minuszeichen (-) gibt die Löschung des Eintrags nach der Ausführung an. Die Verfahren FIRST^{-} und LAST^{-} sollen analog zum Konzept der *Verbrauchsfolgeverfahren* (vgl. [Enge93]) Anwendung finden. Dabei ist *Last In* gleichbedeutend mit *First Out* (LIFO) und *First In* gleichbedeutend mit *First Out* (FIFO). Die dynamischen Basisdaten werden nach der Reihenfolge des Auftretens in die Auflagenliste geschrieben. Im Gegensatz zu den Algorithmen FIRST bzw. LAST ist die Auflagenliste veränderbar. Der LAST^{-}-Algorithmus wählt das zuletzt hinzugefügte Element aus der Liste aus und löscht dieses danach. Im FIRST^{-}-Algorithmus wird das zuerst hinzugefügte Element (das Älteste) ausgewählt und danach aus der Liste entfernt. Analog funktionieren ANY^{-} und ALL^{-}. Bei ANY^{-} kann es passieren, dass aktivierte OE nach der Ausführung des Zielobjektes inaktiv sind. Die Algorithmen erlauben die Definition von modellweiten Regeln für die Auswahl der Basisdaten. Teilweise ist es sinnvoll, die Wahl des Algorithmus von der individuellen OE abhängig zu machen bzw. diese individuell zu gestalten. Wird keine andere Angabe gemacht, wird das LAST-Verfahren als Standard angewandt.

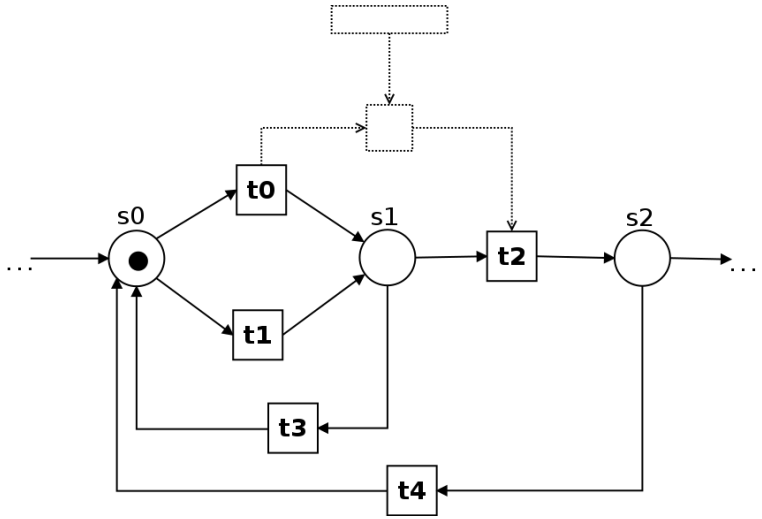


Abb. 5.13: Prozessschleifen in Petri-Netzen

In Abb. 5.13 wird ein Workflow-Modell dargestellt, das über eine indirekte OE verfügt. Innerhalb des Modells ist ein Pfad definiert, der das Zielobjekt vor der Ausführung des Quellobjektes erreichen kann. Andere Pfade sind für die spezielle Ausführung von Quell- oder Zielobjekt integriert. Mit t4 erfolgt ein Rücksprung an den Anfang des Prozesses (Schleife), so dass die Variablen der Auflagen für die OE (im Standard-Fall) zurückgesetzt werden können.

Ein möglicher Verlauf der Workflow-Instanz ist in Tab. 5.9 exemplarisch dargestellt. Die Spalten repräsentieren jeweils einen Zeitpunkt innerhalb der Workflow-Instanz. Die Erzeugte Auflage wird jeweils immer nach diesem Zeitpunkt angegeben. Im Anwendungsbeispiel — nach der Schaltung von Transition t0 — wird eine Auflage „a“ erzeugt. In der dritten Zeile wird die Auflagenliste (für LAST, FIRST, ALL und ANY) abgebildet, die bei der aktuellen Ausführung des Quellobjekts erzeugt werden (siehe Kleinbuchstaben „a“, „b“ und „c“).

Im unteren Teil von Tab. 5.9 werden vier unterschiedliche Verfahren dargestellt, die jeweils keine Manipulation der Auflagenliste anbieten. In der dritten Zeile ist die Auflagenliste dargestellt (abhängig vom Zeitpunkt und ohne Beeinflussung von $LAST^{-}$, $FIRST^{-}$, ALL^{-} und ANY^{-}). Mit $LAST^{-}$, $FIRST^{-}$, ALL^{-} und ANY^{-} werden nach der Ausführung einer Auflage diese aus der Liste entfernt. Der Zustand vor und nach einer Ausführung des jeweiligen aktiven Workflow-Modellobjekts kann anhand der Tab. 5.9 und Abb. 5.13 nachvollzogen werden.

Tab. 5.9: Auswahlverfahren mit/ohne Veränderung der Auflagenliste

Ausgeführte Transitionen (in Reihenfolge)	**t0**	**t2**	**t4, t1**	**t3, t0**	**t3, t0**	**t2**
erzeugte Auflagen	a	-	-	b	c	-
Auflagenliste	a	a	a	a, b	a, b, c	a, b, c
LAST	-	a	a	-	-	c
FIRST	-	a	a	-	-	a
ALL	-	a	a	-	-	a ∩ b ∩ c (∩= logische UND – Verknüpfung)
ANY	-	a	a	-	-	a ∪ b ∪ c (∪= logische ODER – Verknüpfung)
LAST^{-}	-	a	-	-	-	c
FIRST^{-}	-	a	-	-	-	b
ALL^{-}	-	a	-	-	-	b ∩ c
ANY^{-}	-	a	-	-	-	b ∪ c

5.3.2 Reset-Objekte

Bei der Modellierung von OE innerhalb von Workflow-Modellen ist es sinnvoll, einzelne Auflagen (von OE) gezielt zu steuern: Die Ausführung einer Operation kann an den Zeitpunkt, an dem der Kontrollfluss ein Objekt innerhalb des Workflow-Modells erreicht, gebunden werden. Ein solches Objekt wird als Reset-Objekt (für eine OE) bezeichnet. Für interaktive Prozessstrukturen (vgl. Schleifen) ist das Zurücksetzen von Auflagen hilfreich. Der Modellierer kann mit diesem Mechanismus direkt in die Ausführungslogik

von OE eingreifen. Jeder OE können beliebig viel Rücksetz-Objekte zugewiesen werden.

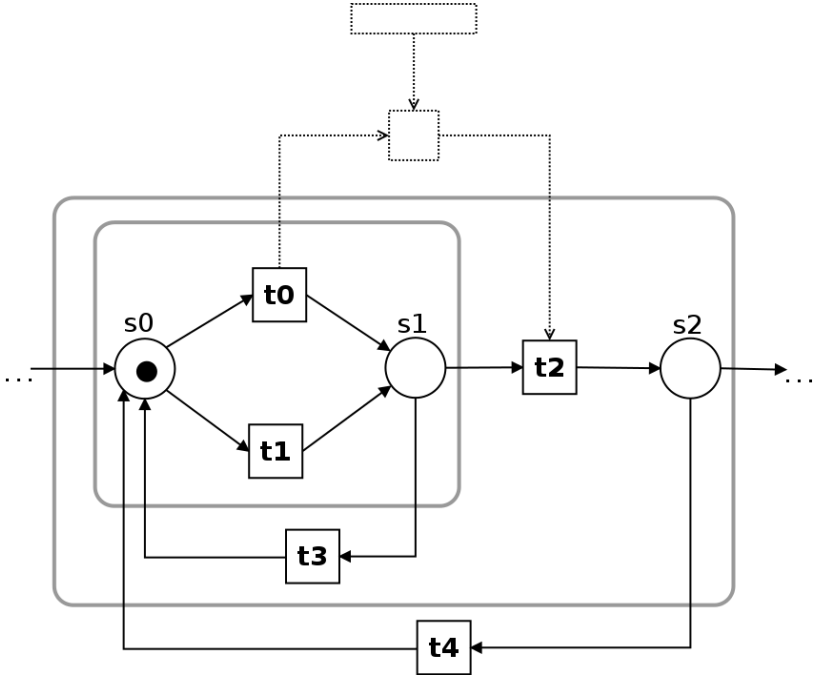


Abb. 5.14: Rücksetz-Objekte in Petri-Netzen

Das in Abb. 5.14 dargestellte Prozessbeispiel beinhaltet zwei Bereiche, die mit abgerundeten Rahmen dargestellt werden. Die Inhalte der Rahmen können jeweils als abgeschlossene Einheiten betrachtet werden, die mehrfach ausführbar sind (vgl. Subprozess BPMN 2.0). Wenn beispielsweise eine Marke in Stelle s2 vorhanden ist und t4 schaltet, startet die Ausführung des Modells (die Instanz) gewissermaßen neu. Sind beispielsweise die vorhandenen Auflagen der Auflagenliste nicht ausführbar, so kann t4 als geeignetes Reset-Objekt verwendet werden. Der Workflow-Modellanteil im inneren Rahmen kann mit Hilfe von t3 (mehrfach) ausgeführt werden.

Falls bei der Ausführung innerhalb einer Instanz die Reset-Objekte t3 und t4 mehrfach schalten, kann es notwendig werden, die OE zu deaktivieren bzw. die Basisdaten zu löschen. Die Handhabung der Rücksetz-Objekte muss genau determiniert werden. Je nach Anwendungsfall macht es Sinn, diese vollständig zu löschen. In anderen Anwendungsfällen ist es sinnvoll, diese nur teilweise zurückzusetzen. Bei modellinternen OE können statische Regelungen bzw. Verfahren Anwendung finden, analog zu den Algorithmen, die in Kapitel 5.3.1 beschrieben wurden (z. B. LAST, FIRST, ALL). Mit Hilfe solcher Verfahren könnten neue Regeln definiert werden: z. B. könnten externe Datenquellen als eigene Auflagen bei modellinternen OE die Entscheidung liefern, ob ein Rücksprung (Reset) ausgeführt wird.

6 Architektur eines mobilen WfMS

Im vorliegenden Kapitel wird ein WfMS mit den Prinzipien einer SOA entworfen. Aus den modellierten Workflows (mit Ortsbezügen) werden Dienste einer SOA abgeleitet und implementiert. Eine SOA liefert strukturelle Vorteile beim Entwurf eines Informationssystems [GoJW04].

Die Architektur des mWfMS wird anhand des „*4+1-View*“ [Kruch95] präsentiert (siehe Kapitel 2.9). Die mobil-spezifischen Szenarien werden in Kapitel 6.1 auf Grundlage von verschiedenen Anwendungsdomänen (vgl. Szenarien, Abb. 2.22) erörtert. In Kapitel 6.2 wird die logische Sicht, in Kapitel 6.3 die prozessorientierte Sicht, in Kapitel 6.4 die Entwicklungssicht und in Kapitel 6.5 die physikalische Sicht auf das mWfMS vorgestellt.

6.1 Szenarien

In den folgenden Anwendungsbeispielen werden Workflows (mit Ortsbezügen) genutzt, um Geschäfts- und Dienstleistungsmodelle zu integrieren. Die Kontextauswertung bzw. die Kontextsensitive Anpassung der Workflowinstanzen sind erforderlich, um die Ausführung der Workflows z. B. ortsabhängig zu gewährleisten. Die sich ändernden Rahmenbedingungen (z. B. Kontextänderungen, gesetzliche Vorgaben) erfordern in den Anwendungsbeispielen einen flexiblen Umgang, so dass Änderungen zu jedem Zeitpunkt integriert werden können. Die Änderungen der Rahmenbedingungen führen ggfs. zum Anpassungsbedarf bei der Leistungserbringung und damit zur Anpassung der Workflows [SWNN06]. Es werden spezifische Varianten der Leistungserbringung bzw. der Workflows benötigt, damit möglichst viele Anwendungsfälle bzw. unterschiedliche Rahmenbedingungen berücksichtigt werden können. Generische Fassungen der Workflows sorgen u.a. für die Gewährleistung der Flexibilität.

Die Workflows müssen neben den Ortsbezügen auf weitere Kontexte (z. B. Ressourcenverfügbarkeit) reagieren. Kontextabhängige Änderungen treten vor und während der Ausführung von Prozessen auf.

Bei technischen Änderungen und gesetzlichen Vorgaben kann von einer bestimmten Vorlaufzeit ausgegangen werden. Die Leistungskonfiguration ändert sich somit nicht während der Ausführungsphase (siehe Kapitel 3.1). Für langlaufende Workflows müssen zusätzliche Mechanismen (Instanz-Migrationen) implementiert werden, die nicht Gegenstand vorliegender Arbeit sind.

Im Folgenden werden Arbeitsprozesse aus drei verschieden betrieblichen Domänen beschrieben, die zu einer allgemeinen bzw. generischen Architektur führen. Mit der Architektur sollen die Domänen Anwendungsgerecht unterstützt werden.

Landwirtschaftliche Arbeitsprozesse

In den landwirtschaftlichen Arbeitsprozessen werden Ernteprozesse (z. B. Düngen, Pflügen, Maisernte, Grünfutterernte) betrachtet. Ernteprozesse werden von mehreren Fahrzeugen innerhalb einer Arbeitsgruppe bzw. Fahrzeuggruppe vollzogen (siehe Abb. 6.1). Neben der Erntemaschine wird z. B. bei der Maisernte ein geländefähiges Umlade-Gefährt (meist landwirtschaftliche Zugmaschine mit Anhänger) benötigt, das während dem Ernten in der sogenannten Parallelfahrt das Erntegut übernimmt. Für diesen Anwendungsfall besitzt die Erntemaschine einen Erntevorsatz, über den das Umlade-Gefährt beladen wird. Eine weitere Umladung findet z. B. zwischen dem am Feldrand stehenden LKW und dem Umlade-Gefährt statt.

Innerhalb einer Arbeitsgruppe werden mehrere Erntemaschinen, Umlade-Gefährte und LKWs eingesetzt. Da die Fahrzeuge einen effizienten Einsatzplan benötigen, wird die Auslastung über das mWfMS gesteuert. Für das Fahrzeuggruppenmanagement werden die Ortsinformationen der mobilen Fahrzeuge in Echtzeit benötigt. Das mWfMS muss die Ortsinformationen von allen Fahrzeugen abgleichen, um z. B. die Parallelfahrtstrecke des Umlade-Gefährts berechnen zu können. Bei der Kommunikation innerhalb der

Arbeitsgruppe muss die Datenübertragung (notwendige Bandbreite) sichergestellt werden. Innerhalb eines mobilen Fahrzeugs müssen die Umgebungskontexte (z. B. Ort, Sensorkontexte) erfasst werden, damit sie im mWfMS ausgewertet werden können. Die verschiedenen Arbeitsprozesse werden vorab von Experten der landwirtschaftlichen Domäne modelliert, damit möglichst viele fachliche Kontexte innerhalb der Workflows beim Ausführen berücksichtigt werden.

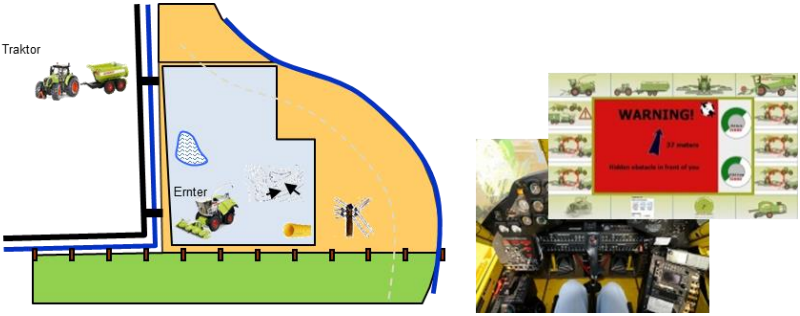


Abb. 6.1: Landwirtschaftliche Anwendungsbeispiel: Feldarbeit in der Arbeitsgruppe (links), Fahrerkanzel bzw. Fahrer-Monitor (rechts)

Baubetriebliche Arbeitsprozesse

Das mWfMS soll baubetriebliche Anwendungsfälle unterstützen. Baubetriebliche Arbeitsprozesse umfassen z. B. die Baustellenorganisation während des Baus einer Straße. Analog zu den landwirtschaftlichen Arbeitsprozessen existieren viele mobile Arbeitsgruppen, die koordiniert werden müssen. Bei baubetrieblichen Arbeitsprozessen werden jedoch die Arbeitsgruppen wiederum in zusätzliche oder weitere Arbeitsgruppen organisiert (z. B. bei Großbauprojekten), was zu einer steigenden Komplexität der Fahr- und Arbeitsgruppenorganisation führt.

Die Baustellenorganisation erfolgt im Allgemeinen über einen sogenannten Vorarbeiter. Mehrere Maschinen und Arbeiter müssen gesteuert und koordiniert werden. Innerhalb des mWfMS müssen die Ortsinformationen der ver-

schiedenen mobilen Geräte vorgehalten werden (z. B. aktueller Standort eines bestimmten Radladers). Bei baubetrieblichen Arbeitsprozessen werden flexible Arbeitsabläufe benötigt, da oft auf Änderungen (z. B. eine blockierte Fahrtstrecke) reagiert werden muss. Die Prozesse müssen in verschiedenen Varianten modelliert werden, so dass Änderungen (Varianten) von Arbeitsabläufen zur Laufzeit möglich werden.

In baubetrieblichen Arbeitsprozessen werden die mobilen Maschinen und die Arbeiter mit Hilfe von mobilen Geräten unterstützt. Der (mobile) Arbeiter kann über das mobile Gerät mit Hilfe einer App beim Verfolgen der Arbeitsprozesse Unterstützung erhalten (vgl. Kapitel 7.3). Spezielle nutzergerechte Endbenutzerschnittstellen sind notwendig, um die entsprechenden Anwendungsfälle durch mobile IKT zu unterstützen (siehe Anhang B).

Arbeitsprozesse bei der mobilen Wartung

Bei der mobilen Wartung sollen mobile Arbeitsprozesse (z. B. Wartung einer IT-Anlage beim Kunden) unterstützt werden. Eine Kunde möchte seine IT-Anlage warten lassen und beauftragt dafür einen (externen) Techniker, der alte bzw. defekte Teile austauscht und die Anlage auf deren Funktionstüchtigkeit prüft. Für die Anfahrt zum Kunden nutzt der Techniker bzw. Wartungsmitarbeiter ein Fahrzeug. Das Fahrzeug soll zur Unterstützung der Arbeitsprozesse mit einem Prozess-basierten System ausgestattet werden. Beim Austausch von Hardware muss der Wartungsmitarbeiter die Ersatzteile teilweise mitführen (z. B. Austausch eines unterbrechungsfreien Strom-versorgungsgeräts).

Zur Unterstützung der Arbeitsprozesse vor Ort soll der Wartungsarbeiter ein mobiles Gerät mit sich führen. Eine Aufgabenliste, Hilfestellungen und Hinweise sollen auf dem mobilen Endgerät angeboten werden (vgl. Kapitel 7.3). Eine Endbenutzer- und Anwendungsgerechte App (siehe Anhang B) kann helfen, die Arbeitsprozesse der mobilen Wartung für den Endbenutzer zu verbessern.

Anforderungen

Die Architektur des mWfMS wird innerhalb der vorab genannten drei Anwendungsdomänen evaluiert. Die mobilen Arbeitsprozesse werden über Kundenaufträge initiiert. Die Aufträge werden innerhalb einer Leistungskonfiguration (vgl. Kundenshop) eingestellt.

Die mobilen Arbeitsmaschinen sollen in Arbeitsgruppen (Bsp.: Erntemaschine, Traktor) organisiert werden und untereinander Informationen austauschen können. Die mobilen Arbeitsmaschinen (Fahrzeuge) erhalten eine Endbenutzerschnittstelle (Monitor) innerhalb der Fahrerkanzel bzw. des Fahrzeugs (siehe z. B. Abb. 6.1), um dem Fahrzeugführer Workflow-relevante Informationen (Kontexte) darzustellen. Eine mobile Recheneinheit muss für die Ansteuerung des Monitors bzw. zur Kommunikation mit den anderen mobilen Arbeitsmaschinen innerhalb des mobilen Fahrzeugs integriert werden.

Mobile Akteure ohne mobile Arbeitsmaschine können mit einem Smartphone unterstützt werden, um Informationen des mWfMS zu erhalten. In vorliegender Arbeit werden verschiedene Anwendungsfälle zur Unterstützung mobiler Aktivitäten bzw. mobiler Benutzeraufgaben untersucht (siehe Kapitel 7.2). Da eine weiterführende Betrachtung mobiler Benutzeraufgaben keine neuen Erkenntnisse liefert, wurde darauf verzichtet, eine App zur Unterstützung dieses Anwendungsfalls in vorliegender Arbeit zu erörtern.

Die Architektur des mWfMS soll nach dem SOA-Paradigma entworfen werden: lose Kopplung, Abstraktion, Standardisierung, Wiederverwendbarkeit, Skalierbarkeit, eindeutige Schnittstellen und große Kohäsion [Heut07, S. 21-62]. Das mWfMS soll plattformunabhängig entwickelt werden, damit es innerhalb von weiteren Projekten mit mobilen Arbeitsprozessen angewendet werden kann.

Bei der mobilen Ausführung von Teilleistungen werden Daten generiert, die für die Erzeugung der Gesamtleistung notwendig sind. Diese sensiblen Daten müssen unter den Gesichtspunkten der Vertraulichkeit, Integrität und Authentizität geschützt werden (Datensicherheit).

Die mobilen Arbeitsmaschinen benötigen jeweils ihre eigenen Ortsinformationen bzw. die Ortsinformationen der anderen Arbeitsmaschinen in der Fahrzeuggruppe, um z. B. die kürzeste Fahrtstrecke berechnen zu können. Die Änderung von Ortsbezügen (vor und während der Ausführung) wird mit den direkten und indirekten OE berücksichtigt (siehe Kapitel 4.1.2 und Kapitel 4.1.3). Weitere Kontextinformation z. B. Bodendichte, Wetterinformationen oder Erntestände müssen zusätzlich vom mWfMS unterstützt werden.

In Tab. 6.1 werden die (groben) nicht-funktionalen und funktionalen Anforderungen an das mWfMS zusammengefasst.

Tab. 6.1: Anforderungen an die mWfMS-Architektur

Bez.	Anforderung	Nicht-fkt.nal	Beschreibung
R1	Service-orientierte Architektur	X	lose Kopplung, Abstraktion, Standardisierung, Wiederverwendbarkeit, Skalierbarkeit, eindeutige Schnittstellen, große Kohäsion.
R2	WfMS	X	Workflow-Modellierung, -Verarbeitung, -Monitoring (Workflow-Lebenszyklus)
R3	Leistungskonfiguration	-	Ein Domänen-Verantwortlicher soll die Möglichkeit erhalten eine Leistungskonfiguration der mobilen Arbeitsprozesse (Leistungen), die gegenüber dem Kunden erbracht werden, über ein geeignetes Frontend vollziehen zu können.
R4	Plattformunabhängig	X	Die Architektur soll plattformunabhängig sein, damit es in vielen Unternehmen mit unterschiedlichen mobilen Anwendungsfallbeispielen angewandt werden kann.
R5	Endbenutzerschnittstelle	-	Der mobile Akteur/Arbeiter soll die Möglichkeit erhalten, die gerade laufenden Prozesse einzusehen und ggfs. zu beeinflussen.

R6	Recheneinheit	-	Das mobile Fahrzeug (der Fahrzeuggruppe) soll jeweils eine eigene Recheneinheit erhalten, um ggfs. autonome Entscheidungen (unabhängig von der Arbeitsgruppe zu treffen). Zusätzlich soll die Ansteuerung der Endbenutzerschnittstelle (siehe Abb. 6.1) erfolgen.
R7	Ortsinformationen	X/-	Eigenortung (siehe Kapitel 2.1.6) des Fahrzeugs; der Domänen-Verantwortliche soll die Möglichkeit erhalten per Modellierung/Konfiguration die Prozessabläufe zu definieren bzw. zu ändern.
R8	Kontextverarbeitung	-	Das mWfMS soll Kontexte verarbeiten und auswerten, um die Workflow-Steuerung zu beeinflussen.
R9	Datensicherheit	X	Sensible bzw. Personenbezogene Daten sollen bzgl. Vertraulichkeit, Integrität und Authentizität geschützt werden.
R10	Multicast-Kommunikation	-	Die mobile Arbeitsgruppe soll einen Informationsaustausch jeglicher Recheneinheiten (auf Basis der Kontextauswertungen) gewährleisten.

Die Anforderungen wurden innerhalb des Verbundprojektes Robot2Business (R2B) zusammen mit den Anwendungsdomänen Baubetrieb, Landwirtschaft und Systemwartung erhoben. Die Generalisierung des mWfMS stand stets im Vordergrund des Verbundforschungsprojektes, so dass das mWfMS in weiteren Domänen mit mobilen Anwendungsfällen angewandt werden kann.

6.2 Logische Sicht

Im mWfMS wird eine Konfiguration der vom Auftraggeber angeforderten Leistung benötigt (siehe Tab. 6.1, „R3“). Zwischen Leistungskonfiguration und den Prozessen wird eine direkte Abbildungsvorschrift etabliert. Die Leistungen ändern sich nicht zur Laufzeit, so dass eine direkte Abbildung

auf konfigurierende Prozesse vor der Ausführung durchführbar ist. Die ausgewählten Produkte (z. B. Sachgüter, Dienstleistungen) werden auf einen bestimmten Workflow (bzw. Workflow-Kombination mit zusätzlicher Parametrisierung) abgebildet. Die Workflows werden abhängig von den zur Verfügung stehenden Ressourcen ausgeführt. Um die Leistungskonfiguration auf die Prozesse abzubilden, wird ein Key-Value-Modell [CoCL11] eingeführt. Dessen Aufgabe ist es die Zuordnung der entsprechenden Leistungskombination auf die notwendige Prozesskombination zu gewährleisten. Das Dienstleistungsprogramm wird direkt mit den Workflows verknüpft (siehe Kapitel 6.1). Die konfigurierte Dienstleistung wählt eine bestimmte Kombination von Prozessen in der Engine aus, konfiguriert ggfs. Bestandteile (Varianten der Prozesse) und stößt die Ausführung an, sobald die Dienstleistung durch den Kunden final bestätigt wurde. Die Komponente zur Gewährleistung der Leistungskonfiguration wird im mWfMS als „*Configurator“* (siehe Abb. 6.2) bezeichnet.

Für die Abbildung der konfigurierten Leistung auf die Prozesse ist ein *Repository* (bzw. eine Datenbank) zur Speicherung der *Key-Value*-Paare [CoCL11] notwendig (siehe Abb. 6.2, „*rules“*). Das Service-Repository wird benötigt (siehe Abb. 6.2, „*Service Modules“*), damit die Konfiguration gespeichert werden kann. Zur Entwicklung eines SOA-basierten mWfMS wird jeder (zustandsbehaftete) Workflow als Dienst bezeichnet. Die angebotenen Leistungen können in weitere Kategorien (Leistungsarten: z. B. Basis-, Zusatzleistung) untergliedert werden (siehe Abb. 6.4). Die Leistungsarten werden über das zur Verfügung stehende Regelwerk miteinander kombiniert, um die vom Kunden geforderten Leistungen zu konfigurieren. Die Auswahl der Leistungen wird durch die Komponente des „*Service Configurator“* unterstützt (siehe Abb. 6.2).

Der Auftraggeber (Kunde) oder Domänen-Verantwortliche fordert im „*Service Configurator“* über eine webbasierte GUI die Leistung an (siehe Abb. 6.2, „*User Interface“*). Das „*User Interface*“ wird über einen Applikationsserver angeboten. Die angeforderten Leistungen werden in das *„Enterprise Resources“* Repository geschrieben, um die Leistungen von dort in die jeweiligen betriebswirtschaftlichen Unternehmenssysteme zu buchen. Das

ist notwendig, um z. B. Rechnungen oder Belege zu erstellen. Die Dienste (bzw. Workflows) werden über den „*Service Configurator*“ konfiguriert und in das „*Configured Service*“ Repository geschrieben.

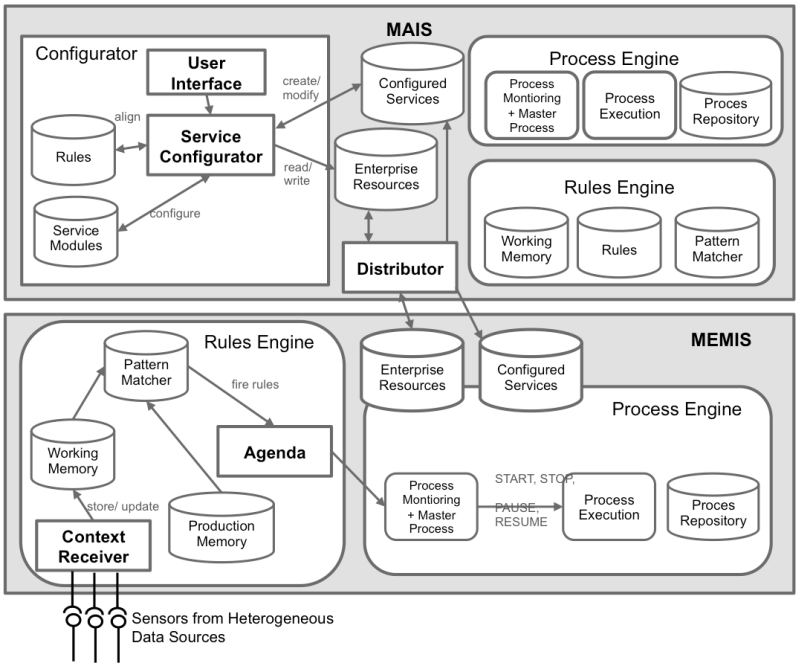


Abb. 6.2: Logische Sicht des mWfMS

Jede Leistung besitzt eine eindeutige Abbildung hinsichtlich einer Dienste-Kombination. Jeder Dienst kann wiederum ein Workflow sein. Mit der Definition der Leistungskombination steht somit eine Workflowkombination bereit, die zur Unterstützung der Leistungserbringung wiederum andere Workflows startet.

Neben dem Konfigurator ist die „*Process Engine*“ und eine (weitere) „*Rules Engine*“ Bestandteil des sogenannten *Management Instance System*

(MAIS[1]). Das MAIS ist die zentrale stationäre Einheit einer Arbeitsgruppe. Die Arbeitsgruppe ist (per WLAN) mit der MAIS am stationären Standort des Fuhrparks verbunden.

Die mobilen Fahrzeuge der Arbeitsgruppe verfügen über ein sogenanntes *Member Management Instance System* (MEMIS). In den Anwendungsfällen kollaborieren mehrere mobile Maschinen automatisiert. Zur Unterstützung verteilter Arbeitsgruppenprozesse (vgl. [ARMC14]) werden jeweils im führenden Fahrzeug der Fahrzeuggruppe (z. B. Erntemaschine) eine mobile Instanz des MEMIS eingespielt. Das MEMIS besitzt die selbe Architektur, wie das MAIS (siehe Abb. 6.2).

Da die Leistungen nur einmal in Auftrag gegeben werden und Enterprise-Ressourcen (siehe Abb. 6.2) benötigen, wird eine unterbrechungsfreie Verbindung zum Internet vorausgesetzt, damit die Kunden jederzeit Änderungen vornehmen können. Der Konfigurator sollte daher nicht auf einer mobilen Arbeitsmaschine implementiert werden. Mit dem Entwurf des MEMIS analog zur MAIS (siehe Abb. 6.2), könnte die MAIS auf einer mobilen Arbeitsmaschine betrieben werden. Diese Fähigkeit wurde innerhalb des Verbundprojektes R2B entwickelt, wurde jedoch in keinem der Anwendungsfälle angefordert.

In dem MAIS musste zusätzlich ein *„Distributor“* implementiert werden, der für die Verteilung der vorab konfigurierten Leistungen (konfigurierte Workflows) auf die MEMIS zuständig ist.

Abhängig von der mobilen Arbeitsmaschine werden Sensoren bzw. Sensordaten verarbeitet. Die mobilen Arbeitsmaschinen sind Spezialmaschinen für einen bestimmten Zweck (z. B. Feldhäcksler). Die Sensordaten werden innerhalb der „*Rule Engine“* angebunden (siehe Abb. 6.2). Der „*Context Receiver“* ist die Eingangsschnittstelle, die je nach Sensordatum entscheidet (vgl. Filter), welche Information in das „*Working Memory“* geschrieben wird. Der „*PatternMatcher“* repräsentiert die eigentliche Regelverarbeitung

[1] Selbst geschaffener Begriff des Verbundprojektes R2B.

in der „*Rules Engine*“. Sobald ein *Pattern* zutrifft, wird die „*Agenda*“ benachrichtigt. Die „*Agenda*“ wiederum ist mit dem sogenannten „*Master Process*“ in der „*Process Engine*“ verbunden und gibt die Agenda vor. Der „*Master Process*“ nutzt eine *START*, *STOP*, *PAUSE* und *RESUME*-Semantik, um die Workflow-Instanzen in der „*Process Execution*“ zu beeinflussen (siehe Abb. 6.2). Mit dem „*Master Process*“ werden, (vgl. START-Operation) basierend auf den Ergebnissen der „*Rules Engine*“, weitere Prozessinstanzen gestartet oder gestoppt. Je nach Regel, die abhängig vom Schwellwert in die „*Agenda*“ kommen, werden innerhalb des „*Master Process*“ verschiedene Workflow-Instanzen über die primitiven Operationen angesteuert. Mit diesem (einfachen) Mechanismus konnte die Abbildung von der „*Rules Engine*“ zur „*Process Engine*“ sichergestellt werden.

6.3 Prozessorientierte Sicht

Die prozessorientierte Sicht veranschaulicht die Entwicklung des mWfMS bis hin zur Anwendung. Die Prozessmodellierung wird verwendet, um anhand der definierten Wertschöpfungsprozesse in einem *Top-down*-Verfahren die Prozesse und Dienste zu definieren. Die Prozesse werden in eine ausführbare Sprache übersetzt und die Dienste werden implementiert. Ein Dienst kann wiederum ein Prozess sein. Die Dienste-Schnittstellenbeschreibung dient als vertragliche Basis für die Entwicklungsprozesse zwischen den verschiedenen Entwicklungsteams. Die Dienste werden von mehreren Teams unabhängig voneinander entwickelt. Die Workflows orchestrieren die Dienste der SOA (siehe Abb. 6.3). Anders formuliert fassen die Prozesse die Dienste einer SOA zu einer Einheit zusammen.

Generell werden auf der obersten Ebene grobe Geschäftsprozessentwürfe entworfen, die der ganzheitlichen Planung dienen. Die groben zeitlichen Phasen werden in Aktivitäten modelliert, die bei der Leistungserbringung stattfinden müssen (siehe Abb. 6.3). Je weiter von oben nach unten verfeinert wird, desto präziser werden die (technischen) Prozesse. Jeder Prozess selbst kann in einer SOA wiederum als Dienst genutzt werden, wenn er über eine entsprechende Schnittstelle angesprochen wird. Beim sogenannten *Service-*

oriented Engineering werden beispielsweise *Managed Services*, *Composite Services* und *Basic Services* als Ebenen eingeführt [PTDL08]. Je mehr Verbundebenen (zwischen oberster und unterster Ebene) eingeführt werden, desto schwieriger wird das Management der Dienste. Die Verbundebenen (zwischen den nativen Diensten und den Prozessen) helfen die Komplexität innerhalb einer SOA zu beherrschen [PTDL08]. In allen Ebenen werden Aktivitäten als Dienste modelliert. An den Systemgrenzen müssen die entsprechenden Ein- und Ausgabedaten (über eine Schnittstelle) definiert werden. Eine Aktivität definiert eine abgeschlossene Einheit. Die Aktivität kann als synchroner oder asynchroner Dienst modelliert werden. Bei der Modellierung eines asynchronen Dienstes kann die Aktivität in zwei Teile (Aktivitäten) abgebildet werden, um den Nachrichtenempfang zu einem anderen Zeitpunkt innerhalb des Prozesses zu gewährleisten.

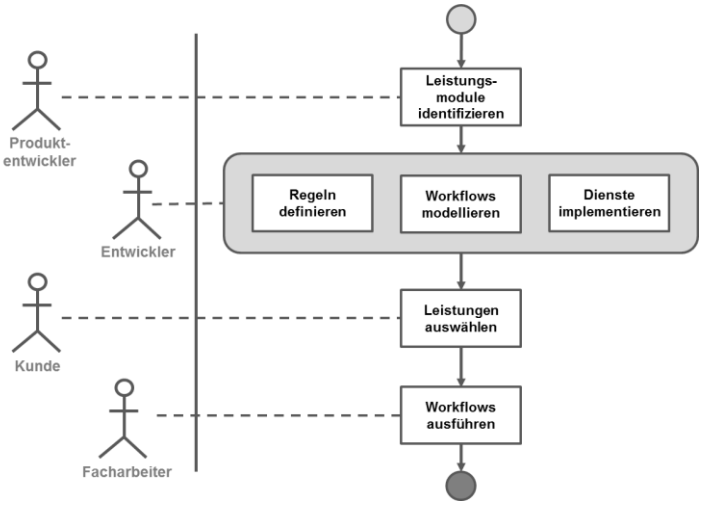


Abb. 6.3: Prozessorientierte Sicht des mWfMS (UML-Aktivitätsdiagramm)

Parallel zur Orchestrierung der Workflows erfolgt die Definition bzw. Konfiguration der verschiedenen Kontextinformationen. Die Modellierung von

z. B. Ortsbezügen kann direkt innerhalb der Modelle erfolgen (siehe Kapitel 4). Die Modelle werden in der Entwurfsphase meist schnell komplex, so dass innerhalb der jeweiligen Anwendungsfälle entschieden werden muss, welche Kontextparameter Anwendung finden. Regeln (bzw. die Definitionen) sind notwendig, um einerseits in Echtzeit auf sich ändernde Umgebungskontexte zu reagieren und andererseits um die Steuerung der verschiedenen parallellaufenden Workflows zu kontrollieren (siehe Abb. 6.3).

