Ermittlung des Industrialisierungsgrades der Anwendungsentwicklung in der Fertigungsindustrie

Inaugural-Dissertation

zur Erlangung des akademischen Grades eines

Doktors der Wirtschaftswissenschaften (Dr. rer. pol.)

an der Wirtschaftswissenschaftlichen Fakultät

der Universität Passau

Vorgelegt von

Dipl.-Inform. Alexander Hildenbrand

Passau

Februar 2010

Gutachter:

Prof. Dr. Peter Kleinschmidt

Prof. Dr. Franz Lehner

Tag der letzten Fachprüfung des Rigorosums:

30. Juli 2010

Danksagung

An dieser Stelle möchte ich allen danken, die zum Gelingen dieser Arbeit beigetragen haben. Insbesondere möchte ich mich bei meinem Doktorvater, Professor Dr. Peter Kleinschmidt, bedanken, dessen Geduld, Anregungen und Kritik viel zu dieser Arbeit beigetragen haben. Professor Dr. Franz Lehner danke ich für die Übernahme des Zweitgutachtens.

Vorwort

Die Industrialisierung ist mittlerweile auch in der Softwarebranche angekommen. In dieser Arbeit wird ein Modell vorgestellt, das eine Einstufung der verschiedenen Ausprägungen der Industrialisierung in der Anwendungsentwicklung ermöglicht.

Ferner wird auf Basis dieses Modells ein Bewertungsverfahren vorgestellt, welches einem Unternehmen erlaubt, den Grad an Industrialisierung in der Entwicklung von Anwendungen zu ermitteln. Das Ergebnis dieser Ermittlung dient dem Unternehmen als Basis für eine Entscheidung zur Optimierung der Softwareentwicklung. Mit der Festlegung eines Sollgrades können zudem die Ziele für die Verbesserung der Anwendungsentwicklung definiert werden. Eine regelmäßige Ermittlung des Industrialisierungsgrades ermöglicht dabei die Dokumentation des aktuellen Fortschrittes.

Eine praktische Bewertung eines Unternehmensbereiches wird im Rahmen einer Fallstudie exemplarisch durchgeführt.

Zur Wahrung der Geheimhaltung der Informationen des evaluierten Unternehmens wurden die Angaben anonymisiert und verändert. Diese Anpassungen haben jedoch keinen Einfluss auf die Aussagen dieser Arbeit.

Der Inhalt dieser Arbeit spricht Frauen und Männer gleichermaßen an. Zur besseren Lesbarkeit wird aber nur die männliche Sprachform verwendet.

Inhaltsverzeichnis

Danksagung **3**

Vorwort **4**

Inhaltsverzeichnis **5**

Abbildungsverzeichnis **9**

Tabellenverzeichnis **10**

Definitionsverzeichnis **11**

Abkürzungsverzeichnis **12**

1 Einleitung **14**

1.1 Motivation 16

1.2 Zielsetzung dieser Arbeit 18

1.2.1 Abgrenzung der Aufgabenstellung 19

1.3 Aufbau der Arbeit 21

2 Grundlagen **23**

2.1 Industrialisierung in der IT 23

2.2 IT-Leistungserstellung als Fertigungsprozess 26

2.2.1 Grenzen der Industrialisierung 30

2.3 Entwicklung von Anwendungen 31

2.3.1 Entwicklungsprozess 33

2.3.2 Artefakte 35

2.3.3 Entwicklungsmethode 38

2.3.4 Entwicklungsinfrastruktur 39

2.3.5 Laufzeitinfrastruktur 41

2.3.6 Gremien 42

2.3.7 Unterstützende Prozesse 43

2.4 Faktoren der Anwendungsentwicklung 43

2.5 Modell 46

2.6 Domänen 48

2.7 Klassifikation von Anwendungen 50

2.7.1 Kategorisierung von Anwendungen 51

2.7.2 Anwendungsklassen 52

2.7.3 Kriterien für eine Klassifikation 52

3 Grad der Industrialisierung **61**

3.1 Stufenmodell für den Grad der Industrialisierung 62

3.2 Reife der eingesetzten Methoden 66

3.2.1 Reifegrade ... 66
3.2.2 Kriterien für die Beurteilung des Reifegrades ... 68
3.2.3 Spezifikation der Reifegrade mittels Kriterien ... 68
3.3 Erläuterungen zum Stufenmodell ... 70
3.4 Stufe 1: Initial (Handwerk) ... 72
3.4.1 Artistic Driven Development ... 72
3.4.2 Erstellung von Artefakten auf Basis des Handwerks ... 73
3.4.3 Rahmenbedingungen für das Handwerk ... 74
3.4.4 Reifegrade des Handwerkes ... 74
3.4.5 Mögliche Einsatzgebiete ... 74
3.4.6 Grenzen des Artistic Driven Developments ... 75
3.5 Stufe 2: Standardisierung ... 75
3.5.1 Standard Driven Development ... 79
3.5.2 Erstellung von Artefakten auf Basis der Standardisierung ... 82
3.5.3 Rahmenbedingungen für die Standardisierung ... 84
3.5.4 Reifegrade der Standardisierung ... 87
3.5.5 Grenzen und Potentiale für Verbesserungen ... 88
3.6 Stufe 3: Wiederverwendung ... 88
3.6.1 Komponentenorientierung ... 89
3.6.2 Arten der Wiederverwendung ... 93
3.6.3 Systematische Wiederverwendung ... 93
3.6.4 Reuse Driven Development ... 95
3.6.5 Erstellung von Artefakten auf Basis der Wiederverwendung ... 97
3.6.6 Rahmenbedingungen für die Wiederverwendung ... 99
3.6.7 Reifegrade der Wiederverwendung ... 101
3.6.8 Grenzen und Potentiale für Verbesserungen ... 102
3.7 Stufe 4: Automatisierung ... 103
3.7.1 Modellierung als wesentliches Element der Automatisierung ... 105
3.7.2 Generierung von Software ... 107
3.7.3 Model Driven Development ... 109
3.7.4 Erstellung von Artefakten auf Basis der Automatisierung ... 112
3.7.5 Rahmenbedingungen für die Automatisierung ... 113
3.7.6 Reifegrade der Automatisierung ... 115
3.7.7 Grenzen und Potentiale für weitere Verbesserungen ... 116
3.8 Stufe 5: Komposition ... 117
3.8.1 Serviceorientierung ... 119
3.8.2 Service Driven Development ... 125
3.8.3 Erstellen von Artefakten auf Basis der Komposition ... 127
3.8.4 Rahmenbedingungen für die Komposition ... 129
3.8.5 Reifegrade der Komposition ... 132
3.8.6 Grenzen der Industrialisierung ... 133

4 Ermittlung des Industrialisierungsgrades ... 134

4.1 Domänenmodell ... 140

4.2 Domänenübergreifende Analyse ... 140
4.2.1 Bewertung von Gremien ... 141
4.2.2 Bewertung von Prozessen ... 142
4.2.3 Bewertung von Entwicklungsinfrastrukturen ... 143
4.2.4 Bewertung von Laufzeitinfrastrukturen ... 144
4.3 Domänenspezifische Analyse ... 144
4.3.1 Erfassung der Anwendungen des Unternehmens ... 145
4.3.2 Klassifikation von Anwendungen einer Subdomäne ... 147
4.3.3 Bestimmung des Sollindustrialisierungsgrades ... 148
4.3.4 Ermittlung der Entwicklungsinfrastrukturen pro Subdomäne ... 149
4.3.5 Bewertung der Anwendungen pro Entwicklungsinfrastruktur ... 149
4.4 Bewertung der Domäne ... 151
4.4.1 Bewertung der Gruppierung nach Entwicklungsinfrastrukturen ... 152
4.4.2 Bewertung der Subdomänen ... 153
4.4.3 Bewertung der Domänen ... 154
4.5 Vergleich des ermittelten Grades mit dem Sollgrad ... 154
4.6 Weitere Anmerkungen ... 154

5 Fallstudie ... 156

5.1 Domänenmodell ... 156
5.1.1 Produktion ... 157
5.1.2 Logistik ... 158
5.1.3 Marketing & Vertrieb ... 159
5.1.4 Forschung und Entwicklung ... 159
5.1.5 Steuerung ... 159
5.1.6 Shared Services ... 160
5.2 Domänenübergreifende Analyse ... 161
5.2.1 Bewertung der vorhandenen Gremien ... 161
5.2.2 Bewertung der vorhandenen Prozesse ... 163
5.2.3 Bewertung der vorhandenen Entwicklungsinfrastrukturen ... 165
5.2.4 Bewertung der vorhandenen Laufzeitinfrastruktur ... 173
5.3 Domänenspezifische Analyse ... 174
5.3.1 Erfassung der Anwendungen des Unternehmens ... 174
5.3.2 Klassifikation der Anwendungen der Subdomänen ... 176
5.3.3 Bestimmung der Sollindustrialisierungsgrades ... 177
5.3.4 Ermittlung der Entwicklungsinfrastrukturen in der Produktion ... 179
5.3.5 Bewertung der Anwendungen in der Subdomäne MES für .Net ... 180
5.4 Bewertung der Domäne Produktion ... 183
5.4.1 Bewertung der .Net Infrastruktur in der Subdomäne MES ... 183
5.4.2 Bewertung der Subdomäne MES ... 185
5.4.3 Vergleich des ermittelten Grades mit dem Sollgrad ... 186
5.4.4 Bewertung der Domäne Produktion ... 187

6 Zusammenfassung und Ausblick ... 188

Glossarverzeichnis..... 191

Literaturverzeichnis 197

Abbildungsverzeichnis

Abbildung 1: Kräfteverhältnis konkurrierender Faktoren in der Entwicklung 17
Abbildung 2: Aufbau der Arbeit 21
Abbildung 3: IT-Leistungserstellung als Fertigungsprozess 27
Abbildung 4: Horizontale, vertikale Integration der IT-Leistungserstellung 28
Abbildung 5: Verschiedene Aspekte der Anwendungsentwicklung 33
Abbildung 6: Artefakte der Anwendung sowie des Entwicklungsumfeldes 35
Abbildung 7: Entwicklungsinfrastruktur 40
Abbildung 8: Mögliche Aufteilung eines Unternehmens in Domänen 49
Abbildung 9: Stufenmodell der Industrialisierung. 64
Abbildung 10: Abnahme der manuellen Erstellung von Quelltext 65
Abbildung 11: Zunahme an Flexibilität 66
Abbildung 12: Reifegrade pro Grad der Industrialisierung 67
Abbildung 13: Schematische Darstellung einer Methode 71
Abbildung 14: Artistic Driven Development 73
Abbildung 15: Standard Driven Development 83
Abbildung 16: Systematische Wiederverwendung 94
Abbildung 17: Reuse Driven Development 97
Abbildung 18: Mehrstufige Transformation bei der Generierung 107
Abbildung 19: Schematische Darstellung des Generierens 108
Abbildung 20: Model Driven Development 112
Abbildung 21: Service Driven Development 128
Abbildung 22: Schritte zur Ermittlung des Industrialisierungsgrades 135
Abbildung 23: Ermittlung des Industrialisierungsgrades 137
Abbildung 24: Detailschritte zur Ermittlung des Industrialisierungsgrades 139
Abbildung 25: Klassifikation von Anwendungen 147
Abbildung 26: Einflussfaktoren für die Bestimmung des Grades pro Domäne 152
Abbildung 27: Domänenmodell des betrachteten Unternehmens 157
Abbildung 28: Bewertung der Gremien im Unternehmen 163
Abbildung 29: Bewertung der Prozesse im Unternehmen 165
Abbildung 30: Bewertung der ABAP-Entwicklungsinfrastruktur 167
Abbildung 31: Bewertung der Java-Entwicklungsinfrastruktur 169
Abbildung 32: Bewertung der SharePoint Entwicklungsinfrastruktur 170
Abbildung 33: Bewertung der .Net Entwicklungsinfrastruktur 172
Abbildung 34: Bewertung der Laufzeitinfrastrukturen 174
Abbildung 35: Bewertung der MES Anwendungen der .Net-Infrastruktur 184

Tabellenverzeichnis

Tabelle 1: Übertragung von Konzepten der Fertigungsindustrie 25
Tabelle 2: Kriterien für die Klassifikation von Anwendungen 54
Tabelle 3: Werte für das Kriterium Verfügbarkeit 55
Tabelle 4: Werte für das Kriterium Katastrophenfall 56
Tabelle 5: Werte für das Kriterium Änderbarkeit 57
Tabelle 6: Werte für das Kriterium Support 58
Tabelle 7: Werte für das Kriterium Integration 58
Tabelle 8: Werte für das Kriterium Zugriff auf die Anwendung 59
Tabelle 9: Werte für das Kriterium Kritikalität von Daten 60
Tabelle 10: Erläuterung der Reifegrade anhand von Kriterien 70
Tabelle 11: Tabelle zur Bewertung einer Anwendung anhand der Elemente 150
Tabelle 12: Reifegradermittlung pro Entwicklungsinfrastruktur 153
Tabelle 13: Zuordnung Anwendungen zu Subdomänen 175
Tabelle 14: Kategorisierung von Anwendungen der Domäne Produktion 175
Tabelle 15: Verteilung der erfassten Anwendungen auf die Klassenkriterien 176
Tabelle 16: Beispielhafte Klassifikation einer Anwendung 176
Tabelle 17: Klassifikation der Anwendungen der Subdomäne MES 177
Tabelle 18: Zuordnung Anwendungen zu einer Entwicklungsinfrastruktur 179
Tabelle 19: Beispielbewertung einer Anwendung 180
Tabelle 20: Ergebnis Analyse der Anwendungen 183
Tabelle 21: Bewertung der Subdomäne MES 185
Tabelle 22: Vergleich des ermittelten Grades mit dem Sollgrad 186

Definitionsverzeichnis

Definition 1: Domäne 48
Definition 2: Artistic Driven Development 72
Definition 3: Standard 76
Definition 4: Standardisierung 76
Definition 5: Standard Driven Development 80
Definition 6: Wiederverwendung 89
Definition 7: Softwarekomponente 90
Definition 8: Reuse Driven Development 95
Definition 9: Automatisierung 103
Definition 10: Modell im Kontext der Automatisierung 106
Definition 11: Model Driven Development 109
Definition 12: Service 120
Definition 13: Serviceorientierte Architektur 123
Definition 14: Service Driven Development 125

Abkürzungsverzeichnis

Abb.	Abbildung
ANSI	American National Standards Institute
ARIS	Architektur integrierter Informationssysteme
ASL	Application Services Library
Aufl.	Auflage
bspw.	beispielsweise
bzgl.	bezüglich
bzw.	beziehungsweise
CBS	Component Build Service
CE	Composition Environment
CMMI	Capability Maturity Model Integration
CMS	Change Management Service
CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
d.h.	das heißt
DTR	Design Time Repository
EAI	Enterprise Application Integration
EJB	Enterprise Java Beans
engl.	englisch
EPK	Ereignisgesteuerte Prozesskette
ERP	Enterprise Resource Planning
erw.	erweiterte
ESA	Enterprise Services Architecture
ESB	Enterprise Service Bus
etc.	et cetera
ggf.	gegebenenfalls
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
Hrsg.	Herausgeber
i.d.R.	in der Regel
IDE	Integrated Development Environment
ISO	International Organization for Standardization
IT	Information Technology
ITIL	IT Infrastructure Library
JEE	Java Platform, Enterprise Edition
KPI	Key Performance Indicator
MDA	Model Driven Architecture
MDSD	Model Driven Software Development
MES	Manufacturing Execution System

Mgmt.	Management
MSDI	Microsoft Development Infrastructure
NATO	North Atlantic Treaty Organization
neubearb.	neubearbeitete
NWDI	Netweaver Development Infrastructure
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
PIM	Platform Independent Model
PNN	Produktionsnahes Netz
PSM	Platform Specific Model
RAD	Rapid Application Development
RTE	Round Trip Engineering
RUP	Rational Unified Process
SaaS	Software as a Service
SEI	Software Engineering Institute
SLD	System Landscape Directory
SOA	Service Orientierte Architektur
SOAP	Simple Object Access Protocol
sog.	so genannt
SPALM	SharePoint Application Lifecycle Management
SPICE	Software Process Improvement and Capability Determination
SPDI	SharePoint Development Infrastructure
TFS	Team Foundation Server
u.a.	unter anderem
u.U.	unter Umständen
überarb.	überarbeitete
URL	Uniform Resource Locator
vgl.	vergleiche
WSDL	Web Services Description Language
XML	Extended Meta Language
XP	eXtreme Programming
z.B.	zum Beispiel

1 Einleitung

Das Erstellen von Software hat sich in den letzten 40 Jahren aufgrund der Entwicklung verschiedener Konzepte, Methoden, aber auch Vorgehensweisen verändert. Der Beginn der Industrialisierungsdebatte in der Softwarebranche reicht nach [GrLO05] bereits in das Jahr 1968 zurück. Damals wurde von dem Science Committee der NATO eine Software-Engineering-Konferenz in Garmisch-Partenkirchen durchgeführt. McIlroy[1] forderte in seinem Vortrag bereits damals, die Softwareentwicklung nach dem Vorbild der industriellen Produktion zu gestalten. Dieser Konferenz ging die Softwarekrise 1965 voraus, bei der die Kosten für die Software erstmalig die Kosten für die Hardware überstiegen. Der Grund hierfür waren die immer komplexer gewordenen Systeme. Damals galt die Herstellung von Software, im Vergleich zur Hardware, nicht als kostspieliger Rohstoff und wurde daher nicht näher beachtet. Die Sicherstellung von Qualität in der Software hatte ebenfalls noch keine Priorität.