Nachdem die Planung (Orchestrierung) des Systems abgeschlossen ist (eine gewisse Stabilität erreicht hat), können die Dienste und deren Anbindung innerhalb des Prozesses implementiert werden. Mit der Implementierung der Dienste/Prozesse kann beispielsweise zusätzlich die Auslastung der implementierenden Entwicklungsteams gesteuert werden.

Im Prozessschritt „Leistungen auswählen“ (siehe Abb. 6.3) wird die Leistungs-konfiguration durch den Kunden (Auftraggeber) im *Configurator* der MAIS vorgenommen. Die gebuchten Leistungen führen dann jeweils zu konfigurierten Workflows (in der MEMIS), die innerhalb der Fahrzeuge ausgeführt werden.

Die prozessorientierte Sicht auf die Architektur des mobilen WfMS darf in diesem Zusammenhang nicht mit der Ausführung der Workflow-Instanzen selbst verwechselt werden. Die prozessorientierte Sicht zeigt den Ablauf des Engineerings des mWfMS bis hin zur Anwendung. Die Verarbeitung der Prozesse findet innerhalb der Workflow-Engine statt.

6.4 Entwicklungssicht

Die Verknüpfung der Leistungen mit Workflows wird über Leistungsmodule erzielt. Die Workflows werden parametrisiert, um eine größtmögliche Wiederverwendbarkeit zu garantieren. Workflows bestehen aus „Prozessen“, „Teilprozessen“ und „Mikroprozessen“ auf drei Ebenen (siehe Abb. 6.4). Teilprozesse sind ausführbare Workflows (können einzeln *deployed* werden). Die orchestrierenden „Prozesse“ lösen wiederkehrende Aufgaben, so dass Teilprozesse (sogenannte „Support“-Prozesse) eingeführt

werden. Die „Support“-Prozesse werden konfigurierbar gestaltet, so dass eine Wiederverwendbarkeit dieser Teilprozesse (als konfigurierter Dienst) garantiert wird. Hinter den Mikroprozessen (*Enabling Services*) verbergen sich einzelne Dienste (z. B. SOAP-Webservices), die in einer nativen Programmiersprache (z. B. Java) mit Hilfe von „Ressourcen- und Funktionsklassen“ implementiert werden (siehe Abb. 6.4).

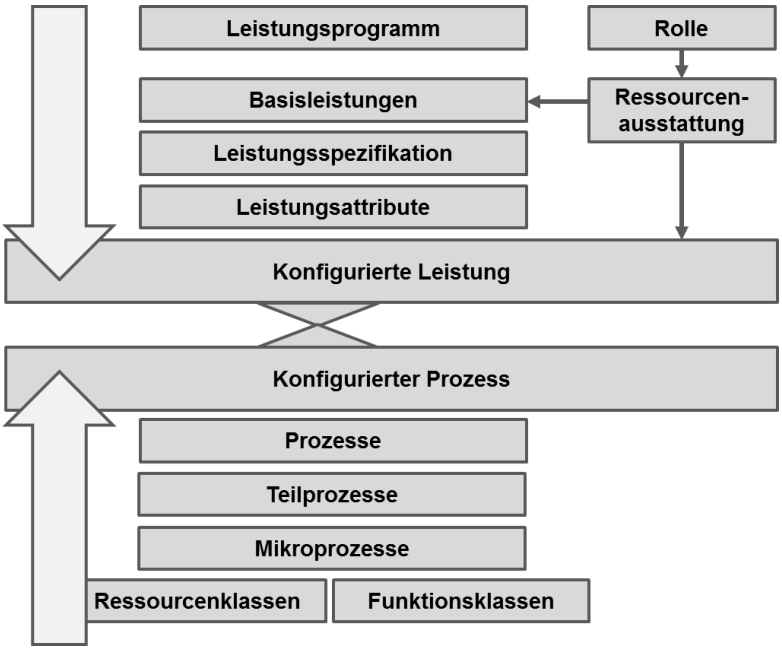


Abb. 6.4: Leistungen und Prozesse im mWfMS

Die Abbildung (siehe Kapitel 6.3) zwischen dem Leistungsprogramm und den ausführbaren Workflows ist in Abb. 6.4 dargestellt. Das „Leistungsprogramm“ setzt sich aus „Basisleistungen“ und „Zusatzleistungen“ zusammen. Die Leistungen können im Ausmaß spezifiziert werden, wodurch „Leistungsattribute“ vergeben werden. Die Attribute wirken sich wiederum auf die „Prozesse“ aus, die damit konfiguriert werden.

Die Prozesshierarchie (siehe Abb. 6.5) zeigt exemplarisch, welche Techniken zur Implementierung eingesetzt wurden (siehe Abb. 6.5, Legende). Die Workflows werden mit der Ausführungssprache BPEL umgesetzt. Die sogenannten *Enabling-Services* (vgl. Mikroprozesse) sind zustandslose Dienste, die eingebunden werden. Die Kommunikation innerhalb der verschiedenen Dienste wird mittels der Webservice-Technik realisiert, die von der BPEL unterstützt wird. Die Struktur ermöglicht es, *Enabling-Services* aus den beiden höheren Ebenen heraus aufzurufen. Die Schnittstellen zwischen den Webservices werden über die *Web Service Description Language*[2] (WSDL) spezifiziert. WSDL ist eine plattform- und protokollunabhängige Beschreibungssprache für Webservices. Der Austausch der Daten erfolgt auf Basis von XML.

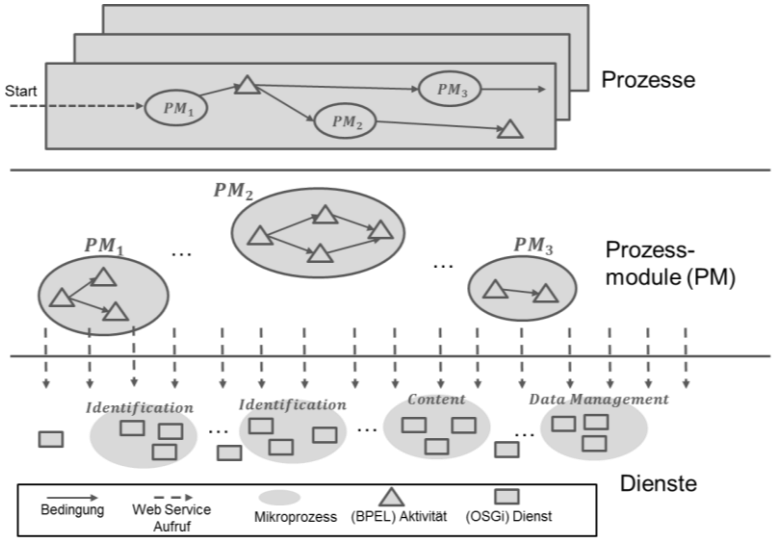


Abb. 6.5: Prozesshierarchie des mWfMS

[2] http://www.w3.org/TR/wsdl

Die Plattformunabhängigkeit des mWfMS wird über Java bzw. über die *Java Virtual Machine* gewährleistet. Das Schichtenmodell des mWfMS (siehe Abb. 6.6) zeigt die eingesetzten Software-Schichten. Auf der mobilen Hardware wird ein Betriebssystem betrieben, das die *Java Virtual Machine* unterstützt. Mit Java als angewandte plattformunabhängige Lösung ist ein Austausch des Betriebssystems (z. B.: Linux, Windows, Mac OSX) möglich. Innerhalb der *Java Virtual Machine* befindet sich die OSGi-Serviceplattform (früher: *Open Services Gateway initiative*). Die OSGi-Plattform definiert innerhalb der Java-Laufzeitumgebung ausführbare Basisdienste. OSGi stellt die Möglichkeit bereit, sogenannte Software-Bundles zur Laufzeit einzuspielen, zu löschen oder zu aktualisieren [Link05]. Es ist im Software-technischen Sinn ein *Class-Loader-Framework* [Link05], mit dem modulare Anwendungen bzw. APIs administriert werden. Die Abhängigkeiten innerhalb der APIs werden durch das „*OSGi-Framework*“ automatisch aufgelöst [Link05].

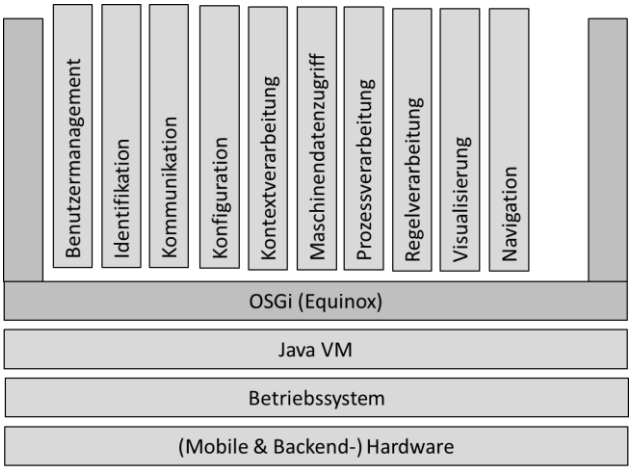


Abb. 6.6: Entwicklungssicht des mWfMS

Jedes *Bundle* spezialisiert eine bestimmte Java-Klasse von System-Aufgaben, wie z. B. GPS-Daten empfangen oder die Can-Bus-Anbindung[3] für die Integration der Sensoren. Die *Bundles*, wie beispielsweise das Benutzermanagement oder die Identifikation der Betriebsparameter (siehe Abb. 6.6), können zur Laufzeit aktiviert und deaktiviert werden. Die einzelnen *Bundles* werden je nach Leistungskonfiguration bzw. Workflow-Konfiguration geladen.

Bei der autonomen Ausführung von Teilleistungen durch die mobilen Komponenten fallen Daten an, die anschließend im MAIS für die Erzeugung der Gesamtleistung benötigt werden. Es handelt sich dabei um sensible Daten, die unter den Gesichtspunkten Vertraulichkeit, Integrität und Authentizität gesichert werden müssen. Eine Analyse des Verbundprojektes R2B ergab drei Ansatzpunkte, um Sicherheitsaspekte umzusetzen (siehe Abb. 6.7). Die Verbindung zum Konfigurator erfolgt über eine SSL-Verschlüsselung und Authentifizierung mittels Benutzername und Passwort. Bei der Übertragung der Daten von der mobilen Komponente zur MAIS sind die Sicherheitsaspekte zu gewährleisten. Die Vertraulichkeit der Daten wird während der Übertragung zur MAIS durch Verschlüsselung sichergestellt. Damit die Daten nicht unbemerkt verändert werden bzw. alle Änderungen nachvollziehbar sind (Integrität), werden die Daten digital signiert [Ecke09, S. 614, S. 569ff]. Mit digitalen Signaturen wird die Verbindlichkeit von Daten (die überprüfbare Zuordnung zu einem Sender) möglich. Die beschriebene Maschine-zu-Maschine-Kommunikation der mobilen Komponenten untereinander erfordert gesonderte Lösungen, etwa eine symmetrische Verschlüsselung und nutzer- bzw. maschinenbezogene Authentifizierung.

[3] http://www.can-cia.org/

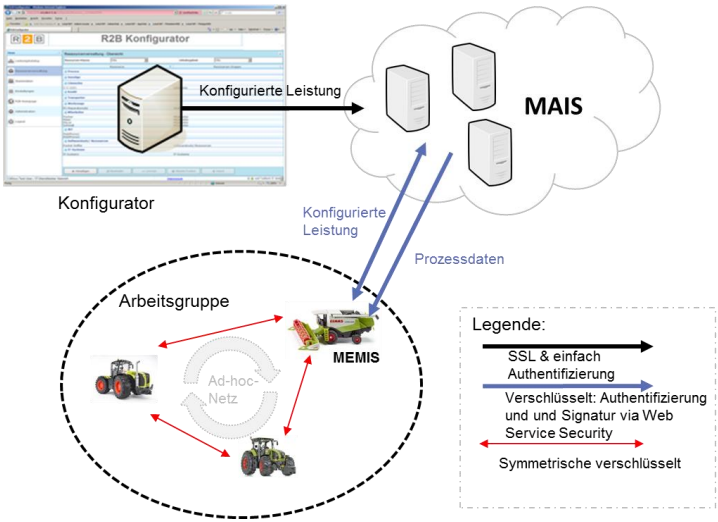


Abb. 6.7: Kommunikation und Sicherheit

Für den Datenaustausch zwischen den Landmaschinen (MEMIS) und dem Backend (MAIS) werden SOAP-Webservices genutzt. Mittels HTTPS kann der Transport von SOAP-Nachrichten verschlüsselt werden. Das ist jedoch nicht ausreichend, wenn z. B. die Daten wegen fehlender WLAN-Netzwerkverbindung zum Backend (MAIS) transportiert werden. Mit dem Einsatz einer Workflow-Modellierungssprache (BPEL) auf Basis von SOAP-Webservices konnten verschiedene Standards der „Web Services Security“ (WS-Security) [NKMH04] und weitere Spezifikationen im Kontext von Web Services genutzt werden, um Verfügbarkeitsprüfungen, Verschlüsselung, digitale Signierung und Authentifizierung zu realisieren. WS-Security nutzt zwei weitere Standards: *XML-Encryption* [ImDS02] und *XML-Signature* [BBLS08]. Die Daten werden damit verschlüsselt und signiert, so dass eine Ende-zu-Ende-Sicherheit zwischen MEMIS und MAIS gewährleistet werden konnte.

Um zu einem bestimmten Zeitpunkt ein bestimmtes Datum in einem bearbeitbaren Zustand (vgl. *java.lang.NullPointerException*, Wert derzeit unbekannt) verfügbar zu haben (siehe Kapitel 4.1.3), wurde ein mehrstufiges Validierungsverfahren innerhalb der Implementierungsphase realisiert: Eine detaillierte Planung des Workflows beinhaltet, an welcher Stelle in der Workflow-Instanz welches Datum gelesen und geschrieben wird. Die Abhängigkeiten können beispielsweise mittels eines *Gantt-Chart* abgebildet werden (vgl. [TaLi08, Fig. 3]). So kann z. B. bei einer nachträglichen Erweiterung des Workflow-Modells schnell in Erfahrung gebracht werden, an welcher Stelle z. B. ein zusätzlicher Service-Call integriert werden kann.

Um die Prozesse vorab zu validieren, sollten verschiedene Simulationen mit den verschiedenen Datenkombinationsmöglichkeiten durchgeführt werden. Die Simulationsumgebung wird gewöhnlich über die entsprechenden Entwicklungs-werkzeuge bereitgestellt. Falls eine Simulation mit gegebener Workflow-Sprache bzw. Workflow-Engine nicht unterstützt wird, kann über diverse Software-Tests eine ähnliche Überprüfung durchgeführt werden [Beck02]. Es ist z. B. möglich, jeden einzelnen Dienstaufruf bzw. den ganzen Prozess mit entsprechend hinterlegten Werten [Beck02, S.35-39] zu prüfen. Die unterschiedlichen Tests werden abhängig von der SOA-Ebene notwendig (siehe Abb. 6.8). Die Hilfskonstruktion einer Workflow-Simulation wird als *Test-getriebenes Workflow-Management* bezeichnet.

Die Tests können automatisiert über einen *Continuous-Integration-Server*[4] ausgeführt werden [Aste03]. Ein solcher Server ist ein System zur kontinuierlichen Integration von Software-Artefakten während der Entwicklung. Das Prinzip wird als kontinuierliche Integration bezeichnet [DuMG07, KKPB15].

[4] Jenkins ist ein webbasierter *Continues Integration Server*: http://jenkins-ci.org/

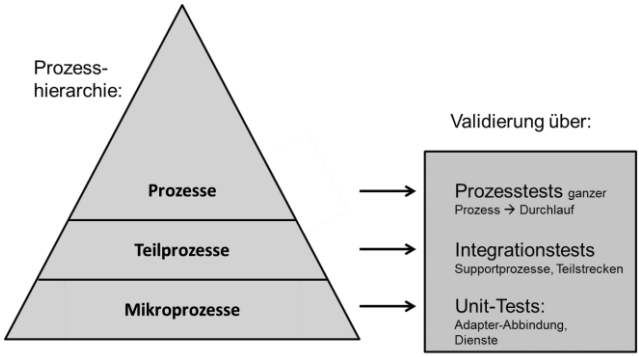


Abb. 6.8: Test-getriebenes Workflow-Management

6.5 Physikalische Sicht

Die physikalische Sicht veranschaulicht die *Elektronische Mobilität* (siehe Kapitel 2.1) des mWfMS (siehe Abb. 6.9). Die drahtlose Kommunikationsform WLAN (siehe Kapitel 2.1.4) wird angewandt, um innerhalb der Arbeitsgruppe Informationen auszutauschen. Ergänzende Informationen (z. B. Wetterdaten) werden über das Mobilfunknetz aus dem Internet bezogen. Die Vorgänge mit der MAIS sind so entworfen worden, dass innerhalb des mWfMS fast jegliche Kommunikation (vgl. Wetterdaten) über das stationäre WLAN durchgeführt wird. Die Konfiguration der Prozesse wird noch im stationären WLAN an die mobilen Arbeitsmaschinen (MEMIS) übertragen. Während der operativen Arbeit der mobilen Arbeitsmaschinen ist das MAIS über das Mobilfunknetz (2G oder 3G) mit den MEMIS verbunden (z. B. für Abbruch/Storno der Vorgänge). Es sind keine hohen Datenraten während der mobilen Arbeit zwischen den MEMIS und der MAIS realisierbar, was im mWfMS berücksichtigt wird.

Die MEMIS sind untereinander in der Arbeitsgruppe über WLAN verbunden (siehe Abb. 6.9). Es ist in der mobilen Arbeitsgruppe unwesentlich, welche mobile Arbeitsmaschine den AP betreibt. Der Nachrichtenaustausch im mWfMS erfolgt in vielen Anwendungsfällen über Broadcast-Kommunikation. In IP-basierten Netzwerken ist jedoch Broadcast-Kommunikation nicht

standardisiert vorhanden. Im mWfMS ist eine zuverlässige Multicast-Kommunikations-Lösung auf Basis von JGroups[5] entwickelt worden. Die Kommunikationslösung erlaubt die Definition einer (Arbeits-)Gruppe über einem entsprechenden Transportprotokoll (TCP) und listet intern die Teilnehmer in einer Liste auf. Sie wird verwendet, um beispielsweise eine Broadcast-Nachricht an alle Teilnehmer zu schicken. Die Kommunikationslösung gewährleistete u. a. die interne Zugriffskontrolle und eine sichere Datenübertragung. Innerhalb des mWfMS werden zusätzlich Ende-zu-Ende-Nachrichten unterstützt. Nachrichten von einem dedizierten Sender zu einem Empfänger wurden über die gleiche Transportschicht von JGroups gewährleistet (z. B. von einem Client zu MEMIS).

Die Zugehörigkeit einer mobilen Maschine (MEMIS oder normaler Client) zu einer Arbeitsgruppe wird über den Aspekt der WLAN-Konnektivität determiniert. Die mobile Arbeitsmaschine befindet sich in einer Arbeitsgruppe, wenn es sich per WLAN zu einem „bekannten" AP verbunden hat. Innerhalb der WLAN-Kommunikation ist es nicht möglich, dass ein mobiles Gerät mit mehreren APs verbunden ist. Es wurde beispielsweise ein Mechanismus implementiert - eine negative OE am Ort der MAIS – der den mobilen AP automatisch ausschaltet, so dass in der WLAN-Reichweite der MAIS alle mobilen Geräte in einer Arbeitsgruppe der MAIS vorhanden sind.

Es existieren zusätzlich Umgebungskontexte, die aus externen Informationssystemen bezogen werden, wie z. B. die Wetterinformationen. Solche Kontextinformationen werden über das Internet bezogen. In jeder MEMIS besteht die Möglichkeit, sich über das Mobilfunknetz zum Internet[6] zu verbinden bzw. IP-basierte Dienste zu nutzen (siehe Abb. 6.9). Das mWfMS stellt eine Ende-zu-Ende-Kommunikation bereit, so dass von der MAIS aus jede einzelne MEMIS über das Mobilfunknetz direkt angesprochen werden kann.

[5] http://www.jgroups.org

[6] Es wurde ursprünglich im Projekt ein Routing über eine dezidierte MEMIS innerhalb der Arbeitsgruppe angedacht. Allerdings sind die Kosten für mobile *Datenflatrates* innerhalb der Mobilfunknetze vernachlässigbar gering geworden, so dass diese Anforderung zum Ende des Projektes nicht mehr relevant war.

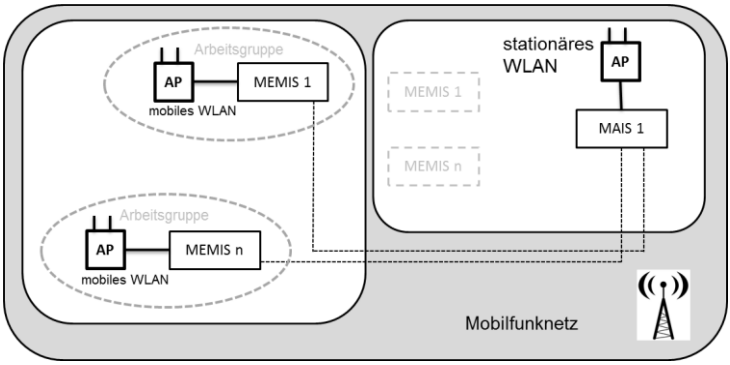


Abb. 6.9: Physikalische Sicht des mWfMS

In Abb. 6.10 ist der physikalische Aufbau eines MEMIS bzw. eines MAIS dargestellt. Zur Speicherung der Workflows bzw. der Regeln wurde jeweils eine Datenbank mit zugehörigem Schema verwendet (z. B. Schema der *ActiveVOS*[7] Workflow Engine).

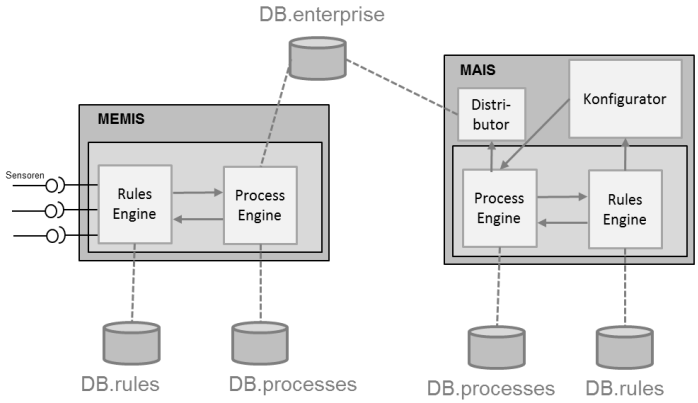


Abb. 6.10: Aufbau einer MEMIS/ MAIS

[7] http://www.activevos.com/

Im Leistungs-Konfigurator wurde eine DB mit Schema verwendet, das eigens innerhalb des Projektes entworfen wurde. Die DB des Konfigurators kann aus physikalischer Sicht direkt über das Internet angebunden werden.

Die Sensoren des MEMIS (siehe Abb. 6.10) sind je nach mobiler Maschine unterschiedlich. Die spezifischen Sensoren müssen für die jeweilige MEMIS über die Regel-Engine integriert werden. Je nach Sensor und Zustand sind Regeln definiert, die wiederum Auswirkungen auf die Workflows haben. In vorliegender Arbeit ist eine Abbildung über einen Master-Prozess gewählt worden (siehe Kapitel 6.2 bzw. Abb. 6.2). Abhängig von den Kontextinformationen aus den Sensoren (und der entsprechenden Leistungskonfiguration) werden Prozessinstanzen eines Modells beeinflusst.

Das MAIS wurde auf einem Standard-PC mit Ethernet-Anschluss (im Unternehmensnetzwerk) installiert. Das MEMIS ist auf einem Embedded-PC mit x86-Prozessorarchitektur betrieben worden. Der *Simatic Microbox Embedded-PC*[8] (die Embedded-Hardware der MEMIS) ist für den industriellen Einsatz konzipiert worden: Resistenz gegen Elektromagnetismus, Temperatur-, Vibrations- und Stoßfestigkeit. Der Embedded-PC muss lediglich vor Wassereinwirkung geschützt werden.

Der Demonstrator einer MEMIS wurde innerhalb des Verbundprojektes R2B entwickelt (siehe Abb. 6.11). Der Monitor der MEMIS bzw. das *Human Machine Interface* (HMI) wurde über den Standard-VGA-Anschluss des PCs angebunden. Der Embedded-PC besitzt ein WLAN-Interface. Eine zusätzliche Antenne verbessert die Reichweite des WLANs auf ca. 500-600 m. Die Verbindung zum Mobilfunknetz ist über einen sogenannten *UMTS-Surfstick* hergestellt worden. Der *Surfstick* enthält eine Standard-SIM-Karte zur Verbindung mit dem Mobilfunknetz. Die Sensoren der mobilen Arbeitsmaschinen werden über einen USB-RS232-Adapter an den mobilen PC angebunden. Eine GPS-Empfangs-einheit ist über USB an den PC angeschlossen. Auch beim GPS-Empfänger empfiehlt sich eine zusätzliche Antenne, um eine stabileres GPS-Signal über die Satelliten zu empfangen.

[8] http://www.pci-card.com/siemens-micro-box-pc.pdf

Mit diesem Aufbau wird deutlich, dass eine Integration der Hardware-Komponenten innerhalb einer landwirtschaftlichen Arbeitsmaschine, eines Baustellenfahrzeugs oder innerhalb eines gewöhnlichen Fahrzeugs möglich ist. Die vollständige elektronische Hardware der MEMIS ist (siehe Abb. 6.11) in der Schaufel des Demonstrators eingepasst (20x30x5cm).

Abb. 6.11: Demonstrator im Projekt R2B

Die Lauffähigkeit des präsentierten mWfMS wird in Kapitel 7.1 mit dem Anwendungsfall der Hinderniswarnung vorgestellt.

6.6 Anforderungsüberprüfung

Im vorliegenden Entwurf einer Software-Architektur für ein mobiles WfMS wurden ausgehend von den Anwendungsfällen verschiedene Sichten der Architektur entwickelt.

Zur Überprüfung der Anforderung **R1** (siehe Tab. 6.1), werden die einzelnen Bestandteile des SOA-Paradigmas überprüft:

„Lose Kopplung“ wurde über die Verwendung der Webservice-Kommunikation sichergestellt. Die „Abstraktion“ innerhalb der SOA wurde mit Hilfe der Workflow-Modellierung erzielt. Es sind standardisierte Sprachen (BPMN 1.2/2.0, BPEL 2.0) und Dienste (SOAP-Webservices) verwendet worden. Mit der Webservice-Orchestrierungs-Sprache[9] (BPEL 2.0) wurden die ausführbaren Modelle erstellt. Mit der standardisierten Programmiersprache Java wurde die Implementierung der Dienste erstellt. Die „Wiederverwendbarkeit“ ist durch parametrisierte Aktivitäten (BPMN 1.2/2.0) bzw. wiederverwendbare Dienste (BPEL 2.0) hergestellt worden. Die Dienste konnten damit innerhalb der Prozesse wiederverwendet werden. So wurden z. B. viele kleine Supportprozesse implementiert, die an verschiedenen Stellen im Prozess eingebunden wurden. Die Software-Artefakte konnten direkt innerhalb der Java-Implementierung wiederverwendet werden. Die „Skalierbarkeit“ der Anwendung ist innerhalb der mobilen Arbeitsgruppen (ca. 10 Fahrzeuge) sichergestellt worden. Die Schnittstellen sind jeweils über die WSDL eines SOAP-Webservices eindeutig spezifiziert. Die „große Kohäsion“ der Architektur wird auf Basis der Programmiersprache Java gewährleistet. In Java bzw. in der realisierten Anwendung ist jede Programmeinheit (Methode, Klasse oder Modul) verantwortlich für genau eine wohldefinierte Aufgabe.

Die Anforderung **R2** verlangt eine ganzheitliche Unterstützung des Workflow-Lebenszyklus (siehe Kapitel 3.1). Die Dienste werden über einen orchestrierenden Prozess modelliert. Eine Sprachtransformation (in BPEL 2.0) ist notwendig, um die Modelle in eine ausführbare Sprache zu übersetzen. Die entworfenen Workflows wurden innerhalb der ActiveBPEL[10]-Workflow Engine ausgeführt. ActiveBPEL beinhaltet als Plattform ein Monitoring der laufenden Workflow-Instanzen auf Basis von KPIs.

[9] Ursprünglich sollte innerhalb des Verbundprojektes R2B eine automatische Transformation von BPMN 1.2 in BPEL 2.0 vorgenommen werden. Jedoch ergeben sich einige Transformations-Probleme von einem flussorientierten Prozessmodell in ein blockorientiertes [Whit05, ODHA06, WDGW08, Duma09].

[10] ActiveBPEL wurde mittlerweile in „ActiveVOS Community Edition“ umbenannt: http://www.activevos.com/learn/open-source

Die Leistungskonfiguration **R3** wird innerhalb einer Webapplikation (Konfigurator) gewährleistet. Der Konfigurator bildet die Leistungsmodule auf Workflows bzw. Workflow-Kombinationen ab. Je nach Anwendungsdomäne existieren unterschiedliche Leistungen, die gegenüber dem Kunden erbracht werden. Der Konfigurator ist in verschiedene Leistungstypen (z. B. Basisleistungen, Zusatzleistungen) untergliedert worden, um unterschiedliche Leistungen (abhängig von der Anwendungsdomäne) zu unterstützen. Hinter den Leistungstypen verbergen sich jedoch standardisierte Workflows, die eine Generalisierung des mWfMS gewährleisten.

Die Plattformunabhängigkeit **R4** wird durch die Verwendung der Programmiersprache *Java* bzw. der *Java Virtual Machine* gewährleistet.

Die Endbenutzerschnittstelle **R5** ist mit einem kleinen 3,5“-Monitor (siehe Abb. 6.11 und Abb. 6.1) bereitgestellt worden. Die graphische Visualisierung der gebuchten Leistungen in Kombination mit den anstehenden Arbeitsprozessen werden innerhalb der Fahrerkanzel-Recheneinheit (**R6**) der MEMIS aufbereitet. Die Endbenutzerschnittstelle wird je nach Anwendungsfall bzw. ausgeführtem Workflow individuell aufbereitet. Der Fahrer der mobilen Arbeitsmaschine soll nicht durch unnötige Hinweise bzw. Visualisierungen von seiner operativen Arbeit abgelenkt[11] werden.

Die Eigenortung **R7** ist mit der GPS-Technik realisiert worden. WLAN-basierte Verfahren wurden genutzt, um bestimmte Aspekte (z. B. Identifizierung des Fahrers der mobilen Arbeitsmaschine) des mWfMS zu unterstützen [KANO10, TsCh08, TRPC09, BoND09].

Jeder Workflow befindet sich innerhalb eines Kontextes (siehe Kapitel 2.2). Mit der Regel-Engine (siehe Abb. 6.2) wird ein ganzheitliches Steuersystem für die Workflows auf Basis der Kontexte **R8** (z. B. gebuchte Leistungen, Sensorinformationen) bereitgestellt. Zusätzlich ist mit der Regel-Engine eine Vorverarbeitung der vielen Sensorinformationen gewährleistet worden.

[11] Die graphische Einheit muss – je nach mobiler Arbeitsmaschine – vom TÜV abgenommen werden. Da es sich innerhalb des Verbundprojekts aber um ein Forschungsprojekt handelte, wurde dies letztendlich vorbereitet, jedoch nicht evaluiert.

Kontexte wie Ortsbezüge sind bereits zum Entwurf der Workflows modelliert (siehe Kapitel 4 bzw. Kapitel 7.1.2) und werden automatisiert ausgeführt.

Zur Unterstützung von Sicherheitsaspekten innerhalb des mWfMS **R9** wurde eine Ende-zu-Ende-Verschlüsselung von MAIS zu MEMIS über die Webservice-Technik realisiert.

Die Multicast-Kommunikation **R10** ist mit Hilfe des JGroups-Frameworks implementiert worden (siehe Kapitel 6.5).

Die Anforderungen **R1-R10** (siehe Kapitel 6.1) werden innerhalb des vorliegenden mWfMS erfüllt. Verschiedene Anwendungsfälle zeigen, wie durch mobile Kontextinformationen das mWfMS angereichert (bzw. der mobile Akteur entlastet) werden kann (siehe Kapitel 7.1.5, Hinderniswarnung), um eine Kontext-basierte Workflow-Steuerung zu ermöglichen. Die Workflows werden über die mobilen Kontextinformationen präziser gesteuert, so dass die mobilen Anwendungsfälle mit Ortsbezügen besser kontrolliert werden können.

6.7 Alternative Architekturen für mWfMS

Bei größeren Software-Projekten kann der Bau der Software mit dem eines Gebäudes verglichen werden. Architekten entwerfen die Grundstruktur der Software (das Gebäude). Der Bauplan ist generisch und kann in vielen Anwendungsfällen verwendet werden. Viele einzelne *Product Owner* bzw. Spezifikateure entwerfen Detailpläne einer bzw. mehrerer Software-Applikationen. Sie bilden die funktionalen Anforderungen der Kunden ab. Die Programmierer realisieren die Software (das Gebäude). Dieses Vorgehen ist beim Entwurf von kleinen Systemen (der Entwicklung von mobilen Apps, vgl. Kapitel 2.1.7) nicht empfehlenswert, da langwierige Vorkonzeptionen Aufwandstreiber sind und sich dies bei kleinen Software-Projekten nach Erfahrung des Autors der vorliegenden Arbeit nicht auszahlt.

Um die in vorliegender Arbeit erstellte „große“ Software-Architektur mit anderen Software-Architekturen zu vergleichen, müssen die spezifizierten Anforderungen bzw. die Realisierung derer mit anderen Realisierungen verglichen werden. Das System wurde innerhalb eines Top-Down-Entwurfs realisiert, da nicht alle Dienstleistungen bzw. Produkte der Anwender vorab bekannt waren. In vorliegender Arbeit wurde folglich von Anwendungsdomänen gesprochen, die wiederrum Dienste bzw. Dienstleistungen unterstützten. Bestimmte Architekturteile bzw -paradigmen des mWfMS lassen sich gut vergleichen: Innerhalb eines mobilen WfMS werden z. B. hohe Ansprüche an die Kommunikation zwischen zwei Diensten (Client und Server) gefordert. Daher wird die Dienste-Orchestrierung der vorliegenden SOA mit einer SOAP- und REST-basierten Architektur betrachtet.

In vorliegender Arbeit wurden die ausführbaren Prozesse in BPEL orchestriert, um diese Sprache in einer sogenannten BPEL-Engine (Komponente des Applikationsservers) auszuführen. BPEL ist eine Workflow-Sprache, die SOAP als Kommunikationsprotokoll verwendet. SOAP ist ein XML-Kommunikationsprotokoll. SOAP ist durch viele Erweiterungen mächtig und entsprechend komplex in der Anwendung geworden. SOAP-Webservices werden als Request in Form einer XML-Datenstruktur an einen Server geschickt. Der Server enthält einen Router, der die XML-Datenstruktur interpretiert (Erkennung der Operationen auf dem Server). Die Parameter der Operation werden aus den XML-Daten ausgelesen und beim Aufruf der Operation mitgegeben. Die Schnittstelle wird in der sogenannten WSDL beschrieben [Josu08].

REST (REpresentational State Transfer) stellt eine andere Herangehensweise zur Erstellung von Webservices (im Vergleich zu SOAP, WSDL) dar. Bei REST ist der Schwerpunkt die Interaktion von zustandsbehafteten Ressourcen. REST ist kein Standard, sondern ein Architekturstil (vgl. Web), daher eignet er sich zum Vergleich mit der erstellten SOAP-Architektur. Für die Umsetzung von REST-Webservices werden Standards des Internets (HTTP) verwendet.

Eigenschaften von REST in einer Client-Server-Betrachtung:

- **Ressourcen/ Repräsentationen:**
  - Jede Ressource besitzt eine URI (Uniform Ressource Identifier) bzw. kann eindeutig darüber identifiziert werden.
  - Eine Ressource kann jegliche Art von Information (Textdokument, Bilder, Audiodatei, XML-Datei) sein.
  - Die Repräsentation einer Ressource führt den Client in einen Zustand über. Die Operationen bzw. Dienste werden als Links abgebildet. Wenn der Client den Link aufruft (per http-GET), liefert die weitere Ressource einen Zustand über die Repräsentation zurück. Repräsentationen verweisen auf Ressourcen, die Repräsentationen liefern und auf andere Ressourcen verweisen usw.
- **Zustandsloser Server/ zustandsbehafteter Client**: Bei der REST-Kommunikation sind die Nachrichten in sich geschlossen. Der Server speichert keinen Status des Clients. Daher müssen innerhalb einer Anfrage jegliche Kontextinformationen des Prozesses enthalten sein. Der Server braucht kein Wissen über vorherige oder spätere Nachrichten zu behalten. Der statuslose Zustand des Dienstes ermöglicht viele parallele Interaktionen. Zusätzlich ist das Beheben von Fehlerzuständen einfach möglich.
- **Caching**: Ein Server kann seine Antwort als cachefähig oder nichtcachefähig charakterisieren. Caching wird benutzt, um die Netzwerkkommunikation gering zu halten.
- **Interaktion/Zugriffsprotokoll:**
  - Mit HTTP als Kommunikationsprotokoll können Daten bzw. Nachrichten ausgetauscht werden. Es stehen die Methoden GET, POST, PUT und DELETE zur Verfügung.

- HTTP sieht eine Abruf-Kommunikation vor. Clients beziehen vom Server Informationen. Der aktive Part ist dem Client vorbehalten, der vom Server Repräsentationen und Ressourcen anfordert.

Eine genauere Betrachtung des Schnittstellenhandlings zeigt die Vor- und Nachteile von SOAP und REST:

- Die **WSDL** beinhaltet eine typisierte Schnittstelle mit SOAP-Operationen und zugehörigen Parametern. Client und Server werden über die WSDL aneinandergekoppelt. Die Clients sind immer vollständig von der Schnittstelle des Dienstes abhängig. Bei einer Änderung (auch nur einzelner Parameter) muss eine neue Version einer WSDL im Server hinterlegt werden. Die Clients müssen folglich synchron umstellen, wenn sie nicht mit einer älteren (abwärtskompatiblen) Version des Dienstes arbeiten können.

- Die **REST-Schnittstelle** ist im Vergleich zu SOAP dynamisch. Sie kann erweitert und angepasst werden (nicht-typisierte Schnittstelle). Die Clients verbinden sich lediglich mit den Teilen, die sie benötigen. Der Client muss folglich nicht angepasst werden, wenn sich Änderungen in den Rückgabewerten o. ä. ergeben.

In folgender Tabelle werden beide Architekturen verglichen:

Tab. 6.2: Vergleich von SOAP und REST

	SOAP	**REST**
Schnittstellen-beschreibung	WSDL	keine
Adressen	URI	URI
Schnittstelle	Anwendungsspezifisch	GET, POST, PUT, DELETE
Zustand/ Status	Server/ Client	Client
Transport	HTTP, SMTP, JMS, ...	HTTP
Zugriffskontrollsteuerung	über Applikation in Server gesteuert	Firewall
Transaktionen	Ja	über Applikation

REST eignet sich, wenn ein Server viele Clients besitz bzw. die Interoperabilität der Anwendung notwendig und die Leistung entscheidend ist. Webserver werden optimiert, um einen Zugriff von vielen Geräten/Usern gleichzeitig zu ermöglichen. Sie zeichnen sich durch eine hohe Skalierbarkeit, hohe Zuverlässigkeit und Transaktionalität aus. In den Anwendungsfällen der vorliegenden Arbeit wird das jedoch nicht gefordert. In den in Kapitel 6.1 vorgestellten Szenarien bzw. Anforderungen werden andere Kriterien gefordert. Zum Beispiel ein zustandsbehafteter Prozess (auf Server-Seite) zur Orchestrierung von Diensten mit gegebener Transaktionalität. Transaktionalität muss innerhalb der Applikation von REST sichergestellt werden, indem die Transaktion selbst als Ressource behandelt wird. Die Transaktion wird mit POST erstellt, mit PUT übergeben (gestartet) und mit PUT und DELETE zurückgerollt. Aufgerufen wird die Transaktion mittels GET.

Die Mitteilung von Zuständen vom Client an den Server (bzw. alle anderen Clients) würden jedoch zu einem größeren Datenaufkommen führen. Jeglicher Prozesskontext muss mit übertragen werden, damit die Clients jegliche

Entscheidungen selbst treffen können. Dadurch wird eine zentrale Steuerung — so wie im betrieblichen Kontext gefordert — verloren gehen. Bei REST entscheidet der Client und kann damit z. B. bei einem Fehler den kompletten Prozess stoppen. Eine zentrale Prozesssteuerung kann auf Basis von REST entwickelt werden, indem ein zusätzlicher Client als Applikation die zentrale Prozesssteuerung erhält. Die Clients müssten jedoch stets beim Server http-PULL-Requests absetzen, um etwaige Änderungen mitzubekommen. Auch diese Lösung würde das Kommunikationsaufkommen vergrößern.

Einige REST-Ansätze können innerhalb von SOAP-Architekturen übertragen werden. Eine SOAP-Schnittstelle kann z. B. mittels einer typisierten Datenstruktur, die nicht-typisierte Daten (vgl. *Java Hashmap*, Objektmodell) enthält, entworfen werden, so dass nicht immer die Schnittstelle geändert werden muss, wenn sich nur kleine Teile des Dateninhalts ändern. Beim Client muss darauf geachtet werden, dass der Datenaustausch weiterhin funktioniert. Die typisierten XML-Daten können in ein Objektorientiertes Modell (z. B. ein DOM-Objekt) überführt werden, um den Datenaustausch zwischen Client und Server gering zu halten. Je nach Datenrepräsentation (gepackt) kann Datenaufkommen eingespart werden. Seit der SOAP Version 1.2 wird eine Binärkompression im Protokoll angeboten [MiLa07]. Letztendlich wird SOAP verwendet, da es sich um eine unternehmenskritische Anwendung handelt, die eine hohe Sicherheit und Zuverlässigkeit fordert, eine flexible Transportschicht (JMS, HTTP, ...) besitzt und Transaktionalität (über das Protokoll) sicherstellt.

Die Erfüllung der Erweiterbarkeit (Skalierbarkeit) ist mit direkten Szenarien evaluiert worden. Solche Szenarien sind innerhalb des Verbundprojektes R2B zahlreich durchgeführt worden [KBSN07]. Die Unterstützung eines indirekten Szenarios, das mit einer Architekturerweiterung realisierbar ist, wird z. B. in Kapitel 7.1.4 und 7.1.5 vorgestellt.

7 Tragfähigkeitsnachweis

Das mWfMS aus Kapitel 6 wird mit Anwendungsfällen aus dem landwirtschaftlichen Bereich ausgeführt. In Kapitel 7.1 werden Ergänzungen der Prozesse des „*hochentwickelte mWfMS*“ (vgl. Kapitel 3.3) vorgestellt. In weiteren Experimenten (siehe Kapitel 7.2) werden „*einfache mWfMS*“ (vgl. Kapitel 3.3) betrachtet. In den Anwendungsfällen wird auf die Automatisierung von (mobilen) Aktivitäten mit Ortsbezügen fokussiert. Mobile Aktivitäten werden mit einer Rolle bzw. einem Nutzer assoziiert. Ziel ist es, eine geeignete Abbildungsform für solche Benutzeraufgaben zu identifizieren. Der Tragfähigkeitsnachweis bzw. die Experimente in Kapitel 7.1 und 7.2 beziehen sich auf die Modellierungs- und Ausführungsphase eines mobilen Workflows (vgl. Kapitel 3.1).

7.1 Das mWfMS in einer Anwendungsdomäne

Die Modellierung und Ausführung der Workflows mit Ortsbezügen werden anhand eines landwirtschaftlichen Anwendungsfalls erläutert. Die landwirtschaftlichen Arbeitsprozesse werden in Kapitel 7.1.1, zusammen mit dem *Precision Farming*, eingeführt. In Kapitel 7.1.2 wird die Modellierung von Ortsbezügen (während der Entwurfsphase der Prozesse) vorgestellt. Die Integration von Kontextinformationen in das mWfMS wird in Kapitel 7.1.3 erläutert. Die Hinderniswarnung (siehe Kapitel 7.1.4), als zusätzliche softwaretechnische Implementierung, wird auf Basis des mWfMS ausgeführt (siehe Kapitel 7.1.5).

7.1.1 Precision Farming mit dem mWfMS

In industriellen Anlagen, bei denen die Bearbeitungsmaschinen i. d. R. an einem festen Standort angebracht sind, werden die zu bearbeitenden Güter

bewegt. Bei landwirtschaftlichen Wertschöpfungsprozessen sind die Maschinen, Betriebsmittel (Saat-, Dünge- und Pflanzenschutzmittel) und die Prozesse mobil. In der Landwirtschaft wirken Umweltfaktoren (viele davon nicht-deterministisch) auf die Arbeitsprozesse ein und ergeben (für einzelne Arbeitsprozessinstanzen) schwankende Rahmenbedingungen. Räumliche und zeitliche Effekte sind die wesentlichen Betrachtungsebenen, die in der Landwirtschaft kombiniert betrachtet werden müssen.

In vielen Ländern werden heutzutage leistungsfähige mobile Arbeitsmaschinen wie z. B. Mähdrescher eingesetzt. Die Arbeitsmaschinen vereinfachen die landwirtschaftlichen Arbeitsprozesse, indem z. B. mit der automatischen Anpassung der Schnittlänge des Erntegutes nur genau die Teile der Pflanze abgeschnitten werden, die benötigt werden. Die maschinentechnische Unterstützung innerhalb der Landwirtschaft wird als *precision agriculture* bzw. *precision farming* bezeichnet [MWAB05, Staf02]. Durch das *precision farming* werden die landwirtschaftlichen Arbeitsprozesse immer effektiver und effizienter. Der Ernte-Ertrag wird gesteigert, während die Kosten bei der Herstellung gesenkt werden. Die mobilen Arbeitsmaschinen werden weiterentwickelt und mit technischen Hilfsmitteln ausgestattet, um den Fahrzeugführer von komplexen Entscheidungen zu entlasten.

Für die landwirtschaftlichen Arbeitsprozesse wird die mobile Recheneinheit (MEMIS) (siehe Abb. 6.11 bzw. Kapitel 6) verwendet, um das mWfMS (siehe Kapitel 6) umzusetzen. Das MEMIS soll dem Fahrzeugführer u. a. Vorschläge bzw. aktuelle Prognosen über Erntemengen auf Basis der aktuell vorliegenden Kontextinformationen liefern. Zusätzlich soll eine Arbeitsgruppe mit mehreren landwirtschaftlichen mobilen Maschinen (abhängig vom Ernteprozess) gesteuert werden. Die MEMIS gewährleisten in der Fahrzeuggruppe die Informationsverarbeitung der tatsächlichen Kontextinformationen und der gebuchten Leistungen (Ernteprozesse). Die Ernteprozesse werden vorab in Form von Geschäftsprozessen mit der BPMN modelliert (siehe Kapitel 6). Die Modelle werden daraufhin in eine ausführbare Sprache (BPEL) transformiert, damit eine direkte Ausführung innerhalb einer Workflow-Engine (beispielsweise während eines Ernteprozesses) möglich wird.

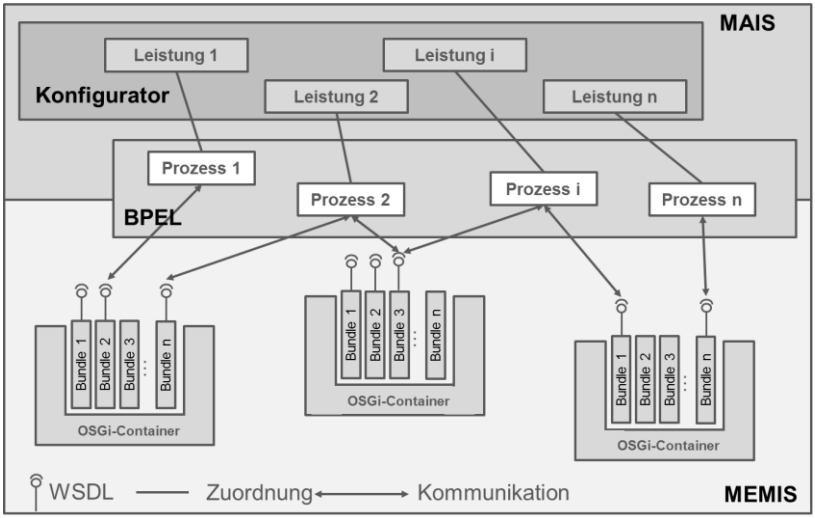


Abb. 7.1: *Snapshot* von Konfigurator und BPEL-Prozess-Engine

Die Kommunikation in der Arbeitsgruppe bzw. in den MEMIS erfolgt über Inter-Prozesskommunikation (SOAP-Webservices vgl. BPEL). In den mobilen Arbeitsmaschinen werden unterschiedliche Prozesse (in sogenannten Bundles) – je nach Aufgabe in der Arbeitsgruppe – vorgehalten (siehe Abb. 7.1). Die gebuchten Leistungen korrelieren mit den Prozessen eindeutig (siehe Abb. 7.1).

Das mWfMS wird zur Anwendung des *Precision Agriculture* verwendet. Die Position der Arbeitsmaschinen wird über GPS ermittelt. Auf Basis der Ortsinformationen können dem Fahrzeugführer einer Erntemaschine z. B. Routenvorschläge gegeben werden. Innerhalb der landwirtschaftlichen Domäne müssen zahlreiche Workflows bzw. Supportprozesse unterstützt werden, um z. B. „Ernten“ als Arbeitsprozess einer Fahrzeuggruppe sicherzustellen.

7.1.2 Modellierung der Workflows mit Ortsbezügen

Die praktische Erprobung der graphischen Notation von Ortsbezügen ist mit der Modellierungssprache BPMN in der Version 2.0 innerhalb des Verbundprojektes R2B durchgeführt worden. Es sind landwirtschaftliche, baubetriebliche und mobile Wartungs-Geschäftsprozesse modelliert worden (vgl. Kapitel 6.1).

Die Modelle der Fachkräfte (siehe Abb. 6.3, „Produktentwickler“) dienten der Beschreibung von betriebswirtschaftlichen Abläufen bei der Leistungserbringung. Mit der Unterstützung durch einen erfahrenen Modellierer konnten Modelle für die Nutzerkonzepte als grobe Architekturbeschreibung in Form von Systemintegrationsprozessen erstellt werden. Die Modellierung der Ortsbezüge halfen, ortsabhängige auslösende Aktivitäten (siehe Kapitel 2.7, vgl. Events) schon mit dem Entwurf innerhalb eines Prozesses zu definieren. Die anderen Kontextinformationen (z. B. Erntestand) wurden innerhalb von Regeln (in der Rules-Engine) definiert und schließlich als Datum in den Prozesskontext integriert. Der Automatismus über das Berechtigungskonzept der OE zeigt gegenüber einer Regel-basierten Definition eine Abbildung vom Modell hin zum Code (ohne weiteren Implementierungsaufwand).

Eine Nutzerevaluation nach der Methode von *AttrakDiff*[1] wurde durchgeführt, um die Attraktivität der entwickelten prototypischen Implementierung [Voge10] zu überprüfen (siehe Anhang A).

Folgende Verbesserungspotentiale wurden für die Werkzeugunterstützung zur Modellierung von Ortsbezügen erkannt:

- Die Bedienbarkeit kann verbessert werden:
    - mehr graphische Symbole und Icons
    - eine Karte zur Darstellung der OE-Bereiche

[1] http://attrakdiff.de

- Die hedonistischen Aspekte[2] können verbessert werden:
    - durch eine semantische Überprüfung zulässiger Darstellungen im Modell

Die Werkzeugunterstützung wird von *AttrakDiff* als „mittelmäßig attraktiv" bewertet. Die Notation von Ortsbezügen innerhalb der Geschäftsprozessmodelle ist, da bisher keine andere Modellierungsmöglichkeit besteht, eine Form der Kontextinformationsdarstellung während der Entwurfsphase und ein durchgängiges Konzept für orts- und kontextabhängige Prozesssteuerung. Mit den in vorliegender Arbeit eingeführten Ortsbezügen ist es innerhalb von Geschäftsprozessmodellen möglich, Raum- und Zeit-Ereignisse (Aktivitäten) gleichzeitig zu spezifizieren.

Die Modellierung der OE zeigt, dass die Modellierer des mWfMS mit der vorliegenden Notation von Ortsbezügen eine wichtige Erweiterung der Geschäftsprozessmodelle erhielten. Eine adäquate Werkzeugunterstützung mit einer semantischen Überprüfung des Modells ist notwendig (vgl. [KRNB94, KRNB94]). „*Correctness by Construction*" wird das Modellierungsparadigma bezeichnet, das eine valide Modellierung innerhalb eines Werkzeugs erlaubt [DRRG09]. Das Modellierungswerkzeug verhindert dadurch einen unzulässigen Einsatz der Sprachelemente.

Das Modellierungswerkzeug wird innerhalb eines sogenannten *Mockups* (GUI-Skizze) dargestellt (siehe Abb. 7.2). Ein Modellierer kann die Modelle aus dem Navigationskasten (siehe Abb. 7.2, links) auswählen und innerhalb des Hauptfensters in der Mitte bearbeiten. Die Sprachelemente einer Geschäftsprozessmodellierungssprache werden per *Drag&Drop* ausgewählt und in das Hauptfenster verschoben. Bei der Modellierung von Ortsbezügen wird dadurch eine einfache Definition der OE ermöglicht. Sobald z. B. Aktivität 1[3] (siehe Abb. 7.2) ausgewählt wird, erscheint ein Hinweis auf die OE

[2] Z. B. Spaßfaktor

[3] Innerhalb von Aktivität 1 ist ein „+" mit Kasten sichtbar. Es handelt sich dabei in BPMN 1.2 um eine Aktivität, die einen Subprozess beinhaltet. OE wirken in vorliegender Arbeit zusätzlich auf Subprozesse (siehe Kapitel 4.3.3).

(kleiner Kasten rechts an Aktivität 1). Die geographische Darstellung innerhalb einer Karte wird zusätzlich eingeblendet, damit der Nutzer die Auswirkungen der OE graphisch mitverfolgen kann (siehe Abb. 7.2, untere Hälfte). Der grau markierte Bereich zeigt den Bereich, auf den die OE wirken soll.

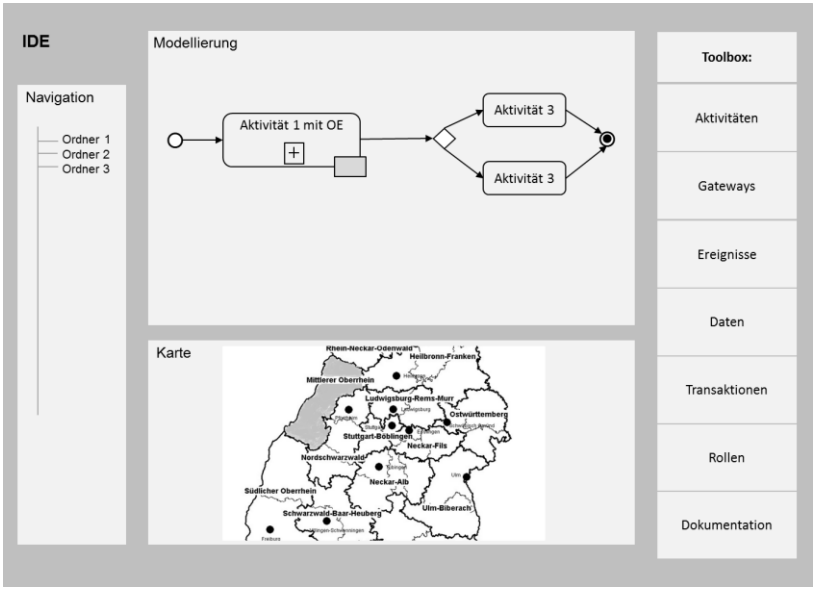


Abb. 7.2: Entwurf einer Modellierungs-IDE für OE

Der Entwurf in Abb. 7.2 zielt nicht darauf ab, Ortsbezüge (wie in Kapitel 4 vorgestellt) zu integrieren, sondern sie mit anderen Anwendungen innerhalb einer IDE bereitzustellen [DaRR11].

7.1.3 Integration von Kontextinformationen in das mWfMS

Mit der Modellierung der Workflows sind die Arbeitsprozesse in einem Top-down-Verfahren identifiziert worden (siehe Abb. 7.3). Anhand der modellierten Aktivitäten bzw. Zustände wurden die Dienstschnittstellen (SOAP-WS) identifiziert. Die Modelle wurden mit den beteiligten Fachkräften (Maschinenbauingenieure) erstellt, so dass in der frühen Entwicklungsphase Probleme und Anforderungen diskutiert und detaillierter spezifiziert werden konnten [Reck10].

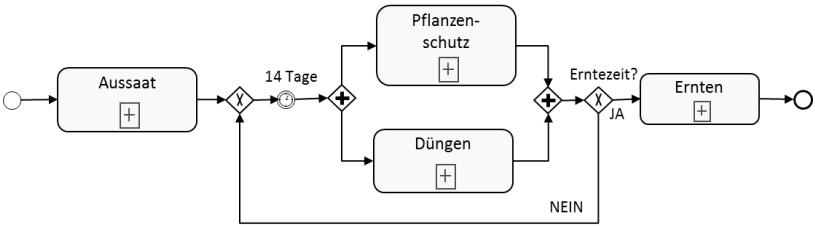


Abb. 7.3: Landwirtschaftliche Arbeitsprozesse in BPMN

Die Modelle für das mWfMS wurden durch Interviews der Fachkräfte von einem erfahrenen Modellierer (mit technischem Hintergrundwissen) erstellt. Die Modelle sind innerhalb der landwirtschaftlichen Domäne in mehreren Iterationen durch Experten bestätigt bzw. ergänzt worden. Bei der fachlichen Modellierung wurden Ortsbezüge mit OE durch den Experten definiert. Andere Kontextbezüge sind primär bei der Modellierung (z. B. nur bei trockenem Wetter ernten) innerhalb von Kommentaren direkt im Modell hinterlegt worden.

Die definierten OE für die Hinderniswarnung wurden zu Beginn des Projektes jeweils vom Entwickler mit BPEL definiert. Dieser Schritt konnte mit der Einführung einer schematischen Transformation des BPMN-Modells

(siehe Kapitel 4) in BPEL automatisiert werden. Direkt nach der Modellierung der entsprechenden BPMN-Modelle konnte die Abbildung in BPEL (mit einer zugehörigen Adapterimplementierung der OE) generiert werden.

Die Umgebungskontexte landwirtschaftlicher Arbeitsprozesse werden über Sensoren erfasst und innerhalb des mWfMS über Regeln definiert. Die Maximierung des Ernteertrags ist jeweils immer die Prämisse, nach der die Prozesse gesteuert werden. Die landwirtschaftlichen Prozesse (siehe Abb. 7.3) besitzen daher (neben den Ortsbezügen) Kontext-basierte Einschränkungen, die über die Regel-Engine gesteuert werden. Jeder (landwirtschaftliche) Workflow beginnt mit der Überprüfung der Voraussetzungen (z. B. Wetter- und Zeitabfragen). Der Wetterdienst wurde beispielsweise per SOAP-Webservice eines Dienstanbieters über das Internet eingebunden. Die Zeitabfrage erfolgt systemintern über einen entsprechenden Dienst auf Betriebssystemebene. Andere Voraussetzungen bzw. Regeln wurden definiert. Beispielsweise konnte nur mit dem Aussaatprozess begonnen werden, wenn Frühling ist und der letzte Frost mindestens zwei Wochen zurücklag. Bei der Aussaat wurden parallel z. B. über Sensoren die Bodengegebenheiten überprüft. Die Prozesse können um Kontexte erweitert werden, indem der neue Kontext/Sensor in die Rules-Engine als Kontextinformation integriert wird. Eine Regel (Algorithmus) mit Schwellwert legt fest, ab wann ein Ereignis auftreten soll (siehe Abb. 6.10). Diese Ereignisse werden innerhalb der Prozessmodelle integriert, so dass eine bestimmte Aktion bzw. ein Prozess ausgelöst werden.

7.1.4 Anforderungen an eine landwirtschaftliche Hinderniswarnung

Im Anwendungsbeispiel soll ein Supportprozess (die Hinderniswarnung) für das MEMIS-System (das vorliegende mWfMS) implementiert werden. Die Hinderniswarnung stellt einen klassischen Supportprozess dar, der innerhalb von vielen landwirtschaftlichen Arbeitsprozessen als (zusätzliche) Leistung gebucht werden kann. Die Hinderniswarnung ist ein wichtiger Supportprozess (vgl. Basisleistung) in der landwirtschaftlichen Arbeitsdomäne. Mit der

Leistung bzw. dem Supportprozess werden Beschädigungen an mobilen Arbeitsmaschinen vermieden. Im Anwendungsbeispiel wird ein Hindernis als negative OE (siehe Kapitel 4) für die landwirtschaftliche Maschine auf dem Feld definiert. Verschiedene Arten von negativen OE müssen unterstützt werden, damit jedes Hindernis adäquat beschrieben wird. Die Hindernisse sollen für jeden „Schlag“ (Erntebereich) gespeichert werden, damit für darauffolgende Ernteprozesse die Informationen bereitstehen. Der Fahrzeugführer bzw. dessen Verantwortlicher soll die Möglichkeit erhalten, die entsprechenden Hindernisse vorab, z. B. mit Hilfe einer Karte (vgl. Modellierungswerkzeug, Abb. 7.2), zu definieren. Das System muss gleichzeitig einen Echtzeit-Betrieb gewährleisten, damit die vom Fahrzeugführer erkannten Hindernisse den anderen Arbeitsmaschinen in der Arbeitsgruppe mitgeteilt werden können. Das mWfMS soll den Fahrzeugführer der Maschine bei der Erkennung kontext-relevanter Hindernisse helfen.

Bei der Anfahrt zum Einsatzort ergeben sich zahlreiche räumliche Einschränkungen. So müssen Fahrzeugmasse, Fahrzeugbreite und -höhe sowie Wendekreis den Streckenbedingungen entsprechen. Der Gegenverkehr muss zu (fast) jedem Zeitpunkt mitberücksichtigt werden. Felder sind gewöhnlich über wenige Zugangspunkte erreichbar. Landwirtschaftliche Flächen können mit zahlreichen Hindernissen von unterschiedlicher Ausprägung gekennzeichnet sein (siehe Abb. 7.4).

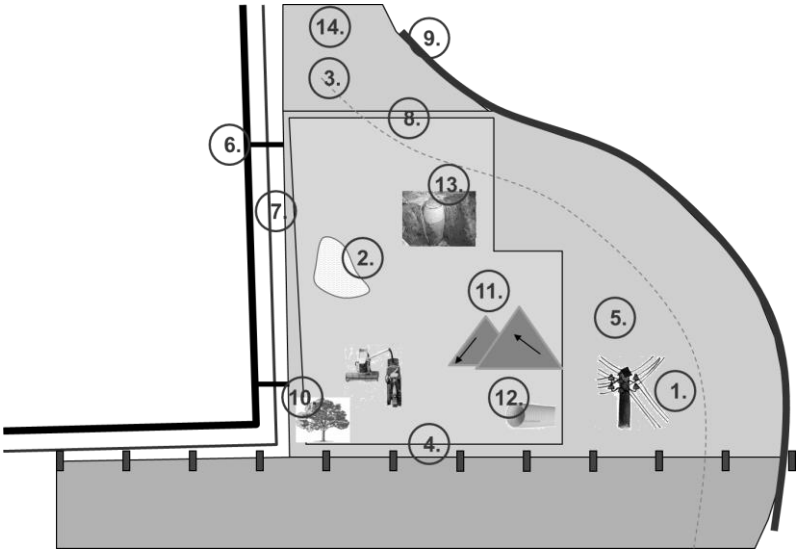


Abb. 7.4: Hindernisse im Feld

In folgender Auflistung werden die Hindernisse (siehe Abb. 7.4) beschrieben:

- Hindernisse sind entweder statisch oder dynamisch (XOR)
  - Statisch: Strommast (1), Baum (10)
  - nicht vorhersagbar dynamisch: Überschwemmungsfläche nach Regen (2)
  - vorhersagbar dynamisch: Resterntefläche (5)
- Hindernisse sind sichtbar oder unsichtbar
  - sichtbar: Strommast (1)
  - unsichtbar: Drainage (12)
- Hindernisse sind permanent sichtbar oder zeitweise unsichtbar
  - permanent: Baum (19)
  - zeitweise: Zaun (4)

- Hindernisse sind physischer Natur oder virtuell
  - physisch: Zaun (4)
  - virtuell: gesetzlich vorgeschriebener Korridor in der Nähe von Fließgewässern (3), z. B. beim Düngen
- Hindernisse können sich aus dem Verfahren ergeben
  - gesetzlich vorgeschriebener Korridor zu Fließgewässern (Düngeverordnung – DüV [BuJu06, §4] (3) bei Pflanzenschutz und Düngemaßnahmen (auch abhängig von Gewässerkategorie, verwendetem Mittel und verwendeter Ausbringungstechnik)
  - Maschineneinstellung Arbeitshöhe (13): ist für die Pflanzenschutzspritze kein Hindernis, aber für die Bodenbearbeitung
  - Maschineneinstellung Arbeitstiefe (12): Eine zu große Arbeitstiefe bei der Bodenbearbeitung kann zur Zerstörung der Entwässerungsdrainage führen
- Hindernisse können sich aus den Eigenschaften der eingesetzten Maschinen ergeben
  - Höhe: Baum (10); (niedrige Maschinen können unter der Baumkrone entlangfahren)
  - Breite: Überfahrungen (6) oder Zäune an Feldzugangspunkten (schmale Maschinen passen durch, breite Maschinen müssen einen geeigneten Zugangspunkt wählen)
  - Masse: Die Masse der Maschine darf die Tragfähigkeit der Überfahrung oder Zuwegung nicht überschreiten.
  - Wendekreis: Zuwegungen mit kleinem Kurvenradius verhindern Zugang (6).

- Hindernisse können sich aus dem vorhandenen Geländerelief in Kombination mit Maschineneigenschaften und -zuständen ergeben

    - Fahrt bergauf in Falllinie: Die Maschine kann aufgrund von Masse (incl. Beladungszustand), Reibung (Art der Bereifung) und Leistung (Motorisierung) den Anstieg (11) nicht bewältigen
    - Fahrt bergab in Falllinie: Die Maschine kann aufgrund von Masse (inkl. Beladungszustand), Reibung (Art der Bereifung) und Bremsleistung die Abfahrt (11) nicht bewältigen.
    - Fahrt entlang der Höhenlinie: Die Maschine driftet in Abhängigkeit von Hangneigung, Masse (inkl. Beladungszustand), Reibung (Art der Bereifung) ab. Die Maschine kippt in Abhängigkeit von Hangneigung und Schwerpunktlage

- Hindernisse können ganze Flächen oder Teilflächen betreffen

    - Teilfläche (2)
    - Ganze Fläche (14): Bsp. Ausbringung von Klärschlämmen: Die Genehmigung wird nur für ein bestimmtes Feld gewährt (nicht für die benachbarte Fläche).

- Hindernisse können bei Maschinenberührung beschädigt werden (10) oder nicht (11)

- Maschinen können bei Hindernisberührung beschädigt werden (13) oder nicht (3)

- Hindernisse können durch Bodeneigenschaften gekennzeichnet sein. Lokal hohe Nährstoffgehalte beschränken oder verbieten die Ausbringung weiterer Nährstoffe

Es gibt weitere Hindernisse, die jedoch nur temporär auftreten. So darf z. B. Gülle laut Gesetz [BuJu06, §4] in bestimmten Zeiträumen nicht ausgebracht werden, der Zeitraum wird auch abhängig davon bestimmt, ob es sich um Ackerland (1. November bis 31. Januar) oder Grünland (15. November bis

31. Januar) handelt. Bestimmte Maßnahmen dürfen in bestimmten Pflanzenentwicklungsstadien nicht durchgeführt werden (z. B. Wartezeiten vor der Ernte beim Einsatz bestimmter Pflanzenschutzmittel zur Vermeidung von Rückständen in Nahrungsmitteln).

Aus der strukturierten Darstellung kann folgendes abgeleitet werden:

- Hindernisse jeglicher Art müssen systematisch beschrieben werden.
- Mobile Maschinen müssen in ihren Merkmalen systematisch beschrieben werden.
- Verfahren bzw. Prozesse und daraus resultierende Maschinenzustände müssen systematisch beschrieben werden.

Aus der systematischen Beschreibung der Punkte 1 bis 3 müssen die möglichen Wechselwirkungen aufgelöst werden. Die Wirkungen werden nach Schadenspotential kategorisiert. Für die Wirkungen werden geeignete Vermeidungsstrategien entwickelt:

1. Gestuftes (priorisiertes) Warnkonzept (HMI)
2. (Teil-) Automatisierte Verfahren zur Vermeidung (z. B. GPS-basierte Tiefensteuerung von Anbaugeräten (12), GPS-basierte Teilbreitensteuerung (3), Automatisierte Lenkung (2))

In Tab. 7.1 sind die Wechselwirkungen der verschiedenen Hindernisse untereinander aufgeführt. Falls eine Wechselwirkung zwischen Hindernis und Eigenschaft vorhanden ist, wird das Feld mit „x“ gekennzeichnet. Falls kein Bezug ableitbar ist, wird das Feld mit „-“ charakterisiert. Wechselwirkungen, die jeweils nur partiell wirken, sind mit „(*)“ gekennzeichnet. Das Zielobjekt (Hindernis) wird mit „o“ gekennzeichnet. Eine mobile Arbeitsmaschine wird mit „m“ innerhalb von Tab. 7.1 dargestellt.

Tab. 7.1: Wechselwirkungen von Hindernissen

		statisch	sichtbar	voraussagbar	temporär	physisch	determiniert über Prozess	determiniert über Maschine	Gesamtfläche	beeinflusstes Objekt
1.	**Strommast**	x	x	x	-	x			-	m/o
2.	**Wasserloch, Morast**	-	x	-	x	x			-	o
3.	**rechtlicher Abstand**	(-)	-	x	-	-	x	x	-	o
4.	**Zaun**	x	(x)	x	-	x			-	m/o
5.	**bearbeiteter Bereich**	-	x	x	x	x	x	x	-	o
6.	**Zugang Fahrzeug**	x	x	x	-	x			-	m/o
7.	**Bewässerung, Wassergraben**	x	x	x	-	x			-	m/o
8.	**Feldgrenze**	x	x	x	-	x			-	o
9.	**fließende Gewässer**	x	x	x	-	x			-	m/o
10.	**andere Maschinen**	-	x	(x)	x	x	x	x	-	m/o
11.	**Geländerelief**	x	x	x	-	x	x	x	(x)	m/o
12.	**Untergrund-Rohre, Drainage**	x	-	x	-	x	x	x	-	o
13.	**Kanaldeckel**	x	(x)	x	-	x	x	x	-	m/o
14.	**gesamte Fläche**	x	x	x	x	x	x	x	x	O

x = positiv - = negativ ( ) partiell m = Maschine o = Zielobjekt

Die hohe Zahl der räumlichen Einschränkungen durch Hindernisse und die Anfälligkeit von Maschinen (im Falle der Nichtbeachtung) begründen die Notwendigkeit einer Unterstützung durch das mWfMS. Vermeidungsstrategien werden durch den Fahrzeugführer durchgeführt, der über das mWfMS/HMI Hinweise bzw. Warnungen erhält.

7.1.5 Ausführung der Hinderniswarnung

Das mWfMS ermöglicht ein serviceorientiertes System mit modularen komponentenorientierten Diensten (siehe Kapitel 6.4). Im Anwendungsfallbeispiel wird die Hinderniswarnung (als paralleler Supportprozess) innerhalb einer Fahrzeuggruppe betrieben. Die Hindernisse werden teilweise vorab definiert, müssen jedoch auch zur Laufzeit über das mWfMS erfasst werden können (vgl. Kapitel 7.1.4). Für jeden Schlag (Feld) wird eine Instanz des Hinderniswarnprozesses geladen (vom Workflow-Typ Hinderniswarnung). Die Hindernisse selbst werden innerhalb einer Datenbank abgespeichert. Die Datenbank wird vom Verantwortlichen um die neu auftretenden Hindernisse kontinuierlich erweitert. Jedes Hindernis (z. B. Temporäre Hindernisse: umgefallener Baum) kann wieder aus dem System entfernt werden.

Für die Hindernisse ist der Mechanismus der (negativen) OE verwendet worden. Die negative OE verbietet die Ausführung einer bestimmten Aktivität bzw. eines Teilprozesses an diesem Ort. D.h. dass der entsprechende Teilprozess von der Engine (mWfMS) nicht ausgeführt wird. Wenn der Fahrer ein Hindernis erkennt, definiert er über das HMI in der MEMIS das Hindernis (siehe Abb. 6.1, rechts). Über den Broadcast-Mechanismus des mWfMS (siehe Kapitel 6.5) werden die anderen Fahrzeuge (MEMIS) über das neue Hindernis in Kenntnis gesetzt.

Polygone werden bei der Hinderniswarnung zur Design-Zeit unterstützt. Zur Laufzeit werden jedoch lediglich Kreise (vgl. *CircleLocation*: siehe Abb. 4.5) definiert. Die Definition von Polygonen war während der Fahrt für den Fahrer der landwirtschaftlichen Maschine nicht praktikabel umsetzbar (am HMI). Die „Hindernis-Kreise“ konnten mittels eines einfachen Kopfdrucks ausgelöst werden. Die Hindernis-Information wird umgehend an die

Arbeitsgruppe (über das mWfMS) weitergeleitet und direkt von den mWfMS (MEMIS) ausgewertet.

Die Dienste sind auf Basis primitiver Funktionen bereitgestellt worden. Die primitiven Funktionen (z. B. GPS-Datenaufbereitung) arbeiten wiederum mit den Rohdaten, die über die Sensoren (z. B. GPS-Empfänger) geliefert werden. Die Rohdaten einer OE bestehen aus der aktuellen GPS-Position der mobilen Maschine, einem sogenannten *Header* (viele unterschiedliche Informationen), der aktuellen Geschwindigkeit und dem Maschinenstatus (weitere Kontextinformationen). Der Maschinenstatus beinhaltet z. B. die aktuelle Spannweite (Breite, Höhe, Länge) der mobilen Arbeitsmaschine. Eine Erntemaschine mit ausgefahrenem Schneidewerkzeug besitzt beispielsweise eine größere Spannweite und muss früher vor einem Hindernis gewarnt werden. Mit Hilfe der vorliegenden Kontextdaten, werden die negativen OE ausgewertet. In Folge können Warnmeldungen (über das HMI) generiert werden.

Für die Warnfunktion der Hinderniswarnung wurde ein BPEL-Prozess erstellt. Die Workflow-Instanzen werden über ein für die Engine bereitstehendes Prozess-Monitoring (siehe Kapitel 3.1) überwacht. Im mWfMS ist damit kontrolliert worden, vor welcher Art von Hindernis gerade gewarnt wird (physisches oder virtuelles Hindernis). Logisch permanente und temporäre Hindernisse werden mit indirekten negativen OE implementiert, da sie aus Polygon-Typen-Beschreibungen abgeleitet werden (siehe Kapitel 7.1.4). Die Erstellung und Verteilung der OE bzw. der Hinderniswarnung muss in Echtzeit erfolgen. Der Verteilungsworkflow wird über das HMI gestartet und beinhaltet die Generierung des temporären Hindernisses (indirekte negative OE) und die Verteilung der OE innerhalb der Fahrzeuggruppe. Die temporären Hindernisse werden vom Fahrer der mobilen Maschine erstellt, wenn ein unbekanntes Hindernis bemerkt wird. Der Fahrzeugführer gibt die Daten des Hindernisses – die grobe Lage des Zentrums mit Richtung und Entfernung aus seiner Sicht – mit der Angabe eines Radius über das HMI in die MEMIS ein. Die Dateneingabe wird durch eine vordefinierte Vorlage (engl. *template*) auf dem HMI der MEMIS unterstützt (siehe Abb. 6.11). Die eingegebenen Werte sind lediglich Richtwerte, abhängig vom aktuellen

Standort der eigenen mobilen Maschine und abhängig von der Einschätzung des Fahrzeugführers.