Seit dieser Zeit wurden Konzepte, Methoden sowie Werkzeuge entwickelt, welche das Erstellen von Software vereinfachen und professionalisieren. Dennoch werden nach [ZaBP05] die Dienstleistungen der IT-Bereiche, wie kaum ein anderer Beitrag zum Unternehmenserfolg, so häufig kritisiert und schlecht bewertet. Signifikant viele Projekte, mit dem Ziel der Entwicklung von Anwendungen, werden nicht im geplanten Zeitrahmen fertig oder überschreiten das vorgegebene Budget. Dies geht aus den Analysen des Beratungsunternehmens Standish Group[2] hervor, das seine Zahlen in dem sogenannten Chaos-Report [Stan04] jedes Jahr veröffentlicht. 2008 wurden demnach lediglich 32% aller Projekte im vorgesehenen Rahmen fertiggestellt [Stan09].

In [DeMa97] wird die Frage behandelt, inwiefern die Erwartungen an die Softwareentwicklung überzogen sind. Niemand hatte erwartet, dass die Softwarebranche den heutigen Stand der Technik je erreichen werde. Die existierende, ungewöhnlich hohe Produktivität und Qualität wurde nicht vorhergesehen. Dennoch werden die Leistungen der IT heute immer noch schlecht bewertet.

Nach [ZaBP05] wird die Effektivität von IT-Lösungen in Unternehmen nicht nachgewiesen und ihre Effizienz ist nicht ausreichend bekannt. Die Unternehmensführung ist somit oft nicht in der Lage, die Leistungen der verschiedenen IT-Bereiche zu beurteilen.

---

[1] Siehe [McIl68]: Mass Produced Software Components.

[2] Standish Group: US-amerikanisches Beratungsunternehmen spezialisiert auf Risikomanagement bei IT-Investitionen. Es erstellt jährlich eine Studie (Chaos-Report) bzgl. Erfolgs- und Misserfolgsfaktoren in IT-Projekten.

Die Unterstützung der Geschäftsprozesse durch IT-Anwendungen ist jedoch unbestritten ein entscheidender Faktor im Wettbewerb mit anderen Unternehmen. Die IT stellt Lösungen für die Reduzierung der Prozesskosten bereit und ermöglicht die Gestaltung von Innovationen.

In [GrSh06] wird bemängelt, dass trotz der existierenden Methoden die Produktivität in der Softwareentwicklung oder die Qualität von Softwareprodukten nur marginal verbessert wurde. Der größte Teil der Software wird heute noch manuell, unter Verwendung von arbeitsintensiven Mitteln, von Grund auf neu entwickelt. Die Probleme liegen im Erfolg der Branche, der durch die wachsende Nachfrage entstanden ist. In [GrSh06] wird die Meinung vertreten, dass die Softwarebranche nur weiter reifen könne, wenn sie die Muster der Industrialisierung übernehme. Eine solche Software-Industrialisierung wird bei [Taub05] als „Entstehung und Ausbreitung der Softwareentwicklung mit industriellen Mitteln“ definiert[3].

Anzeichen für die noch fehlende Industrialisierung bei der Softwareentwicklung werden auch in [Janß05] genannt. Als Beispiel für diese Anzeichen wird der Vergleich eines Architekten der bereits industrialisierten Baubranche mit einem Softwareentwickler beschrieben. Während der Architekt sich nach Fertigstellung eines Gebäudes nicht mehr mit dem Gebäudemanagement beschäftigt, wünscht sich der Softwareentwickler die Begleitung einer Anwendung über deren Lebenszyklus hinweg. Ein weiterer Aspekt liegt in der Ablehnung von Komponenten, die nicht selbst entwickelt wurden. Die Wiederverwendung von vorgegebenen Bausteinen stößt auf emotionalen Widerstand. Hinzu kommt eine teilweise leidenschaftliche Verehrung von Werkzeugen und Technologien. Solche Verhaltensmuster deuten eher auf eine Manufaktur oder ein Kunsthandwerk hin.

Die Softwaretechnik (engl. Software Engineering) hat ihre Wurzeln in der erwähnten NATO-Konferenz und wird seitdem kontinuierlich weiterentwickelt. Es ist eine vergleichsweise junge Disziplin, deren Reife sowohl bzgl. der Konzepte, Methoden und Werkzeuge als auch hinsichtlich der Menschen, die an der Entwicklung beteiligt sind, weiter wachsen wird.

[Balz01] definiert die Software-Technik als „zielorientierte Bereitstellung und systematische Verwendung von Prinzipien, Methoden und Werkzeugen für die arbeitsteilige, ingenieurmäßige Entwicklung und Anwendung von umfangreichen Softwaresystemen“. Der konsequente Einsatz von industriellen Prinzipien und Verfahren ist daher ein weiterer Schritt zur Steigerung der Produktivität der Anwendungsentwicklung sowie zur Sicherung der Qualität von IT-Lösungen.

---

[3] Die Software-Industrialisierung wird in Abschnitt 2.1 näher erläutert. Zu den Prinzipien der Industrialisierung gehören u.a. Standardisierung, Automatisierung, Reduzierung der Fertigungstiefe, Wiederverwendung, Arbeitsteilung.

1.1 Motivation

Aufgrund der auch heute noch zu einem großen Teil manuellen Realisierung von IT-Anwendungen ist nach [GrSh06] die Entwicklung von Software langsam und teuer. Die so implementierten Anwendungen sind fehleranfällig und eingeschränkt in der Brauchbarkeit, der Zuverlässigkeit, der Performanz sowie der Sicherheit.

Aber nicht nur die Entwicklung von Anwendungen ist relevant für die Betrachtung. Der Anteil an wiederkehrenden Kosten in Form von Weiterentwicklungen oder Fehlerbehebungen wird in [ZaBP05] auf ca. 80% der Gesamtkosten[4] einer Anwendung beziffert. Somit ist nicht nur darauf zu achten, eine möglichst effiziente Erstentwicklung einer Anwendung durchzuführen. Vielmehr sind die Kosten über den gesamten Lebenszyklus einer Anwendung zu betrachten. In diesem Zusammenhang wird in [SiSM06] festgestellt, dass knapp die Hälfte des Aufwandes bei der Weiterentwicklung bzw. der Fehlerbehebung durch das Verstehen des Quelltextes einer Anwendung verursacht wird.

Eine weitere Problematik entsteht durch die mittlerweile hohe Vielfalt an Technologien, mit denen Entwickler konfrontiert werden. Meist ist eine Entscheidung für die geeignete Technologie zur Lösung der Problemstellung nicht einfach zu treffen. Teilweise verfügen die Entwickler nicht über die benötigten Kenntnisse oder es besteht eine emotionale Bindung an eine bestimmte Technologie [Janß05].

Zudem werden Entwickler mit erhöhten Sicherheitsanforderungen an die Anwendungen konfrontiert, insbesondere im Zusammenhang mit der Globalisierung sowie zunehmenden gesetzlichen Regulierungen.

Software wird branchenspezifisch nach unterschiedlichen Kriterien und Schwerpunkten entwickelt. Die Softwarebranche entwickelt beispielsweise Anwendungen als Produkte, die an mehrere Kunden verkauft werden. Die Anforderungen an die Erweiterbarkeit sowie die Wartbarkeit sind in diesem Fall sehr hoch.

Das Ziel der Fertigungsindustrie ist die Herstellung von Gütern. Wesentliches Merkmal für den Einsatz von Software ist die effiziente Unterstützung der Geschäftsprozesse. Stellt eine IT-Lösung in einem bestimmten Bereich des Unternehmens ein Differenzierungsmerkmal gegenüber dem Wettbewerb dar, wird die hierfür benötigte Software individuell für dieses Unternehmen erstellt. Handelt es sich jedoch um einen bereits standardisierten Bereich, wie z.B. die Dokumentenverwaltung, liegt der Schwerpunkt auf der Bereitstellung einer kostengünstigen IT-Lösung. Kostengünstig bezieht sich hierbei sowohl auf die Beschaffung als auch auf die Wartung der entsprechenden Software. Solche IT-Anwendungen

[4] Kosten der initialen Entwicklung und Bereitstellung, der Wartung sowie der Abschaltung.

können meist von verschiedenen Softwareherstellern als Produkt eingekauft werden.

Allgemein ist die Entwicklung von Individuallösungen nicht die Kernaufgabe der Fertigungsindustrie. Jedoch besteht der Bedarf, Anwendungen

- mit der geforderten Qualität
- im Rahmen der IT-Strategie
- durch geeignetes Eigen- oder Fremdpersonal
- mit möglichst geringen Kosten und Wartungsaufwand
- und der geforderten Funktionalität

zu erstellen.

Diese Forderungen sind bei der Entwicklung von Anwendungen zu berücksichtigen. Möglichst geringe Kosten bei der Erstellung von Anwendungen sind jedoch meist nicht vereinbar mit der Forderung nach einer möglichst hohen Qualität. Nach [Star02] ist die Realisierung von Anwendungen oft durch Kompromisse gekennzeichnet, die durch die Gewichtung verschiedener Faktoren bestimmt werden. Das Kräftespiel dieser Faktoren wird bei [Star02] anhand eines „Gummiband-Diagramms“ (vgl. Abbildung 1) dargestellt[5].

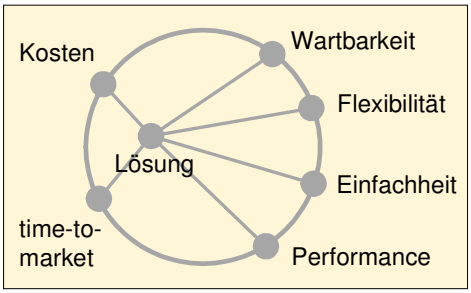

Abbildung 1: Kräfteverhältnis konkurrierender Faktoren in der Entwicklung[6].

In diesem Beispiel werden den Faktoren Kosten sowie time-to-market[7] ein höheres Gewicht als den restlichen Faktoren zugewiesen. Die Wartbarkeit, Flexibilität, Einfachheit und Performanz werden somit bei der Entwicklung der Anwendung[8] weniger berücksichtigt. Die Vernachlässigung von Faktoren kann im Lebenszyk-

[5] Die abgebildeten Faktoren sind [Star02] entnommen. Die in dieser Arbeit verwendeten Faktoren, werden in Abschnitt 2.4 eingeführt.

[6] Gummiband-Diagramm aus [Star02].

[7] Siehe [Star02]: Zeitspanne für die Realisierung und Einführung einer Anwendung.

[8] In Abbildung 1 als Lösung bezeichnet.

lus[9] der Anwendungen zu Problemen führen. Ist eine Anwendung z.B. schlecht wartbar, ist eine Modifikation nur mit erhöhtem Aufwand durchführbar.

Die Prinzipien der Industrialisierung[10] liefern Ansätze, das Kräfteverhältnis auf die Anwendung zu optimieren. Hierfür ist zu klären, welche dieser Ansätze bei der Erstellung von Software in welcher Reihenfolge anzuwenden sind, um die Entwicklung von Anwendungen nachhaltig zu verbessern.

Nach [GrSh06] besteht die Herausforderung der Industrialisierung in der Softwareentwicklung nicht in der hochgradigen Mechanisierung, die von unerfahrenen Arbeitern durchgeführt werden kann. Vielmehr gilt es, die geeigneten Entwickler von mechanischen Aufgaben zu befreien, um mehr Zeit und Möglichkeiten für die Gestaltung der Prozesse zu gewinnen. Die Industrialisierung wird in diesem Kontext daher nicht für die Massenproduktion von Gütern (z.B. die Herstellung von Kopien eines Software-Produktes) betrachtet. Im Fokus steht die Übertragung der Prinzipien der Industrialisierung auf die effiziente Gestaltung der Entwicklung von IT-Anwendungen.

1.2 Zielsetzung dieser Arbeit

Seit dem Start der Diskussion um das Software Engineering wurden viele verschiedene Ansätze zur Verbesserung der Softwareentwicklung erstellt. Es fehlt nicht an Konzepten, welche die Anwendungsentwicklung verbessern. Oft ist es die mangelnde oder nicht konsequente Umsetzung in den Unternehmen, die den Zugewinn an Effizienz verhindern.

Ausgehend von der Analyse, welche Managementkonzepte der Fertigungsindustrie sich auf die Softwareentwicklung übertragen lassen, werden bereits existierende Methoden der Erstellung von Anwendungen betrachtet. Hierbei wird untersucht, inwiefern diese Methoden den Prinzipien der Industrialisierung entsprechen.

In dieser Arbeit wird somit keine neue Methode entwickelt, die eine noch effizientere Realisierung von Anwendungen erlauben würde. Vielmehr werden die bestehenden Konzepte in Beziehung zueinander gesetzt und analysiert, inwiefern sie zur Industrialisierung beitragen.

Der Kern dieser Arbeit ist der Aufbau eines fünfstufigen Modells des Industrialisierungsgrades der Anwendungsentwicklung. Bereits existierende Verfahren bzw. Methoden, welche die verschiedenen Prinzipien der Industrialisierung umsetzen, bilden die Grundlage der fünf aufeinander aufbauenden Stufen. Die Ein-

---

[9] Zeitraum von der Erstellung einer Anwendung über die mehrfache Modifikation der bereitgestellten Funktionalitäten bis zur Abschaltung.

[10] Siehe Abschnitt 2.1: Standardisierung, Automatisierung, Reduzierung der Fertigungstiefe, Wiederverwendung, Arbeitsteilung.

ordnung einer Methode in eine Stufe erfolgt dabei auf Basis des Beitrags zur Steigerung der Industrialisierung der Anwendungsentwicklung.

Dieses Modell dient Unternehmen als Rahmen für eine kontinuierliche Optimierung der Anwendungsentwicklung. Für die Bewertung der im Unternehmen etablierten Entwicklungsmethoden wird in dieser Arbeit ferner ein Verfahren zur Bestimmung des aktuellen Industrialisierungsgrades beschrieben. Auf Basis einer initialen Bewertung kann das Unternehmen den aktuellen Zustand der Anwendungsentwicklung analysieren und Maßnahmen zur Verbesserung beschließen. Eine regelmäßige Durchführung dieser Bewertung gibt dabei Aufschluss über die Wirksamkeit bzw. Zielerreichung der durchgeführten Maßnahmen.

Das Verfahren zur Ermittlung des Industrialisierungsgrades wird durch eine exemplarische Untersuchung in einem Unternehmen der Fertigungsindustrie verifiziert.

1.2.1 Abgrenzung der Aufgabenstellung

Der Software-Entwicklungsprozess ist nach [Dumk03] der gesamte Prozess der Aufgabenstellung, Planung, Realisierung und Bewertung einer Software- oder Hardware-Anwendung.

Die Qualität der Software wird dabei nach [HDHM06] von der Qualität des Entwicklungsprozesses gesteuert. Eine Beurteilung der Prozessqualität wurde, laut [Kneu06], erstmals durch das amerikanische Verteidigungsministerium zur Beurteilung seiner Lieferanten gefordert. Hierzu entwickelte das Software Engineering Institute (SEI)[11] der Carnegie Mellon University 1986 ein entsprechendes Verfahren. Dieses Verfahren wurde später als Capability Maturity Model (CMMI) bekannt und wird auch als Reifegradmodell bezeichnet. Weitere Modelle, wie z.B. SPICE[12], von anderen Organisationen folgten. Mittlerweile können Unternehmen sich nach diesen Modellen zertifizieren lassen.

Ferner existiert mit der Application Services Library (ASL) (siehe [Pols07]) ein Standard für das Management von IT-Anwendungen. Dieser Standard enthält alle Prozesse, die für die Erbringung von umfassenden und hochwertigen Dienstleistungen im Lebenszyklus einer IT-Anwendung nötig sind.

Die Beschreibung bzw. Vorgaben für die Gestaltung der Entwicklungsprozesse enthalten jedoch nach [UHSB07] keine Angaben über die Methoden bzw. Me-

[11] Forschungs- und Entwicklungszentrum an der Carnegie Mellon University in Pittsburgh, Pennsylvania. Es wird zum Großteil vom US-Verteidigungsministerium finanziert (http://www.sei.cmu.edu/).

[12] SPICE – Software Process Improvement and Capability Determination: ein internationaler Standard zur Durchführung von Bewertungen (Assessments) von Unternehmensprozessen mit Schwerpunkt auf der Softwareentwicklung. Siehe auch [HDHM06].

chanismen der industriellen Erstellung von IT-Anwendungen. Die Entwickler können von den bekannten Reifegradmodellen keine Vorgaben über die anzuwendenden Methoden entnehmen. Der Grund dafür ist, dass bei den Reifegradmodellen nicht der Grundgedanke der Industrialisierung im Vordergrund steht. Somit können auch keine Aussagen über den Grad der Industrialisierung abgeleitet werden.

Auch in dem Excellence-Modell von [UHSB07] liegt der Fokus nicht auf der Erstellung von IT-Anwendungen. Vielmehr wird ein Modell zur Bewertung und Einschätzung des Industrialisierungsgrades eines IT-Dienstleisters entwickelt. Dies beinhaltet einen wertkettenorientierten Ansatz, der alle funktionalen Bereiche der IT-Leistungserstellung und deren Unterstützungsprozesse umfasst. Auf die verschiedenen Methoden bei der Anwendungsentwicklung wird dabei nicht eingegangen.

Im Gegensatz zur Prozessbetrachtung oder umfassenden Wertkettenbetrachtung eines IT-Dienstleisters liegt der Schwerpunkt in dieser Arbeit auf den Methoden zur Erstellung von IT-Anwendungen. Entwicklungsprozesse wie Projektmanagement oder Anforderungsmanagement, aber auch Vorgehensmodelle, wie das Wasserfallmodell, werden daher nicht näher beleuchtet.

Die Entwicklung von IT-Anwendungen wird mittlerweile in den meisten Unternehmen, unabhängig von der Branche, durchgeführt. Die einzelnen Unternehmen haben jedoch unterschiedliche Interessen bei der Realisierung von Softwarelösungen. Die Einschränkung auf die Fertigungsindustrie hat den Hintergrund, dass die zu realisierenden IT-Anwendungen lediglich die Geschäftsprozesse unterstützen und nicht als Produkt an andere Firmen verkauft werden. In der Fertigungsindustrie stehen die Produkte (Güter) im Vordergrund, deren Herstellung sowie Verkauf durch IT-Anwendungen zu unterstützen sind. Da Software somit als unterstützendes Werkzeug betrachtet wird, sind die Forderungen nach Rationalisierung bei der Entwicklung hoch.

Der Schwerpunkt in dieser Arbeit liegt somit auf IT-Anwendungen, die individuell für ein Unternehmen erstellt werden. Welche Art von Fachlichkeit realisiert wird bzw. welche Geschäftsprozesse unterstützt werden, spielt dabei eine untergeordnete Rolle. Die in dieser Arbeit vorgestellten Methoden des Stufenmodells sind grundsätzlich für die Entwicklung von Anwendungen unterschiedlicher Fachlichkeit nutzbar.

Sofern Softwareprodukte[13] bereits eine Fachlichkeit abbilden, jedoch die Möglichkeit einer Anwendungsentwicklung erlauben, können hierfür die betrachteten Methoden ebenfalls eingesetzt werden. Beispielsweise bietet das SAP R/3-System neben einer Fachlichkeit in Form von Modulen, wie Buchhaltung, Ver-

[13] Software, die von Herstellern an mehrere Kunden vertrieben werden.

kauf oder Materialwirtschaft, auch die Möglichkeit, diese Module zu modifizieren. Ferner können neue unternehmensspezifische Anwendungen erstellt werden.

In dieser Arbeit wird davon ausgegangen, dass die interne IT für die Entwicklung von Anwendungen zuständig ist. Dabei können sowohl interne wie externe Entwickler an der Realisierung der Anwendungen beteiligt sein. Bei der Einbindung externer Entwickler ist jedoch darauf zu achten, dass die festgelegten Vorgaben, wie Prozesse und Methoden, für die Erstellung von Software im Unternehmen eingehalten werden.

1.3 Aufbau der Arbeit

Diese Arbeit ist in fünf weitere Kapitel aufgeteilt (siehe Abbildung 2). In Kapitel 2 werden die verschiedenen Grundlagen für den Aufbau des Stufenmodells beschrieben. Neben der Erläuterung der Prinzipien der Industrialisierung wird die IT-Leistungserstellung als Fertigungsprozess vorgestellt. Dieser betrachtet die Realisierung von IT-Dienstleistungen unter industriellen Gesichtspunkten. Ferner werden die verschiedenen Elemente bei der Erstellung von Anwendungen erläutert, die für das Stufenmodell relevant sind. Die Grundlagen für eine Analyse der Anwendungsentwicklung im Unternehmen bilden den Abschluss des Kapitels. Hierzu gehören die Faktoren der Anwendungsentwicklung, die Strukturierung der Fachlichkeit sowie die Klassifikation von Anwendungen.

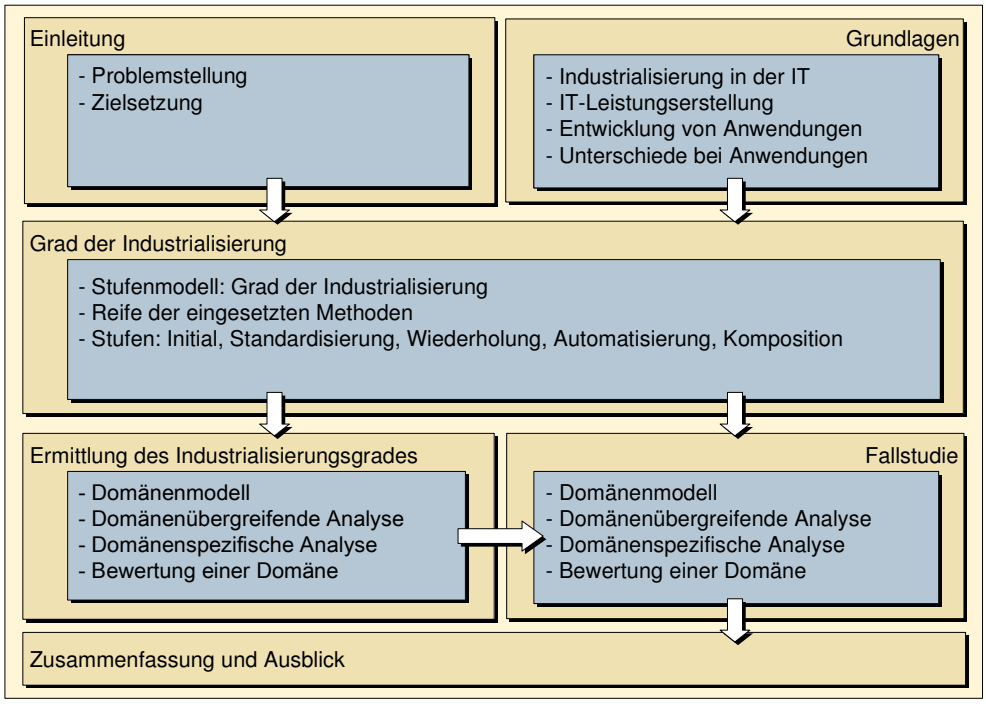

Abbildung 2: Aufbau der Arbeit

Die Einführung des Stufenmodells für den Industrialisierungsgrad erfolgt in Kapitel 3. Neben einer allgemeinen Erläuterung des Stufenmodells werden die Kriterien vorgestellt, die eine Einstufung der Anwendungsentwicklung eines Unternehmens in einen bestimmten Grad vorgeben. Der Kern des dritten Kapitels ist die Vorstellung der einzelnen Stufen. Jeder Stufe wird eine Entwicklungsmethode zugeordnet, die einem Prinzip der Industrialisierung entspricht. Die jeweiligen Abschnitte der einzelnen Stufen enthalten die Beschreibung und die Zielsetzung der zugeordneten Methode sowie die Vorgehensweise bei der Erstellung einer Anwendung. Ferner werden die Voraussetzungen für den Einsatz sowie die Kriterien für die Bestimmung des Reifegrades der jeweiligen Methode vorgestellt.

Für die Ermittlung des Industrialisierungsgrades eines Unternehmens wird in Kapitel 4 ein Bewertungsverfahren eingeführt. Dieses Verfahren basiert auf der Strukturierung der Fachlichkeit in sogenannte Domänen. Eine Domäne stellt einen spezifischen Bereich des Unternehmens dar, der gezielt mit Anwendungen eines bestimmten fachlichen Hintergrunds bebaut wird [EHHJ08]. Die Komponenten der Bewertung setzen sich zusammen aus einer domänenübergreifenden Analyse, z.B. der Entwicklungsinfrastruktur, sowie der domänenspezifischen Analyse, in welcher die verschiedenen Elemente von Anwendungen untersucht werden. Die Ergebnisse der Analysen werden im Anschluss zu einer Bewertung der betrachteten Domäne aggregiert.

Dieses Bewertungsverfahren wird in Kapitel 5 anhand eines Unternehmens der Fertigungsindustrie exemplarisch durchgeführt. Das Ziel ist dabei die Verifizierung des vorgestellten Stufenmodells sowie die Ermittlung des Industrialisierungsgrades auf die Durchführbarkeit in der Praxis.

Das Kapitel 6 schließt diese Arbeit mit einer Zusammenfassung der verschiedenen Ergebnisse ab und stellt mögliche weitere Schritte in einem Ausblick vor.

2 Grundlagen

Die Prinzipien der Industrialisierung sind die Basis für eine Optimierung der Anwendungsentwicklung. Der konsequente Einsatz dieser Prinzipien hat das Potential, die Produktivität in der Erstellung von IT-Lösungen zu erhöhen. Das in dieser Arbeit vorgestellte Stufenmodell verdeutlicht diese Potentiale. Ferner wird ein Vorgehen zur Ermittlung des Industrialisierungsgrades vorgestellt, das eine Bewertung der Anwendungsentwicklung im Unternehmen ermöglicht. Die hierfür benötigten Grundlagen werden in diesem Kapitel erläutert.

2.1 Industrialisierung in der IT

Die Industrialisierung wird als Entstehung und Entwicklung der industriellen Produktion bezeichnet[14]. Den Ursprung führt [Kilg86] auf die Nebentätigkeit der Landwirtschaft in der vorindustriellen Zeit zurück. Aufgrund der saisonalen Beschäftigung in der Landwirtschaft mussten die Bauern die Monate mit geringer Arbeit durch Nebentätigkeiten überbrücken, um den Lebensunterhalt zu sichern. Hierbei wurden Textilien über den eigenen Bedarf hinaus erzeugt und verkauft. Mit zunehmenden Produktmengen wurde der Vertrieb der Waren auf einen neuen Berufsstand, die Aufkäufer, verlagert. Diese wurden später als sogenannte Verleger zu selbständigen Unternehmern[15].

Diese Entwicklung kann bereits als Übergang zur industriellen Produktion betrachtet werden, da die Spezialisierung auf die Produktion bzw. den Vertrieb der Waren dem Prinzip der Arbeitsteilung folgt. Zudem wurden im Unterschied zum damaligen Handwerk die Produkte nicht auf Basis von Kundenaufträgen produziert, sondern größere Stückzahlen für den Markt hergestellt. Die Grundlagen für die Massen- und Serienproduktion waren gelegt.

Einen zusätzlichen Impuls bekam die Industrialisierung durch die Automatisierung der Produktion, also die Ersetzung von menschlicher Arbeitskraft durch Maschinen. Ein Beispiel für eine solche Automatisierung ist die Webmaschine für die Produktion von Geweben. Hierdurch konnten die Textilien schneller hergestellt werden.

Die weitere Entwicklung der Industrialisierung führte über Manufakturen hin zum Fabrikbetrieb. Die Manufaktur ist eine Betriebsform, die noch geprägt ist von Handarbeit. Die Produktion erfolgt jedoch nicht mehr am Wohnort. Ferner zeichnet sich die Manufaktur durch eine weitere Arbeitsteilung aus [Kilg86].

---

[14] Siehe [Kilg86].

[15] Siehe [Rübb72].

Für den Fabrikbetrieb hat sich heute der Begriff des Industriebetriebes durchgesetzt. Dieser Betrieb zeichnet sich nach [Kilg86] durch die folgenden Eigenschaften aus:

- Funktionale Trennung zwischen Leitung und Ausführung
- Weitgehende Arbeitszerlegung im Bereich der Produktion
- Zunehmende Substitution der menschlichen Arbeit durch Betriebsmittelleistungen
- Herstellung von Massen- und Serienerzeugnissen
- Erhöhter Kapitalbedarf und eine daraus folgende Trennung der Kapitalgeber- und Leitungsfunktion
- Wachsende Betriebsgrößen
- Verstärkte Krisenanfälligkeit infolge steigender Fixkosten
- Auftreten sozialer Spannungen infolge zunehmender Abhängigkeit der Arbeitnehmer

Nach [WaBK07] basiert die Industrialisierung „auf technischem und organisatorischem Fortschritt, ist durch Automatisierung der Produktion charakterisiert und führt dadurch zu signifikant sinkenden Kosten für die zentralisierte, spezialisierte und global verteilte Produktion standardisierter Produkte“.

Viele dieser Eigenschaften lassen sich auch in der Softwarebranche wiederfinden. [Taub05] definiert die Software-Industrialisierung mit der „Entstehung und Ausbreitung der Softwareentwicklung mit industriellen Mitteln“. Hierbei werden die in der industriellen Produktion bereits bekannten Mittel auf die Softwareentwicklung übertragen. Auch [WaBK07] bescheinigen eine „erste empirische Evidenz für eine Industrialisierung der IT“.

[Taub05] stellt die typischen industriellen Mittel der Fertigungsindustrie den vorhandenen Mitteln der Softwarebranche gegenüber (siehe Tabelle 1). Dabei übertragen sich einzelne Konzepte direkt, manche heißen anders und manche funktionieren etwas anders.

Dominierender Faktor in der Diskussion zur Industrialisierung ist derzeit die Reduzierung der Fertigungstiefe und damit der Zukauf von Leistungen durch spezialisierte Anbieter. Verschiedene Konzepte des Outsourcings[16] werden zurzeit für die Reduzierung der Kosten in der IT genutzt. Insbesondere das Offshoring, das Outsourcing von Leistungen in das meist asiatische Ausland, bietet hierfür Potentiale. Dies wird begünstigt durch die relativ niedrigen Lohnkosten und die ver-

[16] Ausgliederung von Teilfunktionen der IT und Inanspruchnahme von Drittunternehmen [LeWS08].

gleichsweise gute Ausbildung der Softwareentwickler in diesen Ländern. Ein solches Modell ist allerdings nur erfolgreich, wenn die Zulieferungen der IT-Leistungen effizient in die IT-Landschaft des Unternehmens eingebettet werden können.