Die Warnung selbst wird durch einen Punkt-in-Polygon-Algorithmus [Kuep05, S. 54ff] realisiert. Wenn die mobile Maschine auf Basis ihrer GPS-Position ein Gebiet durch ein Hindernis-Polygon registriert hat, so wird dies mit einer Warnmeldung akustisch signalisiert und am Display dem Maschinenführer angezeigt. Damit der Maschinenführer genügend Zeit hat, um dem Hindernis auszuweichen wird die aktuelle Geschwindigkeit ermittelt. Bei einer schneller fahrenden mobilen Arbeitsmaschine wird somit das akustische/graphische Signal früher abgesandt, damit der Fahrzeugführer auf die Warnung rechtzeitig reagieren kann. Bei solch einer Warnmeldung wird ein dynamischer Bereich auf dem HMI mit einer rot flackernden Anzeige des aktuellen Abstands zum Hindernis visualisiert (siehe Abb. 6.1, rechte Seite). Die Position des Hindernisses wird durch einen Pfeil und den Abstand gegenüber der mobilen Maschine gekennzeichnet.

Ein vorübergehendes Hindernis wird somit immer direkt unter den mobilen Arbeitsmaschinen mittels Ad-hoc-Nachricht verteilt. Die Maschinen bilden über das angebundene WLAN der Fahrzeuggruppe einen Verbund (siehe Abb. 6.7).

Weitere Supportprozesse

Parallel zur Hinderniswarnung und -verteilung wurden weitere Supportprozesse implementiert, die Erkenntnisse über die Wiederverwendbarkeit des mWfMS bzw. der Prozesse für das Verbundprojekt R2B lieferten.

Ein evaluierter Prozess als „Zusatzleistung“ war beispielsweise die automatische Buchung der Leistungen. Mit den Prozessen werden z. B. während der Ernte Prozesszeiten (Ernte- und Anfahrtszeiten), Erntemengen, verbrauchte Spritmengen ermittelt. Diese Daten des Prozesskontextes werden vom mWfMS ausgewertet, um daraus beispielsweise eine „automatische Buchung“ für den Verantwortlichen bzw. Lohnunternehmer bereitzustellen.

Eine weitere Zusatzleistung ist die Düngemittelausbringung. Das mWfMS (mit der Unterstützung der OE) wurde verwendet, um die Düngemittelausbringung zu steuern. Jede abgefahrene Strecke wurde als temporäres Hindernis (negative OE) markiert, über das die mobile Maschine bei der Düngemittelausbringung nicht wieder fahren sollte.

Fazit

Das mWfMS konnte anhand des landwirtschaftlichen Anwendungsfallbeispiels der Hinderniswarnung mit der vorgeschlagenen Architektur aus Kapitel 6 und mit der Modellierung von (negativen) OE (siehe Kapitel 4) als Workflow modelliert und ausgeführt werden. Der Maschinen-Kontext (über Sensoren) der landwirtschaftlichen Fahrzeuge wurde verwendet, um über Regel-Maschinen die Prozesse zu beeinflussen. Die Prozesse wurden mit BPMN in der Design-Phase modelliert und mit BPEL implementiert und ausgeführt. Die verschiedenen Formen der OE wurden angewandt und innerhalb eines bestehenden WfMS implementiert. Das WfMS wurde zu einem mWfMS erweitert, indem Ortsbezüge und weitere Kontexte über verschiedene Mechanismen integriert wurden. Die Grenzen des beschriebenen Ansatzes liegen in der Interaktion zwischen Fahrer und Maschine. Die Festlegung eines temporären Hindernisses während der produktiven Fahrt (z. B. Ernte) erwies sich als fehleranfällig. Jedoch war es für die Fahrgruppe stets hilfreich, über das Hindernis im Feld zu informieren (schlechte Informationen sind besser als keine Informationen). Vergleichbare Systeme werden mit Hilfe von sogenannten Laserscannern am Fahrzeug befestigt. Diese können z. B. mit Hilfe des „*Context Receiver*“ (vgl. Abb. 6.2) mit dem mWfMS verbunden werden, so dass auch detektierte Hindernisse mit aufgeführt werden können.

7.2 Automatisierte Aktivitäten mit Ortsbezügen

In diesem Kapitel werden mobile Aktivitäten mit Ortsbezug innerhalb einer Anwendung (App) betrachtet. Die mobilen LBS-Anwendungsfälle werden

vorgestellt (siehe Kapitel 7.2.1), ein unterstützendes Datenmodell wird entworfen (siehe Kapitel 7.2.2) und es wird eine App zur graphischen Unterstützung der Nutzer realisiert (siehe Kapitel 7.2.3). Die Beiträge des Kapitels basieren auf den in [DeOS10] entwickelten Ursprüngen.

7.2.1 LBS-Anwendungsfälle

Die LBS-Anwendungsfälle enthalten Textdokumente mit Ortsbezügen, auf deren gemeinsamer Basis ein Datenmodell entworfen wird (siehe Kapitel 7.2.2). Für die verschiedenen Anwendungsfälle soll eine App „für eine bestimmte Aufgabe“ (siehe Kapitel 2.1.7) realisiert werden, die alle Anwendungsfälle beim Zugriff auf virtuelle Textdokumente unterstützt.

Informationsdienste

Informationsdienste bzw. Anfragedienste sind häufig genutzte LBS (vgl. POI). Der (mobile) Endbenutzer erhält als Rückgabewert eine Liste mit POI, die entsprechend der Anfrage-Query aufgelöst werden. Die Informationen der Ortung werden automatisch vom Endgerät ausgehend erfasst und dem LBS (meist lokal als App) übermittelt. Die Ortsinformationen können bei einigen LBS auch manuell eingegeben werden. In der Abfrage nach POI werden persönliche Interessen des Endbenutzers (z. B. die nächste Tankstelle in der Umgebung) aufgelöst. Zusätzlich muss der Aktionsradius für die POI (z. B. die Tankstellen) eingegrenzt werden. Diese Informationsdienste werden häufig innerhalb von anderen Diensten (z. B. Navigation) integriert.

Sofortnachrichtendienst

Der Sofortnachrichtendienst (engl. *Instant Messaging*) ist eine Form der Kommunikation über das Internet, die eine Übertragung von Text-basierten Nachrichten in Echtzeit vom Absender zum Empfänger bietet. Es werden Punkt-zu-Punkt-Kommunikation und Multicast-Kommunikation von einem Sender zu vielen Empfängern unterstützt. *Instant Messaging* (IM) existiert meist in Kombination mit sogenannten Benutzergruppen. Die Kommunikation erfolgt über Chat-Foren o. ä. Umgebungen. Die Endbenutzer haben beim *Instant Messaging* die Möglichkeit, persönliche Daten zu hinterlegen

und sehen (in Echtzeit) ob Freunde bzw. Bekannte aus der Kontakteliste gerade „online“, „abwesend“ oder „beschäftigt“ sind. IM-Dienste können auch ortsabhängig zur Verfügung gestellt werden [FrTF03]. Der aktuelle Aufenthaltsort wird beispielsweise in die Nachrichtenanfrage integriert, so dass der Endbenutzer nur diejenigen Kontakte zurückerhält, die sich in unmittelbarer Nähe befinden. Laut [FrTF03] erfordern solche LBS eine ständige Abfrage des aktuellen Zustands bzw. verlangen neue Konzepte für die Wahrung der Privatsphäre der Endbenutzer.

Verkehrstelematik

Verkehrstelematik (engl. *Traffic Telematics*) soll Endbenutzer als Fahrzeugführer beim Fahren unterstützen. Bekannte Dienste aus diesem Bereich sind beispielsweise Navigationsdienste, Pannendienste oder Geschwindigkeitsbegrenzungsanzeigen. Innerhalb einer Navigationslösung werden mehrere *Traffic Telematic* Dienste kombiniert angeboten. Je nach Hersteller des Navigationsgerätes ist es möglich, aktuelle Staumeldungen oder einen Wetterbericht für den aktuellen Aufenthaltsort zu beziehen. Navigationslösungen werden mit weiteren Diensten kombiniert (z. B. Parkplatzfinder). Die elektronische Suche nach einem Parkplatz spart Zeit, Energie, Kosten und reduziert das innerstädtische Verkehrsaufkommen [ScBa02].

Mobiles Marketing

Beim mobilen Marketing werden Werbebotschaften auf den mobilen Geräten potentieller Kunden publiziert. Werbebotschaften werden an die Endbenutzer verschickt, die sich mit ihrem mobilen Endgerät an einem bestimmten Ort befinden und im Besitz einer bestimmten App sind [VePo02]. Der Endbenutzer kann bestimmte Interessen verfolgen, indem er die Produktkategorien eingrenzt [VePe02].

Mobile Marketing-Systeme setzen voraus, dass der Endbenutzer den Wunsch äußert, per Werbung Nachrichten zu bestimmten Angeboten zu erhalten [DBSK05]. Die Werbenachrichten selbst werden in verschiedenen Kategorien mit bestimmten Attributen untergliedert. Die Attribute besitzen eine Vererbungsfunktionalität, um z. B. Attribute wie „Kneipe“ oder „Catering“ der Klasse „Gastronomie“ unterzuordnen [DBSK05]. An den zentralen

LBS werden Anfragen gestellt, die persönliche Daten, Kategorien und den aktuellen Aufenthaltsort mit einbeziehen. Der Endbenutzer erhält daraufhin die „passende“ Werbenachricht z. B. als SMS, MMS, Email oder Sprachnachricht direkt auf das mobile Endgerät. Ein von Google Inc. eingereichtes Patent[4] analysiert gleich mehrere Kontextinformationen (des Gerätes über den Endbenutzer) und versucht, Werbung präzise an den jeweiligen Endbenutzer anzupassen.

Virtuelles Graffiti

Virtuelles Graffiti (engl. *Virtual Graffiti*) ist ein Community-basierter Dienst, dessen Nutzer an bestimmten Orten virtuelle Nachrichten hinterlegen können. Die Nachrichten sollen von jedem Nutzer, der sich an dem festgelegten Ort befindet, gelesen werden können. Das Recht, die Nachricht zu editieren, hat jedoch nur der ursprüngliche Ersteller [BuGa02]. Anhand der Daten, dem Aufenthaltsort und dem Nutzer wird eine Liste der sichtbaren Dokumente erstellt und an das Endgerät geschickt. Bei der Neuerstellung von Textdokumenten hat der Nutzer die Wahl zwischen öffentlichen und privaten Nachrichten. Während öffentliche Dokumente für alle Endbenutzer sichtbar sind, können die privaten auf einen bestimmten Nutzer (bzw. bestimmte Gruppen) eingegrenzt werden. Im *Virtual Graffiti* Konzept von [CCDH04] werden „öffentliche Poster“ (*Plasma Posters*) verwendet, um mit dem Nutzer zu kommunizieren. Die Nutzer können Kommentare anderer Endbenutzer lesen, diese aber nicht ändern. Die Kommentare liegen in Form von Texten, Bildern und Audio-Aufnahmen vor. Es werden im Gegensatz zu [BuGa02] drei unterschiedliche Kommentararten unterstützt [CCDH04]: Kommentare für den persönlichen Gebrauch, Kommentare für Gruppen und öffentliche Kommentare. Die öffentlichen Poster sind miteinander vernetzt, so dass die Informationen an mehreren Orten angezeigt werden [CCDH04].

[4] United States Patent 8,138,930; direkter Link zum United States Patent: http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=8,138,930.PN.&OS=PN/8,138,930&RS=PN/8,138,930

Erinnerungsdienste

CybreMinder ist ein virtueller Erinnerungsdienst, der ortsabhängig Nachrichten an den Nutzer (bzw. das mobile Gerät) absetzt [DeAb00]. Die Nachrichten beziehen sich auf bestimmte Aufgaben oder geplante Aktivitäten im Umfeld des Endbenutzers und bestehen aus zwei Teilen: dem Signal und der Beschreibung. Signale sollen auf die eigentliche Nachricht aufmerksam machen und können zusätzlich audiovisueller Natur sein [DeAb00]. *CybreMinder* besitzt textuelle Beschreibungen von Aufgaben. Zusätzlich können mit Hilfe von *CybreMinder* Prioritäten vergeben und spezielle Kontexte definiert werden, indem mehrere Aktionen überlagert (logische UND-Verknüpfung) als Ereignis angegeben werden (z. B. Benutzer X in Gebäude B zur Uhrzeit C). Es wird X mit B und C verknüpft. Diese Verknüpfungen werden als *„subsituation"* bezeichnet [DeAb00]. Nach der Erfüllung der *subsituation* wird eine Erinnerungsnachricht an den Nutzer verschickt. *CybreMinder* besitzt für jede Aufgabe ein Ablaufdatum [DeAb00].

Content-Managementsysteme

Content-Managementsysteme (CMS) sind Informationssysteme zur gemeinschaftlichen Bearbeitung elektronischer Dokumente [Spör09]. CMS bestehen aus zwei Komponenten, der Applikationskomponente und der Verteilungskomponente. Mit Hilfe der Applikation werden die Daten erstellt, geändert oder gelöscht. Die Verteilungskomponente ist für die eigentliche Erstellung (Kompilierung) und für die Darstellung der Dokumente zuständig. Oft werden CMS z. B. zur Verwaltung von dynamischen Webseiten verwendet. In CMS können Ortsbezüge integriert werden [TuJo05]. Die Nutzer werden in zwei Rollen, die Content-Provider und in die Standardnutzer, aufgeteilt [TuJo05]. Die Content-Provider sind die Benutzer der CM-Applikation zur Verwaltung der Dokumente. Ein Nutzer dieser Gruppe kann Dokumente erstellen, löschen, ändern und über eine bestimmte Werkzeugunterstützung Orte Dokumenten zuweisen. Die Werkzeugunterstützung für die Content-Provider kann über ein mobiles Endgerät oder von einem (stationären) Computer aus bedient werden [TuJo05]. Ein Standardnutzer des CMS kann abhängig vom Ort Informationen über gespeicherte Dokumente einholen [TuJo05].

Dateisysteme

Dateisysteme sind Komponenten die in fast allen Betriebssystemen zu finden sind. Sie stellen verschiedene Operationen zur Dateiverwaltung zur Verfügung. Die Dateien werden in einem Dateisystem meist in einer Baumstruktur angeordnet, worin die Dateien selbst über ihren Pfad einen eindeutigen Bezeichner erhalten. Dateisysteme können Ortsbezüge verwalten [HeCa03]. Eine Region (vgl. Ortstyp) wird in aktive Gebiete untergliedert, die Zugangsberechtigungen enthalten [HeCa03]. Diese aktiven Gebiete können Referenzen auf Dateien erhalten. Mit einem aktiven Gebiet werden immer Nutzer- und Systemdateien assoziiert, z. B. für die Zugriffsberechtigung. Wenn ein Nutzer ein aktives Gebiet betritt, wird ein neues Verzeichnis erzeugt, in das die persönlichen Daten eingefügt werden. Es werden zwei verschiedene Betriebsarten unterstützt [HeCa03]: Im *File Mode* sind die Endbenutzerdaten in den entsprechenden Verzeichnissen für jeden Endbenutzer sichtbar. Im *Context Mode* werden die Inhalte kontext-bezogen dargestellt. In diesem Modus enthalten die Systemdaten Informationen, die eine Kontextauswertung ermöglichen. Ein Kontext ist z. B. der aktuelle Aufenthaltsort eines Endbenutzers, der eine Freigabe oder eine Sortierung der Dateien in einem bestimmten Verzeichnis auslösen kann [HeCa03].

Notizen

Elektronische Notizen können digital und ortsabhängig auf mobilen Geräten erzeugt werden [PEFS03]. Die Nutzer eines solchen Systems haben die Möglichkeit, elektronischen Notizen für bestimmte andere Nutzer oder generell an alle Nutzer (*broadcasting)* freizugeben. Jede virtuelle Notiz beinhaltet eine Signatur (zur Identifikation des Erstellers), die Ortsinformation (Erzeugungsort der Nachricht) und den eigentlichen Inhalt. Auf Grund von Ortungsfehlern werden sogenannte *Place Labels* eingeführt [PEFS03], die die Orte in konkrete natürliche Zeichenketten abbilden (z. B. Bahnhof). Die *Place Labels* sind notwendig, damit auf Grund von Ortungsfehlern die Notizen nicht falschen Orten zugewiesen werden. Beim Speichern der Nachricht kann der Nutzer anhand des *Place Labels* überprüfen, ob er die Nachricht diesem Ort wirklich zuweisen möchte [PEFS03].

Persönliche Dokumente

Stick-e Notes sind persönliche elektronische Dokumente, die einen Ortsbezug und einen Adressaten mit der Nachricht beinhalten [Brow96]. *Stick-e Notes* können am Erzeugungsort eingesehen werden, sofern der Adressat der aktuelle Nutzer ist. Gruppenfunktionen oder ähnliche Berechtigungskonzepte sind bei *Stick-e Notes* nicht vorgesehen.

Ortsabhängige Wikis

Ein Wiki ist ein System für Webseiten, deren Inhalte von Nutzern gelesen und direkt im Webbrowser verändert werden [EGHW08]. Wikis werden in Unternehmen häufig verwendet, um kooperative Dokumentationen zu erstellen. Ortsabhängige Wikis (bzw. Wikis mit Ortsbezügen) werden genutzt, um den Informationszugang zu Verwaltungssystemen zu erleichtern [ABHK05]. In ortsabhängigen Wikis wird der dreidimensionale Raum in sogenannte *Landings* unterteilt, die wiederum im Wiki eine eigene URL bzw. Webseite besitzen [ABHK05]. Der Nutzer wird bei einem durchgeführten Ortswechsel automatisch auf die richtige Wiki-Seite navigiert. Die Benutzung bzw. die Editierung des Wikis erfolgt nach dem „jeder darf schreiben"-Prinzip [EGHW08].

In [SLBG07] werden zusätzlich vier kontextbezogene Listen je nach Aufenthaltsort auf dem mobilen Gerät angeboten, um ein effizientes Arbeiten zu gewährleisten. Die vier Listen bestehen aus der Liste der Nachbarorte, der Liste der populären Seiten, der Liste der neu erzeugten bzw. bearbeiteten Seiten und einer zufallsgenerierten Liste von Wiki-Seiten.

7.2.2 Datenmodell

Das Datenmodell für die LBS-Anwendungsfälle gewährleistet mit bestimmten Benutzerrechten ein Zugriffskontrollmodell auf die Datenobjekte, um die verschiedenen LBS-Anwendungsfälle auszuführen (siehe Kapitel 7.2.1). Die verwaltenden Informationsobjekte werden mit Attributen und Zusammenhängen modelliert, um einen Überblick über die Datensicht innerhalb

der Anwendung zu erhalten. Das Datenmodell für die LBS-Anwendungsfälle basiert auf einem ACL[5]-Ansatz, bei dem jeder Endbenutzer eine eindeutige Kennung erhält. Endbenutzer werden in Gruppen eingeteilt, wobei nicht ausgeschlossen ist, dass ein Endbenutzer in mehreren Gruppen vorhanden ist. Ein Nutzer besitzt daher immer genau eine aktive Gruppe, damit im Modell zu einem gegebenen Zeitpunkt eindeutige Zugriffsrechte identifiziert werden können. Alle Endbenutzer sind Mitglieder der Gruppe „Benutzer" (vgl. Entitäten in Abb. 7.5). Die Gruppe „Benutzer" und die jeweiligen Gruppen sind sogenannte „Rechtsträger". „Rechtsträger" vergeben Zugriffsrechte für die entsprechenden Textdokumente. Bei der Rechtevergabe ist der zugehörige Ort zu identifizieren. Für die Erstellung eines Dokuments muss der „Dokumententyp" festgelegt werden. Der Dokumententyp definiert abhängig vom Typ einen Ort als Kreis (mit Punkt) oder als Polygonzug und legt die Standardrechte fest (vgl. Kapitel 4.2). Standardrechte beinhalten die Rechte des Dokumentenbesitzers, der besitzenden Gruppe und der restlichen Endbenutzer (falls vorhanden). Ein Dokumentenbesitzer ist der Ersteller des Dokuments. Analog dazu wird die zu diesem Zeitpunkt aktive Gruppe als „besitzende Gruppe" bezeichnet. Im Unix-Betriebssystem wird für solche Zwecke eine spezielle Umgebungsvariable *umask* eingesetzt, die je nach Benutzer differenziert [Tane09]. Im Vergleich dazu sind in diesem Modell die Standardrechte für alle Endbenutzer gleich und nicht personalisierbar. Ein Endbenutzer kann nur Dokumententypen wählen, die für die aktive Gruppe (oder ihn speziell) freigeschalten wurden. Damit ist es beispielsweise möglich, einzelnen Personen bzw. Gruppen besondere Administrationsrechte zuzuweisen. Falls der Dokumententyp für die aktive Gruppe andere Rechte als für die restlichen Endbenutzer definiert, sollen diese Rechte für die besitzende Gruppe gelten. Eine Ortsdefinition kann für den Dokumententyp als Kreis mit Punkt und Radius oder als Polygon definiert werden. Der Dokumententyp bestimmt die zugehörige Ortsklasse (-typen), bei der die Operationen gelten. Orte gleicher Ortsklassen (-typen) dürfen sich nicht überschneiden.

[5] Eine Access Control List (ACL) ist eine Zugriffssteuerungsliste, mit der das Betriebssystem (Applikationen) Zugriffe auf Daten und Funktionen eingrenzt. Eine ACL legt fest, welcher Benutzer welche Dienste und Dateien nutzen darf [Tane09].

In einer „Rechte-Liste“ (vgl. Abb. 7.6, „Screen 3“) kann der Nutzer die entsprechenden Rechte für ein Dokument ändern. Die Zugriffsrechte werden während des Zugriffs auf ein Dokument bestimmt, indem zuerst ermittelt wird, ob der Endbenutzer der Besitzer ist. Ist dies nicht der Fall und besitzt der Endbenutzer keinen Eintrag in der Rechte-Liste, werden die expliziten Rechte (vgl. „Benutzer“, „ist in“, „Gruppe“) verwendet. Sind diese Rechte auch nicht definiert, gelten die Rechte der derzeit aktiven Gruppe des Endbenutzers. Falls keines dieser Rechte vorhanden ist, besitzt der Endbenutzer keine Rechte. Das bedeutet dann z. B. im Anwendungsfall, dass der Benutzer auf das Dokument nicht zugreifen kann.

Die ortsabhängigen LBS-Szenarien (aus Kapitel 7.2.1) unterscheiden sich lediglich über die Standardrechte, die für die jeweils im Anwendungsfall verwendeten Textdokumente unterschieden werden. Durch eine eindeutige Festlegung dieser Standardrechte können die LBS-Anwendungsfälle mittels einer App unterstützt werden. Soll zum Beispiel das Szenario mit den ortsabhängigen **Wikis** unterstützt werden, müssen Lese- und Schreibrechte an jeden Endbenutzer vergeben werden. Bei einer Unterstützung von **Virtual Graffiti** werden die Rechte nur für eine bestimmte Gruppe vergeben. Falls das Dokument nur für den Ersteller einsehbar ist, handelt es sich um ein **Persönliches Dokument**. Mit den **Erinnerungsdiensten** kann analog verfahren werden. Lediglich die Kommunikation per *Push* und *Pull* sollte für diesen Anwendungsfall vorhanden sein.

Mit diesem Zugriffskontrollmodell kann das Datenmodell eingeführt werden, wobei die Entitäten des Datenmodells mit dem Zugriffskontrollmodell interagieren. Das Datenmodell ist innerhalb eines *Entity-Relationship*[6]-Diagramm (ER) in Abb. 7.5 dargestellt. Entitäten werden durch Rechtecke repräsentiert und verfügen über Attribute, die als ovale Kreise abgebildet sind. Die Schlüssel (Attribute) einer Entität sind unterstrichen (siehe Abb. 7.5). Schlüsselattribute sind notwendig zur Bestimmung geeigneter Attribute eines Entitätstyps. Die semantischen Beziehungen der Entitäten werden über

[6] Ein Entity-Relationship-Modell (ER-Modell) wird zur semantischen Datenmodellierung verwendet. Ein ER-Modell beinhaltet ein Abbild der realen Welt, indem Datenstrukturen und Inhalte bzw. die Semantik der Daten dargestellt wird [ElNa06].

Rauten innerhalb eines ER-Modells dargestellt. Die Beziehungen können wiederum Attribute besitzen. Die Kardinalität (Zahl zwischen den Verbindungslinien in Abb. 7.5) spezifiziert für jede Instanz des Entitätstyps, wie viele Beziehungen er besitzen kann. In Abb. 7.5 wird die „Ist in“-Beziehung mit Hilfe einer fetten gerichteten Kante dargestellt. Jeder Endbenutzer („Benutzer“) besitzt

- eine „Mail“-Adresse,
- ein Schlüsselattribut des Benutzernamens „BName“,
- ein „Passwort“,
- einen „Status“ (eingeloggt oder ausgeloggt),
- einen „Name“,
- einen „Vorname“,
- einen „Zeitstempel“ des letzten gültigen Login-Versuchs,
- eine Markierung „Banned“ zur expliziten Verweigerung eines Dienstes,
- ein „Geschlecht“,
- einen zusätzlichen „Schlüssel“ zur (asymmetrischen oder symmetrischen) Verschlüsselung der Kommunikation und „Koordinaten“,

um die ortsabhängigen Zugriffsrechte zu identifizieren (siehe Abb. 7.5). Jeder „Benutzer“ gehört zu einer „Gruppe“ und kann Mitglied mehrerer Gruppen sein. Leere Gruppen sind nicht erlaubt. Zu jedem Zeitpunkt ist ein „Benutzer“ Mitglied genau einer aktiven Gruppe (siehe Abb. 7.5, Raute „Aktive Gruppe“).

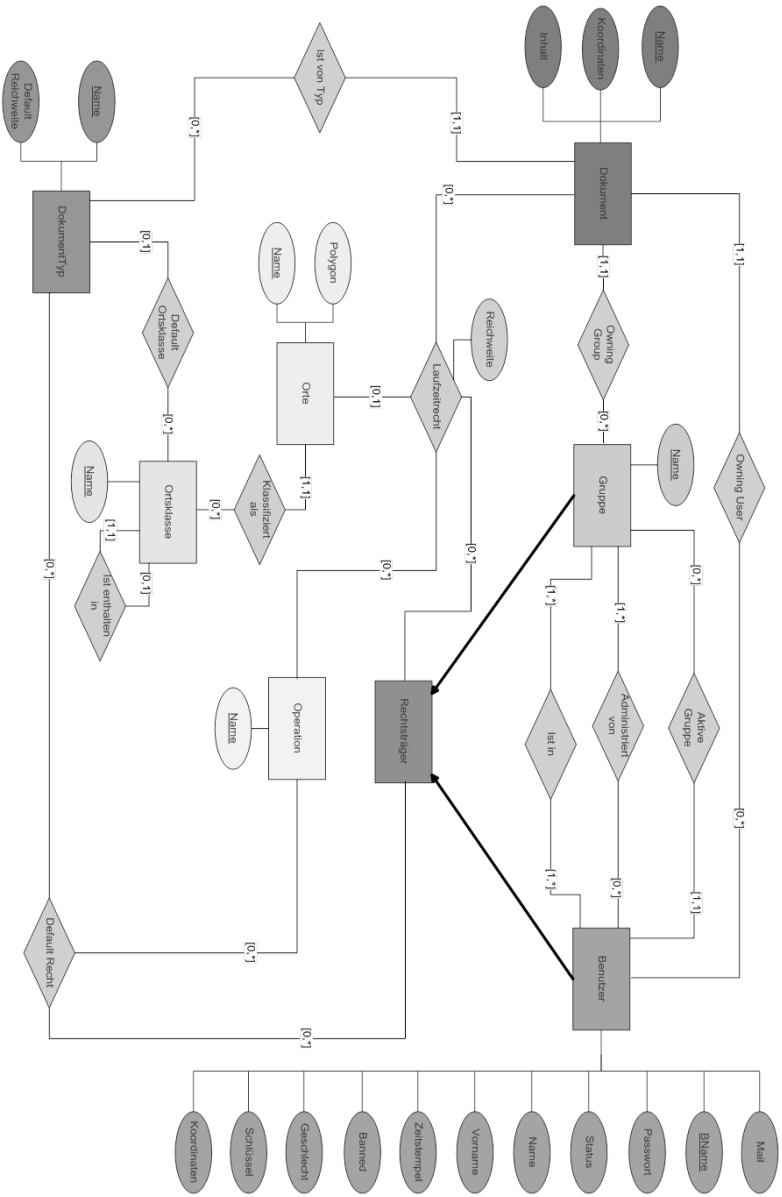


Abb. 7.5: ER-Datenmodell aus [DeOS10]

Die eindeutige Identifizierung einer „Gruppe“ erfolgt anhand des Attributs „Name“. „Gruppen“ und „Benutzer“ sind über die „Ist in“-Beziehung zu „Rechtsträgern“ dargestellt (siehe Abb. 7.5). Beide sind somit „Rechtsträger“ und können Rechte auf Dokumente zuweisen. Ein „Dokument“ (siehe Abb. 7.5) besitzt einen Schlüssel „Name“, eine „Koordinate“ des Erzeugungsortes und ein Attribut „Inhalt“. Zusätzlich hat jedes Dokument einen Besitzer („Owning User“) und eine zugehörige Gruppe („Owning Group“). Standardrechte („Default Recht“) sind von der Auswahl des „Dokument-Typ“ direkt abhängig und werden als Beziehungen zwischen „Rechtsträger“, „Operation“ und „DokumentTyp“ modelliert. Dokumententypen („DokumentTyp“) haben einen Schlüssel „Name“. Entweder haben Dokumententypen eine „Default Reichweite“ oder sind einer „Ortsklasse“ (über „Default Ortsklasse“) zugehörig. Ortsklassen werden kaskadiert verwendet, was an der Beziehung „Ist enthalten“ zu erkennen ist. „Orte“ haben auch einen eindeutigen „Name“ eine zugehörige „Ortsklasse“ (über „Klassifiziert als“) und werden als „Polygon“ charakterisiert. „Laufzeitrechte“ werden verwendet, um die Standardrechte anzupassen. Zusätzlich werden „Laufzeitrechte“ als Beziehung zwischen „Dokument“, „Operation“ und „Rechtsträger“ im Modell dargestellt. Ein Ort ist valide, wenn er über einen Radius (Attribut „Reichweite“) oder mit Hilfe eines Ortes (z. B. anhand des Erzeugungsortes) beschrieben ist.

7.2.3 iLBS-App

Die App, mit der die LBS-Anwendungsfälle aus Kapitel 7.2.1 evaluiert worden sind, besteht aus einer Server- und Client-Komponente (mit SOAP-Anbindung).

Server

Zur Implementierung wird ein Datenbankmanagementsystem (DBMS) verwendet, das geometrische Datentypen und Operationen unterstützt. Mit dem

DBMS PostGIS[7] ist das vorgestellte Datenbankmodell (siehe Abb. 7.5) erstellt worden. Die Implementierung des Servers erfolgt mit Java, da es plattformunabhängig auf allen Betriebssystemen eingesetzt werden kann. Der Einsatz einer anderen Programmiersprache (mit anderen Modulen) wäre auch möglich gewesen. Die Verbindung zur mobilen App ist über einen SOAP-Webservice realisiert worden. Eingesetzt wurde die SOAP-Engine *Apache Axis2*[8] mit einem *Tomcat*[9] Applikationsserver. Die Server-Applikation wurde auf einer *Ubuntu Linux*[10] Installation (auf einer virtuellen Maschine) installiert. Die Eingabe von geometrischen Ortsangaben erfolgt mit der PostGIS-Funktion *AddGeometryColumn()*. Folgende Parameter werden für die Funktion benötigt:

- Tabellenname, in der die geometrische Spalte einsortiert werden soll
- Name der geometrischen Spalte
- *Spatial Reference Identifier (SRID)*: Da GPS-Koordinaten benutzt wurden, ist das *WGS84*-Koordinatensystem mit dem *SRID-4326* verwendet worden
- Typen der geometrischen Spalten (Punkt, Polygon und Kreis)
- Dimensionen der Spalte (Länge, Breite, Höhe)

Damit die Server-Applikation mit der Geo-Datenbank kommunizieren kann, sind JDBC[11]-Treiber für das DBMS installiert worden. Jede Funktion, die vom Server ausgeführt wird, muss vom Client ausgelöst werden. Die Antworten auf die ausgelöste Funktion werden direkt an den Client, von dem die Anfrage kam, zurückgeschickt. Der Aufruf (des Clients) wird direkt in eine SQL-Datenbankoperation übersetzt, um möglichst schnelle Berechnungen zu gewährleisten. Die Beziehung (von Applikation zur Datenbank) wird über eine asynchrone Verbindung (Instanz der Klasse *java.sql.Connection*) gehalten. Um eventuellen Verklemmungen in der Datenbank vorzubeugen, wurde immer nur eine aktive Instanz (hin zur Datenbank) erlaubt. D.h. wenn

[7] http://postgis.net
[8] http://axis.apache.org/axis2/java/core/
[9] http://tomcat.apache.org
[10] http://www.ubuntu.com
[11] http://www.oracle.com/technetwork/java/javase/jdbc/index.html

mehrere Zugriffe auf eine Datei zur gleichen Zeit eingehen, werden diese sequentiell abgearbeitet.

Client

Die Implementierung der Client-App erfolgte mit dem *iPhone-SDK 3.0*[12] (später *iOS*) mit der Programmiersprache *Objective-C* [Dunc03]. Um selbstentwickelte Apps auf einem Apple-Gerät zu installieren, muss ein gerätespezifisches Zertifikat erstellt werden. Dieses Zertifikat wurde über das von Apple für iPhone-Dozenten erstellte Universitätsprogramm[13], an dem das Institut AIFB teilnimmt, erstellt. Implementierungsdetails der App [Arbi09] werden in vorliegender Arbeit nicht weiter betrachtet, da sie keine weiteren Erkenntnisse liefern. In folgender Abb. 7.6 werden die *Screens* der App dargestellt:

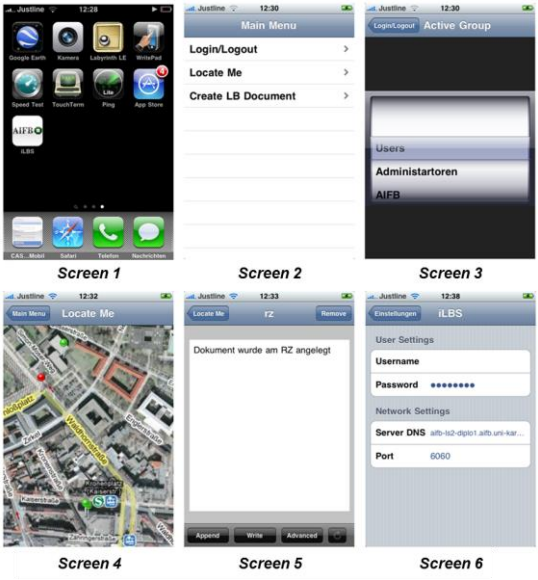


Abb. 7.6: Screenshots der iLBS-App

[12] https://developer.apple.com/devcenter/

[13] https://developer.apple.com/programs/ios/university/

Im „*Screen 1*“ (siehe Abb. 7.6) ist das App-Icon der Applikation erkennbar. „*Screen 2*“ wird automatisch nach dem Start der App angezeigt. Die App besitzt drei native Funktionen: ein „Login/Logout“, ein „Locate Me“ und „Create LB Document“. Unter „Login/Logout“ ist eine Anmeldung, ein Benutzerwechsel oder die Abmeldung des Endbenutzers möglich. Gleichzeitig wird in diesem *Screen* die obligatorische aktive Gruppe zugeordnet (siehe Abb. 7.6, „*Screen 3*“). Mit Anklicken von „Locate Me“ gelangt der Endbenutzer zu einer Kartendarstellung. Der Nutzer wird über eine rote Stecknadel innerhalb der Karte visualisiert. Die Dokumente mit Ortsbezug werden mit grünen Stecknadeln innerhalb der Karte dargestellt (siehe Abb. 7.6). In „*Screen 5*“ wird das kleine implementierte Textbearbeitungstool sichtbar. Mit diesem Werkzeug werden die elektronischen Textdokumente mit Ortsbezug bearbeitet, sofern der Nutzer die entsprechenden Rechte besitzt. In „Screen 6“ ist der Konfigurations-*Screen* dargestellt, mit dem die Benutzer-*Credentials* und die Zugangsdaten des Servers zur Speicherung der elektronischen Textdokumente angegeben werden.

Fazit

Die in Kapitel 7.2.1 beschriebenen LBS-Anwendungsfälle konnten mit Hilfe der iLBS-App und einer Serverapplikation basierend auf einer geographischen Datenbank unterstützt werden [DeOS10, PaVP17]. Allerdings war es bei diversen Feldversuchen für den jeweiligen Endbenutzer schwierig zu verinnerlichen, in welchem LBS-Anwendungsfall er sich gerade befindet. Diejenigen Nutzer, die keinerlei Vorkenntnisse der LBS-Anwendungsfälle hatten, konnten die App nicht anwenden. Mit dem Experiment, ein Zugriffskontrollmechanismus mit Hilfe eines Datenmodells einzuführen und damit eine App bzw. den Datei-Zugriff zu steuern, ist in vorliegendem Kapitel gezeigt worden. Es ist möglich ein größeres Spektrum an Funktionalität mit einfachen Prinzipien über eine explizit gesteuerte ortsabhängige Zugriffskontrolle zu unterstützen. Gleichzeitig wurde durch die Feldversuche deutlich, dass bei mobilen Apps der Handlungsraum für den Nutzer verkleinert werden muss, damit der jeweilige LBS-Anwendungsfall verstanden wird. Ein kleinerer Handlungsraum für die App bedeutet eine eindeutige graphi-

sche Benutzerführung für den entsprechenden Anwendungsfall. Diese Erkenntnis entspricht den *iOS Human Interface Guidelines* von Apple beschriebenen Vorgehensmodellen zur Entwicklung einer App [Appl15].