Fertigungsindustrie	Softwarebranche
Massenproduktion	→ Software-Produkte
Arbeitsteilung, Fließband	→ Zerlegung der Arbeit, Vorgehen
Spezialisierung	→ Spezialisierung
Rationalisierung	→ Rationalisierung
Automatisierung	→ Werkzeuge, MDA
Kontinuierliche Verbesserung	→ ISO900x, CMMI Level 5
Standardisierung	→ Standardisierung (IP, J2EE, . . .)
Plattformstrategie (z. B. Autoindustrie)	→ z. B. Common IT-Plattform
Modulkomponenten (z. B. Autoindustrie)	→ komponentenbasierte Architektur
Verringerung der Fertigungstiefe	→ Outsourcing
Nutzung globaler Märkte	→ Nutzung globaler Märkte
Globale Nutzung von Lohngefälle	→ Offshoring

Tabelle 1: Übertragung von Konzepten der Fertigungsindustrie[17]

Allgemein ist in der Literatur keine eindeutige Definition zu finden, mit welchen industriellen Mitteln die Softwareentwicklung industrialisiert werden kann. Häufig wiederkehrende Prinzipien in der Literatur[18] sind:

- *Standardisierung, Plattformstrategie:* Reduktion der Individualität von IT-Lösungen, um die Komplexität der IT-Landschaft zu senken. Die Standardisierung von Produkten ermöglicht die Standardisierung von Prozessen, deren Automatisierung sowie Verteilung auf verschiedene Wertschöpfungspartner. Die Plattformstrategie ist eine Standardisierung auf Unternehmensebene.
- *Wiederverwendung, Modularisierung:* Erhöhung der Qualität der Anwendung sowie die Reduzierung der Entwicklungskosten durch die Wiederverwendung existierender Komponenten.
- *Automatisierung:* Ersetzen menschlicher Arbeitskraft durch Maschinen bzw. Generatoren. Die Entwickler von Anwendungen werden von routinemäßigen Arbeiten entlastet.
- *Arbeitsteilung, Spezialisierung:* Aufteilung der verschiedenen Arbeitsschritte bei der Leistungserstellung auf mehrere spezialisierte Personen. Im Gegensatz zum Kunsthandwerk, bei dem eine Person alle Arbeitsschritte beherrschen muss, können sich die Arbeitskräfte bei der Arbeitsteilung auf die spezifischen Aufgaben spezialisieren.

[17] Siehe [Taub05].
[18] Z.B. in [Taub05], [HBMK07], [WaBK07], [PfWi07].

- *Verringerung der Fertigungstiefe:* Unternehmen verzichten auf eigene Fertigung bestimmter Leistungen, die von spezialisierten Lieferanten zu niedrigeren Preisen zugekauft werden können. Hierdurch wird die Auslagerung von gering wettbewerbsfähigen Teilen der Wertschöpfung an spezialisierte Anbieter ermöglicht.

Aufgrund der Mehrfachnennung dieser Prinzipien in der Literatur stellen sie somit eine gewisse Grundvoraussetzung für die Industrialisierung dar. Es gibt verschiedene Abbildungen dieser Prinzipien auf Methoden der Softwareentwicklung (siehe [Taub05]). Welcher Grad an Industrialisierung jedoch durch welche Methoden der Softwareentwicklung erreicht wird, ist in der Literatur nicht zu finden.

Weitere Eigenschaften eines Industriebetriebes wie ein erhöhter Kapitalbedarf, wachsende Betriebsgrößen, eine verstärkte Krisenanfälligkeit sowie das Auftreten sozialer Spannungen aufgrund zunehmender Abhängigkeiten (vgl. [Kilg86]) sind auch in der Softwarebranche zu beobachten. Diese Eigenschaften betreffen jedoch eher das Umfeld der Entwicklung und weniger die Methoden der Entwicklung. Sie werden daher in dieser Arbeit zunächst nicht betrachtet. In einer weiterführenden Abhandlung könnte jedoch der Einfluss dieser Eigenschaften auf die Etablierung der Methoden der einzelnen Industrialisierungsstufen untersucht werden.

2.2 IT-Leistungserstellung als Fertigungsprozess

Die IT hat in den letzten Jahren einen hohen Beitrag zur weiteren Industrialisierung der Unternehmen geleistet. Insbesondere in der Automatisierung, z.B. am Fließband durch die Unterstützung von Robotertechnik oder der Erfassung von Kundenaufträgen über das Internet, werden die Prozesse weiter optimiert. Die IT selbst ist jedoch noch am Anfang der Industrialisierung.

Wichtige Voraussetzung für die Industrialisierung der IT ist ein geändertes Verständnis der Leistungserbringung. Nach [ZaBP05] sollte sich die IT-Abteilung als IT-Dienstleister verstehen, der auf Basis von IT-Produkten den verschiedenen Geschäftsbereichen (den internen Kunden) Dienstleistungen anbietet.

Hierbei genügt es nicht mehr, lediglich die angeforderten Projekte abzuwickeln. Vielmehr muss die Schnittstelle zwischen der IT und den Bereichen neu definiert werden, indem das Portfolio an IT-Dienstleistungen von beiden Seiten aktiv gestaltet wird. Das Management zur Gestaltung des Portfolios an IT-Dienstleistungen stellt damit das Bindeglied zwischen der IT und den Geschäftsbereichen dar.

In dieser Arbeit wird davon ausgegangen, dass die betrachteten Unternehmen jeweils über eine eigene interne IT-Abteilung verfügen, die IT-spezifische Anfor-

derungen der Geschäftsbereiche bearbeiten[19]. Die Umsetzung dieser Anforderungen kann auch mit externer Unterstützung erfolgen. Voraussetzung ist jedoch insbesondere für die Anwendungsentwicklung, dass die externen Entwickler an die internen Vorgaben zur Realisierung von Anwendungen gebunden sind.

Wesentlich bei der Industrialisierung der IT ist die Anwendung der verschiedenen Prinzipien im Prozess der Erstellung von IT-Leistungen. In [ZaBP05] wird die IT-Leistungserstellung als Fertigungsprozess betrachtet, der einige Analogien mit der industriellen Fertigung hat. Dieser Fertigungsprozess setzt sich aus den drei Hauptaktivitäten Portfoliomanagement, IT-Entwicklung sowie IT-Produktion zusammen (siehe Abbildung 3).

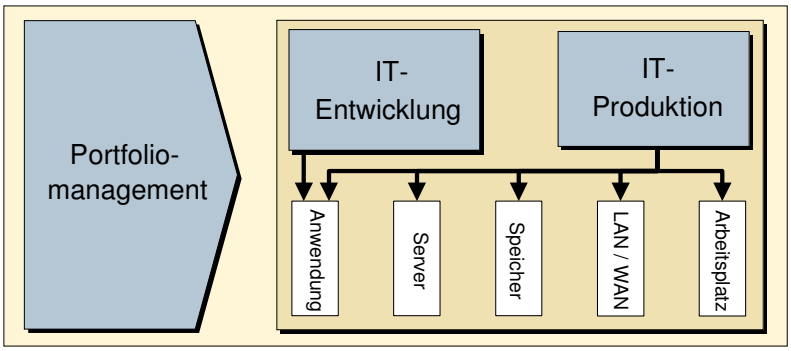

Abbildung 3: IT-Leistungserstellung als Fertigungsprozess[20]

Das Portfolio-Management legt die Eigenschaften der Leistungen fest und definiert die Anforderungen an die IT-Entwicklung und die IT-Produktion. Die technische Gestaltung, die sich im Wesentlichen auf die Anwendungsentwicklung bezieht, erfolgt in der IT-Entwicklung. Die IT-Produktion ist für die Gestaltung der Produktionsinfrastruktur zuständig.

Die Betrachtung als Fertigungsprozess ermöglicht die Übertragung erfolgreicher Managementansätze und -methoden aus der industriellen Fertigung auf die IT-Leistungserstellung [ZaBP05]. Dabei kann der IT-Leistungserbringer die Leistungen nur wirtschaftlich herstellen, wenn die drei Aktivitäten Portfoliomanagement, IT-Entwicklung und IT-Produktion eng miteinander verzahnt werden. Beispielsweise kann eine aus Sicht der Entwicklung ideal gestaltete Lösung ohne Blick auf die Produktionsanforderungen zu Qualitäts- und Kostenproblemen führen. Ferner kann ein schlechtes Portfoliomanagement, trotz guter Leistung in Entwicklung und Produktion, zu Lösungen mit geringem Kundennutzen führen.

[19] Ist die IT komplett an einen externen Dienstleister ausgelagert, ist die Einschränkung in dieser Arbeit auf die Anwendungsentwicklung der Fertigungsindustrie nicht mehr gegeben.

[20] Siehe [ZaBP05].

Die Verzahnung dieser drei Aktivitäten erfolgt aufgrund einer horizontalen und vertikalen Integration (siehe Abbildung 4). In [ZaBP05] wird dies als integriertes Management bezeichnet.

Die horizontale Integration bezieht sich auf die Schnittstellen zwischen den drei Aktivitäten. Das Ziel ist die Entwicklung von durchgängigen Managementkonzepten über alle drei Aktivitäten hinweg.

Die vertikale Integration betrachtet die unterschiedlichen Handlungsebenen der IT-Leistungserstellung. In jeder Aktivität gibt es strategische, planerische sowie operative Aufgaben, die nicht isoliert betrachtet werden dürfen.

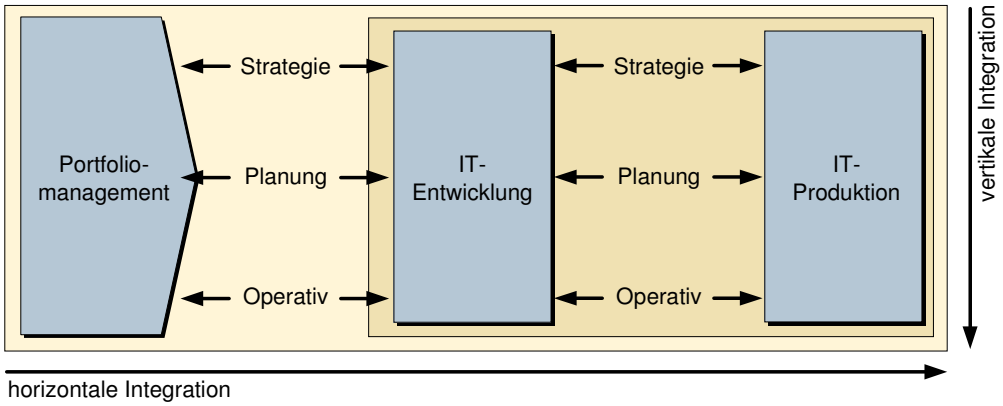

Abbildung 4: Horizontale, vertikale Integration der IT-Leistungserstellung[21]

Bei der horizontalen Integration spielt insbesondere die optimale Abstimmung der strategischen Ebene der drei Aktivitäten eine wichtige Rolle. [ZaBP05] verdeutlichen dies mit folgendem Beispiel: Ein Unternehmen hat eine Umfrage bei seinen Kunden bzgl. der Zufriedenheit seiner Dienstleistungen durchgeführt. Ein Ergebnis war die verstärkte Nachfrage nach Internet-Anwendungen. Als Konsequenz aus dieser Umfrage wurde im Portfolio ein neues Leistungssegment „Internet“ eingerichtet. Für die IT-Entwicklung hat dies zur Folge, eine entsprechende Strategie für die Entwicklung von Internet-Anwendungen zu erstellen. Beispielsweise wird festgelegt, dass alle Internet-Anwendungen auf Basis des Microsoft .Net-Frameworks realisiert werden. Diese Entscheidung hat für die Produktion wiederum zur Konsequenz, dass Microsoft-basierte Serverplattformen zur Verfügung gestellt werden müssen. Sofern die Produktionsstrategie dies nicht vorsieht oder sogar ausschließt, ist die Entwicklungsstrategie neu auszurichten.

Im Einzelnen sind den drei Aktivitäten für die strategische Ebene nach [ZaBP05] die folgenden Aufgaben zugeordnet:

[21] Siehe [ZaBP05].

Die *Portfoliostrategie* legt langfristig den Aktionsbereich des Leistungserbringers fest und definiert damit die Strategie für das Leistungsportfolio. Im Einzelnen sind dies die folgenden Aufgaben:

- Identifikation von Leistungssegmenten, die einen langfristigen Markterfolg versprechen. Dies beinhaltet neben der Betrachtung der Kundenanforderungen auch Informationen aus Entwicklungs- und Produktionsstrategie.
- Bewertung und Positionierung der identifizierten Leistungssegmente.
- Formulierung von Strategien für den langfristigen Umgang mit den Leistungssegmenten.

Die langfristigen, strategischen Rahmenbedingungen für den Entwicklungsprozess werden in der *Entwicklungsstrategie* festgelegt. Diese umfasst die folgenden Aufgaben:

- Festlegung von Positionen und Verantwortungsbereichen für die Organisation der Entwicklung. Hierbei ist insbesondere die Bildung von Gremien für die Spezifikation von Verfahren, Standards und Richtlinien wichtig.
- Festlegung der Entwicklungsprinzipien und Standards. Entwicklungsprinzipien sind allgemeingültige Grundsätze, die für die Entwicklung von Anwendungen gelten. Hierzu zählt neben der Auswahl einer Methode zur Implementierung von Anwendungen auch die Gestaltung bzw. der Aufbau einer Anwendung. Die Standards legen zudem fest, welche Technologien, Plattformen etc. in der Entwicklung zu nutzen sind.
- Rahmenbedingungen für Entwicklungswerkzeuge und -sprachen. Die in der Strategie festgelegten Prinzipien der Anwendungsentwicklung haben Einfluss auf die Werkzeuge. Diese Werkzeuge sowie die Entwicklungssprachen werden durch die verantwortlichen Gremien festgelegt.
- Strategische Ausrichtung des Anwendungsportfolios. Ähnlich dem Leistungs-Portfolio sind die verschiedenen Anwendungen strategisch zu planen, um die Vielzahl an Anwendungen zu beherrschen. Die Anwendungsarchitektur beschreibt dabei das Soll-Anwendungsportfolio aus fachlicher und technischer Sicht.

Für eine effiziente Gestaltung der Produktionsinfrastruktur bedarf es einer Strategie, die mit den Strategien von Entwicklung und Portfolio abgestimmt ist. Die *Strategie* im Umfeld der *Produktion* hat die folgenden Aufgaben:

- Die Organisation der Produktion ist analog der Entwicklung auszurichten. Hierbei sind insbesondere die räumliche Gestaltung der Infrastruktur, die Entscheidung über die Beschaffung von Anlagen sowie deren Kapazitätseigenschaft und Instandhaltung eine zentrale Fragestellung.

- Designprinzipien und Standards sind für die Gestaltung der Produktionsinfrastruktur wesentliche Faktoren. Für den Aufbau ist beispielsweise die Modularität, die Skalierbarkeit, die Flexibilität, die Sicherheit oder Fehlertoleranz der Systeme in dieser Infrastruktur zu bestimmen.
- Die strategische Ausrichtung der Systemarchitektur definiert die langfristige Gestaltung der Produktionsinfrastruktur.
- Die Rahmenbedingungen für Werkzeuge für die Steuerung und Überwachung der Produktionsinfrastruktur sind in der Strategie festzulegen.

Auf Basis dieser Überlegungen wird die IT-Leistungserstellung als industrieller Fertigungsprozess betrachtet. Die Entwicklung von Anwendungen ist im Rahmen der IT-Entwicklung ein wesentlicher Bestandteil dieses Prozesses. Welche Managementkonzepte der Fertigungsindustrie auf die Anwendungsentwicklung angewandt bzw. übertragen werden können, ist Gegenstand der Untersuchungen dieser Arbeit. Der Grad der Industrialisierung gibt dabei Auskunft über die Etablierung industrieller Methoden bei der Realisierung von Anwendungen.

In diesem Zusammenhang wird auch die IT-Produktion betrachtet, da die strategischen Entscheidungen für die IT-Entwicklung Auswirkungen auf die Beschaffenheit der IT-Produktion haben.

Das Portfoliomanagement der IT-Leistungen ist eine strategische Aufgabe der IT-Leitung in Abstimmung mit dem (internen) Kunden. Die Festlegung auf ein Portfolio hat zwar Auswirkungen auf die Entwicklung von Anwendungen aufgrund des gewählten Leistungsspektrums, für die Methoden in der Anwendungsentwicklung spielt sie jedoch keine Rolle. Daher wird das Portfolio-Management in dieser Arbeit nicht näher betrachtet.

Für die planerische sowie operative Ebene der drei Aktivitäten werden in [ZaBP05] ebenfalls die entsprechenden Aufgaben definiert. Sie spielen im Kontext dieser Arbeit jedoch keine Rolle, da der Schwerpunkt der Betrachtung auf den Prinzipien bzw. Methoden für die Entwicklung von Anwendungen liegt. Diese Ebenen werden daher nicht näher erläutert.

2.2.1 Grenzen der Industrialisierung

Die Prinzipien der Industrialisierung bieten Potentiale für die Optimierung sowie den Ausbau von IT-Leistungen. Eine undifferenzierte Etablierung dieser Prinzipien bei der Anwendungsentwicklung in allen Bereichen eines Unternehmens sollte jedoch hinterfragt werden.

Unternehmen, die in direktem Wettbewerb stehen, müssen sich in Bezug auf ihre Produkte und Dienstleistungen differenzieren. Nach [KaSc07] sind die hierfür notwendigen IT-Leistungen ebenso zu unterscheiden. Diese lassen sich in die folgenden Sichten gliedern:

- *Sicherstellung der funktionalen Differenzierung*. Bestimmte Bereiche des Unternehmens wie z.B. die Buchhaltung lassen sich hochgradig standardisieren. Dies ergibt sich bereits aus der Gesetzgebung, welche die Einhaltung bestimmter Regeln einfordert.

  Für die spezifischen Prozesse, die ein Unternehmen von anderen unterscheidet, können meist für den Prozess als Ganzes keine Standards verwendet werden. Die Individualität der Entwicklung kann hierbei ein Vorteil gegenüber der Industrialisierung sein.

- *Sicherstellen der temporalen Differenzierung*. Die zeitnahe Vermarktung neuer Produkte kann ein entscheidender Wettbewerbsvorteil sein. Unternehmen benötigen für diese Differenzierung spezifische Prozesse und Werkzeuge, die meist nicht den industriellen Prinzipien entsprechen.

- *Beherrschbarkeit der Steuerkomplexität*. Die Arbeitsteilung ist eine wichtige Eigenschaft der Industrialisierung. Sofern diese Arbeitsteilung jedoch zu filigran wird, steigt der Aufwand für die Steuerung bis zu dem Punkt, an dem sie nicht mehr beherrschbar ist.

Jedes Unternehmen hat für sich zu entscheiden, in welchen Bereichen welche Prinzipien der Industrialisierung angewandt werden sollen. In dieser Arbeit wird daher nicht der Industrialisierungsgrad für das ganze Unternehmen bestimmt, sondern spezifisch für ausgewählte Bereiche ermittelt.

2.3 Entwicklung von Anwendungen

Die effiziente Unterstützung der Entwicklung von Anwendungen mit industriellen Mitteln ist das zentrale Thema dieser Arbeit. In der Literatur[22] finden sich viele, zum Teil sehr unterschiedliche Ansätze, das Thema Anwendungsentwicklung zu gestalten.

Vereinfacht ausgedrückt besteht die Entwicklung von Anwendungen in der Aufnahme und Umsetzung von Anforderungen sowie dem Installieren der erstellten Anwendung auf einem System. Strukturiert wird diese Tätigkeit durch die folgenden Aspekte:

- *Entwicklungsprozess:* Strukturierung der Aktivitäten in der Entwicklung nach einem bestimmten Vorgehensmodell sowie Installation der Anwendung auf ein produktives System.

- *Artefakte:* Bei der Entwicklung einer Anwendung werden verschiedene Ergebnisse, die Artefakte, erzeugt. Hierzu gehören u.a. die Anforderungen an eine Lösung, die Tests zur Qualitätssicherung der entwickelten

[22] Siehe u.a. [Andr04], [Balz01], [Dumk03], [GrSh06],[LiRW02].

Anwendung, eine technische und fachliche Dokumentation sowie die Anwendung selbst.

- *Entwicklungsmethode:* Das Erzeugen von Artefakten kann auf unterschiedliche Arten erfolgen. Eine Methode beschreibt den Ablauf bei der Erstellung von Artefakten einer Anwendung.
- *Entwicklungsinfrastruktur:* Die Entwicklung von Anwendungen basiert auf einer Infrastruktur, die Werkzeuge und Richtlinien für die Erstellung von Artefakten sowie Systeme zur Verwaltung der erstellten Artefakte enthält.
- *Laufzeitinfrastruktur:* Der Einsatz von Anwendungen erfolgt in einer Laufzeitumgebung, die alle notwendigen Dienste für den Betrieb und die Nutzung der Anwendung zur Verfügung stellt.
- *Gremien:* Die Erstellung von Vorgaben für die Entwicklung von Anwendungen wird durch zentrale Gremien koordiniert. Vorgaben werden u.a. für Methoden, Infrastrukturen sowie Prozesse erstellt.
- *Unterstützende Prozesse:* Die zentrale Festlegung von Abläufen, Standards etc. für die Realisierung von Anwendungen wird über zentrale Prozesse gesteuert, die durch die zuständigen Gremien koordiniert werden.

Abbildung 5 setzt die verschiedenen Aspekte in Beziehung zueinander. In den hier vereinfacht dargestellten Phasen des Entwicklungsprozesses der Anwendungsentwicklung sind verschiedene Artefakte zu erstellen. Je nach Grad der Industrialisierung erfolgt die Erzeugung dieser Artefakte nach der dafür vorgesehenen Methode. Für eine effiziente Betreuung einer Anwendung über ihren Lebenszyklus ist die Erstellung weiterer Artefakte wichtig, wie z.B. die Dokumentation.

Die Entwicklungsinfrastruktur ermöglicht die Konstruktion der Anwendung anhand der erstellten Artefakte. Sie stellt Mechanismen für den Transport der Anwendung zu einem Testsystem sowie zu einem produktiven System zur Verfügung.

Die Gremien sowie die unterstützenden Prozesse stellen die Grundlagen für den Ablauf der Entwicklung von Anwendungen sicher.

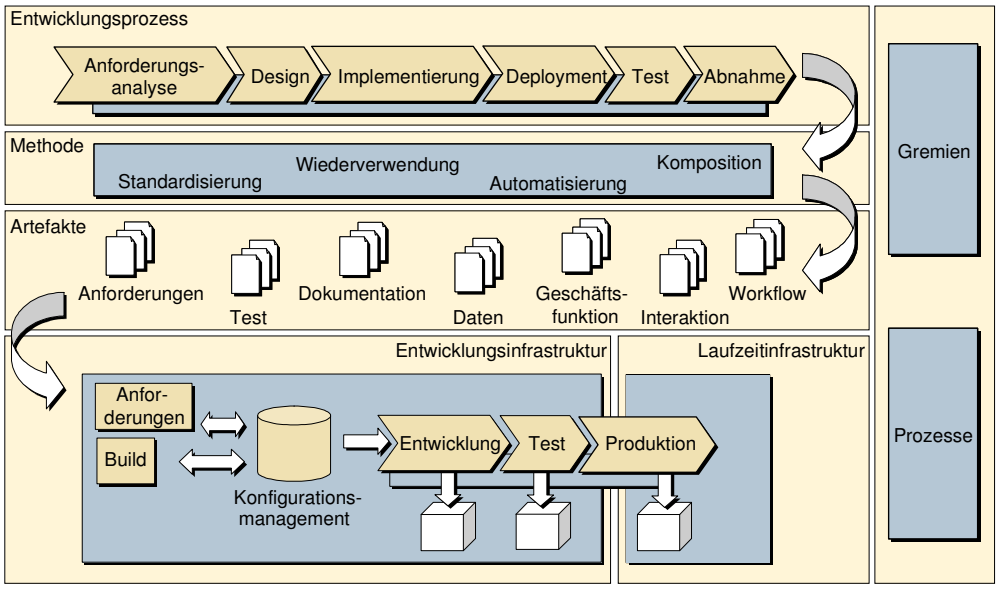

Abbildung 5: Verschiedene Aspekte der Anwendungsentwicklung

2.3.1 Entwicklungsprozess

Der Software-Entwicklungsprozess ist nach [Dumk03] der gesamte Prozess der Aufgabenstellung, Planung, Realisierung und Bewertung einer Anwendung. Es existieren verschiedene Vorgehensmodelle[23], die jeweils den Entwicklungsprozess mit unterschiedlichen Ausprägungen durchlaufen. In [BuKn02] werden die folgenden Familien von Vorgehensmodellen vorgestellt:

- *Phasen-, Wasserfall-, Schleifenmodelle:* Bei diesen Modellen werden Aktivitäten bei der Entwicklung einzelnen Phasen zugeordnet, die sequentiell durchgeführt werden. Beim Übergang zwischen den einzelnen Phasen werden die erstellten Ergebnisse an die nächste Phase übergeben Die vorhergehende Phase wird dabei abgeschlossen. Ein bekannter Vertreter dieser Familie ist das V-Modell (siehe [Dumk03]).
- *Prototypische Vorgehensmodelle:* Die Phasen der Entwicklung werden in diesem Modell nicht sequentiell, sondern wiederholt durchlaufen. Zu verschiedenen Zeitpunkten ist die Entwicklung von Prototypen vorgesehen. Die gewonnenen Erfahrungen aus den Prototypen führen zu einer erneuten Ausführung bereits abgearbeiteter Phasen. Ein Beispiel dieses Vorgehensmodells ist das Extreme Programming (XP) (siehe [LiRW02]).
- *Inkrementelle, evolutionäre, rekursive und iterative Verbesserungsmodelle:* Die Entwicklung von Anwendungen bei diesen Modellen erfolgt in In-

[23] Effizienter und effektiver Ablaufplan zur Erstellung von Software [LeWS08].

krementen. Zunächst wird eine Teilmenge, ein erstes Inkrement, der Anforderungen sequenziell in Phasen bearbeitet. Anschließend wird dieses Inkrement um zusätzliche Anforderungen erweitert. Die Weiterentwicklung der Anwendung ist damit bereits Bestandteil des Vorgehensmodells. Zu dieser Familie gehört u.a. der Rational Unified Process (RUP) (siehe [Kruc00]).

Die Vorgehensmodelle unterscheiden sich in der Durchführung der verschiedenen Aktivitäten bei der Entwicklung von Anwendungen. Die Arten der Aktivitäten sind jedoch bei allen Modellen ähnlich. Diese Arten sind:

- *Bestimmung der Anforderungen:* Identifizierung und Erfassung der fachlichen Anforderungen des Auftraggebers.
- *Analyse und Design:* Transformation der fachlichen Anforderungen in eine technische Konzeption für die Realisierung der geforderten Anwendung.
- *Implementierung:* Realisierung der Anwendung auf Basis der erstellten Konzeption.
- *Deployment:* Bereitstellen der implementierten Anwendung auf einem Entwicklungs-, Test- sowie einem Produktivsystem.
- *Test:* Test der bereitgestellten Anwendungen auf Basis der erfassten Anforderungen.
- *Abnahme:* Freigabe der getesteten Anwendung durch den Auftraggeber.

Entwicklungsprozesse werden seit der Softwarekrise sehr genau betrachtet. Mittlerweile gibt es Maßnahmen zur Steigerung der Produktivität und Qualität der Softwareentwicklung durch die effiziente und effektive Gestaltung des Entwicklungsprozesses. Der Reifegrad dieses Prozesses kann durch etablierte Verfahren wie SPICE [HDHM06] oder CMMI [Kneu06] bestimmt werden. Bei diesen Verfahren werden verschiedene Prozessgebiete bei der Entwicklung von Anwendungen, z.B. die Projektplanung oder das Management von Anforderungen, unterschieden. Diese Prozessgebiete werden im Rahmen von Assessments[24] betrachtet und bewertet. Die Erkenntnisse aus diesen Untersuchungen geben Auskunft über mögliche Optimierungen der Prozesse in der Entwicklung.

Der Schwerpunkt dieser Arbeit liegt weniger in der Optimierung von Prozessen der Entwicklung als vielmehr der Optimierung von Methoden zur Transformation von Anforderungen in ausführbare Anwendungen. Der Entwicklungsprozess wird daher nicht weiter betrachtet.

[24] Durchführung einer Bewertung des Entwicklungsprozesses eines Unternehmens. Wird in der Regel durch zertifizierte externe Prüfer durchgeführt [Kneu06].

2.3.2 Artefakte

Artefakte sind allgemein die greifbaren Informationselemente, die von einem Prozess erzeugt, geändert oder genutzt werden [Kruc00]. Sie existieren in verschiedenen Formen wie Modellen bzw. Modellelementen, Dokumenten, Quelltexten oder auch als ausführbare Anwendungen. Abbildung 6 stellt die in dieser Arbeit relevanten Artefakte in einer Übersicht dar. Die Erläuterung der dargestellten Schichten erfolgt im Abschnitt 2.3.2.1 und 2.3.2.2.

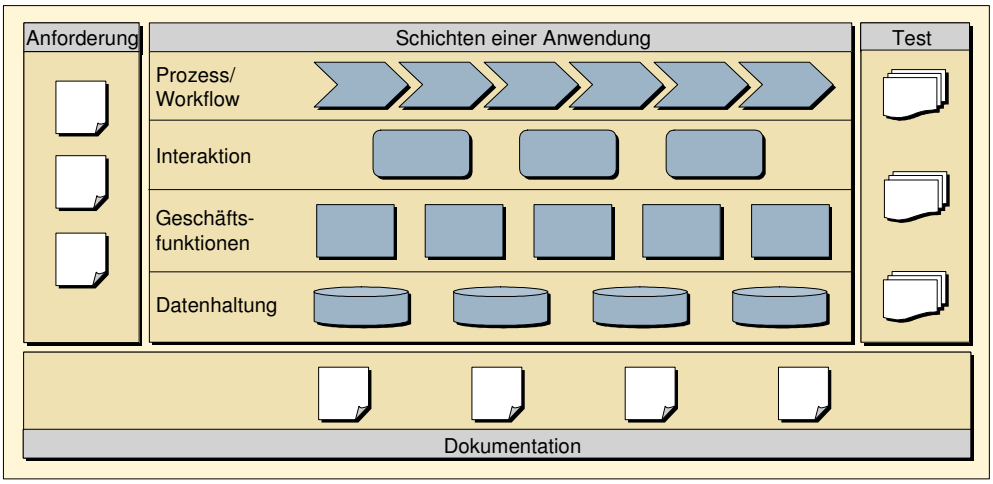

Abbildung 6: Artefakte der Anwendung sowie des Entwicklungsumfeldes

Aus dieser Grafik lassen sich zwei unterschiedliche Kategorien von Artefakten entnehmen:

- *Artefakte der Anwendungsarchitektur*: Diese Kategorie enthält alle Artefakte aus den Schichten einer Anwendung. Sie sind direkte Bestandteile einer Anwendung, die für eine Ausführung benötigt werden.
- *Artefakte des Entwicklungsumfeldes*: Anforderungen, Dokumentation, Test-Spezifikationen sowie Test-Protokolle sind bei der Realisierung sowie der Wartung von Anwendungen wichtige Artefakte. Sie sind nicht direkter Bestandteil einer Anwendung, sondern gehören zum Umfeld der Entwicklung.

2.3.2.1 Artefakte der Anwendungsarchitektur

Wesentliches Merkmal einer Anwendung ist die Architektur. Sie bildet die konzeptionelle Grundlage und ist ein Rahmenwerk für Veränderungen. Die Art und Weise des Aufbaus einer Anwendung gibt Aufschluss über ihre Qualität.

Nach [Star02] hat sich in Software-Architekturen die Schichtenbildung als ein wichtiges Mittel zur Strukturierung von Anwendungen etabliert. Jede Schicht bie-

tet der darüberliegenden Schicht Dienste an und kapselt dabei den internen Aufbau. Jede Schicht kann ein beliebig komplexes Subsystem darstellen.

Die wesentlichen Vorteile der Organisation in Schichten sind: (vgl. [Star02])

- Unabhängigkeit: Die verschiedenen Schichten sind voneinander unabhängig sowohl in der Erstellung als auch in der Betreuung.
- Abhängigkeiten zwischen Komponenten werden minimiert.
- Möglichkeit zum Austausch: Einzelne Schichten können gegen andere ausgetauscht werden, sofern die angebotenen Dienste gleich bleiben.
- Leicht verständliches Strukturkonzept.

Die Aufteilung der Schichten nach Präsentation, Fachlichkeit und Infrastruktur hat sich in der Praxis bewährt [Star02]. Diese Schichtung ist ein wichtiger Bestandteil bei der Industrialisierung der Softwareentwicklung. Durch die Aufteilung einer Anwendung in Artefakte verschiedener Schichten können industrielle Prinzipien wie Arbeitsteilung oder Wiederverwendung eingesetzt werden. Dieser Zusammenhang wird in Kapitel 3 näher beschrieben.