Das Datenmodell bzw. das ortsabhängige Zugriffskontrollmodell kann innerhalb von verschiedenen Apps wiederverwendet werden. Die graphische Aufbereitung innerhalb einer GUI muss auf den jeweiligen Anwendungsfall angepasst werden.

7.3 Automatisierte Aktivitäten als Benutzeraufgaben

Im vorliegenden Kapitel wird das Interaktionsdesign (siehe Kapitel 2.1.7) zur automatisierten Bearbeitung von (mobilen) Benutzeraufgaben untersucht. In Kapitel 7.3.1 werden mobile Übertragungsdienste bzw. Techniken zur Bearbeitung von (Tasklisten-orientierten) Benutzeraufgaben vorgestellt (vgl. Apps zur Informationsverarbeitung, Kapitel 2.1.7). Verschiedene Darstellungen mobiler Tasklisten zeigen, welche Darstellung zur Bearbeitung von Benutzeraufgaben auf Smartphones geeignet ist (siehe Kapitel 7.3.2).

7.3.1 Dienste zur Bearbeitung von mobilen Benutzeraufgaben

Im vorliegenden Kapitel werden verschiedene Dienste zur informationstechnischen Unterstützung von mobilen Benutzeraufgaben auf mobilen Geräten betrachtet. Voraussetzung zur Nutzung der Dienste ist die Hardwareseitige Unterstützung (siehe Kapitel 2.1.4).

SMS

Die SMS (engl. *short message service*) kann als Mitteilungsdienst innerhalb eines mWfMS eingesetzt werden, um Benutzeraufgaben als SMS zu verteilen. Die Taskliste ist somit die SMS-Posteingangsliste. Die Liste der eingegangenen Nachrichten kann je nach Mobiltelefon nach Namen, Größe oder

Typ sortiert werden. Gewöhnlich wird die chronologische Reihenfolge als Listensortierung unterstützt. In vielen mobilen Endgeräten können die einzelnen Nachrichten in Verzeichnissen organisiert werden. Eine SMS ist auf 160 Zeichen pro Nachricht eingeschränkt. Eine Benutzeraufgabe kann daher nur „knapp“ beschrieben werden. Für Illustrationen oder Graphiken sind SMS nicht geeignet. In einer SMS kann nur Text übertragen werden. Die Kommunikation über SMS verläuft asynchron und ist robust gegenüber vorübergehenden Netzausfällen. SMS ist ein zentraler Dienst (innerhalb eines Mobilfunknetzes), der über ein sogenanntes *Short Message Service Center* (SMSC) des entsprechenden Providers betrieben wird [PSGS00]. Da SMS vorübergehende Netzausfälle erkennt und die Nachricht so lange übermittelt, bis das mobile Gerät diese erfolgreich empfängt, eignen sich SMS-basierte mWfMS in Bereichen schlechter Netzabdeckung.

Email

Email war zeitweise der am meiste genutzte Dienst im Internet [Sieg08]. Für mobile Endbenutzer, die über ein mobiles Gerät Emails lesen und beantworten möchten, stehen Email-Clients zur Verfügung. Der Email-Dienst besteht meist aus mehreren Übertragungsprotokollen [Dent04]: SMTP[14] und IMAP[15] bzw. POP3[16]. Zur Nutzung von Email auf dem mobilen Gerät ist ein Zugang zum Internet notwendig. Analog zum SMS-Posteingang listet ein Email-Client die Emails in chronologischer Reihenfolge auf und erlaubt es, diese Liste nach bestimmten Kriterien zu sortieren. Bei der Nutzung von IMAP wird lediglich der Email-Header heruntergeladen (beim Öffnen die komplette Email). IMAP ist für die mobile Nutzung geeignet, da kleinere Datenpakete transportiert werden und ein Verbindungsabbruch durch wiederholtes Senden besser kompensiert werden kann. Bei POP3 wird stets die

[14] Simple Mail Transfer Protocol (SMTP) ist ein Internet-Protokoll, das zum Austausch von E-Mails in Netzen dient [Dent04].

[15] Internet Message Access Protocol (IMAP) ist ein Anwendungsprotokoll das den Zugriff und das Management von empfangenen E-Mails erlaubt, die sich in einem Postfach auf einem Mailserver befinden [Dent04].

[16] Post Office Protocol (POP) ist ein Übertragungsprotokoll, über welches ein Client E-Mails von einem E-Mail-Server abholen kann. Version 3 (POP3) wird im RFC 1939 beschrieben [Dent04].

komplette Email heruntergeladen. Emails lassen sich in verschiedene Verzeichnisstrukturen sortieren und erlauben ein Management von z. B. relevanten Benutzeraufgaben. Email-Clients ermöglichen mehrere Email-Konten gleichzeitig zu verwalten, so dass beispielsweise ein mWfMS mit einem speziellen Email-Account eingerichtet werden kann. Einige Email-Clients bzw. Email-Server (z. B. Microsoft Exchange Server 2010) besitzen eine Benutzeraufgaben- bzw. Tasklisten-Unterstützung, um eine Kategorisierung und Bearbeitung der Benutzeraufgaben integrativ zu ermöglichen. Emails lassen sich leicht um Illustrationen und Graphiken ergänzen, sind jedoch nicht entworfen worden, um eine (effiziente) Datenübertragung zu gewährleisten [Sieg08].

WAP

Das *Wireless Application Protocol* (WAP) war eine der ersten Möglichkeiten, von einem mobilen Endgerät aus auf das Internet zuzugreifen [Dulz00]. WAP ist ein offener Standard für die Darstellung von Inhalten aus dem Internet, der speziell an die mobil-spezifischen Eigenschaften angepasst worden ist. Das WAP berücksichtigt …

… langsame Übertragungsraten,

… lange Antwortzeiten und

… kleine Displays.

Die Inhalte werden mit Hilfe der *Wireless Markup Language* (WML) dargestellt. Über einen Proxy-Server, das sogenannte WAP-Gateway, kommuniziert der Client (das mobile Gerät) mit dem Server. Über dieses Gateway werden die Daten von HTTP bzw. *Transmission Control Protocol/ Internet Protocol* (TCP/IP) in die WAP-Protokolle übersetzt [WAPF01]. Die Darstellung der Internetinhalte über WAP erfolgt über einen gewöhnlichen Browser. WAP garantiert über Komprimierungen (innerhalb des Protokolls) eine effiziente Datenübertragung [Dulz00].

Im Anwendungsbeispiel [ChCK02] erhält der Nutzer eine WAP-basierte Mitteilung, dass eine Bestellung eines Kollegen freigegeben werden muss.

Diese Freigabe kann direkt vom mobilen Gerät aus durchgeführt werden [ChCK02]. Das Anwendungsbeispiel [ChCK02] ist ein *hochentwickeltes* mWfMS (vgl. Kapitel 3.3).

MMS

Der *Multi Messaging Service* (MMS) ist die Weiterentwicklung der SMS, um multimediale Inhalte zu übertragen. Die Datenübertragung bei MMS erfolgt über das WAP. Bei MMS werden bei der Übertragung max. 300kB/s erreicht. Die Inhalte werden vor der Übertragung (über das WAP) komprimiert. Die Zwischenspeicherung wird endgeräteunabhängig über das *Multimedia Messaging Service Center* (einen Proxy-Server) durchgeführt. Eine Benutzeraufgabenverwaltung innerhalb einer Taskliste kann (analog zur SMS) innerhalb des Nachrichteneingangs geführt werden. Je nach mobilem Gerät, kann im Posteingang die MMS nach verschiedenen Kriterien sortiert werden. Durch die größere (multimediale) Ausdrucksmächtigkeit einer MMS (gegenüber einer SMS) können Benutzeraufgaben ergänzend aufbereitet werden. Illustrationen, Graphiken, Tonaufzeichnungen oder Videos können helfen, den Zusammenhang einer Benutzeraufgabe zu beschreiben.

Im Anwendungsbeispiel eines MMS-basierten mWfMS [Rein03] soll eine Pipeline instandgesetzt werden. Die Benutzeraufgabe mit Ortsbezug (Ort der Pipeline-Reparatur) ist innerhalb einer Karte visualisiert worden. Der Nutzer wird visuell zum entsprechenden Ort geführt, an dem sich die gebrochene Pipeline befindet [Rein03].

Apps

Benutzeraufgaben können als *native App*, als *Web App* oder als *hybride App* auf einer mobilen Endgeräteplattform abgebildet werden (siehe Kapitel 2.1.7). Innerhalb von Apps lassen sich Benutzeraufgaben bzw. Tasklisten auf unterschiedliche Art darstellen (siehe Kapitel 7.3.2). Voraussetzung ist der Zugang zum Internet/Intranet. Bei Apps wird das mobile Gerät als Mediator zwischen dem Menschen und anderen Informationsobjekten, die über das Internet angebunden werden, beschrieben [MaFl10]. Durch die starke Verbreitung der Smartphones, werden immer mehr Apps und Dienste über

das Internet genutzt. Die virtuellen App-Stores haben die Installation neuer Apps vereinfacht und damit einen neuen Markt geschaffen [HuHM10].

Die Darstellung einer Benutzeraufgaben *Web App* erfolgt innerhalb des mobilen Browsers. Die Benutzeraufgabe einer *Web App* kann nur bei bestehender Verbindung zum Internet (bzw. zum mWfM-Server) aktualisiert werden. Mit HTML5 [Hick11] können jedoch Daten innerhalb des Caches mitgeführt werden, so dass eine Nutzung unabhängig von der Internetverbindung möglich ist. Die graphische Aufbereitung der Benutzeraufgabe innerhalb einer *Web App* ist abhängig von der zur Verfügung stehenden Funktionalität des Browsers. Die Benutzeraufgabe kann auf den entsprechenden Anwendungsfall, jedoch nicht explizit auf den Nutzer, angepasst werden (im Gegensatz zur *nativen* oder *hybriden App*). Es können beispielsweise keine Gerätefunktionen genutzt werden. Die API der entsprechenden Plattform (*iOS, Android*) steht lediglich für eine native Implementierung bereit (siehe Kapitel 2.1.7).

Native mobile Apps können Endbenutzer-spezifisch implementiert werden, so dass die abgebildeten Benutzeraufgaben eine gute *Usability* erhalten (siehe Kapitel 2.1.7). Durch die Nutzungsmöglichkeit jeglicher zur Verfügung stehender Gerätefunktionen kann der Softwareingenieur die App so vorbereiten, dass beispielsweise unnötige Eingaben durch den Endbenutzer verhindert werden. Eine entsprechende Kontextanalyse auf dem mobilen Gerät kann z. B. die Steuerung der App übernehmen oder dem Nutzer zusätzliche Informationen zur Bearbeitung der Benutzeraufgabe bereitstellen. Die relativ hohen Kosten (im Vergleich zur *Web App*) bei der Entwicklung einer *nativen mobilen App* müssen berücksichtigt werden. Eine Kosten-Nutzen-Analyse sollte bei nativen Apps immer durchgeführt werden [Hugh10]. Eine *native App* wird für eine entsprechende Endgeräteplattform (in einer speziellen Programmiersprache) entwickelt. D. h. eine iOS-App wird nicht auf einem *Android*-Gerät funktionieren und umgekehrt. Vor der Implementierung sollte zusätzlich eine (Geräte- bzw.) Nutzergruppenanalyse durchgeführt werden, damit die App für die richtige Anwendergruppe realisiert wird (siehe Anhang B).

Hybride Apps sind eine Kombination aus *Web App* und *nativer App*. *Hybride Apps* werden innerhalb einer Plattform-spezifischen Sprache entwickelt. Bei

der Anwendung der App werden jedoch Teile der App (der sogenannte *Content*) über Webtechnologien (nach-)geladen. Der erstellte *Content*, der meist über einen sogenannten Webserver bereitgestellt wird, kann in den verschiedenen Plattformen wiederverwendet werden (siehe Kapitel 7.2.3). Die native Basis der App wird explizit auf die Benutzeraufgabe und die Plattform bzw. das mobile Gerät angepasst. Das Verhältnis zwischen nativen und webbasierten Anteilen innerhalb der App kann variabel gestaltet werden. *Hybride Apps* können (wie native Apps) über die Plattform-abhängigen App Stores heruntergeladen und installiert werden.

Fazit

In Tab. 7.2 werden die Dienste zur Bearbeitung mobiler Benutzeraufgaben nach ihren wichtigsten Kriterien bzw. in Bezug auf die Verwendung innerhalb eines mWfMS ausgewertet:

Tab. 7.2: Auswertung der Dienste zur Bearbeitung mobiler Benutzeraufgaben

	SMS	**MMS**	**Email**	**WAP**	**App**
Mitteilungsdienst	X	X	X	-	-
Bilder/ Graphik	-	X	X	X	X
Serverbetrieb (Cloud-, Web-, Mailserver)	-	-	X	X	X (bei Hybride App und Web App)
Mobiles Internet	-	-	X	X	X
Kosten	gering	mittel	gering	moderat	hoch
Vorteile	Auf jedem Handy verfügbar	Multimedia-Inhalte über Mobilfunknetz	weite Verbreitung	Einfache Website-Gestaltung	Maximale Gestaltungsfreiheit

7.3.2 Ausführung von (mobilen) Benutzeraufgaben auf Smartphones

Die graphische Symbolik bei der Benutzeraufgabenverwaltung orientiert sich an Analogien, die eine bestimmte Bedeutung assoziieren. Zur Differenzierung verschiedener Kategorien der Tasklisten (z. B. innerhalb von Apps) werden unterschiedliche Farben verwendet. Bei Unterkategorisierungen werden etwas dunklere bzw. hellere Farbtöne angewandt, so dass der Nutzer die Unterkategorie zur „Ursprungsfarbe“ assoziiert [Clar10]. Die Signalfarbe rot wird für besonders wichtige Aufgaben verwendet. Gelb symbolisiert meist einen Übergangsbereich (die Bedeutung steigert sich). Grün hingegen symbolisiert keine Handlungsaufforderung, sondern eine neutrale Haltung. Der Einsatz von Farben innerhalb mobiler Apps dient dem Hervorheben oder Verbergen von wichtigen oder unwichtigen Merkmalen der Benutzeraufgabe. Analogien zum Straßenverkehr (z. B. Schilder, Ampeln, Zebrastreifen) finden oft Anwendung innerhalb der Symbolik [Clar10]. Mit den drei „Ampelfarben“ wird beispielsweise der Status einer Benutzeraufgabe gekennzeichnet. Ein Stoppschild kann z. B. kennzeichnen, dass eine Aufgabe nicht weiterbearbeitet werden sollte (siehe Abb. 7.7).

Aufgabenlisten

Benutzeraufgaben werden innerhalb von Aufgabenlisten (Tasklisten) zur Bearbeitung bereitgestellt. Über einen Dienst (siehe Kapitel 7.3.1) können auf dem mobilen Gerät verschiedene Tasklisten gefüllt werden. Zur Darstellung von Aufgabenlisten existieren verschiedene Methoden, die je nach Anwendungsfall (Kontext des Workflows) dem Nutzer bei der Bearbeitung der Aufgabe helfen. Eine Aufgabenliste kann z. B. nach einer bestimmten Merkmalsausprägung (z. B. Fälligkeitsdatum) sortiert werden, um dem Nutzer des mobilen Gerätes nur die fälligen Benutzeraufgaben anzuzeigen. Wenige mWfMS liefern eine Priorisierungsverwaltung von Benutzeraufgaben (siehe Kapitel 3.2.1 und Kapitel 3.2.2). Zur Tasklistenverwaltung auf mobilen Geräten ist es wichtig, eine Listensortierung nach mehreren Merkmalen anzubieten, damit diese Merkmalskombinationen verknüpft werden können (z. B. Aufgaben mit hoher Priorität in einer bestimmten Umgebung).

Die Benutzeraufgabe wird im vorliegenden mWfMS-Anwendungsbeispiel innerhalb eines *Screen* dargestellt (siehe Abb. 7.7, „*Screen 3*“). Die an den Nutzer adressierten Aufgaben werden innerhalb einer Aufgabenliste verwalten (siehe Abb. 7.7, „*Screen 2*“).

Abb. 7.7: mWfMS *Tasklister Lite*

Dem Nutzer der App *Tasklister Lite* (siehe Abb. 7.7, „*Screen 1*“) werden mehrere Tasklisten bzw. Kategorisierungen angeboten, um die Benutzeraufgaben zu verwalten (z. B. Liste offener Benutzeraufgaben (*Pending)*, siehe Abb. 7.7, „*Screen 2*“). Zur Konfiguration der Merkmale der Benutzeraufgabe (Status, Wichtigkeit, etc.) dient ein weiterer *Screen* (siehe Abb. 7.7, „*Screen 3*“). Die Benutzeraufgaben werden in einer Liste nach der jeweiligen Kategorisierung aufgelistet (siehe Abb. 7.7). Das rote Ausrufezeichen markiert, dass eine Aufgabe umgehend bearbeitet werden muss. Im Anwendungsbeispiel werden die einzelnen Aufgabenlisten differenziert nach *Overdue* (überfällig), *Important* (wichtig), *Pending* (bevorstehend) und *Completed* (abgeschlossen) aufgeführt. In der Liste *All* (alle) werden alle Aufgaben zur Übersicht dargestellt. Zusätzlich kann mit Hilfe der App nach Aufgaben

bzw. deren Beschreibung gesucht werden (*Find tasks, „Screen 1“*). Die Farbunterstützung der Benutzeraufgaben im Anwendungsbeispiel erfolgt mit den „Ampelfarben“ grün, gelb und rot (siehe Abb. 7.7). Die Liste *Overdue* wird z. B. gelb (mit Warnhinweis) dargestellt, um die Dringlichkeit zu verdeutlichen. Der rote Kreis mit weißem Ausrufezeichen (siehe Abb. 7.7) hebt die Wichtigkeit der Aufgabe hervor. Ein grüner Punkt mit weißem Häkchen kennzeichnet dahingegen, dass eine Aufgabe abgeschlossen bearbeitet wurde (siehe Abb. 7.7).

Kacheln als Kategorisierung von Benutzeraufgaben

In der Kacheldarstellung werden Benutzeraufgaben innerhalb von Verzeichnissen verwaltet. Über das Verzeichnis gelangt der Nutzer meist direkt zur Listendarstellung. Die Kacheln werden u. a. verwendet, um z. B. Apps innerhalb des Betriebssystems anzuzeigen (siehe z. B. *iOS*, *Android*). Bei der initialen Kategorisierung von Benutzeraufgaben wird die Kacheldarstellung angewandt (siehe Abb. 7.8). Die verschiedenen Kategorien werden durch sogenannte *Icons* (kleine aussagefähige Bildchen oder Symbole) abgebildet, die wiederum als Navigationsobjekte (bzw. Verzeichnisse) dienen.

Abb. 7.8: Screenshots von Applikation *eTodo* und *Awesome Note*

Die Anwendungsbeispiel-Apps *eTodo* und *Awesome Note* verwenden Kacheln, um zur Darstellung der Aufgabenliste zu gelangen (siehe Abb. 7.8). Es werden für den Nutzer verschiedene Kategorien angeboten. Hinter den Kategorien (Kacheln) befinden sich Aufgabenlisten. Die Kategorien sind Verzeichnisse, die in Analogie zu einem „echten Büro" als Ordner, Korb oder Box dargestellt werden. Mit dem Symbol „Inbox" (Abb. 7.8, „*Screen 1*") wird beispielsweise ein Eingangspostfach für Benutzeraufgaben abgebildet. Unter „Completed" werden die erledigten Aufgaben gespeichert. Die Symbolik des Hakens für „Completed" betont graphisch, dass sich in diesem Verzeichnis abgeschlossene Benutzeraufgaben befinden.

Der Nutzer hat in der App „Awesome Note" die Möglichkeit, eine Aufgabenliste je nach Aufgabentyp individuell zu gestalten. Die Benutzeraufgaben werden in besonders markierten Listen als Kacheln dargestellt, die an einen Notizzettel erinnern sollen (siehe Abb. 7.8). Wichtigkeit oder Prioritäten werden mittels Bewertungsschemas definiert (siehe Abb. 7.8).

Benutzeraufgaben mit Ortsbezug innerhalb von Karten

Die Darstellung von Benutzeraufgaben als z. B. Stecknadeln in einer Landkarte, unterstützen Nutzer, die sich innerhalb eines (mobilen) Anwendungsfalls befinden (vgl. Kapitel 7.2.3). Für die geographische Darstellung innerhalb einer Karte werden Ortsangaben für die Benutzeraufgaben benötigt. Mit dieser Darstellungsform werden Arbeitszeit und damit Kosten eingespart, da die Benutzeraufgaben innerhalb der Karte integriert dargestellt werden [LeAH08]. Die geographische Verteilung der Benutzeraufgaben wird in einer Land- oder Stadtkarte visualisiert, um bestimmte Aufgaben (z. B. Kundentermine) anzuzeigen. Farben helfen, um z. B. unterschiedliche Aufgabenkategorien in der Karte zu markieren. Zur eigentlichen Bearbeitung der Benutzeraufgabe ist die Kartendarstellung nicht geeignet. Die Karte gewährleistet dem Nutzer gegenüber einen „geographischen Überblick", der zur Navigation bzw. Orientierung benutzt werden kann. Zeitliche Abhängigkeiten können beispielsweise mittels eines Pfades (Wegeplans) innerhalb der Karte dargestellt werden. Über eine Navigationsunterstützung innerhalb der App kann die Orientierung zusätzlich erleichtert werden. Bei der Wegeplanung

können Benutzeraufgaben, die zeitlich nicht mehr in die Planung integrierbar sind, an andere Mitarbeiter adressiert werden [JHHS00].

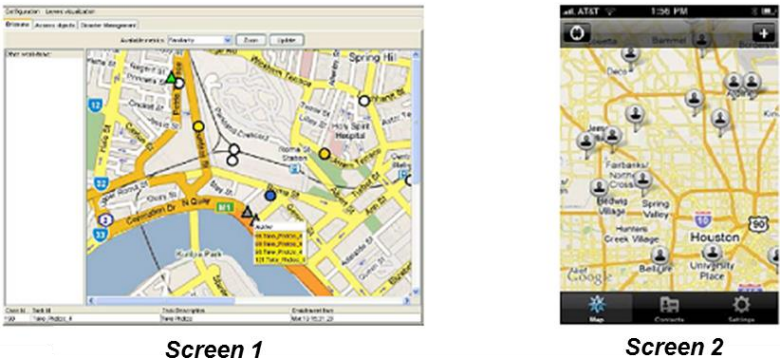


Abb. 7.9: Geographische Darstellung von Benutzeraufgaben, „Map Contacts“

Mit den Kreisen bzw. Punkten werden die Benutzeraufgaben in den Anwendungsbeispielen (siehe Abb. 7.9) markiert. Die Dreiecke (siehe Abb. 7.9, „*Screen 1*“) symbolisieren spezielle Ressourcen, die zu Bearbeitung der Benutzeraufgabe benötigt werden [LeAH08]. Die Stecknadelköpfe visualisieren im zweiten Anwendungsbeispiel Kundentermine (Abb. 7.9, „*Screen 2*“), so dass der Nutzer direkt aus der Kartenansicht die Ausprägungen der Benutzeraufgabe sehen kann.

Benutzeraufgaben in Kalendern

Kalender sind Werkzeuge für die zeitliche Planung von Terminen und Ressourcen. Die Benutzeraufgaben werden als Termine innerhalb des Kalenders (der Tages-, Monats- und Jahresansicht) integriert (siehe Abb. 7.10).

Die Beispiele (siehe Abb. 7.10) zeigen unterschiedliche farbige Markierungen, um die Benutzeraufgaben voneinander unterscheiden zu können (z. B. anhand von Prioritäten). Eine Monatsansicht der App „Awesome Note“ (siehe Abb. 7.10, „*Screen 1*“) und eine Tagesansicht der App „myTimes“

(siehe Abb. 7.10, „*Screen 2*“) sind Anwendungsbeispiele der (integrierten) Benutzeraufgaben.

Abb. 7.10: Benutzeraufgaben in Kalendern: „Awesome Note“, „myTimes“.

Benutzeraufgaben im Prozessmodell

Benutzeraufgaben könnnen innerhalb eines Prozessmodells dargestellt werden, um verschiedene Zusammenhänge übersichtlich darzustellen bzw. zu planen. In einem Prozessmodell werden u. a. Ressourcen, Rollen und Beziehungen modelliert (siehe Kapitel 2.3). Die Darstellung von Benutzeraufgaben innerhalb von Workflow-Modellen kann Nutzern beim Bearbeiten der Aufgabe helfen [KoRR12]. Durch die Darstellung des Kontextes, in dem die Benutzeraufgabe bearbeitet werden soll, kann der Nutzer die Aufgabe kreativ und ggfs. effizienter lösen. Die übersichtliche Darstellung von Prozessmodellen auf mobilen Geräten ist maßgeblich von der Displaygröße abhängig (siehe Abb. 7.11). Smartphones eigenen sich auf Grund des kleinen Bildschirms (<5“) beispielsweise weniger, um viele Kontexte gleichzeitig darzustellen. Tablets hingegen besitzen eine Displaygröße, die sich zur übersichtlichen Darstellung von Prozessmodellen und ihren Kontexten eignet (siehe

Kapitel 2.1.5). Falls das Prozessmodell dennoch auf einem Smartphone dargestellt wird, können Techniken wie z. B. das *Pinch-to-Zoom*[17] helfen, eine bessere *Usability* gegenüber dem Nutzer zu erzielen (siehe Abb. 7.11).

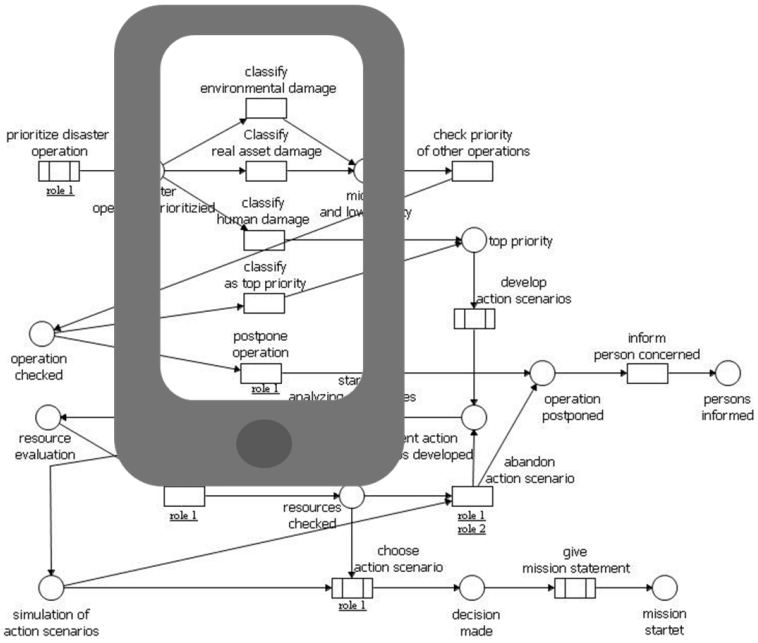


Abb. 7.11: Aufgaben in Workflow-Modelldarstellung

Farbliche Markierungen von bestimmten Modellobjekten verbessern die Übersicht im Modell. Die Überschaubarkeit des Workflow-Modells hängt zusätzlich von der Anzahl der Modellobjekte ab. Je mehr Objekte im Workflow-Modell enthalten sind, desto schneller verliert der Nutzer den Überblick, da nur Teilbereiche des Modells auf einmal betrachten werden können (siehe Abb. 7.11).

[17] Patentierte Multitouch-Funktion der Firma Apple, um mit zwei Fingern den Inhalt eines Screens bzw. ein Workflow-Modell auseinanderzuziehen oder zu verkleinern.

Benutzeraufgaben mit *Augmented Reality*

Augmented Reality (AR) ist eine bekannte und seit langem angewandte Technik [Suth68, SMRL97, FMHW97, HFTR99]. *Augmented Reality* bedeutet so viel wie „angereicherte Realität“ oder „erweiterte Wirklichkeit“. Bei AR werden zum Kamerabild ergänzende Kontextinformationen in Echtzeit (z. B. Video, Audio, Aufgabenliste) angeboten [HöFe04]. Während in [Suth68] jeweils eine spezielle Apparatur mit Brille notwendig war, ist AR heutzutage auf jedem Smartphone nutzbar [MoBD12]. Die jeweiligen Plattformhersteller bieten mittlerweile den App-Entwicklern die Nutzung von AR über das zur Verfügung *gestellt Software Development Kit* (SDK) an.

Benutzeraufgaben (bzw. Informationen, die mit diesen Aufgaben im Zusammenhang stehen) werden zum Kamerabild in Echtzeit eingeblendet. Voraussetzung für die Nutzung von AR-Benutzeraufgaben sind Ortsinformationen. Es ist es notwendig, Ortsinformationen an die Benutzeraufgabe zu binden (vgl. Kartendarstellung). Die Darstellung von Benutzeraufgaben innerhalb einer AR-App auf dem Smartphone eignet sich nicht für jeden Anwendungsfall, da der Nutzer zum aktuellen Kamerabild seines Smartphones die Benutzeraufgabe anhand der Ortsinformation angezeigt bekommt. Befindet sich der Nutzer in großer Distanz zur Ortsangabe der Benutzeraufgabe, wird diese beispielsweise (meist) nicht eingeblendet. Gerade bei AR ist die Darstellung von Farben und Symbolen besonders wichtig, da der Nutzer (wie beispielsweise bei einer Navigations-App) gleichzeitig die Umgebung wahrnehmen muss.

Im Hintergrund der beiden AR-Anwendungsbeispiele (siehe Abb. 7.12) ist das Kamerabild des mobilen Geräts zu sehen, das mit Texten und Symbolen annotiert ist. Die Ausrichtung des Gerätes wird jeweils zur Orientierungsunterstützung des Nutzers eingeblendet. Das Symbol (siehe Abb. 7.12, rechts oben in „*Screen 1*“) ist als Analogie zu einem Kompass entworfen worden.

Abb. 7.12: AR-Benutzeraufgaben: Screenshots O2-Werbung, "Layar"-App

„Layar“ (siehe Abb. 7.12, „*Screen 2*“) ist eine App für Wohnungssuchende, die zum Kamerabild in Echtzeit die aktuellen Immobilienangebote einsehen können. Die Größe der Kreise zeigt die relative Entfernung vom aktuellen Ort ausgehend an (siehe Abb. 7.12, „*Screen 2*“). In dieser App ist ein kleiner Kompass als Orientierungsunterstützung eingeblendet. Die aktuellen Wohnungstreffer werden im virtuellen Kompass zweidimensional dargestellt (siehe Abb. 7.12, kleine blauen Punkte in „*Screen 2*“).

Fazit

Für die Bearbeitung von mobilen Benutzeraufgaben (durch einen Nutzer bzw. Akteur) werden Tasklisten-orientierte mobile Dienste eingesetzt (siehe Kapitel 7.3.1). Mobile Akteure sind z. B. Handelsvertreter, Handwerker, Software-Entwickler etc. Zur Bearbeitung einer mobilen Aufgabe auf einem mobilen Endgerät bzw. Smartphone können verschiedene Darstellungstechniken verwendet werden (siehe Kapitel 7.3.2).

Die Visualisierung von Benutzeraufgaben ist immer vom zugehörigen Workflow bzw. dem Anwendungsfall, vom Nutzer, vom Endgerät und vom verwendeten Dienst (siehe Kapitel 7.3.1) abhängig. Ein Kontext-abhängiges Interaktionsdesign innerhalb der App kann einen Anwendungsfall besser un-

terstützen [AnD'I02]. Bei oft wiederkehrenden Arbeitsabläufen ist eine Abarbeitungsroutine in Form einer Vorlage bzw. eines geführten Arbeitsablaufes geeignet. Benutzeraufgaben können innerhalb von Karten dargestellt werden, sofern die Ortsinformationen der jeweiligen Benutzeraufgaben (bzw. Aktivitäten) vorliegen [LeAH08, BrPa05].

Eine Kombination aus mehreren Darstellungen (engl. *Screens*), verknüpft über einen innerhalb der App intuitiv gestalteten einfachen Wechsel (Interaktionsdesign), ermöglicht dem Nutzer Flexibilität bei der Bearbeitung der Benutzeraufgaben. Parallel zu einer Kartendarstellung, innerhalb derer die Benutzeraufgaben angezeigt werden, kann z. B. eine Aufgabeliste in einem weiteren *Screen* angeboten werden, um zu den Details der Benutzeraufgabe zu gelangen. Der Nutzer soll selbst auswählen können, welche Ansicht ihm mehr Überblick bzw. Vorteile bietet. Dadurch kann einerseits die Kontextanalyse innerhalb der App eingeschränkt werden und andererseits wird die Benutzerführung nicht als zu eng empfunden.

Letztendlich sollte eine App immer bzgl. der *Usability* evaluiert werden, damit eine möglichst gute Unterstützung der Benutzeraufgabe gewährleistet wird. Die Benutzeraufgabe sollte innerhalb der App so bearbeitet werden können, dass der Nutzer einen Mehrwert gegenüber konventioneller (mobiler) Aufgabenbearbeitung (z. B. mit Notizbuch) erkennt.

8 Zusammenfassung und Ausblick

In diesem Kapitel werden die Erkenntnisse dieser Arbeit zusammengefasst und weitere forschungsrelevante Themen in Form eines Ausblicks auf weiterführende Arbeiten beschrieben.

8.1 Zusammenfassung

In vorliegender Arbeit sind Ortsbezüge respektive ortsbasierte Kontextinformationen in Form von sogenannten Ortseinschränkungen innerhalb von Geschäftsprozessmodellierungssprachen integriert worden. Die Spracherweiterung bzw. das ganze Konzept ist innerhalb von bestehenden WfMS erstmals erfolgreich erprobt worden. Die Geschäftsprozesse, die von Unternehmen bzw. Organisationen verwenden werden, werden mit Kontextinformationen gesteuert, indem sie über sogenannte Workflows innerhalb eines mobilen Workflow-Managementsystems (mWfMS) ausgeführt werden. Mit den OE erhalten Modellierer bzw. Systemdesigner von WfMS ein neues Konzept, um ortsbasierte Kontextinformationen auszuwerten bzw. abhängig davon die Workflow-Modelle zu steuern.

Für die Integration von Ortsbezügen in Workflow-Modelle wurde ein Konzept auf Basis von Ortseinschränkungen entwickelt. Ein Ortsmodell wurde definiert, um die Annotation auf mehrere Workflow-Sprachen anwenden zu können. Die graphische Annotation der Ortsbezüge wurde unter Berücksichtigung unterschiedlicher Sprachaspekte (z. B. Aggregationen, Hierarchische Modellelemente, Verknüpfungen) entworfen. Eine Erweiterung der Petri-Netze und der BPMN 2.0 zeigt, dass dieses Konzept in verschiedenen Sprachen innerhalb des Geschäftsprozessmanagements integriert werden kann. Das entwickelte Konzept der Ortseinschränkungen kann mit den eingeführten OCL-Definitionen (siehe Kapitel 4.2, 5.2) innerhalb von Programmiersprachen angewandt werden, indem mittels Sprachtransformation (z. B.

in Java[1]) eine Übersetzung in die jeweilige Zielsprache durchgeführt wird. Damit kann das Konzept der Ortsbezüge in weitere bestehende Modellierungssprachen integriert werden.

Das Referenzmodell der WfMC (siehe Kapitel 3.3) ist zu einem mobilen WfMS erweitert worden, um die Workflow-Modelle mit Ortsbezügen innerhalb einer Workflow-Engine automatisiert auszuführen. Die Workflow-Modelle wurden auf einem mWfMS, das für die Anwendungsfälle ausreichend generisch und erweiterbar war, basierend auf einer SOA ausgeführt. Das entwickelte *Advanced* mWfMS ist innerhalb von mehreren Unternehmen mit unterschiedlichen mobilen Anwendungsfällen im Verbundprojekt R2B erfolgreich getestet worden (vgl. Kapitel 7.1). Verschiedene Anwendungsfälle zeigen, wie durch die Integration der Ortsbezüge und weiterer Kontextinformationen (vgl. Kapitel 7.1.3) das mWfMS angereichert wurde, um die Workflow-Modelle Kontext-basiert zu steuern. Ferner wurde eine Workflow-Steuerung auf Basis eines Regel- bzw. Schwellwert-basierten Mechanismuses in Feldversuchen erfolgreich erprobt (siehe Kapitel 7.1.5). Die Implementierungen bzw. Erweiterungen des WfMS verdeutlichen, dass die vorgestellten Konzepte und Methoden in weitere bestehende WfMS integrierbar sind. Damit zeigt die vorliegende Arbeit, dass bestehende WfMS zu mWfMS erweitert werden können (vgl. Abb. 1.1).

Die Kontext-basierte Erweiterung des mWfMS zeigt, dass Kontextinformationen ganzheitlich erfasst und ausgewertet werden müssen, damit Mehrwerte für den Nutzer bzw. das Unternehmen entstehen. Die Auswertung und Aufbereitung der Kontextinformationen[2] wird zur maßgebenden Funktionalität eines mWfMS, wenn die Worklflows davon abhängig ausgeführt bzw. gesteuert werden. Die praktischen Versuche zeigen, dass sich das Konzept der Ortsbezüge innerhalb von Workflow-Modellen mit vielen geographischen Beziehungen besonders eignet. Gerade in der Landwirtschaft, im Baubetrieb, in der Logistik, im Tourismus, in der Automobilindustrie und in vielen weiteren Branchen existieren diese Geschäftsprozesse mit Ortsbezügen.

[1] http://www.dresden-ocl.org

[2] Durch die Daten-Sammlung innerhalb des mWfMS müssen besondere Schutzmaßnahmen vor Datendiebstahl und –missbrauch getroffen werden.