Im Rahmen dieser Arbeit werden die folgenden Schichten einer Anwendung insbesondere für die Analyse betrachtet (siehe Abbildung 6):

- *Workflow – Abbildung von Prozessen:* Die Abbildung von Prozessen erfordert die Bearbeitung von einzelnen Prozessschritten durch verschiedene Personen. Diese Abfolge wird durch einen Workflow abgebildet. Hierbei werden die für den Prozess benötigten Eingaben, z.B. von Genehmigungen, von den zuständigen Personen angefordert. Der Fortschritt eines Prozesses hängt damit von der Bearbeitung der einzelnen Schritte ab.
- *Interaktion – Präsentation und Bedienung der Anwendung:* Die Präsentation von Daten aber auch deren Eingabe und Modifikation erfolgt über die Präsentationsschicht der Anwendung. Hier tritt der Benutzer in Interaktion mit der Anwendung.
- *Geschäftsfunktionen – Abbildung der Geschäftslogik:* Die eigentlichen Geschäftsfunktionen einer Anwendung werden unabhängig von der Interaktion in einer eigenen Schicht realisiert. Dies ermöglicht z.B. die Nutzung verschiedener Technologien der Interaktionsschicht.
- *Daten – Zugriff und Verwaltung von Daten:* Die Verwaltung von Daten erfolgt ebenfalls in einer eigenen Schicht. Hierbei ist darauf zu achten, dass die Daten verfügbar sind und einen konsistenten Zustand haben. Diese Schicht enthält auch die Zugriffslogik auf Daten, die in unterschiedlichen Datenquellen organisiert sein können.

Die Artefakte der einzelnen Schichten werden bei der Analyse für die Bestimmung des Industrialisierungsgrades betrachtet. Dabei wird überprüft, welche der in Kapitel 3 beschriebenen Methoden auf die Artefakte angewandt wurde.

2.3.2.2 Artefakte des Entwicklungsumfeldes

Anwendungen werden im Laufe ihres Lebenszyklus mehrfach modifiziert, indem Fehler behoben oder neue Funktionalitäten hinzugefügt werden. Um diese Arbeiten effizient durchführen zu können, ist es wichtig, neben den Artefakten der Anwendungsarchitektur auch weiterführende Informationen zur Verfügung zu haben. Diese sind:

- *Anforderungen:* Die Entwickler einer Anwendung benötigen die Anforderungen in einer Form, welche den Umfang sowie die Qualität der erwarteten Funktionalität möglichst genau beschreibt. Die Anforderungen sind auch im Rahmen des Lebenszyklus fortzuschreiben, um jederzeit ein klares Verständnis der Anforderungen an die Anwendung zu haben.
- *Dokumentation:* Die Beschreibung der Anwendung ist eine wichtige Voraussetzung für das Verständnis des Aufbaus sowie des Ablaufs einer Anwendung. Dies gilt sowohl für den Anwender, den Entwickler sowie den Anwendungsverantwortlichen.

  Bestandteile der Dokumentation sind die Beschreibung der Außensicht und der Innensicht. In der Außensicht wird die Anwendung in ihrer Außenwirkung zum Benutzer oder zu anderen Anwendungen beschrieben. Sie enthält z.B. die Benutzerdokumentation. In der Innensicht werden der innere Aufbau sowie das Zusammenwirken der verschiedenen Bestandteile der Anwendung erläutert.
- *Test:* Die Sicherung der Qualität ist ein wichtiger Bestandteil der Industrialisierung. Sie erfolgt im Kontext der Anwendungsentwicklung auf Basis der Planung und Durchführung von Tests. Es lassen sich vier verschiedene Arten von Tests identifizieren, die sowohl während der Entwicklung als auch zur Abnahme der Anwendung erfolgen:
  - *Modultest (Unit Test):* Modultests, engl. unit tests, werden direkt während der Programmierung von Entwicklern durchgeführt. Sie beziehen sich auf die einzelnen Einheiten, die Module, und nicht auf die gesamte Anwendung. Diese Einheiten können dabei eine unterschiedliche Granularität aufweisen, wie z.B. Methoden, Programmklassen oder Subsysteme.

    Eine weitere Anwendung des Modultests erfolgt in der regelmäßigen Durchführung von zentralen Tests. Hierbei werden die einzelnen Modultests gesammelt ausgeführt, um festzustellen, ob Änderungen an einzelnen Modulen einer Anwendung die Korrektheit der restlichen

Module beeinflusst haben. Modultests sind geeignete Vorstufen zum Integrationstest.

- *Integrationstests:* Insbesondere bei größeren Anwendungen wird die Implementierung auf mehrere Entwickler aufgeteilt. In regelmäßigen Abständen ist dabei zu prüfen, ob die verschiedenen Teile als Ganzes ablauffähig sind. Der Integrationstest stellt sicher, dass die verteilten Entwicklungen reibungsfrei zusammenarbeiten.
- *Funktionale Tests:* Die geforderte Funktionalität einer Anwendung wird über diese Art des Tests sichergestellt. Hierbei prüfen ein oder mehrere Tester, ob die realisierten Funktionalitäten vollständig und fehlerfrei sind.
- *Lasttests:* Anwendungen werden unter Laborbedingungen entwickelt. Der Entwickler testet die implementierten Funktionen aus Sicht eines einzelnen Anwenders. Die gleichzeitige Nutzung der Anwendung durch viele Anwender kann jedoch eine Last erzeugen, die zu Problemen in der Ausführung der Anwendung führen kann. Diese Situationen werden durch Lasttests simuliert.

Die Durchführung dieser Tests erfolgt auf eine jeweils unterschiedliche Art. Bei der Analyse des Industrialisierungsgrades ist lediglich das Erzeugen der verschiedenen Testartefakte relevant. Bei der Betrachtung der verschiedenen Methoden der Industrialisierung werden die Tests nicht weiter differenziert, da das Erstellen dieser Testartefakte nach den gleichen Mustern erfolgt.

2.3.3 Entwicklungsmethode

Die Realisierung der Anforderungen einer Anwendung kann auf recht unterschiedliche Art und Weise erfolgen. Nach [LeWS08] beschreibt eine Methode ein systematisches und zielgerichtetes Vorgehen, das zur Lösung einer Klasse von Problemen verwendet wird. Im Kontext dieser Arbeit definiert eine Methode die Mechanismen und Regeln zur Erstellung von Artefakten. Ausgehend von den Anforderungen werden diese

- *manuell erstellt:* Die benötigten Artefakten werden vom Entwickler individuell von Hand erstellt. Hierbei hat er für die Gestaltung der einzelnen Artefakte die meisten Möglichkeiten, jedoch auch den meisten Aufwand.
- *automatisch erstellt bzw. generiert:* Die Artefakte werden anhand von Regeln und Vorlagen durch Generatoren automatisiert erstellt[25]. Der Ent-

[25] Siehe auch Abschnitt 3.7.2.

wickler spezifiziert bzw. modelliert hierbei die Anwendung in der für den Generator vorgesehenen Syntax.

- *wiederverwendet:* Komponenten mit mehrfach benötigter Funktionalität werden zentral implementiert und können in verschiedene Anwendungen eingebunden werden.
- *komponiert:* Die für die Anwendung geforderte Funktionalität ist bereits in einem produktiven Umfeld vorhanden. Sie kann z.B. als Service über ein entsprechendes Service-Verzeichnis zur Verfügung gestellt werden. Die Aufgabe des Entwicklers ist die Komposition dieser Funktionalitäten zu einer neuen Anwendung.

Die jeweiligen Ausprägungen der Mechanismen können sehr unterschiedlich ausfallen. Dies ist abhängig von verschiedenen Aspekten wie der gewählten Technologie, der Entwicklungsplattform, aber auch dem Entwicklungsprozess. Auch die Kombination der Mechanismen ist in der Praxis anzutreffen. Es liegt in der Aufgabe des Unternehmens festzulegen, welche Methode für die Entwicklung von Anwendungen zu verwenden ist und welche Regeln dabei gelten.

2.3.4 Entwicklungsinfrastruktur

Neben den Elementen der Anwendungsarchitektur ist für die Erstellung von Anwendungen auch das Entwicklungsumfeld wesentlich. Nach [ZaBP05] entsteht der größte Anteil der Kosten einer Anwendung in der Phase der Wartung. Nach der erstmaligen Realisierung der Anforderungen werden im Verlauf des Lebenszyklus einer Anwendung weitere Funktionen implementiert und Fehler korrigiert. Daher sind auch die Elemente der Entwicklungsinfrastruktur ein wesentlicher Faktor für den Grad der Industrialisierung.

Die Entwicklungsinfrastruktur stellt Werkzeuge und Systeme für die Realisierung von Anwendungen zur Verfügung. Diese Realisierung muss nachvollziehbar und wiederholbar sein. Abbildung 7 stellt die verschiedenen Elemente der Entwicklungsinfrastruktur dar. Hierbei wird zwischen lokaler und zentraler Entwicklungsumgebung unterschieden.

Die lokale Umgebung ist der Entwicklungsrechner des jeweiligen Entwicklers. Dieser enthält alle notwendigen Werkzeuge sowie Informationen für die Implementierung von Anwendungen. Über eine zentrale Umgebung laufen alle Entwicklungen zusammen, die anschließend auf die Test- und Produktiv-Systeme ausgebracht werden.

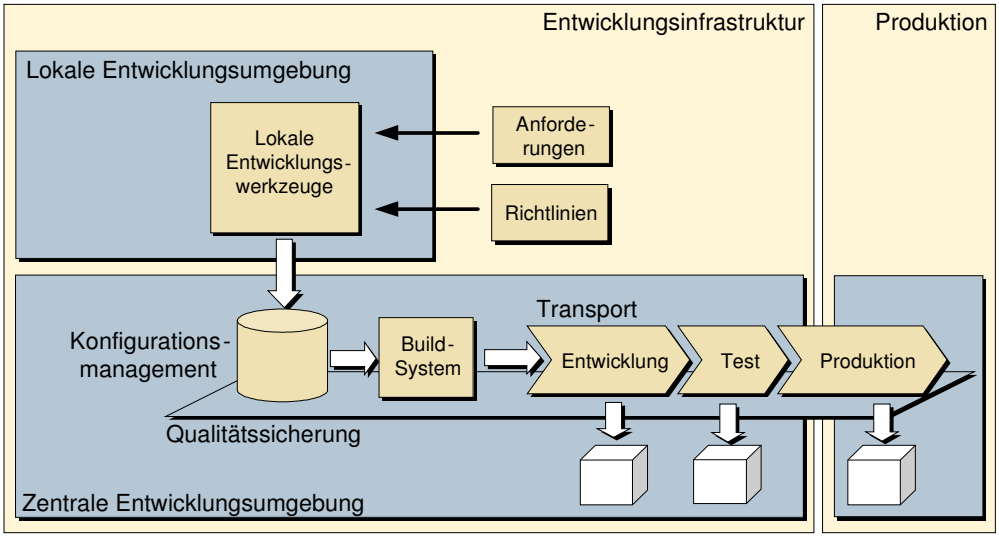

Abbildung 7: Entwicklungsinfrastruktur

Die Entwicklungsinfrastruktur enthält die folgenden Elemente:

- *Entwicklungswerkzeuge:* Für die Erstellung von Artefakten stellt die Entwicklungsinfrastruktur Werkzeuge zur Verfügung, welche die betrachteten Methoden unterstützen und lokal installiert werden. Hierzu gehören neben der integrierten Entwicklungsumgebung (IDE[26]) zur Erstellung von Quelltexten auch Werkzeuge zur Generierung oder Modellierung von Anwendungen.
- *Zentrales Konfigurationsmanagement:* Die erstellten Quelltexte einer Anwendung werden zentral durch eine Quelltext-Versionsverwaltung organisiert. Damit wird sichergestellt, dass alle Entwickler auf den aktuellen Stand der Anwendung zugreifen können. Ferner besteht die Möglichkeit, alte Stände wiederherstellen zu können.

  Neben Quelltexten kann jede Art von Datei in diesem System hinterlegt und damit zentral verwaltet werden.
- *Zentraler Build Service:* Verschiedene Technologien, wie z.B. Java[27], benötigen nach der Programmierung des Quelltextes einen weiteren Schritt zum Erzeugen einer ausführbaren Anwendung. Dieser Schritt wird im Folgenden auch als „Build“[28] bezeichnet.

[26] IDE – Integrated Development Environment: Anwendungsprogramm zur Entwicklung von Software. Es enthält u.a. einen Texteditor, Compiler sowie Werkzeuge zur Fehleranalyse.

[27] Java: objektorientierte Programmiersprache.

[28] Gängige Bezeichnung in der Softwareentwicklung.

Die automatische Erstellung einer solchen Anwendung aufgrund der verfügbaren Quelltexte sowie spezifischer Parameter wird durch einen zentralen Service, den Build Service, zur Verfügung gestellt. Dies ermöglicht ein nachvollziehbares Erstellen von Anwendung ohne die speziellen Einstellungen der lokalen Umgebungen von Entwicklern.

- *Transport und Deployment*[29]*:* Die Entwicklung einer Anwendung, das Testen der Funktionalität sowie die produktive Nutzung erfolgt auf unterschiedlichen Systemen. Diese Trennung vermeidet unerwünschte Effekte bei der Entwicklung und Nutzung von Anwendungen. Ein zentrales Transportsystem zwischen den Systemen sowie ein automatisiertes Deployment ermöglicht eine nachvollziehbare Ausbringung einer Anwendung auf das Test- sowie das Produktiv-System.
- *Entwicklungsrichtlinien:* Richtlinien sind Vorgaben für die Entwicklung von Anwendungen. Sie dienen der Sicherstellung eines Mindestmaßes an Qualität der zu erstellenden Anwendungen und müssen für jeden Entwickler zur Verfügung stehen.

Je nach Technologie und Plattform kann es unterschiedliche Entwicklungsinfrastrukturen geben. Die grundsätzlichen Bestandteile, die hier beschrieben wurden, müssen jedoch in jeder Umgebung vorhanden sein.

2.3.5 Laufzeitinfrastruktur

Die Entwicklungsinfrastruktur stellt Werkzeuge und Systeme zur Verfügung, um eine Anwendung zu realisieren. Eine fertigerstellte Anwendung wird auf eine Umgebung ausgebracht, über welche sie der Anwender ausführen kann. Dies kann sowohl auf einem Desktop-Rechner, z.B. einem Büroprogramm wie Microsoft Word, als auch auf einem Server erfolgen, wie z.B. einer webbasierten Anwendung. Diese Umgebung wird auch als Laufzeitumgebung bezeichnet. Sie stellt eine Softwareschicht zwischen der Anwendung und dem Betriebssystem dar, die der Anwendung verschiedene Basisfunktionen, wie z.B. den Zugriff auf das Dateisystem oder eine Datenbank, zur Verfügung stellt.

Verschiedene Dienste in Form von Systemen oder Komponenten, die zur Laufzeit von einer Anwendung genutzt werden, bilden die Laufzeitinfrastruktur. Neben technischen Basisfunktionen werden hierbei auch fachliche Funktionen bereitgestellt, die von verschiedenen Anwendungen zur Laufzeit aufgerufen werden können.

---

[29] Deployment: Die Ausbringung bzw. Installation von Anwendungen auf einem Server wird auch als Deployment bezeichnet (gängige Bezeichnung in der Softwareentwicklung).

Je nach Reifegrad der Infrastruktur sind diese Funktionen fest mit der Anwendung verbunden oder lose miteinander gekoppelt. Sofern sie fest verbunden sind, werden sie bereits während der Entwicklung in die Anwendung integriert.

Eine lose Koppelung erlaubt dagegen die Einbindung der Fachlichkeit zur Laufzeit. Welche Vorteile sich hieraus ergeben, wird näher in Kapitel 3 (Stufe 5: Komposition) beschrieben. Hierfür sind jedoch zusätzliche Komponenten einer Infrastruktur notwendig.

Die Beschaffenheit dieser Infrastruktur ist damit für den Grad der Industrialisierung ebenfalls ein wichtiger Faktor.

2.3.6 Gremien

Je komplexer und umfangreicher eine IT-Anwendungslandschaft ist, desto mehr besteht die Notwendigkeit der Koordination der darin enthaltenen Komponenten. Hierfür werden in größeren Unternehmen Gremien mit der Aufgabe eingerichtet, Regeln bei der Bebauung der IT-Landschaft aufzustellen und für deren Einhaltung zu sorgen. Gremien müssen bzgl. ihrer Mitglieder nicht disjunkt sein, da verschiedene Aspekte in mehreren Gremien bearbeitet werden können.

Die folgenden Gremien sind insbesondere bei größeren Unternehmen bzw. Konzernen anzutreffen:

- *Architektur – Architektur-Board:* Die IT-Architektur besteht aus verschiedenen Ebenen. Hierzu gehört die Systemarchitektur, Anwendungsarchitektur sowie die Geschäftsarchitektur.

  In der Systemarchitektur wird der Aufbau von Rechnernetzen sowie der darin enthaltenen Systeme bestimmt. Die Anwendungsarchitektur regelt den Aufbau von Anwendungen, aber auch die Interaktion mit anderen Anwendungen. Die Geschäftsarchitektur beschreibt den Aufbau und das Zusammenspiel von Prozessen.

  Das Architektur-Board hat die Aufgabe, die verschiedenen Ebenen in ihrem Zusammenspiel zu regeln bzw. zu koordinieren.

- *Sicherheit – Security-Board:* Bei Anwendungen mit sensiblen Daten ist sicherzustellen, dass nur berechtigte Personen Zugriff haben. Welche Vorkehrungen hierfür zu treffen sind, wird im Security-Board festgelegt. Ebenso ist der Schutz der gesamten IT-Infrastruktur vor unberechtigten Zugriffen sicherzustellen.

- *Strategie – Strategie-Board:* Die künftige Ausrichtung der IT-Anwendungslandschaft wird in einem Strategie-Board festgelegt. Hierbei ist eine Abstimmung mit dem Gremium für die Unternehmensstrategie wichtig.

- *Standardisierung – Technologie-Board:* Die standardkonforme Realisierung von Anwendungen ist die Grundalge für eine effiziente Wartung und Weiterentwicklung der realisierten Funktionalität. Was zu einem unternehmensweit gültigen Standard erklärt wird, ist Aufgabe dieses Gremiums. Über dieses Technologie-Board werden ferner die verschiedenen Standards koordiniert.

Die genannten Gremien haben entscheidenden Einfluss auf die Entwicklung von Anwendungen. Im Kapitel 3 wird erläutert, dass mit zunehmendem Industrialisierungsgrad der Einfluss der Gremien in die Anwendungsentwicklung steigt.

Die Bezeichnung der Gremien sowie die zugehörige Definition der Aufgabenstellungen unterscheiden sich von Unternehmen zu Unternehmen. Für diese Arbeit sind die Aufgabenbereiche der geforderten Gremien wichtig. Wie diese in den Unternehmen bezeichnet werden, ist dabei nicht relevant, da dies je nach Bedarf geregelt wird. Die Anzahl der Gremien kann sich zwischen den verschiedenen Unternehmen ebenfalls unterscheiden.

2.3.7 Unterstützende Prozesse

Die Einführung von industriellen Prinzipien wie der Standardisierung oder Wiederverwendung erfordert die zentrale Organisation verschiedener Aufgaben. Die Festlegung von Standards sollte beispielsweise nicht willkürlich erfolgen, sondern in einem für alle Beteiligten transparenten Prozess ablaufen.

Diese Prozesse unterstützen die Aufgaben der genannten Gremien und sind in jeder Stufe des Industrialisierungsgrades festzulegen. Die Form sowie eine mögliche Unterstützung durch Werkzeuge sind in dieser Arbeit nicht relevant und können in Unternehmen unterschiedlich gestaltet werden. Wesentlich ist die Abdeckung der verschiedenen Aufgaben durch etablierte Prozesse. Die erarbeiteten Ergebnisse sind nachvollziehbar zu dokumentieren.

2.4 Faktoren der Anwendungsentwicklung

Die Anwendungsentwicklung wird beeinflusst von verschiedenen, konkurrierenden Faktoren, wie beispielsweise den möglichen Kosten oder der zur Verfügung stehenden Zeit [Star02]. Diese und weitere Faktoren definieren den Rahmen, in welchem eine Entwicklung stattfindet. Diese Faktoren haben dabei ein unterschiedliches Gewicht bzgl. der Auswirkung auf die Anwendungsentwicklung und können in Konkurrenz zueinander stehen. In Abschnitt 1.1 wurde bereits anhand des Gummiband-Diagrammes von [Star02] das Kräfteverhältnis der verschiedenen Faktoren auf die Entwicklung einer Anwendung vorgestellt.

Ist der Einsatz einer Software sehr fehleranfällig, wurde vermutlich den Faktoren Zeit oder Kosten ein höheres Gewicht zugewiesen als der Qualität der Anwen-

dung. Da die Zeit sowie die Kosten wichtige Planungsgrößen für die Beauftragung zur Erstellung einer Anwendungen für den Auftraggeber sind, ist diese Gewichtung zunächst nachvollziehbar. Meist wird dabei jedoch ignoriert, dass eine schlechte Qualität der Anwendung hohe Kosten im laufenden Betrieb bzw. in der Wartung verursachen kann.

Eine Optimierung des Kräfteverhältnisses der einzelnen Faktoren kann auf Basis der Prinzipien der Industrialisierung erreicht werden. Hierzu sind bei der Entwicklung von Anwendungen Methoden anzuwenden, die diesen Prinzipien entsprechen. Diese Methoden werden in Kapitel 3 in den einzelnen Stufen des Industrialisierungsgrades eingeführt. Durch den konsequenten Einsatz dieser Methoden wird die Qualität inhärenter Bestandteil der Anwendung und bleibt nicht länger ein optionaler und somit meist vernachlässigter Faktor. Die nach [Star02] benötigten Kompromisse bei der Erstellung der Anwendung werden hierdurch reduziert. Die Auswirkungen der jeweiligen Methode und damit der Industrialisierung auf die Faktoren werden in den einzelnen Stufen in Kapitel 3 analysiert.

Zunächst ist jedoch zu klären, welche Faktoren für die Anwendungsentwicklung relevant sind. Die drei wesentlichen konkurrierenden Faktoren sind Zeit, Kosten und Qualität. Nach [SiSM06] ist Qualität „die Gesamtheit von Eigenschaften und Merkmalen eines Produktes, die sich auf dessen Eignung zur Erfüllung gegebener Erfordernisse beziehen“. Qualität als Faktor ist in dieser Form jedoch zu unspezifisch für eine Analyse. Eine Möglichkeit zur Konkretisierung des Qualitätsbegriffs ist die Bildung von Qualitätsmodellen. Nach [Wall01] „bieten diese eine differenzierte und den Erfordernissen von Software-Produkten angepasste Möglichkeit zur Spezifikation von Qualitätsanforderungen“. Ein solches Modell wird in der Norm ISO[30] 9126[31] definiert. Darin werden sechs Qualitätsmerkmale benannt [SiSM06], [Reza04]:

- *Funktionalität*: Bei diesem Merkmal wird geprüft, inwieweit die Software die geforderten Funktionalitäten besitzt. Diese Funktionalitäten decken dabei verschiedene Aspekte ab: Eignung der Funktionen für die spezifizierten Aufgaben (Angemessenheit), Bereitstellung der korrekten Werte (Richtigkeit), Vermeidung unberechtigten Zugriffs (Sicherheit), Einhaltung anwendungsspezifischer Normen (Ordnungsmäßigkeit) sowie der Fähigkeit, mit vorgegebenen Systemen zu kommunizieren (Interoperabilität).
- *Zuverlässigkeit*: Hierbei wird die Fähigkeit analysiert, ob die Anwendung das spezifizierte Leistungsniveau unter festgelegten Bedingungen über einen definierten Zeitraum aufrechterhalten kann. Betrachtet werden da-

[30] International Organization for Standardization (ISO): internationale Vereinigung von Normungsorganisationen.

[31] Die Norm ISO 9126 definiert Qualitätsmerkmale für Softwareprodukte. Sie ist mittlerweile in die Norm ISO 25000 aufgegangen.

bei die Versagenshäufigkeit aufgrund von Fehlerzuständen (Reife), die Fähigkeiten, trotz Softwarefehler das Leistungsniveau zu halten (Fehlertoleranz) sowie bei Versagen das Leistungsniveau wiederherzustellen und die betroffenen Daten wiederzugewinnen (Wiederherstellbarkeit).

- *Benutzbarkeit:* Betrachtet wird bei diesem Merkmal der Aufwand, der zur Benutzung der Anwendung erforderlich ist. Dies unterteilt sich in den Aufwand, die Anwendung zu verstehen (Verständlichkeit), den Aufwand, die Anwendung zu erlernen (Erlernbarkeit) sowie den Aufwand, die Anwendung zu bedienen (Bedienbarkeit).
- *Effizienz*: Das Verhältnis der eingesetzten Betriebsmittel zum Leistungsniveau unter definierten Bedingungen wird bei diesem Merkmal analysiert. Dies betrifft die Antwort- und Verarbeitungszeiten sowie den Durchsatz bei Funktionsausführung (Zeitverhalten). Ferner werden Anzahl und Dauer der benötigten Betriebsmittel betrachtet, die für die Ausführung der Funktionen benötigt werden (Verbrauchsverhalten).
- *Änderbarkeit*: Die Eignung der Software, vorgegebene Änderungen ohne unerwünschte Nebeneffekte durchzuführen, sowie der hierfür nötige Aufwand stehen im Fokus dieses Merkmals. Betrachtet wird dabei der Aufwand bzw. die Eignung der Software für die Diagnose von Fehlern oder der Analyse von Änderungen (Analysierbarkeit) sowie der Aufwand bzw. die Eignung zur Ausführung von Änderungen (Modifizierbarkeit). Weitere Aspekte sind die Wahrscheinlichkeit für das Auftreten von Fehlern durch unerwartete Zustandsänderungen oder Eingaben des Anwenders (Stabilität) sowie der Eignung durchgeführte Änderungen zu validieren (Prüfbarkeit).
- *Übertragbarkeit*: Die Fähigkeit der Software, von einer Umgebung in eine andere übertragen zu werden, ist der Kern dieses Merkmals. Dabei kann die Umgebung sich sowohl auf die Hardware- als auch Softwareumgebung beziehen. Eingeschlossen wird hierbei auch die Änderung organisatorischer Umgebungen, die z.B. bei Unternehmensfusionen auftreten können. In diesem Kontext werden die Fähigkeiten analysiert, die Software an verschiedene Umgebungen anzupassen (Anpassbarkeit) oder in einer bestimmten Umgebung zu installieren (Installierbarkeit). Ferner wird die Möglichkeit betrachtet, die Software anstelle einer anderen Software für den gleichen Zweck einzusetzen (Austauschbarkeit). Bei der Übertragung wird zudem analysiert, ob vorhandene Normen oder Vereinbarungen eingehalten werden (Konformität).

Diese Merkmale haben bzgl. der Qualität einer Anwendung eine hohe Relevanz. Sie werden daher in dieser Arbeit als Faktoren der Anwendungsentwicklung verwendet. Neben diesen Qualitätsfaktoren werden in jeder Stufe auch die Aus-

wirkungen der Methoden auf die Faktoren Zeit und Kosten betrachtet, die in Konkurrenz zur Qualität bei der Entwicklung von Anwendungen stehen:

- *Zeit:* Insbesondere für die Auftraggeber sollten Anwendungen nach einer möglichst kurzen Zeitspanne zur Verfügung stehen. Oft werden für diesen Anspruch andere Faktoren vernachlässigt, wodurch die Qualität der Anwendung gemindert wird oder die Kosten steigen.[32] Bei diesem Faktor ist zu klären, wie die Methoden der einzelnen Stufen des Industrialisierungsgrades die Zeit für die Realisierung reduziert.
- *Kosten:* Die Kosten für die Realisierung einer Anwendung sind ein weiterer wesentlicher Faktor, an dem häufig Veränderungen stattfinden. Sie enthalten neben der reinen Entwicklung häufig auch Lizenzen für eingesetzte Softwareprodukte sowie den Aufwand für die Bereitstellung der benötigten Infrastruktur.

  Die Einführung einer neuen Methode ist meist mit einer Erhöhung der Kosten verbunden. Beispielsweise erfordert die Automatisierung eine Investition in Modellierungssprachen, Editoren und Generatoren. Ferner müssen die Entwickler in diese neuen Verfahren eingewiesen werden. Bei der Kostenbetrachtung in den einzelnen Stufen des Industrialisierungsgrades werden jedoch nur die projektspezifischen Kosten analysiert, die für die Umsetzung einer Anwendung benötigt werden. Es wird davon ausgegangen, dass die Methode bereits eingeführt wurde.

2.5 Modell

Verschiedene Beschreibungen in dieser Arbeit erfolgen auf der Basis von Modellen. Modelle bilden die Realität auf einer abstrahierten Ebene ab. Dabei werden nur diejenigen Attribute der Realität in das Modell integriert, die für die Betrachtung relevant erscheinen [Kell07]. Modelle werden u.a. für die Erläuterung des Industrialisierungsgrades (Stufenmodell – vgl. Abschnitt 3.1), die Strukturierung eines Unternehmens (Domänenmodell – vgl. Abschnitt 2.6) oder die Vorgabe für die Generierung von Quelltexten (Meta-Modell – vgl. Abschnitt 3.7.2) verwendet. Die genannten Modelle unterscheiden sich jedoch in ihren Strukturen und Aussagen. Daher werden in diesem Abschnitt die verschiedenen Arten von Modellen genauer spezifiziert.

Allgemein wird durch ein Modell die Realität bzw. ein Realitätsausschnitt abgebildet. Diese Abstraktion muss nach [LeWS08] eine oder mehrere der folgenden Eigenschaften besitzen:

[32] Durch Bereitstellung weiterer Ressourcen z.B. in Form von zusätzlichen Entwicklern.

- strukturelle Ähnlichkeit
- funktionelle Ähnlichkeit
- Verhaltensähnlichkeit

Zwischen den einzelnen Wissenschaftsdisziplinen lassen sich Unterschiede in der Verwendung des Modellbegriffes sowie im Begriffsverständnis feststellen. In der Wirtschaftsinformatik werden nach [LeWS08] die Modelle nach den modellimmanenten Zielen klassifiziert und beschrieben:

- *Beschreibungs- und Erfassungsmodelle:* Möglichst präzise und leicht verständliche Darstellung eines Sachverhalts oder eines Phänomens. Beispiel: Landkarte.

  Beschreibungsmodelle werden in dieser Arbeit bei dem Domänenmodell sowie bei der Modellierung verwendet. Das Domänenmodell aus Abschnitt 2.6 beschreibt die Struktur eines Unternehmens nach fachlichen Aspekten. Die Modelle für die Modellierung von Anforderungen oder IT-Lösungen (vgl. Abschnitt 3.7.1) beschreiben die jeweiligen Strukturen in einer für Maschinen auswertbaren Form.

- *Erklärungsmodelle:* Mit dieser Art von Modellen werden die Ursachen für bestimmte Abläufe, z.B. betriebliche Prozessabläufe, erklärt. Dieses Modell basiert auf dem Beschreibungsmodell, das durch Beobachtungen im Zeitverlauf auf erkennbare Gesetzmäßigkeiten untersucht wird. Diese Modelle werden daher u.a. für die Identifikation von Zustandskonstellationen oder von Ereignisfolgen verwendet. Beispiel: Prognosemodelle.

  Diese Klasse von Modellen wird in dieser Arbeit nicht verwendet.

- *Gestaltungsmodelle.* Zur Erfüllung von Anforderungen oder Aufgaben enthalten diese Modelle alle erforderlichen Einflussgrößen. Sie fassen z.B. speziell für Entscheidungsträger das Verständnis bestimmter Phänomene oder Problembereiche zusammen oder beinhalten Ansatzpunkte für die Manipulation an einem System. Normative Modelle sind eine spezielle Form dieser Art von Modellen. Sie erweitern den Ausschnitt der betrachteten Phänomene um eine Sinn-, Zweck- oder Zielkomponente. Die festgelegte Messgröße wird hierbei in Relation zum Bedarf gesetzt. Dies ermöglicht die Festlegung einer Norm sowie den Weg dorthin. Beispiel: Phaseneinteilung bei Vorgehensmodellen.

  Das Stufenmodell aus Abschnitt 3.1 zur Bestimmung des Industrialisierungsgrades ist ein normatives Modell. Es enthält die verschiedenen Einflussgrößen, die für die Bestimmung der Stufe benötigt werden. Zudem werden Ansatzpunkte für die Verbesserung der Anwendungsentwicklung beschrieben. Es stellt ferner eine mögliche Norm für die Anwendungsentwicklung in einem Unternehmen bereit, die abhängig vom Bedarf er-

reicht werden soll. Im Zusammenspiel mit der Bewertungsmethode aus Abschnitt 4 werden zudem Pfade für die Erreichung der angestrebten Norm aufgezeigt.

- *Meta-Modelle und generische Modelle:* Diese Form der Modelle hat eine übergeordnete Funktion. Sie dient als systematische Beschreibung von Modellen. Ein Meta-Modell ist damit quasi ein Modell eines Modells. Diese Modelle werden insbesondere bei Generatoren zur Erzeugung von Modellen eingesetzt.