Mit dem vorgestellten Konzept ist es möglich, diese geographischen Beziehungen innerhalb von Workflow-Modellen abzubilden und automatisiert innerhalb von mWfMS auszuführen. Im Umfeld betrieblicher Softwareentwicklung birgt das Konzept enormes Marktpotential, wenn die genannten Branchen diese Erkenntnisse der vorliegenden Arbeit nutzen und in ihren betrieblichen Abläufen integrieren. Die Unternehmen können mit einer solchen Softwareunterstützung bzw. der Automatisierung von geschäftlichen Abläufen Ressourcen einsparen und potentiell neue Märkte erschließen (siehe Kapitel 5.2.2).

Einfache ortsbasierte mobile Anwendungsfälle werden mit einer mobilen Applikation auf einer mobilen Geräteplattform (vgl. iOS, Android) umgesetzt. Solche *Einfachen* mWfMS (siehe Kapitel 3.3) wurden innerhalb von Experimenten erprobt. Die Erkenntnisse daraus lassen sich zum Software-Engineering mobiler Applikationen nutzen. Die Experimente im Umfeld mobiler Softwareentwicklung zeigen, dass mobile Apps auf Basis eines gemeinsamen Datenmodells entwickelt werden können. Eine mobile App kann z. B. mit solch einem zentralen Datenmodell verknüpft werden. Die WfMS sind in vielen Unternehmen bereits heute im Einsatz. Es bedarf somit lediglich einer Systemerweiterung, um die Ortsbezüge innerhalb der betrieblichen Workflows zu nutzen und aus einem WfMS ein mWfMS zu erweitern. Die mobile App hingegen, muss (bis auf das Datenmodell) auf den jeweiligen betrieblichen Anwendungsfall angepasst werden. Abhängig vom mobilen Gerät (bzw. der Displaygröße) müssen die graphischen Elemente der App so angeordnet werden, dass die Nutzung nicht als unnötiger Mehraufwand wahrgenommen wird.

8.2 Ausblick

Weiterführende Arbeiten bei der Modellierung und Ausführung von Workflows unter Berücksichtigung mobiler Kontextinformationen ergeben sich durch die Erforschung weiterer Werkzeugunterstützung und durch die Unterstützung weiterer Kontextinformationen innerhalb des mWfMS. Folgende potentielle Themen ergeben sich auf Basis der vorliegenden Arbeit:

- Die Integration Sekundärkontexten (siehe Abb. 2.6) in Workflow-Sprachen sollte untersucht werden, um eine kontextabhängige Steuerung von Workflows zu gewährleisten. Für diese Integration müssen geeignete Modellierungsformen und unterstützende Komponenten (des WfMS) entworfen und evaluiert werden.

- Die Modellierungswerkzeuge sollten die „*Correctness by Construction*"-Vorgehensweise unterstützen, damit nur valide Modellierungen erlaubt werden [DRRG09]. Die Modellierungswerkzeuge verhindern dadurch einen fehlerhaften Einsatz der Sprachelemente der Modellierungssprache und der Ortsbezüge. Während der Modellierung muss hierzu (ständig) das Metamodell überprüft werden, damit basierend auf diesen Informationen eine falsche Anwendung des Metamodells vermieden wird bzw. verschiedene Hilfestellungen angeboten werden können.

- Das Modellierungswerkzeug sollte um weitere Funktionalitäten des Ortsmodells erweitert werden, um z. B. das Gebiet innerhalb einer Karte darzustellen, auf das die Ausführung einer Aktivität eingeschränkt ist (siehe Abb. 7.2). Auf Basis dieser Kartendarstellung sollte die Integration von weiteren Kontexten (Umgebungskontext, Benutzerkontexte) untersucht werden.

- Die entworfene mWfMS-Architektur sollte in weiteren Domänen mit weiteren unterschiedlichen Geschäftsprozessen evaluiert werden. Dadurch kann die vorgestellte Architektur (vgl. Kapitel 6), weiter ausgebaut werden und für viele weitere Domänen einen Mehrwert liefern.

- Die in vorliegender Arbeit präsentierte Verknüpfung von Regel-Engine und Prozess-Engine sollte erweitert werden: Applikationsserver, die mit Regel-Engines (z. B. Drools[3]) ausgeliefert werden, zeigen, dass die Verknüpfung von Regel-Engine und Prozess-Engine in vielen betrieblichen Anwendungsfällen benötigt wird. Die Abbildung von Regeln innerhalb der Prozesse sah in vorliegender Arbeit jeweils nur vier primitive Operationen bzgl. der Prozessbeeinflussung vor (siehe Kapitel 6.2). Die Rückrichtung, also die Beeinflussung von Regeln basierend auf Prozessen, wurde nicht realisiert. Mit dieser weiteren Funktionalität könnten weitere komplexe Anwendungsbeispiele unterstützt werden bzw. die Prozess-Engine (je nach Anwendungsfall) entlastet werden.

[3] http://www.jboss.org/drools

Anhang A: Nutzerevaluation

Untersuchungsbericht zum Produkt

„Annotation von Ortsbezügen in Geschäftsprozessmodellen“

Untersuchte Fragestellung: Wie bedienbar und attraktiv wird das Produkt wahrgenommen?

Inhalt des Berichts

- Untersuchungsmethode
- Kenndaten der Untersuchung
- Ergebnisüberblick - Portfolio
- Das Diagramm der Mittelwerte
- Das Profil der Wortpaare
- Konfidenzintervalle

Untersuchungsmethode

AttrakDiff[1] ist ein Instrument zur Messung der Attraktivität interaktiver Produkte.

Nutzer (oder potenzielle Nutzer) ihres Produkts geben mit Hilfe gegensätzlicher Adjektivpaare an, wie sie das Produkt wahrnehmen. Diese Adjektivpaare lassen sich den untersuchten Beurteilungsdimensionen zuordnen.

Beurteilt werden folgende Dimensionen des Produkts:

Pragmatische Qualität (PQ): Sie beschreibt die Benutzbarkeit eines Produktes, und verdeutlicht, wie gut der Nutzer seine Ziele mit Hilfe des Produkts erreichen kann.

[1] http://attrakdiff.de/

Hedonische Qualität - Stimulation (HQ-S): Menschen haben das Bedürfnis, sich weiterzuentwickeln. Die Dimension bildet ab, inwieweit ein Produkt diese Entwicklung unterstützen kann, indem es neuartige, interessante und anregende Funktionalitäten, Inhalte, Interaktions- und Präsentationsstile bietet.

Hedonische Qualität - Identität (HQ-I): Sie beschreibt, inwieweit ein Produkt seinem Nutzer ermöglicht, sich mit dem Produkt zu identifizieren.

Attraktivität (ATT): Sie beschreibt eine globale Bewertung des Produkts auf der Basis der wahrgenommenen Qualität.

Die Pragmatische und hedonische Qualität sind unabhängig voneinander und tragen in etwa gleichstark zum Attraktivitätsurteil bei.

Kenndaten der Untersuchung

Anzahl Bewertungen: 8 Probanden (aus den Anwendungsdomänen, siehe Kapitel 6.1)

Annotation von „Ortsbezügen in Geschäftsprozessmodellen“ Einzelbewertung, d. h. jeder Teilnehmer gibt genau eine Bewertung ab.

Ergebnisüberblick – Portfolio

Abb. A.1: Portfolio mit der durchschnittlichen Ausprägung der Dimensionen PQ und HQ und dem Konfidenz-Rechteck des Produkts „Annotation von Ortsbezügen in Geschäftsprozessmodellen“

P

Mittlere Ausprägung der Dimensionen bei Produkt „Annotation von Ortsbezügen in Geschäftsprozessmodellen“

In der Portfolio-Darstellung ist vertikal die Ausprägung der hedonischen Qualität zu sehen (unten = geringe Ausprägung). Horizontal ist die Ausprägung der pragmatischen Qualität zu sehen (links = geringe Ausprägung).

Je nach Ausprägung der beiden Dimensionen, fällt das Produkt in einen oder mehrere "Charakterbereiche".

Je größer das Konfidenz-Rechteck ausfällt, desto geringer ist die Sicherheit, mit der das Produkt einem bestimmten Bereich zugeordnet werden kann. Ein kleines Konfidenz-Rechteck ist von Vorteil, da dies bedeutet, dass die Untersuchungsergebnisse mit höherer Sicherheit auf das Produkt zutreffen und weniger zufällig sind.

Zudem spiegelt das Konfidenz-Rechteck auch wieder, wie „einig" sich die Nutzer bei der Beurteilung des Produkts sind. Je größer das Konfidenz-Rechteck ist, desto unterschiedlicher wird das Produkt bewertet (weitere Details finden sich im Anhang).

Interpretationshilfe

Die Benutzeroberfläche des Produkts wurde als „eher selbstorientiert" eingestuft.

Diese Zuordnung ist für die pragmatische Qualität nicht eindeutig, da das Konfidenzintervall über den Charakterbereich hinausgeht. Der Nutzer wird durch das Produkt zwar unterstützt, allerdings erreicht die Ausprägung der pragmatischen Qualität bei dem Produkt lediglich mittlere Werte.

Auswertung bzw. Fazit von AttrakDiff:

Für die hedonische Qualität trifft die Charakterzuordnung nicht eindeutig zu, da das Konfidenzintervall über den Charakterbereich hinausgeht. Der Nutzer wird durch das Produkt zwar angeregt, allerdings erreicht die Ausprägung der hedonischen Qualität bei dem Produkt lediglich mittlere Werte. Es besteht somit noch Verbesserungspotenzial hinsichtlich der Bedienbarkeit und der hedonischen Aspekte.

Das Konfidenzintervall PQ ist groß. Dies kann auf eine geringe Stichprobengröße oder sehr unterschiedliche Beurteilungen des Produkts zurückgeführt werden.

Das Diagramm der Mittelwerte

Im Diagramm der Mittelwerte sind die mittleren Ausprägungen der Dimensionen des AttrakDiff bei dem untersuchten Produkt dargestellt.

In dieser Darstellung wird die hedonische Qualität zusätzlich nach den Gesichtspunkten Stimulation und Identität differenziert. Außerdem wird auch die Beurteilung der Attraktivität dargestellt.

Interpretationshilfe

Hinsichtlich der hedonischen Qualität – bzgl. Identität und Stimulation befindet sich das Produkt im überdurchschnittlichen Bereich.

Der Attraktivitätswert des Produkts befindet sich im durchschnittlichen Bereich.

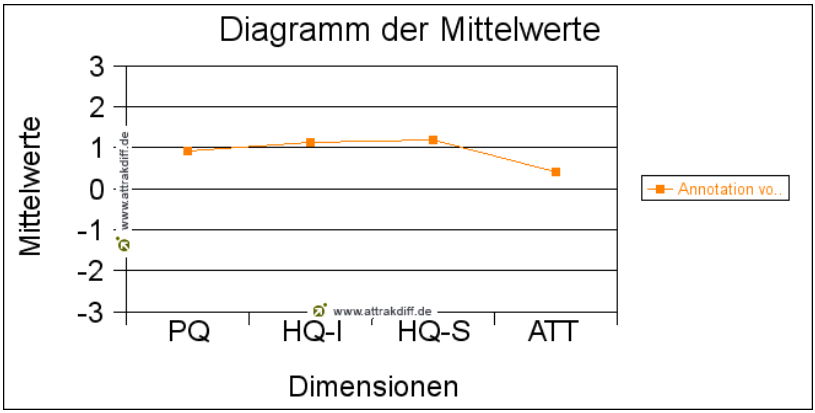


Abb. A.2: Mittlere Ausprägung der vier Dimensionen des AttrakDiff für das Produkt „Annotation von Ortsbezügen in Geschäftsprozessmodellen“

Insgesamt wirkt das Produkt auf die Nutzer mittelmäßig attraktiv.

Das Profil der Wortpaare

Im Profil der Wortpaare sind die mittleren Ausprägungen der einzelnen Wortpaare des AttrakDiff für das untersuchte Produkt dargestellt. Hier sind vor allem Extremwerte interessant. Sie zeigen, welche Eigenschaften besonders kritisch sind, oder besonders gut gelöst sind.

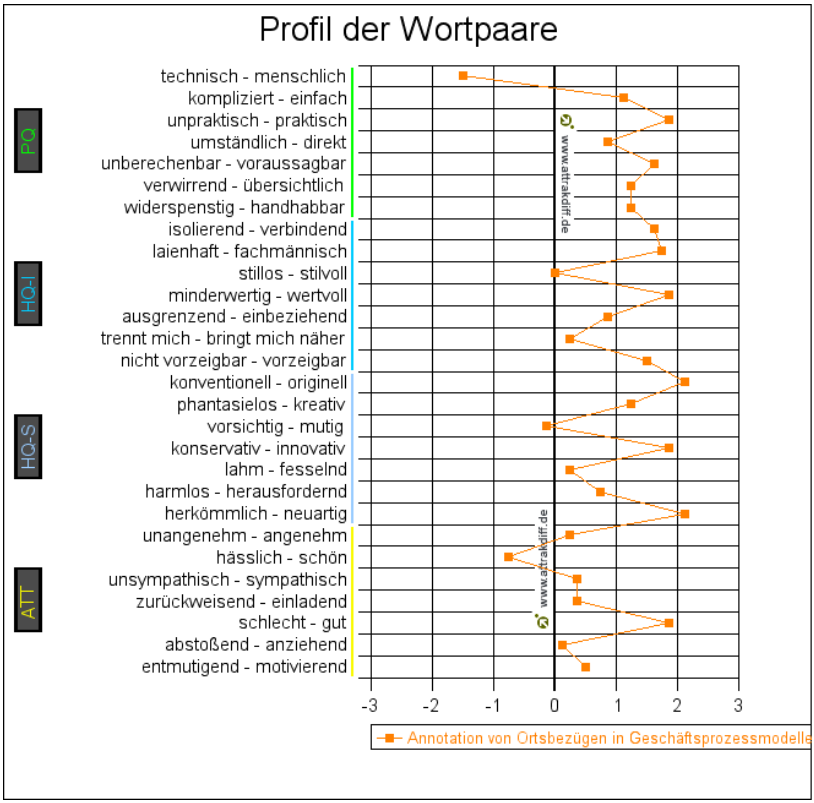


Abb. A.3: Mittlere Ausprägung der Wortpaare des AttrakDiff für das Produkt „Annotation von Ortsbezügen in Geschäftsprozessmodellen“

Anhang B:
Mobile Endbenutzergruppen

Innerhalb eines mWfMS können mobile Apps als *Worklist Handler*, als *Client Apps* oder als (entkoppelte) Methode über den sogenannten *Toolagent* bzw. die *Invoked Applications* bereitgestellt werden (siehe Abb. 3.12). Im Umfeld mobiler Dienste existieren viele Produkte, die sich auf Grund von fehlender Nutzerakzeptanz nicht durchgesetzt haben [Sigu01, HsLH07, Dela08]. Die Fehlentwicklungen sind meist in einer fehlerhaften Anforderungserhebung der Endbenutzergruppe begründet [SeWC10]. Mobile Apps (siehe Kapitel 2.1.7) müssen den Nutzern (Kunden, Mitarbeiter, usw.) gegenüber so bereitgestellt werden, dass eine effektive Arbeitsweise gewährleistet wird. Mobile Apps werden an den Nutzer angepasst, um Wünsche, Nutzungsgewohnheiten, Verhaltensweisen und Fähigkeiten (z. B. über eine Kontextanalyse vom mobilen Gerät aus) antizipieren zu können.

Mobile Apps sind Endbenutzerschnittstellen eines mWfMS, die auf den jeweiligen Anwendungsfall und den Endbenutzer angepasst werden (siehe Kapitel 7.2). Das mWfMS (siehe Kapitel 6) kann für unterschiedliche betriebliche/öffentliche Anwendungsfälle verwendet werden. Vor Entwurf und Implementierung einer App wird eine entsprechende Endbenutzergruppe identifiziert, damit die App im jeweiligen Kontext des Endbenutzers (und dem Anwendungsfall) anforderungsgerecht entwickelt werden kann.

Im weiteren Verlauf werden Nutzerakzeptanzmodelle als Erweiterungen des TAM und Modelle auf Basis rationaler Beweisführungen analysiert, um die Nutzerakzeptanz neuer mobiler Dienste zu untersuchen.

Erweiterungen des TAM

Das TAM (siehe Kapitel 2.8) ist für die Evaluierung mobiler Nutzerdienste nicht geeignet, da keine sozialen Einflüsse betrachtet werden [Davi86]. Soziale Einflüsse haben Auswirkungen auf die Haltung der Endbenutzer [MaGa99] gegenüber einer Technologie. Das TAM muss erweitert werden,

damit wichtige Faktoren zur Analyse mobiler Dienste bzw. Apps berücksichtigt werden [WuWa05].

In [HaYC07] ist eine Erweiterung des TAM mit 1011 Teilnehmern evaluiert worden, um die Akzeptanz einer mobilen Spiele-App zu bestimmen. Das TAM wurde um die Faktoren „Flow Experience“, „Perceived Enjoyment“, „Perceived Attractiveness“ und „Perceived Lower Sacrifices“ ergänzt (siehe Abb. B.1). „Flow Experience“ verdeutlicht die „Versunkenheit“ des Nutzers in die App. Mit „Perceived Enjoyment“ soll der Spaßfaktor veranschaulicht werden. „Perceived Attractiveness“ betont, wie attraktiv die App visuell und akustisch ist. Mit „Perceived Lower Sacrifices“ sollen monetäre Faktoren („das Geld, das man bereit ist zu opfern“) Einfluss auf das Modell nehmen [HaYC07].

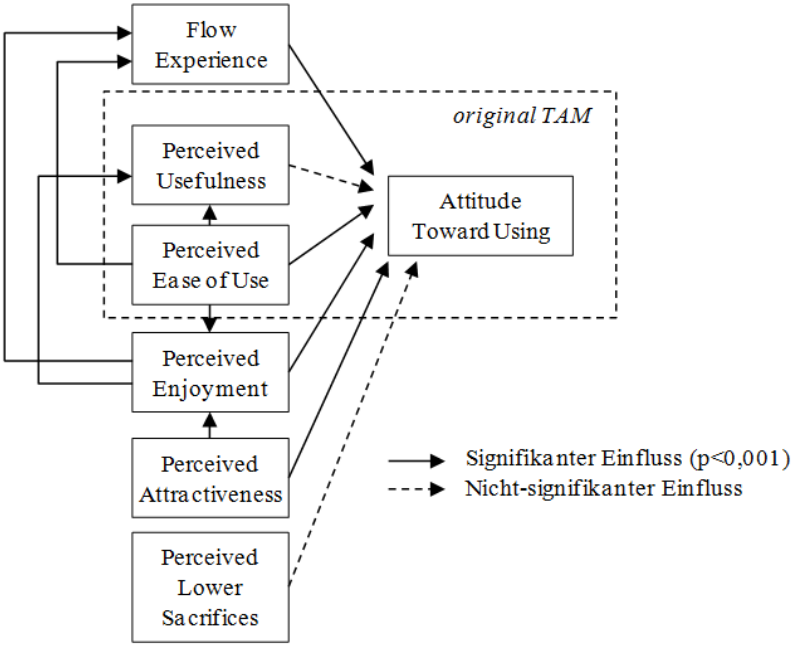


Abb. B.1: Erweitertes TAM nach [HaYC07]

Das „Perceived Enjoyment“ beeinflusst die Nutzung der App [HaYC07]. Mit zunehmendem Alter der Nutzer nimmt jedoch die Bedeutung dieses Faktors ab. „Perceived Enjoyment“ ist maßgeblich abhängig von „Perceived Usefulness“. Die Nützlichkeit (einer App) steht beim Nutzer im Vordergrund. „Perceived Ease of Use“ ist, analog zu „Perceived Enjoyment“, vom Alter abhängig, wobei die Bedeutung mit steigendem Alter zunimmt. „Perceived Enjoyment“ ist abhängig von „Perceived Ease of Use“. „Perceived Usefulness“ hat keinen Einfluss auf die Haltung der Endbenutzer gegenüber der App [HaYC07, HsLu07]. Für jüngere Nutzer ist Spaß ein maßgeblicher Erfolgsfaktor zur Nutzung von Spiele-Apps. Mit steigendem Alter nimmt die Bedeutung der intuitiven Benutzung der App zu. Der Nutzer muss die App als „praktisch“ empfinden, damit er sie nutzt. Auf Grund der Unterschiede zwischen „jung“ und „alt“, wird eine demographische Kategorisierung der Endbenutzergruppen empfohlen [HaYC07].

In [Paga04] werden Multimedia-Dienste innerhalb von mobilen Apps untersucht (siehe Abb. B.2).

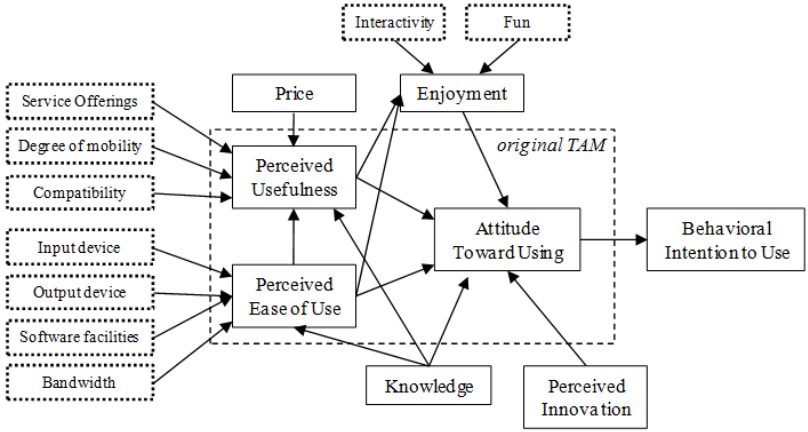


Abb. B.2: Erweitertes TAM nach [Paga04]

An der Studie nahmen 1000 Probanden (aus Italien) teil. „Usefulness“ beeinflusst die höheren Altersgruppen nicht. „Ease of Use“ hingegen gewinnt mit ansteigendem Alter an Bedeutung. Das Interesse an mobilen Apps sinkt mit dem Alter. Die Unterschiede unterhalb der Merkmalsausprägungen [Paga04, Table 2+4] sind in Bezug auf die Demographie sichtbar. Am Beispiel der „Usefulness“ wird eine Differenzierung nach „simple lifetime organisation“, „enjoy“, „time saving“ und „money saving“ vorgenommen (siehe Tab. B.1). Die „lifetime organisation“ und „time saving“ sind den 25-34 Jahre alten Nutzern wichtiger als „enjoy“ und „money saving“ der Generation im Alter zwischen 18-24 Jahre [Paga04].

Tab. B.1: Signifikanz des Merkmals *Usefulness* nach [Paga04]

	18–24	**25–34**
Simple lifetime organisation	17,2%	35,5%
Enjoy	38,8%	15,7%
Time saving	18,1%	25,4%
Money savings	25,9%	19,8%

Nutzer, die über technisches Fachwissen verfügen, zeigen neuen Diensten gegenüber mehr Interesse [Paga04]. Nutzer, die über weniger technisches Wissen verfügen, sind statistisch nachweisbar gegenüber neuen Diensten desinteressierter [Paga04]. Die persönliche Einstellung der Nutzer ist verantwortlich dafür, ob ein (neuer) Dienst genutzt wird [Paga04].

In [HoTa06] sind mobile Datendienste mit 808 Teilnehmern evaluiert worden. Die Erweiterung des TAM ist mit Spiele-Apps und Ticket-Buchungssystem-Apps getestet worden [HoTa06].

Tab. B.2: Aufbau des Modells nach [HoTa06, Table1]

Construct categories	**Constructs**
General technology perceptions	Perceived usefulness, perceived ease of use
Technology-specific perceptions	Perceived service availability, perceived monetary value
Psychographics	Perceived enjoyment, need for uniqueness
Social influence	Social influence
Demographics	Gender, age

Technische, psychologische, soziale und demographische Faktoren (siehe Tab. B.2: „construct categories“) sind innerhalb der Studie evaluiert worden [HoTa06]. Im Modell werden die Faktoren auf messbaren Konstrukten („Constructs“) abgebildet (siehe Tab. B.2).

Eine statistische Signifikanz [HoTa06, Fig. 2] ist zwischen den meisten Faktoren gegeben. Nicht signifikant („ns“) sind Einflüsse von „Age” zu „Behavioral Intention”, von „Perceived Ease of Use“ zu „Perceived Usefulness“ und von „Need for Uniquness“ zu „Perceived Usefulness“. Wenig Einflüsse existieren zwischen „Perceived Usefulness“ und „Behavioral Intention“. „Perceived Usefulness“ ist in diesem Modell für die Nutzung eines neuen Dienstes kein ausschlaggebender Faktor [HoTa06]. Das Alter und das Geschlecht zeigen fast keinen Einfluss auf „Behavioral Intention“ [HoTa06, Fig. 2]. Dieser schwache Einfluss wird mittel- bis langfristig entfallen [HoTa06]. Der Einfluss von „Perceived Ease of Use“ auf die Nutzer wird derzeit noch unterschätzt [HoTa06]. „Perceived Service Availability“ und dessen Einflüsse auf andere Faktoren zeigen, dass Dienste ständig (z. B. unabhängig von Verbindungsabbrüchen) verfügbar sein müssen. Die Endbenutzer erwarten eine reibungslose Funktionsweise von einem neuen Dienst [HoTa06]. „Need for Uniqueness“ und „Social Influence“ sind Faktoren, die neue Produkte und Dienste maßgeblich beeinflussen werden [HoTa06].

In [NyPT05] sind mobile Sofortnachrichtendienste (siehe Kapitel 7.2.1) mit 684 Probanden (aus Norwegen) evaluiert worden. Ziel dieser Studie war, die geschlechtlichen Unterschiede zu identifizieren. Der soziale Faktor ist in [NyPT05] als „Normative Pressure“ bezeichnet worden (vgl. „Social Influence“, Tab. B.2). Der Faktor „Perceived Expressiveness“ entspricht dem „Need for Uniqueness“. „Perceived Enjoyment“ wird analog zu den bereits vorgestellten Modellen verwendet.

Die Ergebnisse der Studie sind in zwei nach Geschlechtern differenzierten Modellen (siehe Abb. B.3 und Abb. B.4) ausgewertet worden [NyPT05].

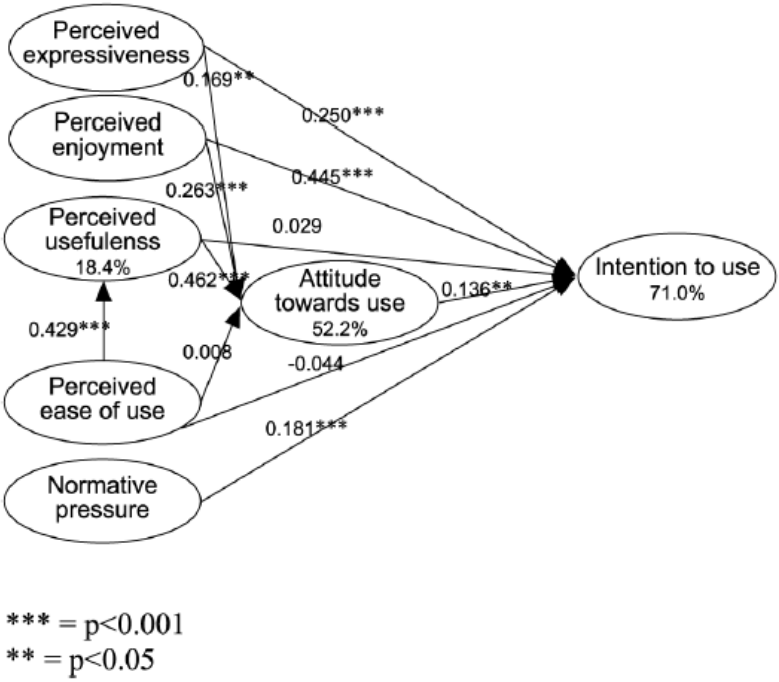


Abb. B.3: Erweitertes TAM; weibliche Endbenutzer nach [NyPT05]

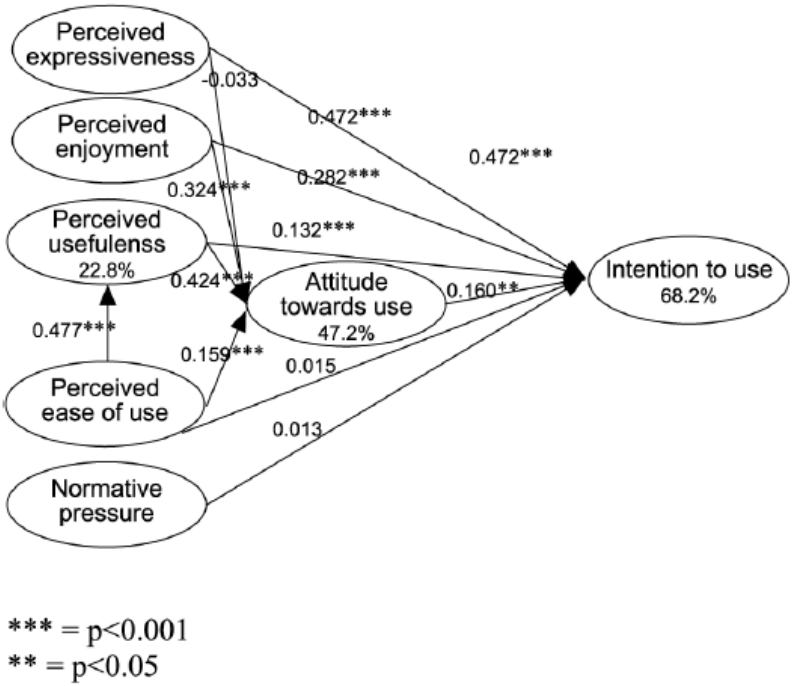


Abb. B.4: Erweitertes TAM; männliche Endbenutzer nach [NyPT05]

Der Vergleich beider Modelle (siehe Abb. B.3 und Abb. B.4) zeigt, dass „Normative Pressure“ und „Perceived Enjoyment“ Frauen stärker beeinflusst als die männlichen Endbenutzer [NyPT05]. Männer lassen sich stark von „Perceived Expressiveness“ beeinflussen [NyPT05]. „Perceived Usefulness“ hat für Männer eine minimal größere Bedeutung als für Frauen [NyPT05]. „Perceived Ease of Use“ hat Einfluss auf „Perceived Usefulness“, was jedoch keine Außenwirkung auf die Nutzung des Dienstes hat [NyPT05]. „Perceived Expressiveness“ beeinflusst zwar männliche Nutzer, jedoch im Gegensatz zu Frauen nicht unter bestimmten Umständen das „Normative Pressure“. [NyPT05] empfehlen, „Perceived Expressiveness“ und „Normative Pressure“ differenziert zu betrachten. Für männliche Endbenutzer ist es wichtig, ihre Persönlichkeit über neue Dienste (bzw. Geräte)

auszudrücken, wohingegen Frauen auf sozialen Druck reagieren. Es existieren somit geschlechtsspezifische Unterschiede [NyPT05].

Modelle auf Basis rationaler Beweisführungen

In [SeWC10] wird eine Klassifikation in fünf Nutzergruppen (Segmente) vorgenommen. Die Nutzer wurden zur Anwendung mobiler Dienste befragt. Die Segmente [SeWC10] werden in vorliegender Arbeit als Klassen einer Klassifikation interpretiert. Die Klassen sind nicht überschneidungsfrei und beinhalten demographische Merkmale, Gefühle und persönliche Einstellungen der Endbenutzer [SeWC10]:

- **Skillful** – erfahren, lernwillig
- **Efficient** –fühlen sich durch Nutzung mobiler Diensten bei der Arbeit unterstützt
- **Trendy** – modebewusst, will Eindruck vermitteln, durch Aktualität/Trends geprägt
- **Basic** – nutzt nur Basisfunktionen des mobilen Gerätes/Dienstes (z. B.: Telefonieren, SMS)
- **Social** – nutzt Dienst/Gerät zur Pflege sozialer Kontakte

An der Studie nahmen insgesamt 350 Anwender aus Finnland teil. Mit den fünf Klassen wurde eine Gleichverteilung erreicht [SeWC10].

Die Eigenschaften bzw. die Anforderungen der Nutzergruppen bei der Nutzung mobiler Dienste werden in Bezug auf das persönliche Interesse ausgewertet (siehe Tab. B.3).

Tab. B.3: Anforderungen, Eigenschaften und Interessen nach [SeWC10]

Kategorie	Anforderungen	weitere Eigenschaften	Interesse an mobilen Diensten
Skillful	Neue, innovative Dienste	Benötigen wenig Hilfe und Unterstützung; Lernprozess dauert nicht zu lang und ist nicht zu aufwendig	hoch
Efficient	Verbesserte Effizienz der Kommunikation; Hilfe und Unterstützung; Statussymbol	Mobile Dienste werden überwiegend aus geschäftlichen Gründen genutzt; Sozialen Kontakten wird die Nutzung der Dienste ebenfalls nahegelegt; Werden Dienste auch in der Zukunft nutzen	hoch
Trendy	Zeitvertreib; Neuste Mobiltelefone mit vielfältigen, neuen Funktionen; Lebensstil; Innovation	Werden Dienste auch in der Zukunft nutzen	hoch
Basic	Einfache Benutzbarkeit	Nutzen Mobiltelefone eher selten und wollen diese auch nicht ihrem Geschmack nach anpassen	sehr wenig
Social	Kommunikation mit sozialen Kontakten	Haben dennoch eher ältere Mobiltelefone	mäßig

Für ältere Nutzer (über 50 Jahre) müssen eigene Produkte/Dienste erstellt werden [SeWC10]. Ältere Nutzer verfügen über die notwendigen finanziellen Mittel, brauchen jedoch länger, um Zusammenhänge bzw. Interaktionsdesigns zu verstehen. Neue mobile Dienste müssen anhand der Anforderungen bzw. am Verhalten der Kunden (am Endbenutzer-Kontext) erstellt werden (siehe Tab. B.3). Hindernisse bei der Nutzung sollten nur innerhalb der Zielgruppe betrachtet werden [SeWC10].

In [GiKe03] wurden 300 Studierende zur Nutzung des WAP befragt. Bei der Auswertung der Studie wurden (wie in [SeWC10]) fünf Nutzergruppen identifiziert. Die Gruppen wurden jedoch um zwei Nutzergruppen ergänzt, die sich vor allem durch die Nicht-Anwendung von WAP auszeichneten [GiKe03]:

- **Mobile Professionals** – Mobiler Dienste soll Arbeit/Beruf erleichtern
- **Sophisticates** – Statusbewusste
- **Socialites** – zur Pflege sozialer Kontakte
- **TechnoToy** – will Erfahrungen über aktuelle technische Entwicklungen sammeln
- **Lifestylers** – Verwendung oft zur Verkürzung der Wartezeit, Trendsetter
- **Misers** – „Geizhals"
- **Laggards** – „Nachzügler" (aus unterschiedlichen Gründen)

Bei der Evaluation der sieben Nutzergruppen mit 433 Teilnehmern (aus Malaysia), sind lediglich „Socialites", „Lifestylers", „Misers" und „Mobile Professionals" identifiziert worden [GiKe03]. Die drei Nutzergruppen („Sophisticates", „TechnoToys" und „Laggards") sind in Malaysia nicht vorhanden. Dies wird als Beweis für kulturelle Unterschiede angeführt [GiKe03].

In [GiKe03] werden die mobilen Apps bzw. Dienste in drei Kategorien („Entertainment", „Information" und „Functionality") aufgeteilt. Mit den Umfrageergebnissen wurden Erkenntnisse über das Verhalten der Endbenutzer herausgearbeitet [GiKe03]. Das größere Wissen über den Endbenutzer-Kontext (das Verhalten) führte jedoch nicht zu einer besseren Endbenutzerklassifikation [GiKe03]. Mobile Dienste werden dennoch falsch eingeschätzt, da die meisten Forscher ihre Vergleiche mit der Einführung des Internets verknüpfen, statt auf das Personenbezogene Alleinstellungsmerkmal mobiler Tech-

nologie zu fokussieren [GiKe03]. Ein Endbenutzer baut eine innige Beziehung zu seinem mobilen Gerät bzw. dem verwendeten Dienst auf. Ein mobiles Endgerät wird z. B. stets (am Körper) mitgeführt. Dadurch handelt der Nutzer bei der Nutzung mobiler Dienste „emotionaler“ [GiKe03].

In [GiHa05] werden sieben Endbenutzergruppen um drei neue Klassen, den „Dedicated Gamers“, den „Social Gamers“ und den „Casual Gamers“, ergänzt. Untersuchungsgegenstand waren mobile Spiele-Apps. Für den Bereich der mobilen Spiele-Apps, der in vorliegender Arbeit nicht speziell betrachtet wird, sind feingranulare Endbenutzergruppen identifiziert worden [GiHa05].

In [CoDK07] wurden 1103 (dänische) Nutzer zur Anwendung neuer mobiler Dienste befragt. Die folgenden vier Endbenutzerklassen wurden bestimmt [CoDK07]:

- Talkers – nutzen nur Sprachdienste
- Writers – nutzen zusätzlich zu den Sprachdiensten nur SMS
- Photographers – nutzen Sprachdienste, SMS und versenden teilweise MMS
- Surfers – nutzen Sprachdienste, SMS, MMS und verwenden mobiles Internet

Die Nutzergruppen wurden anhand der Dienstnutzung identifiziert, wobei innerhalb der Studie versucht wurde, die Nutzer zur Verwendung der Dienste (Sprachtelefonie, SMS, MMS und mobiles Internet) zu stimulieren [CoDK07]. Die Nutzergruppen ähneln dem Diffusionsmodell [Roge03]. „Talkers“ sind vergleichbar mit „Nachzüglern“, „Writers“ mit „Späte Mehrheit“, „Photographers“ mit „Frühe Mehrheit“ und „Surfers“ mit „Innovatoren“ bzw. „frühe Mehrheit“. Die größeren technischen Möglichkeiten neuerer Geräte haben nicht dazu geführt, dass Nutzer neue Dienste nutzten [CoDK07]. Größtes Potential ist bei den „Photographers“ vorhanden, die von den entsprechenden Dienstanbietern in die Gruppe der „Surfer“ überführt werden sollen [CoDK07].

Tab. B.4: Mobile Endbenutzerklassen mit demografischen Eigenschaften nach [CoDK07]

		Talkers	**Writers**	**Photographers**	**Surfers**
Gender	male	82%	58%	59%	92%
	female	18%	42%	41%	8%
Age	>50	42%	14%	9%	10%
	50-41	34%	16%	12%	7%
	40-31	18%	24%	27%	37%
	30-20	5%	45%	49%	37%
	<20	0%	1%	3%	10%
Education	Primary & Secondary	42%	34%	39%	27%
	Tertiary	37%	36%	33%	27%
	Quaternary	21%	30%	29%	46%
Occupation	Private sector	53%	28%	33%	42%
	Students	7%	43%	44%	34%
	Semi public or public sectors	40%	29%	23%	24%
Monthly Household Income	<3.500 €	19%	41%	38%	29%
	3.500 €-8.000 €	38%	31%	30%	34%
	> 8.000 €	25%	15%	16%	25%
	No response	18%	13%	16%	12%
	Monthly payment	~10€	~10€	~15-30€	~15-30€
	Total	**6%**	**56%**	**33%**	**4%**

Die demographische Auswertung der Studie (siehe Tab. B.4) zeigt, je älter der Endbenutzer desto konservativer ist das Nutzungsverhalten bzgl. mobiler

Dienste/Apps. In den beiden Altersgruppen (30-20 und 40-31) sind die meisten Innovatoren zu finden (siehe Tab. B.4). In der Nutzergruppe der „Innovatoren“ sind die vier definierten Einkommensklassen gleich verteilt. Der Bildungsstand („Education“) der „Surfer“ ist im direkten Vergleich zu den anderen Nutzergruppen höher. Ein „Innovator“ ist meist männlich (siehe Tab. B.4). Bei „Writers“ und „Photographers“ existieren keine demographischen Unterschiede [CoDK07]. Beide Nutzergruppen nehmen ca. 90% aller Endbenutzer ein. [CoDK07] empfehlen jedoch die Nutzergruppen differenziert zu umwerben. Die „Writers“ sollen demnach „aggressiv“ angegangen werden, wohingegen bei den „Photographers“ ein eher defensives Vorgehen empfohlen wird [CoDK07].

In [AnD'I02] ist die Demographie mobiler Endbenutzergruppen untersucht worden. Mit der Studie sollte die Bereitschaft Geld für mobile Dienste auszugeben und mit 445 Endbenutzern (aus Finnland) überprüft werden. Es konnten jedoch keine demographischen Unterschiede bei der Zahlungsbereitschaft festgestellt werden [AnD'I02]. Mobile Apps müssen bestimmte Bedürfnisse bzw. Aufgaben erfüllen, um vom Endbenutzer angewandt zu werden [AnD'I02]. Es werden in [AnD'I02] zeitkritische Aufgaben (z. B. Absprachen), spontane Bedürfnisse, Unterhaltungswünsche, effiziente Aufgabenbearbeitung und mobile Bedürfnisse – Dinge, die mit der Mobilität selbst zu tun haben - genannt. Die Kontextanalyse der Endbenutzer bzw. die informationstechnische Interpretation ist somit die wichtigste Eigenschaft einer mobilen App [AnD'I02]. Eine Kontextanalyse auf dem mobilen Gerät kann die Bedienung erleichtern und dadurch ältere Endbenutzergruppen motivieren. Ältere Menschen sind meist nicht „innovativ“, so dass eher jüngere Endbenutzer sich mit Unterhaltungs- bzw. Spiele-Apps beschäftigen [AnD'I02]. Endbenutzer unterscheiden nicht zwischen mobilen Apps und anderen Diensten. Die Erwartungen der Nutzer gegenüber einer App sind mindestens so hoch, wie bei einer gewöhnlichen PC-Applikation [AnD'I02].

In [Wils03] sind speziell junge mobile Nutzer beobachtet worden. Das dem Nutzer zur Verfügung stehende Geld hat keinen Einfluss auf das Konsumentenverhalten (vgl. auch [CoDK07]). Das „Elternhaus“ zeigt auch keinen

Einfluss auf das Konsumentenverhalten [Wils03]. Das Konsumentenverhalten mobiler Dienste verhält sich analog zum generellen Konsumentenverhalten in der Gesellschaft [Wils03].

In [IFAK09] wird von zwei verschiedenen Endbenutzertypen mobiler Dienste gesprochen. Die Nutzergruppe der innovativen Dienste und die Nutzergruppe der konservativen Dienste (z. B. SMS und Sprachtelefonie). In [IFAK09] wurden 1011 Endbenutzer zwischen 14-69 Jahren (in einer Online-Umfrage) befragt. Als Ergebnis der Studie sind sieben verschiedene Endbenutzergruppen identifiziert worden [IFAK09]:

- **Mobile Innovation Type** – männlich, zwischen 30- 39 Jahren, aktuelles mobiles Endgerät, nutzt viele Funktionen und Dienste.
- **Multi Entertainment Type** – zwischen 14-20 J., nutzt meist Unterhaltungsdienste, -programme.
- **Open Minded Type** – meist weiblich, zwischen 20-39 J., starkes Interesse an neuen Diensten und Funktionen.
- **Pragmatic Business Type** – meist männlich, zwischen 40-59 J., starker Fokus auf Erreichbarkeit und Business, nutzt viel SMS und Sprachtelefonie, aber auch andere Dienste.

- **Standard Basic Type** – meist weiblich, zwischen 40-49 J., nutzt Sprachtelefonie und SMS, Potential für neue Dienste gering.
- **Conservative Assurance Type** – meist > 40 J., mobiles Endgerät meist nur für SMS und Sprachtelefonie, Erreichbarkeit wichtigstes Kriterium.
- **Simple Backup Type** – meist > 40 J., mobiles Endgerät nur für Notfälle.

(Die Trennlinie markiert den hohen (oben) und niedrigen (unten) Marktanteil [IFAK09]).

Fazit

Mobile Dienste müssen im Kontext eines mWfMS auf den Endbenutzer, den Benutzeraufgaben (Aufgaben während der Arbeit) und auf das Endgerät selbst angepasst werden. Während der Ausführung einer App (eines mobilen Dienstes) können die Umgebungskontexte des Endgerätes über Sensoren des mobilen Endgerätes analysiert werden, so dass eine mobile Benutzeraufgabe als App adäquat (je nach mobilem Anwendungsfall) gegenüber dem Nutzer angeboten wird [AnD'I02].

[Paga04, NyPT05, HoTa06, HaYC07, HsLH07] verwenden das TAM als Basis für die Nutzergruppenanalyse mobiler Dienste. Das Diffusionsmodell muss, analog zum TAM, zur Nutzergruppenanalyse mobiler Dienste erweitert werden [Paga07]. Die zweite Gruppe [AnD'I02, GiKe03, Wils03, GiHa05, CoDK07, IFAK09, SeWC10] umfasste jeweils eigens erstellte Klassifikationen auf Basis von rationalen Begründungen.

Die Erweiterungen des TAM [NyPT05, HoTa06, HaYC07] sind geeignet, um **geschlechtliche Unterschiede** zu identifizieren. Während Männer beispielsweise mobile Endgeräte als Statussymbole benutzen [Econ04], lassen sich Frauen auf Grund des „sozialen Drucks" zur Nutzung eines mobilen Dienstes motivieren. Die Entwicklung sozialer Netzwerke hat gezeigt, dass sozialer Druck maßgeblich dazu beiträgt, dass bestimmte Anwendungen genutzt werden. Mobile Endgeräte werden langfristig ihren Stellenwert als Statussymbole verlieren [HoTa06].

Kulturelle Aspekte müssen bei der Erstellung einer App bzw. einer Nutzerstudie betrachtet werden [PaCh02, CLKJ05]. Die Semantik vieler Zeichen und Symbole ist je nach Region und kultureller Zugehörigkeit unterschiedlich. Viele Hersteller versuchen diesem Aspekt gerecht zu werden, indem die Endbenutzerschnittstellen weitestgehend neutral gestaltet werden [Appl11].

Die **Altersunterschiede** werden in fast allen Studien erörtert. Im Bezug auf mobile Apps muss berücksichtigt werden, dass Nutzer über 50 Jahre eine einfachere Interaktionssteuerung bzw. einfache Symbole und Analogien verlangen. Nutzer bzw. Anwender im Alter von 20-50 Jahren präferieren eine klare Fokussierung auf Nützlichkeit [CHRW05]. Jüngere Anwender (unter 20 Jahren) können zur Nutzung über Werbung oder Unterhaltungsaspekte motiviert werden [NyPT05, HoTa06, HaYC07].

Bei der **Anwendung eines mWfMS** bzw. einer mobilen App werden (gewöhnlich) Aufgaben bearbeitet (vgl. Kapitel 7.3). Die effiziente Bearbeitung der Aufgabe durch den Endbenutzer muss immer im Vordergrund stehen. Die App bzw. der mobile Dienst wird nur dann wertvoll für das Unternehmen bzw. die Organisation, wenn ein Mehrwert gegenüber konventioneller Arbeit (ohne mobiles Endgerät) durch den Anwender erkannt wird. Die mobile App muss so entworfen werden, dass die identifizierte Nutzergruppe anhand der Nutzung der App einen Mehrwert erkennt. Die Prüfung mittels eines Usability-Tests wird vor dem Vertrieb der App vorausgesetzt [Appl11].

Kritik

Einige Faktoren bzw. Merkmale aus [NyPT05, HoTa06, HaYC07] zeigen maßgebliche Änderungen der Wirkungszusammenhänge innerhalb der Modelle. Diese Einflüsse haben jedoch keine Auswirkung auf die beabsichtigte Nutzung, die wiederum keinen Einfluss auf die tatsächliche Nutzung haben müssen [Quir06].

Den Studien mobiler IKT bzw. des Mobile Business mangelt es an deskriptiven und empirischen Untersuchungen [ScBH05, Wohl04]. Innerhalb der Studien werden zu wenig Nutzer befragt. Die Studien wurden meist nur im skandinavischen [AnCW03, Wils03, CHRW05, NyPT05, BCMW07, CoDK07, SeWC10] und asiatischen Raum [GiKe03, GiHa05, WuWa05, HsLH07, KiPO08] durchgeführt. Da kulturelle Unterschiede die Ergebnisse beeinflussen, müssen die Ergebnisse der Studien relativiert werden [PaCh02, Pavl02, CLKJ05]. Es existiert ein Forschungsdefizit in Nordamerika und Europa.

Literaturverzeichnis

[AAAM97] G. Alonso, D. Agrawal, A. El Abbadi und C. Mohan: Functionality and Limitations of Current Workow Management Systems. IEEE Expert, Vol. 12, Iss. 5, 1997, S. 632-635.

[AaHD05] W. M. P. van der Aalst, A. ter Hofstede, M. Dumas: Patterns of Process Modeling. Dumas, M.; van der Aalst, W.; ter Hofstede, A. (Hrsg.): Process-Aware Information Systems, Bridging People and Software through Process Technology, John Wiley & Sons, Hoboken, NJ, USA, 2005, S. 179-203.

[Aals13] W. M. P. van der Aalst: Business Process Management: A Comprehensive Survey. ISRN Software Engineering, Vol. 2013, Article ID 507984, 2013.

[Aals96] W. M. P. van der Aalst: Three Good reasons for Using a Petrinetbased Workflow Management System. Navathe, S.; Wakayama, T. (Hrsg.): Proceedings of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96), Cambridge, MA, USA, 1996, S. 179-201.

[Aals98] W. M. P. van der Aalst: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers, Vol. 8, Iss. 1, 1998, S. 21-66.

[ABHK05] N. Anderson, A. Bender, C. Hartung,G. Kulkarni, A. Kumar, I. Sanders, D. Grunwald, B. Sanders: The Design of the Mirage Spatial Wiki. Proceedings of the First International Conference on Web Information Systems and Technologies, Miami, USA, Mai 2005, S. 48-55.

[ACDG03] T. Andrews, F. Curbera, H. Dholakia,Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller,D. Smith, S. Thatte, I. Trickovic, S. Weerawarana: Business Process Execution Language for Web Services, Version 1.1.

2003, http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-bpel/ws-bpel.pdf, letzter Abruf: 07.07.2014.

[AdAH98] N.R. Adam, V. Atluri, W.-K. Huang: Modeling and Analysis of Workflows Using Petri Nets. Journal of Intelligent Information Systems, Vol. 10, 1998, S. 131-158.

[AGKA95] G. Alonso, R. Gunthor, M. Kamath, D. Agrawal, A. El Abbadi and C. Mohan: Exotica/FMDC: Handling Disconnected Clients in a Workflow Management System, 1995, S. 99-110.

[AGKA96] G. Alonso, R. Glünthör, M. Kamath, D. Agrawal, A. El Abbadi, C. Mohan: Exotica/FMDC: A Workflow Management System for Mobile and Disconnected Clients. Distributed and Parallel Databases, Vol. 4, 1996, S. 229-247.

[AhIl10] S. A. Ahson, M. Ilyas (Hrsg.): Location-Based Services Handbook: Applications, Technologies, and Security. Crc Pr Inc Verlag, 2010.

[AHKB03] W. M. P. van der Aalst, A. ter Hofstede, B. Kiepuszewski, A. Barros: Workflow Patterns. Distributed and Parallel Databases, Springer Netherlands, Vol. 14, Nb. 1, 2003, S. 5-51.

[Ajze91] I. Ajzen: The theory of planned behavior. Organizational Behavior and Human Decision Processes, Vol. 50, Iss. 2, 1991, S. 179-211.

[AnCW03] B. Anckar, C. Carlsson, P. Walden: Factors affecting consumer adoption decisions and intents in mobile commerce: Empirical insights. Proceedings of the 16th Bled eConference, Bled, Slowenien, 2003, Paper 28.

[AnD'I02] B. Anckar, D. D'Incau: Value creation in mobile commerce: Findings from a consumer survey. Journal of Information Technology Theory and Application, Vol. 4, Iss. 1, 2002, S. 43-64.

[Anto05] P. Antoniac: A Taxonomy of Mobility: Some Implicatios and Requirements for Mobile Information Appliances. CEAI, Vol. 7, Iss. 4, 2005, S. 3-10.

[Appl11] apple.com: iOS Human Interface Guideline (siehe PDF Datei); http://developer.apple.com/appstore/guidelines.html, letzter Abruf: 07.07.2014.

[Appl15] Apple: iOS Human Interface Guidelines, https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/, letzter Abruf: 22.10.2015.

[Arbi09] F. el Arbi: Entwicklung eines mobilen Informationssystems für ortsabhängige Textdokumente. Diplomarbeit, 2009.

[ARMC14] R. Alcarria; T. Robles; A. Morales; E. Cedeño: Resolving coordination challenges in distributed mobile service executions, International Journal of Web and Grid Services (IJWGS), Vol. 10, No. 2/3, 2014.

[Aste03] D. Astels: Test Driven Development: A Practical Guide. Prentice Hall Professional Technical Reference, 2003.

[Bago07] R. P. Bagozzi: The legacy of the technology acceptance model and a proposal for a paradigm shift. Journal of the Association for Information Systems, Vol. 8, Iss. 4, 2007, S. 243-254.

[Balz09] H. Balzert: Lehrbuch der Softwaretechnik - Basiskonzepte und Requirements Engineering. 3. Auflage, Spektrum Akademischer Verlag, Heidelberg, 2009.

[BBLS08] M. Bartel, J. Boyer, B. LaMacchia, E. Simon: XML Signature Syntax and Processing (Second Edition). W3C Recommendation, 2008, http://www.w3.org/TR/xmldsig-core/, letzter Abruf: 07.07.2014.

[BCKM04] R. Bill, C. Cap, M. Kofahl, T. Mundt: Indoor and Outdoor Positioning in Mobile Environments – a Review and some Investigations

on WLAN Positioning. Geographic Information Sciences, Vol. 10, No. 2, Dez. 2002, S. 91-98.

[Beck02] K. Beck: Test Driven Development. By Example. Addison-Wesley, Longman, Amsterdam, 2002.

[BeDi91] B. Berthomieu, M. Diaz: Modeling and verification of time dependent systems using time Petri nets. Software Engineering, IEEE Transactions, CNRS, Toulouse, 1991, S. 259-273.

[Beng14] G. Bengel: Grundkurs Verteilte Systeme: Grundlagen und Praxis des Client-Server und Distributed Computing. Springer Vieweg, 4. Auflage, 2014.

[Bier92] P. Bieri: Was macht das Bewusstsein zu einem Rätsel?. Spektrum der Wissenschaft, Vol. 10, 1992, S. 48–56.

[BNST08]N. Blinn, M. Nüttgens, M. Schlicker, O. Thomas, P. Walter: Lebenszyklusmodelle hybrider Wertschöpfung - Modellimplikationen und Fallstudie an einem Beispiel des Maschinen- und Anlagenbaus, Multikonferenz Wirtschaftsinformatik (MKWI), 2008, S. 711-722.

[Bolc00] G. A. Bolcer: Magi: An architecture of mobile and disconnected workflow. IEEE Journal on Internet Computing, Vol. 4, Iss. 3, 2000, S. 46-54.

[BoND09] M. Borenovi, A. Nevskovic, B. Djuradj: Cascade-connected ANN structures for indoor WLAN positioning. Proceedings of the 10th international conference on Intelligent data engineering and automated learning, Springer-Verlag, Burgos, Spain, 2009, S. 392-399.

[Bons96] G. Bonsiepe: Interface: Design neu begreifen. Bollmann, Mannheim, 1996.

[Brow96] P. J. Brown: The stick-e document: a framework for creating contextaware applications. Proceedings of EP"96, Palo Alto, CA, USA, Januar 1996, S. 259-272.

[BrPa05] R. Brown, H.-Y. Paik: Resource-Centric Worklist Visualisation. Proceedings of CoopIS/DOA/ODBASE, 2005, S. 94-111.

[BuGa02] J. Burell, G. K. Gay: E-graffiti: Evaluating real-world use of a context-aware system. Interaction with Computers, Vol. 14, No. 1, 2002, S. 301-312.

[BuJu06] Bundesministerium für Justiz: Verordnung über die Anwendung von Düngemitteln, Bodenhilfsstoffen, Kultursubstraten und Pflanzenhilfsmitteln nach den Grundsätzen der guten fachlichen Praxis beim Düngen (Düngeverordnung – DüV), 2006, http://www.gesetze-im-internet.de/bundesrecht/d_v/gesamt.pdf, letzter Abruf: 07.07.2014.

[Busb94] U. Busbach: Mobiler Zugang zu Task Management Systemen, S. Kirn, K. Klöckner (Hrsg.): Betrieblicher Einsatz von CSCW-Systemen, GMD-Studien Nr. 230, GMD, Sankt Augustin, 1994, S. 113-132.

[Buss95] C. Bussler: User Mobility in Workflow-Management-Systems. Proceedings of the Telecommunications Information Networking Conference (TINA '95), Australia, 1995.

[CBLT06]S. Chen, Y. Bu, J. Li, X. Tao, J. Lu: Toward Context-Awareness: A Workflow Embedded Middleware. Ubiquitous Intelligence and Computing IEEE 2006, Vol. 4159, 2006, S. 766-775.

[CCDH04] S. Carter, E. Churchill, L. Denoue, L. Helfman, L. Nelson: Digital graffiti: public annotation of multimedia content. CHI'04 extended abstracts on Human factors in computing systems, Session: Late braking result papers, Wien, Österreich, April 2004, S. 1207-1210.

[CEGV00] K. Czarnecki, U. Eisenecker, R. Glück, D. Vandevoorde, T. Veldhuizen: Generative Programming and Active Libraries. LNCS, Vol. 1766, 2000, S. 25-39.

[ChCK02]D. K. W. Chiu, S. C. Cheung, E. Kafeza: Three-Tier View-based Support for Mobile Workflows. Proceedings of M-Business 2002 – The First International Conference on Mobile Business, Athen, Griechenland, 2002.

[ChDe10] H. Che, M. Decker: Anomalies In Business Process Models For Mobile Scenarios With Location Constraints. IEEE (Hrsg.): Proceedings of the IEEE International Conference on Automation and Logistics (ICAL 2010), Hong Kong, China, 2010, S. 306-313.

[ChLe04] D. Charkraborty, H. Lei: Pervasive Enablement of Business Processes. Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom), 2004, S. 87-98.

[ChLe11] A. Charland, B. Leroux: Mobile application development: web vs. native. Commun. ACM, Vol. 54, Iss. 5, 2011, S. 49-53.

[ChNB16] C. Chang, S. Srirama, R. Buyya: Mobile Cloud Business Process Management System for the Internet of Things: A Survey. ACM Computing Surveys, Vol.49, No. 4, 2016.

[ChNC06]S. Chakrabarti, E. Nordmark, D. Cohen: Enterprise Mobility. Tech Report SMLI TR-2006-158, Sun Microsystems, Inc. Mountain View, CA, USA, 2006, S. 1-26.

[CHRW05] C. Carlsson, K. Hyvönen, P. Repo, P. Walden: Adoption of mobile services across different technologies. Proceedings of the 18th Bled eConference - eIntegration in Action, Bled, Slowenien, 2005, S. 166-178.

[ChSL00] J. Chan, A. Seneviratne, B. Landfeldt: The challenges of provisioning real-time services in wireless Internet. Telecommunications Journal of Australia, Vol. 50, No. 3, 2000, S. 37-48.

[CJKM08] R. Carbon, G. Johann, T. Keuler, D. Muthig, M. Naab, S. Zilch: Mobility in the virtual office: a document-centric workflow approach. Proceedings of the 1st international workshop on Software architectures and mobility (SAM '08). ACM, New York, NY, USA, 2008, S. 21-26.

[Clar10] J. Clark: Tapworthy: Designing Great iPhone Apps. O'Reilly, Canada, 2010.

[CLKJ05] B. Choi, I. Lee., J. Kim, Y. Jeon: A qualitative cross-national study of cultural influences on mobile data service design. CHI 2005: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, New York, 2005, S. 661-670.

[CoCL11] A. G. Connor, P. K. Chrysanthis, A. Labrinidis: Key-key-value stores for efficiently processing graph data in the cloud. Data Engineering Workshops (ICDEW), 2011 IEEE 27th International Conference, Pittsburgh, PA, USA, 2011, S. 88-93.

[CoDK07] I. D. Constantiou, J. Damsgaard, L. Knutsen: The four incremental steps toward advanced mobile service adoption. Communications of the ACM archive, Vol. 50, Iss. 6, Smart business networks, 2007, S. 51-55.

[Coop01] G. Cooper: The Mutable Mobile: Social Theory in the Wireess World. (Hrsg.) Brown, B., Green, N. und Harper, R.; Wireless World. Social and Interactional Aspects of the Mobile Age. Springer, London, 2001, S. 19-31.

[DaBW89] F. D. Davis, R. P. Bagozzi, P. R. Warshaw: User Acceptance of Computer Technology: A Comparison of Two Theoretical Models. Management Science, Vol. 35, Iss. 8, 1989, S. 982-1003.

[DaBW92] F. D. Davis, R. P. Bagozzi, P. R. Warshaw: Extrinsic and Intrinsic Motivation to Use Computers in the Workplace. Journal of Applied Social Psychology, Vol. 22, Iss. 14, 1992, S. 1111-1132.

[Dahm05] M. Dahm: Grundlagen der Mensch-Computer-Interaktion. Pearson Studium, München, 2005.

[DaRR11] P. Dadam, M. Reichert, S. Rinderle-Ma: Prozessmanagementsysteme: Nur ein wenig Flexibilität wird nicht reichen. Informatik-Spektrum. Vol. 34, Iss. 4, 2011, S. 364-376.

[Dave93] T. H. Davenport: Process Innovation: Reengineering Work through Information Technology. Harvard Business School Press, Boston, MA, USA, 1993.

[Davi86] F. D. Davis: A technology acceptance model for empirically testing new end-user information systems: theory and results. Doctoral dissertation, MIT Sloan School of Management, Cambridge, MA, USA, 1986.

[Davi89] F. D. Davis: Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, Vol. 13, Nb. 3, 1989, S. 319-340.

[DBSK05] M. Decker, R. Bulander, G. Schiefer, B. Kölmel, B.: A System for Mobile and Wireless Advertising. Mobile Information Systems II (Proceedings of the IFIP TC8 Working Conference on Mobile Information Systems 2005 (MOEIS)), IFIP, Springer, Leeds, U.K., December 2005, S. 287-301.

[DCOS10] M. Decker, H. Che, A. Oberweis, P. Stürzel, M. Vogel: Modeling Mobile Workflows with BPMN. 9th International Conference on Mobile Business (ICMB 2010), IEEE, Athens, Greece, Juni, 2010, S. 272-279.

[DeAb00] A. K. Dey,G. D. Abowd: CybreMinder: A Context-Aware System for Supporting Reminders. Proceedings of the second International Symposium on Handheld and Ubiquitous Computing (HUC), Bristol, Großbritanien, 2000, S. 172-186.

[Deck11] M. Decker: Modellierung ortsabhängiger Zugriffskontrolle für mobile Geschäftsprozesse. Dissertation, KIT Scientific Publishing, 2011.

[DeHa95] M. DeBellis, C. Haapala: User-centric Software Engineering. IEEE Expert, Vol. 10, Iss. 1, 1995, S. 34-41.

[Dela08] J. Delaney: MMS five years on. Journal of Telecommunications Management, Vol. 1, Iss. 1, 2008, S. 69-78.

[Dent04] K. D. Dent: Postfix. O'Reilly Verlag, 1. Auflage, 2004.

[DeOS10] M. Decker, A. Oberweis, P. Stürzel: Ortsabhängiger Dokumentenzugriff mit Discretionary Access Control. Markus Bick, Stefan Eulgem, Elgar Fleisch, J. Felix Hampe, Birgitta König-Ries, Franz Lehner, Key

Pousttchi, Kai Rannenberg, (Hrsg): Technologien, Anwendungen und Dienste zur Unterstützung von mobiler Kollaboration, Köllen Druck+Verlag GmbH, LNI, Bonn, Februar, 2010, S. 153-166.

[Dese92] J. Desel: Struktur und Analyse von Free-Choice-Petrinetzen. Deutscher Universitäts-Verlag GmbH, Wiesbaden, 1992.

[Dey01] A. K. Dey: Understanding and Using Context. Personal and Ubiquitous Computing, Vol. 5, Iss. 1, 2001, S. 4-7.

[DHTZ14] S. Deng, L. Huang, J. Taheri, A. Zomaya: Computation offloading for service workflow in mobile cloud computing, IEEE Transactions on Parallel and Distributed Systems, 2014, S. 3317 - 3329.

[DiHe95] N. Diehl, A. Held: Mobile Computing. Systeme, Kommunikation, Anwendung. In: International Thompson Publishing, Bonn, 1995.

[Dijk82] E. Dijkstra: On the role of scientific thought. Dijkstra, E.: Selected writings on computing: a personal perspective. Springer-Verlag, New York, USA, 1982, S. 60-66.

[DJJA04] H. D. Chon, S. Jun, H. Jung, S. W. An: Using RFID for Accurate Positioning. Journal of Global Positioning Systems, Vol. 3, No. 1-2, 2004, S. 32-39.

[DKKO09] M. Decker, B. Keuter, S. Klink, A. Oberweis, P. Stürzel: Workflow-Management mit Mobile Computing: Ein Überblick. Alexander Zipf, Sandra Lanig, Michael Bauer (Hrsg.): 6. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste (2009), Geographisches Institut der Universität Heidelberg, Heidelberger Geographische Bausteine, Heidelberg, September, 2010, S. 145-154.

[DLNW13] H. T. Dinh, C. Lee, D. Niyato, P. Wang: A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing, Vol. 13, Iss. 18, 2013, S. 1587-1611.

[DMPD99] H. J. Domingos, J. L. Martins, N. M. Preguiça, S. M. Duarte: A Workflow Architecture to Manage Mobile Collaborative Work. Encontro Português de Computação Móvel 1999 (EPCM 99), 1999, S. 49-61.

[DRRG09] P. Dadam, M. Reichert, S. Rinderle-Ma, K. Göser, U. Kreher, M. Jurisch: Von ADEPT zur AristaFlow BPM Suite - Eine Vision wird Realität: Correctness by Construction und flexible, robuste Ausführung von Unternehmensprozessen. EMISA Forum, Vol. 29, Iss. 1, 2009, S. 9-28.

[DSBG06] J. Davis, D. Sow, D. Bourges-Waldegg, C. J. Guo, C. Hoertnag, M. Stolze, B. White Eagle, Y. Yin: Supporting Mobile Business Workflow with Commune. Proceedings of the 7th IEEE Workshop on Mobile Computing Systems & Applications, Washington, DC, USA, 2006, S. 10-18.

[DSKO09] M. Decker, P. Stürzel, S. Klink, A. Oberweis: Location Constraints for Mobile Workows. J.E. Agudo et al. (Hrsg.): International Conference on Techniques and Applications for Mobile Commerce (TAMoCo 09), Band 201 der Frontiers in Artificial Intelligence and Applications, Merida, Spain, September 2009, IOS Press, S. 93-102.

[DuGa03] S. Dustdar, H. Gall: Architectural concerns in distributed and mobile collaborative systems. Journal of Systems Architecture, Vol. 49, 2003, S. 457-473.

[Dulz00] W. Dulz: WAP: Wireless Application Protocol – Aktuelles Schlagwort. Informatik Spektrum, Bd. 23, Nr. 4, 2000, S. 271-273.

[Duma09] M. Dumas: Case Study: BPMN to BPEL Model Transformation. Oryx Journal, 2009, S. 6-9.

[DuMG07] P. M. Duvall, S. Matyas, A. Glover: Continuous Integration: Improving Software Quality and Reducing Risk. Addison-Wesley, 2007.

[Dunc03] A. M. Duncan: Objective-C Pocket Reference. O'Reilly Media, 2003.

[Ecke09] C. Eckert: IT-Sicherheit: Konzepte, Verfahren, Protokolle. Oldenbourg Wissenschaftsverlag, 6. Auflage, 2009.

[Econ04] Economist: Why phones are replacing cars - And why this is a good thing, 29.04.2004, http://www.economist.com/node/2628969?story_id=2628969, letzter Abruf: 07.07.2014.

[EGHW08] A. Ebersbach, M. Glaser, R. Heigl, A. Warta: Wiki: Kooperation im Web. Springer-Verlag Berlin et al., 2. Auflage, 2008.

[ElNa06] R. Elmasri, S. B. Navathe: Fundamentals of Database Systems. Addison Wesley, 5. Auflage, 2006.

[FeHS06] H. Feng, T. Hoegler, W. Stucky: Exploring the Critical Success Factors for Mobile Commerce. Proceedings of the International Conference on Mobile Business (ICMB) 2006, 2006, S. 40-47.

[FiAj75] M. Fishbein, I. Ajzen: Belief, attitude, intention and behavior: An introduction to theory and research. MA: Addison-Wesley, 1975.

[FiCh97] A. F. Firat, C. J. Shultz: From segmentation to fragmentation: Markets and marketing strategy in the postmodern era. European Journal of Marketing, Vol. 31, Iss. 3/4, 1997, S. 183-207.

[FlNa14] H. Flores, S. Narayana Srirama: Mobile Cloud Middleware, Journal of Systems and Software, Vol. 92, 2014, S. 82-94.

[FMHW97] S. Feiner, B. MacIntyre, T. Höllerer, A. Webster: A Touring Machine - Prototyping 3D Mobile Augmented Reality Systems for exploring the urban environment. Personal and Ubiquitous Journal, Bd. 1., Nr. 4, 1997, S. 208-217.

[FrRH10] J. Freund, B. Rücker, T. Henninger: Praxishandbuch BPMN. Carl Hanser Verlag, München, Wien, 2010.

[FrTF03] N. Fremuth, A. Tasch, M. Fränkle: Mobile communities – New Business Opportunities for Mobile Operators?. Proceedings of the 8th International Workshop on Mobile Multimedia Communications (MoMuc 03), München, Deutschland, 2003, S. 341-346.

[FWAC17] J. Fan, J. Wang, W. An, B. Cao, T. Dong: Detecting Difference between Process Models Based on the Refined Process Structure Tree. Mobile Information Systems, Article ID 6389567, 2017.

[Gada01] A. Gadatsch: Prozesskostenrechnung als Element des Workflow-Life-Cycle. EMISA Forum, Vol. 11, Iss. 2, 2001, S. 13-20.

[Gall97] J. Galler: Vom Geschäftsprozessmodell zum Workflow-Modell. Wiesbaden, 1997.

[GaSc95] J. Galler, A.-W. Scheer: Workflow-Projekte: Vom Geschäftsprozessmodell zur unternehmensspezifischen Workflow-Anwendung. Information Management, Vol. 10, Iss. 1, 1995, S. 20-27.

[GeCa02] L. Gentry, R. Calantone: A comparison of three models to explain shop-bot use on the web. Psychology and Marketing, Vol. 19, Iss. 11, 2002, S. 945-956.

[GeHS95] D. Georgakopoulos, M. Hornick, A. Sheth: An Overview of Workflow Management: From Process Modeling to Workow Automation Infrastructure. Distributed and Parallel Databases, Vol. 3, Iss. 2, 1995, S. 119-153.

[Gier98] O. Gierhake: Integriertes Geschäftsprozessmanagement: Effektive Organisationsgestaltung mit Workflow-, Workgroup- und Dokumentenmanagement-Systemen. 2. Auflage, Friedrich Vieweg Verlag, Braunschweig, Wiesbaden, 1998.

[GiHa05] A. L. Gilbert, H. Han: Understanding mobile data services adoption: Demography, attitudes or needs? Technological Forecasting & Social Change 72, 2005, S. 327–337.

[GiKe03] A. L. Gilbert, J. L. Kendall: A Marketing Model for Mobile Wireless Services. Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003, S. 9-17.

[GiVa03] C. Girault, R. Valk: Petri Nets for System Engineering - A Guide to Modeling, Verification, and Applications. Springer-Verlag, Berlin, 2003.

[GoJW04]J. Gortmaker, M. Janssen, R. W. Wagenaar: The advantages of web service orchestration in perspective. M. Janssen, G. Henk, R. W. Wagenaar (Hrsg.): Proceedings of the 6th international conference on Electronic commerce (ICEC '04), ACM, New York, USA, S. 506-515.

[GoMe03]L. Gorlenko, R. Merrick: No wires attached: Usability challenges in the connected mobile world. IBM Systems Journal, Vol. 42, Nr. 4, 2003, S. 639-651.

[GPWK07] T. Gunarathne, D. Premalal, T. Wijethilake, I. Kumara, A. Kumar: BPEL-Mora: Lightweight Embeddable Extensible BPEL Engine. Emerging Web Services Technology, Birkhäuser, Basel, 2007.

[GrKe95] V. Grover, W. J. Kettinger. Business Process Change – Reengineering, Concepts, Methods, Technologies. Idea Group Publishing, Hershey, USA, 1995.

[GSWM13] C. Gröger, S. Silcher, E. Westkämper, B. Mitschang: Leveraging Apps in Manufacturing. A Framework for App Technology in the Enterprise. Procedia CIRP, Vol. 7, 2013, S. 664-669.

[HaCh93] M. Hammer, J. Champy: Reengineering the Corporation: A Manifesto for Business Revolution. Harper Business, New York, USA, 1993.

[Hamm08] N. Hammer: Mediendesign für Studium und Beruf. Springer-Verlag, Berlin, 2008.

[HaSt04] C. Richter-von Hagen, W. Stucky: Business-Prozess und Workflow-Management: Prozessverbesserung durch Prozess-Management. B. G. Teubner, Stuttgart, Deutschland, 2004.

[HaYC07]I. Ha,Y. Yoon, M. Choi: Determinants of adoption of mobile games under mobile broadband wireless access environment. In: Information & Management, Vol. 44, Iss. 3, 2007, S. 276-286.

[HeCa03] C. K. Hess, R. H. Campbell: An application of a context-aware file system. Personal and Ubiquitous Computing, Vol. 7, Nr. 6, Springer-Verlag London, Großbritannien, 2003, S. 339-352.

[Heil00] S. Heilbronner: Konzeption einer Architektur für das integrierte Management der Ressourcennutzung nomadischer Systeme in Datennetzen, Herbert Utz Verlag, 2000.

[Heil94] H. Heilmann: Workflow Management: Integration von Organisation und Informationsverarbeitung. Handbuch der modernen Datenverarbeitung (HMD), Vol. 31, Iss. 176, 1994, S. 8-21.

[Heil96] H. Heilmann: Die Integration der Aufbauorganisation in Workflow-Management-Systeme. H. Heilmann, L. J. Heinrich, F. Roithmayr (Hrsg.): Information Engineering, München, 1996, S. 147-165.

[HeKi09] R. Hewett, P. Kijsanayothin: Location Contexts in Role-based Security Policy Enforcement. Proceedings of the 2009 International Conference on Security and Management (SAM'09), Las Vegas, Nevada, July 2009, S. 404-410.

[Heut07] R. Heutschi: Serviceorientierte Architektur – Architekturprinzipien und Umsetzung in die Praxis. H. Österle, R. Winter, W. Brenner (Hrsg.): Business Engineering, Springer, Berlin, 2007.

[HeVa02] H. van der Heijden, P. Valiente: Mobile Business Processes: Cases from Sweden and the Netherlands. SSE/EFI Working Paper Series in Business Administration, 2002.

[HFTR99]T. Höllerer, S. Feiner, T. Terauchi, G. Rashid, D. Hallaway: Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system. Computers & Graphics Journal, Bd. 23, Nr. 6, 1999, S. 779-785.

[HHGR06] G. Hackmann, M. Haitjema, C. Gill, G.-C. Roman: Sliver: A BPEL Workflow Process Execution Engine for Mobile Devices. Conference on Service Oriented Computing 2006, Vol. 4294, 2006, S. 503-508.

[HHLL10] K. Häussermann, C. Hubig, P. Levi, F. Leymann, O. Simoneit, M. Wieland, O. Zweigle: Understanding and designing situation-aware mobile and ubiquitous computing systems - an interdisciplinary analysis on the recognition of situation with uncertain data using situation templates. Proceedings of International Conference on Mobile, Ubiquitous and Pervasive Computing, March 2010, S. 329-339.

[Hick11] I. Hickson: Specification for HTML5. 2011. http://dev.w3.org/html5/spec/, letzter Abruf: 07.07.2014.

[HöFe04] T. Höllerer, S. Feiner: Mobile Augmented Reality. Karimi H. A., Hammad A. (Hrsg.): Telegeoinformatics: Location-Based Computing and Services, Taylor & Francis Books Ltd., USA, 2004.

[Holl95] D. Hollingsworth: Workflow Management Coalition: The Workflow Reference Model. Specification TC00-1003, Workflow Management Coalition, Winchester, Großbritannien, 1995.

[HoTa06] S. Hong, K. Y. Tam: Understanding the adoption of multi-purpose information appliances: The case of mobile data services. Information Systems Research, Vol. 17, Iss. 2, 2006, S. 162-179.

[HsLH07] C.-L. Hsu, H.-P. Lu, H.-H. Hsu: Adoption of the mobile Internet: An empirical study of multimedia message service (MMS). Omega, Vol. 35, Iss. 6, 2007, S. 715-726.

[HsLu07] C.-L. Hsu, H.-P. Lu: Consumer behavior in online game communities: A motivational factor perspective. Computers in Human Behavior, Vol. 23, Iss. 3, 2007, S. 1642-1659.

[Hugh10] J. Hughes: IPhone and IPad Apps Marketing: Secrets to Selling Your IPhone and IPad Apps. Que Verlag, 2010.

[HuHM10] K. A. Hummel, A. Hess, H. Meyer: Mobilität im „Future Internet“ - Geräte- und Ressourcenmobilität: Herausforderungen und Techniken im Überblick. Informatik Spektrum, Sonderheft: Future Internet, 2010, S. 143-159.

[Huhn03] M. N. Huhns: Software Agents: The Future of Web Services. Agent Technology Workshops 2002, LNAI 2592, Springer-Verlag, Heidelberg, 2003, S. 1-18.

[IBM10] Übersichtswebsite über IBM-Veröffentlichungen zu Pervasive Computing, http://domino.research.ibm.com/comm/wwwr_thinkresearch.nsf/pages/pervasive.html, letzter Abruf: 07.07.2014.

[IFAK09] IFAK Mobile Scan 2009 – eReport Basic, http://www.izmf.de/sites/default/files/download/Studien/ifak_mobile-scan-2009_ereport-basic.pdf, letzter Abruf: 07.07.2014.

[ImDS02] T. Imamura, B. Dillaway, E. Simon: XML Encryption Syntax and Processing. W3C Recommendation, 2002, http://www.w3.org/TR/xmlenc-core/, letzter Abruf: 07.07.2014.

[JHHS00] J. Jing, K. Huff, B. Hurwitz, H. Sinha, B. Robinson, M. Feblowitz: WHAM: Supporting Mobile Workforce and Applications in Workflow Environments. Research Issues in Data Engineering (RIDE), San Diego, California, USA, 2000, S. 31-38.

[JHSH99] J. Jing, K. Huff, H. Sinha, B. Hurwitz, B. Robinson: Workflow and Application Adaptations in Mobile Environments. In: Mobile Computing Systems and Applications, 1999, Proceedings. WMCSA '99. IEEE Workshop, New Orleans, LA, USA, 1999, S. 62-69.

[Josu08] N. Josuttis: SOA in der Praxis – System-Design für verteilte Geschäftsprozesse. dpunkt.verlag GmbH, 2008.

[KANO10] T. J. S. Khanzada, A. R. Ali, S. A. Napoleon, A. S. Omar: Use of super resolution algorithms for indoor positioning keeping novel designed WLAN signal structure. G. Saake and V. Küppen (Hrsg.):Proceedings of the

First International Workshop on Digital Engineering (IWDE '10), ACM, New York, USA, 2010, S. 59-63.

[KaSo01] M. Kakihara, C. Sorensen: Expanding the mobility concept. ACM SIGGROUP Bulletin, Vol. 22, Iss. 3, 2001, S. 33-37.

[KaSo04] M. Kakihara, C. Sorensen: Practiving Mobile Professional Work: Tales of Locational, Operational and Interactional Mobility. The Journal of Policy, Regulation and Strategy for Telecommunication, Information and Media, Vol. 6, Nr. 3, 2004, S. 180-187.

[KayA72] A. C. Kay: A Personal Computer for Children of All Ages. Xerox PARC. August, 1972.

[KBSN07] M. L. M. Kameng, V. Brandt, T. Steckel, W. Nüßer: Prozessintegration mobiler Landmaschinen mittels automatisch erzeugter BPEL-Prozesse. Wirtschaftsinformatik, Vieweg Verlag, Vol. 49, Iss. 4., 2007, S. 289-294.

[Kech09] C. Kecher: UML 2.0 - Das umfassende Handbuch. 3. Auflage, Galileo Press, Bonn, 2009.

[KeNS92] G. Keller, M. Nüttgens, A. Scheer: Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). A. Scheer (Hrsg.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89, Saarbrücken, 1992.

[KhPW03] D. Khodawandi, K. Pousttchi, C. Winnewisser: Mobile Technologie braucht neue Geschäftsprozesse. 2003. http://wi2.wiwi.uni-augsburg.de/pics/mobile/Uni-Augsburg_WI2-MC_MP-16-11.pdf, letzter Abruf: 27.02.2013,

[KiHe06] W. R. King, J. He: A meta-analysis of the technology acceptance model. In: Information and Management, Vol. 43, Iss. 6, 2006, S. 740–755.

[KKPB15] S. Klepper, S. Krusche, S. Peters, B. Bruegge, L. Alperowitz: Introducing Continuous Delivery of Mobile Apps in a Corporate Environment: A Case Study. 2015 IEEE/ACM 2nd International Workshop on Rapid Continuous Software Engineering, 2015, S. 5-11.

[Klei95] L. Kleinrock: Nomadic Computing — an Opportunity. ACM SIGCOMM, Computer Communication Review, Vol. 25, No. 1, 1995, S. 36–40.

[KöGr04a] A. Köhler, V. Gruhn: Lösungsansätze für verteilte mobile Geschäfts-prozesse. P. Horster (Hrsg.): Elektronische Geschäftsprozesse, Klagenfurt, Österreich, 2004, S. 243-255.

[KöGr04b] A. Köhler, V. Gruhn: Mobile Process Landscaping am Beispiel von Vertriebsprozessen in der Assekuranz. Mobile Economy - Transaktionen, Prozesse, Anwendungen und Dienste. K. Pousttchi, K. Turowski (Hrsg.): Proceedings zum 4. Workshop Mobile Commerce, Buchreihe LNI, GI-Lecture Notes in Informatics, Augsburg, Deutschland, 2004, S. 12-24.

[KoRR12]J. Kolb, B. Rudner, M. Reichert: Towards Gesture-Based Process Modeling on Multi-touch Devices. Advanced Information Systems Engineering Workshops, Lecture Notes in Business Information Processing, Vol. 112, 2012, S. 280-293.

[Kosc07] A. Koschmider: Ähnlichkeitsbasierte Modellierungsunterstützung für Geschäftsprozesse. Dissertation, Universitätsverlag Karlsruhe, 2007.

[KrLj00] S. Kristoffersen, F. Ljungberg: Mobility: From stationary to mobile work. Planet Internet, Schweden, 2000, S. 41-64.

[KRNB94] K. Kuhn, M. Reichert, M. Nathe, T. Beuter, C. Heinlein, P. Dadam: A Conceptual Approach to an Open Hospital Information System. Proceedings of the 12th International Congress on Medical Informatics, MIE'94, Lissabon, Portugal, 1994, S. 373-378.

[Kruch95] P. Kruchten: The 4+1 View Model of Architecture. IEEE Software, Vol. 12, No. 6, Nov. 1995, S. 42-50.

[Kuep05] A. Küpper: Location-based Services: Fundamentals and Operation. John Wiley & Sons, Chichester, U.K., 2005.

[KuGo12] M. Kuhlmann, M. Gogolla: From UML and OCL to Relational Logic and Back. Model Driven Engineering Languages and Systems, Lecture Notes in Computer Science, Vol. 7590, 2012, S. 415-431.

[KuXi16] A. Kumar, B. Xie: Handbook of Mobile Systems Applications and Services, CRC Press, 2016.

[KuZL06] C. P. Kunze, S. Zaplata, W. Lamersdorf: Mobile Process. Description and Execution, DAIS '06, 2006.

[KuZL07] C. P. Kunze, S. Zaplata, W. Lamersdorf: Mobile Processes: Enhancing Cooperation in Distributed Mobile Environments. Journal of Computers, Vol. 2, 2007, S. 1-11.

[LeAH08] M. d. Leoni, W. M. P. v. der Aalst, A. H. M. t. Hofstede: Visual Support for Work Assignment in Process-Aware Information Systems. Proceedings of Business Process Management, 2008, S. 67-83.

[Lehn09] F. Lehner: Wissensmanagement: Grundlagen, Methoden und technische Unterstützung. 3. Auflage, Hanser Verlag, München, 2009.

[LeLN11] T. van Lessen, D. Lübke, J. Nitzsche: Geschäftsprozesse automatisieren mit BPEL, dpunkt Verlag, Januar, 2011.

[Link05] M. Linke: Das OSGI-Framework. Java-Spektrum, Ausgabe 2, Java für mobile und eingebettete Systeme, 2005, S. 35-38.

[LöPe01] H. Löbler, H. Petersohn: Kundensegmentierung im Automobilhandel zur Verbesserung der Marktbearbeitung. K. Wilde (Hrsg.): Handbuch Data Mining im Marketing, Knowledge Discovery in Marketing Databases, Vieweg, Wiesbaden, 2001, S. 623-641.

[LyYo02] K. Lyytinen, Y. Yoo: Issues and Challenges in Ubiqitous Computing. Communications of the ACM, Vol. 45, Nr. 12, 2002.

[Mack99] J. Macker: Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations. 1999.

[MaFl10] F. Mattern, C. Flörkemeier: Vom Internet der Computer zum Internet der Dinge. Informatik Spektrum, Bd. 33, Nr. 2, 2010, S. 107-121.

[MaGa99]Y. Malhotra, D. F. Galletta: Extending the technology acceptance model to account for social influence: theoretical bases and empirical validation. Proceedings of the 32nd Annual Hawaii International Conference on System Sciences, IEEE Computer Society Press, Los Alamitos, 1999, S. 1006-1020.

[MaMo04] A. Maurino, S. Modafferi: Workflow Management in Mobile Environments. Ubiquitous Mobile Information and Collaboration Systems, Lecture Notes in Computer Science, Vol. 3272, 2005, S. 83-95.

[Mans09] W. Mansfeld: Satellitenortung und Navigation; Grundlagen, Wirkungsweise und Anwendung globaler Satellitennavigationssysteme. 3. Auflage, Vieweg+Teubner Verlag, GWV Fachverlage GmbH, Wiesbaden, 2009.

[McDu07]M. McDonald, I. Dunbar: Market segmentation: how to do it, how to profit from it. Butterworth-Heinemann, Oxford, 2007.

[Mevi06] M. Mevius: Kennzahlenbasiertes Management von Geschäftsprozessen mit Petri-Netzen. Dr. Hut Verlag, 2006.

[MiLa07] N. Mitra, Y. Lafon: SOAP Version 1.2 Part 0: Primer (Second Edition). In: http://www.w3.org/TR/2007/REC-soap12-part0-20070427, 2007, letzter Abruf: 07.07.2014.

[MnSY04] A. B. Mnaouer, A. Shekhar, Z. Yi Liang: A Generic Framework for Rapid Application Development of Mobile Web Services with Dynamic Workflow Management. Proceedings of the 2004 IEEE International Conference on Services Computing (SCC '04), 2004, S. 165-171.

[MoBD12] S. Mora, A. Boron, M. Divitini: CroMAR: Mobile Augmented Reality for Supporting Reflection on Crowd Management. International Journal of Mobile Human Computer Interaction (IJMHCI), Vol. 4, Iss. 2, 2012, S. 88-101.

[Mone07] V. Monebhurrun: Fast Global Characterization of 2G and 3G Mobile Phones using Reverberation Chamber. The Second European Conference on Antennas and Propagation, 2007, S. 1-7.

[Mueh02] M. zur Muehlen: Workfow-based Process Controlling – Foundation, Design, and Application of Workflow-driven Process Information Systems, Logos Verlag Berlin, 2002.

[Müll05] J. Müller: Workflow-based Integration. Springer-Verlag, 2005.

[MuWL00] S. Müller-Wilken, F. Wienberg, W. Lamerdorf: On Integrating Mobile Devices into a Workflow Management Scenario. Proceedings of the 11th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, Washington, DC, USA, 2000, S. 186-191.

[MWAB05] A. McBratney, B. Whelan, T. Ancev, J. Bouma: Future Directions of Precision Agriculture. Precision Agriculture, Vol. 6, Iss. 1, 2005, S. 7-23.

[NePW02] S. Neumann, C. Probst, C. Wernsmann: Kontinuierliches Prozessmanagement. In: Prozessmanagement. J. Becker, M. Kugeler, M. Rosemann (Hrsg.): 3rd Edition, Berlin, 2002, S. 297-323.

[NeSt07] T. Neubauer, C. Stummer: Extending business process management to determine efficient it investments. Proceedings of the 2007 ACM symposium on Applied computing (SAC'07), Seoul, Korea, 2007, S. 1250-1256.

[NKMH04] A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker: Web Services Security: SOAP Message Security 1.1 (WS-Security 2004). OASIS Standard Specification, 2004, http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf, letzter Abruf: 07.07.2014.

[NyPT05] H. Nysveen, P. E. Pedersen, H. Thorbjørnsen: Explaining intention to use mobile chat services: moderating effects of gender. Journal of Consumer Marketing, Vol. 22, Iss. 5, 2005, S. 247-256.

[OASI06] Organization for the Advancement of Structured Information Standards (OASIS): Web Services Human Task Version 1.1. 2006, http://docs.oasis-open.org/bpel4people/ws-humantask-1.1-spec-cd-06.pdf, letzter Abruf: 07.07.2014.

[OASI07] Organization for the Advancement of Structured Information Standards (OASIS): Web Services Business Process Execution Language Version 2.0. 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html, letzter Abruf: 07.07.2014.

[Ober90] A. Oberweis: Zeitstrukturen für Informationssysteme - Inauguraldissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften der Universität Mannheim, 1990.

[Ober96] A. Oberweis: Modellierung und Ausführung von Workflows mit Petri-Netzen. B. G. Teubner Verlag, Stuttgart, Leipzig, 1996.

[ObMG09] Object Management Group (OMG): Business Process Model and Notation (BPMN) – Version 1.2, http://doc.omg.org/formal/09-01-03.pdf, 2009, letzter Abruf am 22.10.2015.

[ObMG10] Object Management Group (OMG): Object Constraint Language Version 2.2; 2010, http://www.omg.org/spec/OCL/2.2/, letzter Abruf: 07.07.2014.

[ObMG10a] Object Management Group (OMG): Business Process Model and Notation (BPMN) – Version 2.0, http://www.omg.org/spec/BPMN/2.0/, 2010, letzter Abruf am 22.10.2015.

[ODHA06] C. Ouyang, M. Dumas, A. H. M. ter Hofstede, W. M. P. van der Aalst: From BPMN Process Models to BPEL Web Services. Web Services, IEEE International Conference on, IEEE International Conference on Web Services (ICWS'06), 2006, S. 285-292.

[Oest09] B. Oestereich: Die UML-Kurzreferenz 2.3 für die Praxis - Kurz, bündig, ballastfrei. 5. Auflage, Oldenbourg Wissenschaftsverlag, München, 2009.

[OMGS07] Object Management Group: OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.1.2. URL: http://www.omg.org/spec/UML/2.1.2/, letzter Abruf: 07.07.2014.

[OpGC07] Open Geospatial Consortium Inc.: Geography Markup Language (GML) Version 3.2.1. 2007, http://www.opengeospatial.org/standards/gml/, letzter Abruf: 07.07.2014.

[Ostr02] J. Ostrem: Palm OS User Interface Guidelines, Doc. Nb. 3101-001, 2002, http://devel.archefire.org/docs/PalmOS/5/Palm%20OS%20UI%20Guidelines.pdf, letzter Abruf: 08.04.2011.

[PaCh02] A. Pavlou, L. Chai: What drives electronic commerce across cultures? A cross- cultural empirical investigation of the theory of planned behavior. Journal of Electronic Commerce Research, Vol. 3, Iss. 4, 2002, S. 240-253.

[PaCh07] L. Pajunen, S. Chande: Developing Workflow Engine for Mobile Devices. In: 11th IEEE International Enterprise Distributed Object Computing Conference, 2007, S. 279.

[Paga04] M. Pagani: Determinants of adoption of third generation mobile mul-timedia services. In: Journal of Interactive Marketing, Vol. 18, Iss. 3, 2004, S. 46-59.

[PaHC06] S.-H. Park, Y.-J. Han, T.-M. Chung: Context-Role Based Access Control for Context-Aware Application. High Performance Computing and Communications, LNCS, Vol. 4208, 2006, S. 572-580.

[PaPC97] S. Paul, E. Park, K. Jarir, K. Chaar: RainMan: A Workflow System for the Internet. USENIX Symposium on Internet Technologies and Systems, 1997.

[Pavl02] P. A. Pavlou: What Drives Electronic Commerce? A Theory of Planned Behavior Perspective. Proceedings of the Academy of Management Conference, Denver, CO, 2002.

[PaVP17] P. Pandey, H. Viswanathan, D. Pompili: Robust Orchestration of Concurrent Application Workflows in Mobile Device Clouds. CoRR, Vol. abs/1702.02903, 2017.

[PCDG02] C. Pappas, E. Coscia, G. Dodero, V. Gianuzzi, M. Earney: A Mobile E-Health System Based on Workflow Automation Tools. Proceedings of the 15th IEEE Symposium on Computer-Based Medical Systems (CBMS '02). IEEE Computer Society, Washington D.C., USA, 2002, S. 271-277.

[PCMa10]Smartphone definition von PC Magazine Encyclopedia, http://www.pcmag.com/encyclopedia_term/0%2C2542%2Ct%3DSmartphone&i%3D51537%2C00.asp, letzter Abruf: 07.07.2014.

[PEFS03] P. Persson, F. Espinoza, P. Fagerberg, A. Sandin, R. Cöster: Geo Notes: A Location-based Information System for Public Spaces. Designing information spaces: the social navigation approach, Springer Verlag London, 2003, S. 151-173.

[Pete77] J. Peterson: Petri Nets. ACM Computing Surveys, 1977, Vol. 9, Iss. 3, S. 223-252.

[PiRo95] A. Picot, P. Rohrbach: Organisatorische Aspekte von Workflow-Management-Systemen, Information Management 1, 1995.

[POSB01] M. Perry, K. O ʻhara, A. Sellen, B. Brown, R. Harper: Dealing with mobility: Understanding access anytime, anywhere. ACM Transactions on Human-Computer Interactions, Vol. 8, Iss. 4, 2001, S. 323-347.

[PSGS00] C. Peersman, S. Cvetkovic, P. Griffiths, H. Spear: The Global System for Mobile Communications Short Message Service. Personal Communications, IEEE, Vol. 7, Nr. 3, 2000, S. 15-23.

[PTDL08] M. P. Papazoglou, P. Traverso, S. Dustdar, F. Leymann: Service-oriented Computing: A Research Roadmap. International Journal of Cooperative Information Systems (IJCIS), Vol. 17, Iss. 2, 2008, S. 223-255.

[PZZZ13] B. Pi, G. Zou, C. Zhong, J. Zhang, H. Yu: Decentralized Workflow Execution on Mobile Phone. In: Proceedings of the 2013 IEEE Second International Conference on Mobile Services (MS '13). IEEE Computer Society, Washington D.C., USA, 2013, S. 78-85.

[Quir06] O. Quiring: Methodische Aspekte der Akzeptanzforschung bei interaktiven Medientechnologien. Münchener Beiträge zur Kommuni-kationswissenschaft, Nr. 6, München, 2006, S. 1-29.

[Ramc74] C. Ramchandani: Analysis of Asynchronous Concurrent Systems by Timed Petri Nets. Technical Report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1974, http://dspace.mit.edu/handle/1721.1/13739, letzter Abruf: 07.07.2014.

[Rask01] J. Raskin: Das intelligente Interface. Neue Ansätze für die Entwicklung interaktiver Benutzerschnittstellen, Addison-Wesley, 1. Auflage, 2001.

[RDUD13] M. La Rosa, M. Dumas, R. Uba, R. Dijkman: Business Process Model Merging: An Approach to Business Process Consolidation. ACM Trans. Softw. Eng. Methodol., Vol. 22, Iss. 2, Article 11, 2013.

[Reck10] J. Recker: Opportunities and constraints: the current struggle with BPMN. Business Process Management Journal, Vol. 16, Iss. 1, 2010, S. 181-201.

[Rein03] J. Reini: MMS (Multimedia Messaging Service) enabled web map dispatching solution for location enhanced fieldwork management. Proceedings of the 9th Scandinavian Research Conference on Geographical Information Science, Department of Surveying, Helsinki, University of Technology, 2003, S. 31-40.

[Reis10a] W. Reisig: Petrinetze - Modellierungstechnik, Analysemethoden, Fallstudien. Vieweg+Teubner Verlag, Springer Fachmedien, Wiesbaden, 2010.

[Reis10b] W. Reisig: Carl Adam Petri 1926-2010 - Visionär und bedeutender Wissenschaftler. Informatik-Spektrum, 2010, Vol. 33, Nb. 5, S. 514-521.

[ReRo98] W. Reisig, G. Rozenberg: Lectures on Petri Nets: Basic Models, Band 1491, Lecture Notes in Computer Science. Springer, 1. Auflage, 1998.

[RGKR07] J. Raper, G. Gartner, H. Karimi, C. Rizos: A critical evaluation of location based services and their potential. Journal of Location Based Services, Vol. 1, No. 1, März, 2007, S. 5-45.

[RHAM06] N. Russell, A. ter Hofstede, W. M. P. van der Aalst, N. Mulyar: WORKFLOWCONTROL-FLOWPATTERNS - A Revised View - BPM Center Report BPM-06-22, 2006, http://wwwis.win.tue.nl/~wvdaalst/BPMcenter/reports/2006/BPM-06-22.pdf, letzter Abruf: 07.07.2014.

[Roge03] E. M. Rogers: Diffusion of Innovations. 5th Edition, The Free Press, New York, London, 2003.

[RoHS06] G. C. Roman, R. Handorean, R. Sen: Tuple space coordination across space and time. Coordination Models and Languages, Vol. 4038, 2006, S. 266-280.

[Roth05] J. Roth: Mobile Computing: Grundlagen, Technik, Konzepte. 2. Auflage. Heidelberg, dpunkt.verlag, 2005.

[RRLH14] M. Rahimi, J. Ren, C. Liu, V. Harold, A. Vasilakos, N. Venkatasubramanian: Mobile Cloud Computing: A Survey, State of Art and Future Directions. Mobile Networks and Applications, New York, 2014, S. 133-143.

[Rump04] B. Rumpe: Modellierung mit UML - Sprache, Konzepte und Methodik. Springer Verlag, Berlin, 2004.

[SaSa10] G. Saake, K.-U. Sattler: Algorithmen und Datenstrukturen: Eine Einführung mit Java. dpunkt Verlag, 4. Auflage, 2010.

[ScBG99] A. Schmidt, M. Beigl, H.-W. Gellersen: There is more to context than location. Computers & Graphics, Vol. 23, Iss. 6, 1999, S. 893-901.

[ScBH05] E. Scornavacca, S. J. Barnes, S. L. Huff: Mobile Business Research, 2000-2004: Emergence, Current Status, and Future Opportunities. Proceedings of ECIS 2005, Paper 59, 2005.

[ScDe08] G. Schiefer, M. Decker: Taxonomy for Mobile Terminals - A Selective Classification Scheme. J. Filipe, D. Marca, B. Shishkov, M. van Sinderen (Hrsg.): Proceedings of the International Conference on e-Business (ICE-B 2008), Porto, Portugal, INSTICC Press, 2008, S. 255-258.

[Schi03] J. Schiller: Mobilkommunikation. Pearson Verlag, 2. Auflage, München, 2003.

[ScTA05] A. Scheer, O. Thomas, O. Adam: Process Modeling using Event-Driven Process Chains. M. Dumas, W. van der Aalst, A. ter Hofstede

(Hrsg.): Process-Aware Information Systems - Bridging People and Software through Process Technology, John Wiley & Sons, Hoboken, NJ, USA, 2005, S. 119-145.

[ScVo04] J. Schiller, A. Voisard: Location-based services. Morgan Kaufmann Publishers, San Francisco, CA, USA, 2004.

[SeAP06] K. Sethom, H. Afifi, G. Pujolle: A distributed and secured architecture to enhance smooth handoffs in wide area wireless IP infrastructures. Newsletter ACM SIGMOEILE, Mobile Computation and Communication Review, Vol. 10, 2006, S. 46-57.

[SeWC10] A. Sell, P. Walden, C. Carlsson: Are you Efficient, Trendy or Skillful? An exploratory segmentation of mobile service users. Mobile Business and 2010 Ninth Global Mobility Roundtable (ICBM-GMR), 2010, S. 116-123.

[SHHR07] R. Sen, G. Hackmann, M. Haitjema, G.-C. Roman, C. Gill: Extending BPEL for Interoperable Pervasive Computing. International Conference on Pervasive Services, IEEE International Conference on Pervasive Services, Vol. 4467, 2007, S. 249-267.

[Sieg08] P. F. Siegert: Die Geschichte der E-Mail. Erfolg und Krise eines Massenmediums, Transcript Verlag, 2008.

[Sigu01] J. Sigurdson: WAP OFF - Origin, Failure and Future. Working Paper No. 135, Originally prepared for The Japanese-European Technology Studies (JETS) at University of Edinburgh, 2001.

[Silv11] B. Silver: BPMN Method and Style, 2nd Edition, with BPMN Implementer's Guide: A structured approach for business process modeling and implementation using BPMN 2.0, Cody-Cassidy Press, 2011.

[SLBG07]R. Schuler, N. Laws, S. Bajaj, S. Grandhi, Q. Jones: Finding your Way with CampusWiki: A Location-Aware Wiki. Proceedings of CHI 2007, San Jose, CA, USA, 2007, S. 2639-2644.

[SMRL97] T. Starner, S. Mann, S. Rhodes, J. Levine, J. Healey, D. Kirsch, R. Pi-card, A. Pentland: Augmented Reality Through Wearable Computing. Presence: Teleoperators and Virtual Environments, Vol. 6, Nr. 4, 1977, S. 386-398.

[SOSF04] S. Sadiq, M. Orlowska, W. Sadiq, C. Foulger: Data flow and validation in workflow modelling. K.-D. Schewe, H. Williams (Hrsg.): Proceedings of the 15th Australasian database conference, Vol. 27. Australian Computer Society, Inc., Darlinghurst, Australia, 2004, S. 207-214.

[Spör09] S. Spörrer: Content Management Systeme: Begriffsstruktur und Praxisbeispiel. Kölner Wissenschaftsverlag, 1. Auflage, 2009.

[SpSc08] M. Speck, N. Schnetgöke: Sollmodellierung und Prozessoptimierung. J. Becker, M. Kugeler, M. Rosemann (Hrsg.): Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung. 6. Auflage, Springer Verlag, Berlin, Heidelberg, 2008, S. 185-220.

[SRWN12] S. Smirnov, H. A. Reijers, M. Weske, T. Nugteren: Business process model abstraction: a definition, catalog, and survey. Distributed and Parallel Databases, Vol. 30, Iss. 1, 2012, S. 63-99.

[SSPM14] J. Schobel, M. Schickler, R. Pryss, F. Maier, M. Reichert: Towards Process-Driven Mobile Data Collection Applications: Requirements, Challenges, Lessons Learned. 10th International Conference on Web Information Systems and Technologies (WEBIST 2014), Special Session on Business Apps, 2014, S. 371-382.

[Stach73] H. Stachowiak: Allgemeine Modelltheorie. Springer Vienna, 1973.

[Staf02] J. V. Stafford: Implementing Precision Agriculture in the 21st Century. Journal of Agricultural Engineering Research, Vol. 76, Iss. 3, 2000, S. 267-275.

[Stapl07] T. Stapelkamp: Screen- und Interfacedesign. Springer-Verlag, Berlin, 2007.

[StDe95] R. Striemer, W. Deiters: Workflow Management – Erfolgreiche Planung und Durchführung von Workflow-Projekten. Arbeitsbericht des Frauenhofer Instituts für Software- und Systemtechnik, Dortmund, 1995.

[StKn01] H. Stormer, K. Knorr: PDA- and Agent-based Execution of Workflow Tasks. Proceedings of Informatik 2001, 2001, S. 968-973.

[StLi04] T. Strang, C. Linnhoff-Popien: A Context Modeling Survey. In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham, England, 2004.

[Suth68] I. E. Sutherland: A head-mounted three dimensional display. Proceedings of the Fall Joint Computer Conference, San Francisco, USA, 1968, S. 757-764.

[SVOK11] F. Schönthaler, G. Vossen, A. Oberweis, T. Karle: Geschäftsprozesse für Business Communities – Modellierungssprachen, Methoden, Werkzeuge. Oldenbourg Verlag München, 2011.

[SWNN06] C.-F. Sørensen, A. I. Wang, J. O. Nødtvedt, M. H. Nguyen: Requirements for Context-Aware, Mobile Workflow Systems. Software Engineering and Applications, 2006, S. 529-536.

[TaLi08] L. Taxen, J. Lilliesköld: Images as action instruments in complex projects. International Journal of Project Management, Vol. 26, Iss. 5, July 2008, S. 527-536.

[Tane09] Andrew S. Tanenbaum: Moderne Betriebssysteme. Pearson Studium, 3. Auflage, 2009.

[Tous89] G. Toussaint: On Separating Two Simple Polygons by a Single Translation. Discrete Computational Geometry, Vol. 4, Nb. 1, 1989, S. 265-278.

[TRPC09]N. O. Tippenhauer, K. B. Rasmussen, Christina Pöpper, S. Capkun: Attacks on public WLAN-based positioning systems. Proceedings of

the 7th international conference on Mobile systems, applications, and services (MobiSys '09). ACM, NY, USA, 2009, S. 29-40.

[TsCh08] C.-H. Tseng, S.-T. Cheng: Location management scheme with WLAN positioning algorithm for integrated wireless networks. Computing Communication, Vol. 31, Iss. 18, 2008, S. 4304-4311.

[TuJo05] H. Tummala, J. Jones: Developing Spatially-Aware Content Management Systems for Dynamic, Location-Specific Information in Mobile Environments. Proceedings of the 3rd ACM international Workshop on Wireless mobile applications and services on WLAN hotspots, Köln, Deutschland, 2005, S. 14-22.

[TuPo04] K. Turowski; K. Pousttchi: Mobile Commerce: Grundlagen und Techniken. Springer Verlag Berlin Heidelberg New York, 2004.

[VeDa00] V. Venkatesh, F. D. Davis: A Theoretical Extension of the Technolo-gy Acceptance Model: Four Longitudinal Field Studies. Management Science, Vol. 46, Iss. 2, 2000, S. 186-204.

[VePo02] C. Ververidis, G. C. Polyzos: Mobile Marketing using a Location Based Service. In: Proceedings of the 1st International Conference on Mobile-Business, Athen, Griechenland, Juli, 2002.

[VMDD03] V. Venkatesh, M. G. Morris, G. B. Davis, F. D. Davis: User acceptance of information technology: Towards a unified view. MIS Quarterly, Vol. 27, Iss. 3, 2003, S. 425-478.

[Voge10] M. Vogel: Integration von Ortseinschränkungen in Geschäftsprozessmodelle. Diplomarbeit, 2010.

[WaDi05] M. Wallbaum, S. Diepolder: Benchmarking Wireless LAN Location Systems Wireless LAN Location Systems. Proceedings of the Second IEEE International Workshop on Mobile Commerce and Services (WMCS '05), IEEE Computer Society, Washington, DC, USA, 2005, S. 42-51.

[WaKl03] J. Warmer, A. Kleppe: The Object Constraint Language - Getting Your Models Ready for MDA. Addison-Wesley, Boston, MA, USA, 2003.

[Wang98] J. Wang: Timed Petri – Theory and Application Nets. Kluwer Academic Publishers, 1998.

[WAPF01] Wireless Application Protocol Forum Ltd.: WAP Architecture, 2001.

[WDGW08] M. Weidlich, G. Decker, A. Großkopf, M. Weske: BPEL to BPMN: The Myth of a Straight-Forward Mapping. On The Move To Meaningful Internet Systems (OTM 2008), LNCS, Vol. 5331, 2008, S. 265-282.

[Weile01] A. Weilenmann, C. Larsson: Local use and sharing of mobile phones. B. Brown, N. Green, R. Harper (Hrsg.): Wireless World. Social and Interactional Aspects of the Mobile Age, London, Springer-Verlag, 2001, S. 92-107.

[Weile03] A. Weilenmann: Doing Mobility. Doctoral Thesis, 2003. http://gupea.ub.gu.se/bitstream/2077/910/1/gupea_2077_910_1.pdf, letzter Abruf: 07.07.2014.

[Wein05] R. Weinstein: RFID: A Technical Overview and Its Application to the Enterprise. IT Professional, Vol. 7, No. 3, 2005, S. 27-33.

[Weis91] M. Weiser: The Computer for the Twenty-First Century. Scientific American, Vol. 265, Iss. 3, 1991, S. 94-104.

[Wess02] I. Wessels: GUI-Design. Hanser-Verlag, 2. Auflage, München, 2002.

[WfMC08] Workflow Management Coalition (WfMC): Process Defnition Interface XML Process Definition Language. Document Number WFMC-TC-1025 Document Status Final Approved Version 2.1a. Workflow Management Coalition, Winchester, UK, 2008.

[WfMC99] Workflow Management Coalition: Terminology & Glossary, Version 3.0. http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf, letzter Abruf: 07.07.2014.

[Whit04] S. A. White: Introduction to BPMN. IBM Cooperation, 2004, S. 2008-029.

[Whit05] S. A. White: Using BPMN to Model a BPEL Process. BPTrends, 2005, http://bptrends.com/publicationfiles/03-05%20WP%20Mapping%20BPMN%20to%20BPEL-%20White.pdf, letzter Abruf: 07.07. 2014.

[WhMi08] S. A. White, D. Miers: BPMN Modeling and Reference Guide – Understanding and Using BPMN – Develop rigorous yet understandable graphical representations of business processes. Future Strategies Inc., Lighthouse Pt, FL, USA, 2008.

[WiBP11] H. Winkelhofer, C. Bauer, D. Palalic: MMA Austria Communication Report, Mobile Marketing Association Austria, 2012. http://startmobile.mmaaustria.at/html/img/pool/2012_MMA_Communication_Report_kostenfreie_Praesentation.pdf, letzter Abruf: 07.07.2014.

[Wils03] T.-A. Wilska: Mobile Phone Use as Part of Young People's Consumption Styles. Journal of Consumer Policy, Vol. 26, Iss. 4, 2003, S. 441-463.

[Wohl04] J. Wohlfahrt: Akzeptanz und Wirkungen von Mobile-Business-Anwendungen. Dissertation, Kovac-Verlag, Hamburg, 2004.

[WuWa05] J.-H. Wu, S.-C. Wang: What drives mobile commerce?: An empirical evaluation of the revised technology acceptance model. Information & Management, Vol. 42, Iss. 5, 2005, S. 719-729.

[WWWC04] World Wide Web Consortium (W3C): XML Schema 1.1. 2004, http://www.w3.org/XML/Schema, letzter Abruf: 07.07.2014.

[ZaBV09] S. Zaplata, D. Bade, A. Vilenica: Service-based Interactive Workflows for Mobile Environments. H. R. Hansen, D. Karagiannis, H.-G. Fill (Hrsg.): Business Services: Konzepte, Technologien, Anwendungen – 9. Internationale Tagung Wirtschaftsinformatik (WI 2009), Österreichische Computer Gesellschaft, 2009, S. 631-640.

[Zand09] P.A. Zandbergen: Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning. Transactions in GIS, Vol. 13, S. 5-25.

[Zimm99] J. B. Zimmermann: Mobile Computing: Characteristics, Business Benefits, and the Mobile Framework, Technical Report INSS 690 CC, University of Maryland, April 1999.

[ZoKJ02] P. Zoche, S. Kompeler, M. Joepgen: Virtuelle Mobilität: Ein Phänomen mit physischen Konsequenzen?. Springer, Berlin, 2002.

[Zuku97] O. Zukunft: Adaptation in Mobile Workflow Management Systems. Springer Verlag, Vol. 1, Iss. 3, 1997.

[Zuku99] O. Zukunft: Waterloo-M: Ein datenbankbasiertes Workflow-Managementsystem für mobile Benutzer. Informatik aktuell, Datenbanksysteme in Büro, Technik und Wissenschaft, Springer, 1999, S. 410-428.

Stichwortverzeichnis

Mobile Kontexte

Ortseinschränkungen

WfMS

Bestehende Geschäftsmodellierungssprachen (bzw. deren Erweiterungen) besitzen Zeit-, Identität- und Aktivitätsbezogene Kontexte bzw. Objekte zur Modellierung von Geschäftsprozessen. Zur Unterstützung eines sogenannten Context Aware Service werden ortsbezogene Kontexte benötigt, damit die Anwendung bzw. der Prozess auf Basis dessen gesteuert werden kann. Für die Integration von Ortsbezügen in Workflow-Modelle wird ein Konzept auf Basis von Ortseinschränkungen entwickelt. Mit der Erweiterung der Petri-Netze und der BPMN 2.0 wird gezeigt, dass dieses Konzept auf weitere Sprachen innerhalb des Geschäftsprozessmanagements angewendet werden kann. Um einen möglichst hohen Grad an Automatisierung zu erzielen, werden in vorliegender Arbeit bestehende WfMS zu mobilen WfMS erweitert.

ISBN 978-3-7315-07543

Gedruckt auf FSC-zertifiziertem Papier