  In der Stufe 4: Automatisierung (siehe Abschnitt 3.7) wird diese Form der Modelle u.a. benötigt, um aus plattformunabhängigen Modellen plattformspezifische Modelle zu erzeugen.

- Sonstige Modelle

  Alle anderen Modelle, die sich nicht den bisher genannten Klassen zuordnen lassen. Beispiel: Optimierungsmodelle der Verifikationsmodelle.

2.6 Domänen

Die Grenzen der Industrialisierung wurden bereits in Abschnitt 2.2.1 dargestellt. Nicht jeder Bereich eines Unternehmens eignet sich gleichermaßen für die Einführung von industriellen Prinzipien. Während die Prozesse der Buchhaltung weitestgehend standardisiert werden können, sind die Prozesse im Bereich Forschung und Entwicklung eines Unternehmens sehr individuell gestaltet.

Für die Analyse der Industrialisierung der Anwendungsentwicklung ist daher eine bereichsspezifische Betrachtung aussagekräftiger und bietet mehr Potentiale für Optimierungen. Eine mögliche Strukturierung könnte anhand der Aufbauorganisation erfolgen. Dies hat jedoch den Nachteil, dass Geschäftsfunktionalitäten wie z.B. Buchhaltung mehrfach zu betrachten sind, wenn dieser Service nicht zentral, sondern dezentral in den verschiedenen Geschäftsbereichen organisiert ist.

Für die Strukturierung der verschiedenen Geschäftsfunktionen eines Unternehmens hat sich die Einführung eines Domänenmodells etabliert. In [EHHJ08] werden Domänen folgendermaßen definiert:

> Domänen gruppieren die Komponenten einer Anwendungslandschaft. Die Gruppierung erfolgt nach fachlichen Gesichtspunkten. Domänen können hierarchisch geschachtelt sein. Komponenten der Anwendungslandschaft werden jeweils den am tiefsten geschachtelten Domänen zugeordnet.

Definition 1: Domäne

Die geschachtelten Domänen werden auch als Subdomäne bezeichnet. In Abbildung 8 wird eine mögliche Aufteilung eines Unternehmens in die verschiedenen Domänen dargestellt. Dieses Beispiel ist [EHHJ08] entnommen und stellt ein fiktives Reiseunternehmen dar, das Individual- und Pauschalreisen verkauft.

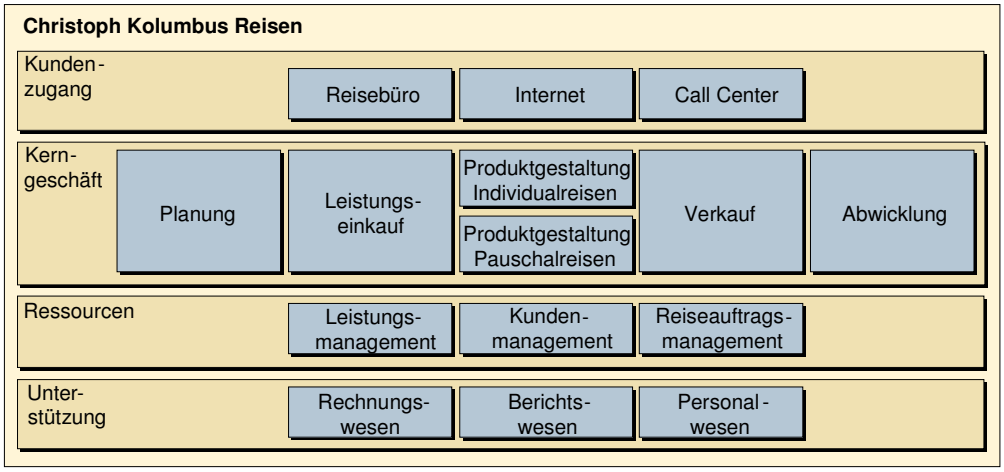

Abbildung 8: Mögliche Aufteilung eines Unternehmens in Domänen[33]

Das Domänemodell ist ein Beschreibungsmodell (vgl. Abschnitt 2.5), das ein Unternehmen nach fachlichen Aspekten strukturiert. Domänen sind ein wichtiges Werkzeug für die Planung und Durchführung der Evolution von Anwendungslandschaften. Wird beispielsweise ein Teil eines Unternehmens verkauft, geben Domänen Auskunft über mögliche Auslagerungen von Systemen. Bei der Fusion von Unternehmen können anhand der Zuordnung von Anwendungen zu Domänen Redundanzen schnell erkannt und optimiert werden.

Eigenschaften von Domänen: (vgl. [EHHJ08], [GrSh06])

- Vollständige und redundanzfreie Strukturierung der Fachlichkeit auf logischer Ebene.
- Ermöglichen die gezielte Zuweisung von Anforderungen, Daten und IT-Systemen zur Unterstützung übergreifender Steuerungsprozesse des Managements.
- Kapseln fachlich eng gekoppelte Funktionen und Geschäftsobjekte und stellen Leistungen in Form von Services bereit.

Die Funktionalitäten und Daten, die über Anwendungen einer Domäne zur Verfügung stehen, sollten in keiner anderen Domäne bereitgestellt werden.

Die Art der Aufteilung in Domänen ist abhängig von der Branche aber auch von jedem Unternehmen. Es gibt daher keine generelle Vorgabe, welche Anwendun-

[33] Siehe [EHHJ08].

gen zu einer Domäne gehören. Domänen sind immer am Geschäft des Unternehmens orientiert und daher spezifisch zu erstellen. Wesentlich bei der Gestaltung der Domänen ist die enge Abstimmung mit den Geschäftsbereichen.

Es gibt verschiedene Wege, die Domänen eines Unternehmens zu identifizieren. Eine Methode zum Entwurf von Domänen ist in [EHHJ08] beschrieben, in welcher Hinweise auf Domänen systematisch untersucht werden. Hierzu gehören die Kerngeschäftsservices, Geschäftsobjekte sowie die unterstützenden Geschäftsservices. Auch die Prozesslandkarte liefert einen wichtigen Input. Dieses Verfahren ist jedoch weniger ein Rezept, das in eine gute Domänenaufteilung mündet. Es gibt vielmehr Hinweise für die Identifizierung von Domänen. Für einen guten Domänenschnitt spielt die Erfahrung von IT-Architekten eine wesentliche Rolle, die einen Überblick über die Anwendungslandschaften sowie die Geschäftsprozesse besitzen.

Für die Bestimmung des Industrialisierungsgrades werden die Domänen als strukturierender Rahmen verwendet. Für jede Domäne wird der entsprechende Grad ermittelt (siehe Kapitel 4 und 5).

2.7 Klassifikation von Anwendungen

Unternehmen setzen unterschiedliche Arten von Anwendungen für die Unterstützung von Prozessen ein. Bisher wurde in dieser Arbeit der Begriff „Anwendung“ nicht näher spezifiziert. Eine eindeutige Definition zu finden, ist sehr schwierig. Jede Definition schränkt Anwendungen auf bestimmte Sichtweisen ein. Verwaltet ein Administrator mehrere Systeme, wird er beispielsweise ein Werkzeug für die Überwachung dieser Systeme als Anwendung bezeichnen. Für einen Verkäufer spielt diese Art von Anwendung jedoch keine Rolle. Eine Anwendung aus seiner Sicht ist beispielsweise die Verwaltung von Kunden oder die Erfassung von Aufträgen. Eine Möglichkeit, diesen Konflikt aufzulösen, ist die Aufteilung von Anwendungen auf verschiedene Kategorien, in der gleichartige Anwendungen gebündelt werden.

Auch die Relevanz von Anwendungen für das Unternehmen ist sehr unterschiedlich. Systeme wie Microsoft Access[34] erlauben beispielsweise die Erstellung von Anwendungen auch durch Personen mit sehr wenig Erfahrung in der Softwareentwicklung. Diese Art von Anwendungen wird meist für lokale Aktivitäten wie die Verwaltung von Aufgaben, Büchern etc. erstellt. Sie haben meist wenig Relevanz aus Unternehmenssicht.

Eine Optimierung der Anwendungsentwicklung sollte insbesondere für die relevanten Anwendungen im Unternehmen erfolgen. Welche Anwendungen hierzu

[34] Datenbankmanagementsystem der Firma Microsoft zur Verwaltung von Daten und zur Entwicklung von Datenbankanwendungen.

näher zu betrachten sind, kann anhand einer Klassifizierung festgelegt werden. Wie eine solche Klassifizierung aufgebaut werden kann, wird in den folgenden Abschnitten beschrieben.

2.7.1 Kategorisierung von Anwendungen

Für das Management der IT-Anwendungslandschaft sind alle Einheiten[35] zu betrachten, die für die Unterstützung von Geschäftsprozessen relevant sind und verwaltet werden müssen. Diese Einheiten erzeugen Kosten, die entsprechend dem jeweiligen Nutzen zu optimieren sind.

Eine Unterteilung von Anwendungen anhand der spezifischen Eigenschaften in unterschiedliche Kategorien ermöglicht eine einsatzorientierte Analyse der vorhandenen Anwendungen. Für die einzelnen Kategorien können jeweils unterschiedliche Methoden zur Entwicklung der Anwendungen zum Einsatz kommen. Dies ist durch das Unternehmen festzulegen.

Bei der Erfassung werden Anwendungen einer der folgenden Kategorien zugeordnet:

- *Business-Anwendungen:* Anwendungen, die basierend auf einer Logik die Daten des Unternehmens durch den Anwender verändern und damit direkt einen Geschäftsprozess unterstützen. Sie stellen eine Interaktionsschicht zur Verfügung, um die verschiedenen Funktionen ausführen zu können. Beispiel: Auftragsverwaltung.
- *Kleine Business-Anwendungen*: Anwendungen mit der gleichen Intention wie bei Business-Anwendungen, jedoch mit einem geringen Umfang. Beispiel: Verwaltung von Unternehmensbroschüren.
- *Reporting:* Anwendungen, die basierend auf einer konkreten Zusammenstellung von Daten spezifische Analysen erlauben. Der Zugriff auf diese Daten ist rein lesend. Beispiel: Monatliche Finanzberichte.
- *Technische Anwendungen:* Diese Art Anwendung hat keine direkte Sichtbarkeit für den Anwender. Sie stellen jedoch notwendige Dienste für die Business-Anwendungen zur Verfügung. Beispiel: Zentrale Benutzerverwaltung.
- *Operative Anwendungen*: Der Betrieb einer IT-Infrastruktur ist wesentliche Voraussetzung für die Verfügbarkeit von Business-Anwendungen. Die Elemente dieser Kategorie unterstützen jedoch nicht direkt die Geschäftsprozesse. Vielmehr stellen sie sicher, dass die Systeme der IT-Infrastruktur zur Verfügung stehen. Beispiel: Monitoring-Systeme.

[35] Software, die in unterschiedlichen Formen (Anwendungen, Komponenten, Bausteine etc.) Prozesse unterstützen.

- *Schnittstellen*: Der Austausch insbesondere von größeren Datenmengen zwischen verschiedenen Systemen erfolgt meist autark im Hintergrund auf Basis von Schnittstellen-Anwendungen. Beispiel: Einbindung eines externen Marktplatzes.

2.7.2 Anwendungsklassen

Bei der Bestimmung des Industrialisierungsgrades sollten nur die Anwendungen betrachtet werden, die eine gewisse Relevanz für das Unternehmen darstellen. Für die Bestimmung dieser Relevanz gibt es in der Literatur keine eindeutigen Richtlinien. In dieser Arbeit werden daher verschiedene Kriterien von Anwendungen analysiert, die einen Rückschluss auf die Relevanz für das Unternehmen erlauben. Die Relevanz wird dabei durch folgende Klassifikation ausgedrückt.

- *Business kritisch:* Anwendungen dieser Klasse haben die oberste Priorität. Ein Ausfall hat schwerwiegende Konsequenzen für das Unternehmen.

  Beispiele: Logistik, Anlagensteuerung, Waagensystem

- *Business relevant:* Anwendungen sind sehr wichtig, haben jedoch bei einem kurzfristigen Ausfall nur geringe Auswirkung auf die betrieblichen Prozesse

  Beispiele: Intranet, Internet, Telefonbuch

- *Business unterstützend:* Der Einsatz dieser Anwendungen hat konkreten Nutzen für das Business, sind jedoch hinsichtlich der Verfügbarkeit sowie den Daten nachrangig zu bewerten.

  Beispiele: Meinungsumfragen, Urlaubsantrag

- *Gering Business relevant:* Diese Klasse enthält Anwendungen mit einem eingeschränkten Einsatzbereich. Sie sind oft temporärer Natur. Eine Zuordnung zu Geschäftsprozessen ist nur indirekt vorhanden.

  Beispiel: Persönliche Aufgabenliste.

Die Zuordnung einer Anwendung zu einer Klasse erfolgt durch die gewichtete Bewertung von Kriterien[36]. Der Vorgang der Bewertung wird in Abschnitt 4.3.2 erläutert.

2.7.3 Kriterien für eine Klassifikation

Für die Einstufung von Anwendungen in eine der genannten Klassen werden Kriterien benötigt, die einen Rückschluss auf die Relevanz für das Unternehmen

[36] Siehe Abschnitt 2.7.3.

zulassen. Folgende Kriterien werden für die Zuordnung bei jeder Anwendung analysiert:

- *Verfügbarkeit:* Software, aber auch die genutzte Hardware, ist im laufenden Betrieb anfällig für Fehler unterschiedlicher Natur. Ein potentieller Ausfall ist somit für die Bereitstellung der Anwendung zu berücksichtigen. Je nach Anforderung sind entsprechende Maßnahmen notwendig, den Ausfall auf ein Mindestmaß zu reduzieren. Dieses Mindestmaß ist zu definieren.
- *Wiederherstellung im Katastrophenfall (Disaster Recovery):* In einem Katastrophenfall bedingen äußere Einflüsse den Ausfall einer oder mehrerer Anwendungen. Beispielsweise werden bei einem Brand z.B. die Netzwerkverbindungen oder sogar die Hardware zerstört. Da dieser Fall eher selten eintritt, ist für jede Anwendung zu prüfen, welche Maßnahmen zu ergreifen sind.
- *Änderbarkeit:* Im Lebenszyklus einer Anwendung werden nach der ersten Produktivsetzung meist weitere Funktionalitäten gefordert oder es werden Fehler erkannt, die zu beheben sind. Je nach Umfang der Anwendung sind entsprechende Voraussetzungen zu schaffen, die eine Änderung der Software ermöglichen.
- *Support:* Anwender mit Fragen zu einer Anwendung können sich an den meist zentral organisierten Support wenden. Die Entwickler dieser Anwendungen sind hierdurch von wiederkehrenden Fragestellungen befreit. Sie können sich somit auf die Wartung oder Weiterentwicklung der Anwendung fokussieren. Die Einbindung des Supports ist jedoch erst ab einer bestimmten Anwenderzahl sinnvoll.
- *Integration:* Für die Unterstützung von Geschäftsprozessen werden Anwendungen zunehmend mit anderen Anwendungen vernetzt. Der Grad der Vernetzung stellt wiederum Anforderungen z.B. an die Infrastruktur oder die Wartung. Mit zunehmender Integration der Anwendung in die IT-Landschaft steigt der Aufwand zur Pflege und Koordination der Anwendung.
- *Zugriff auf Anwendungen:* Aufgrund der steigenden, unternehmensübergreifenden Prozessunterstützung werden Anwendungen auch außerhalb des Unternehmensnetzes bereitgestellt. Je nach Art des Zugriffs (lesend, schreibend) sowie des Ortes (innerhalb, außerhalb des Unternehmens) sind entsprechend unterschiedliche Sicherheitsvorkehrungen zu treffen.
- *Datenkritikalität:* Die für eine Unternehmung wertvollen Daten unterliegen höheren Sicherheitsanforderungen als öffentlich zugängliche Daten. Ent-

sprechende Sicherheitsmechanismen sind für den Zugriff auf diese Daten bzw. deren Verarbeitung zu treffen.

Bei der Analyse einer Anwendung ist für jede dieser Kriterien eine Bewertung von 1 bis 4 durchzuführen. Der Wert 1 hat dabei die höchste Relevanz für das Kriterium und erfordert den höchsten Aufwand bei der Umsetzung. Wird beispielsweise die höchstmögliche Verfügbarkeit einer Anwendung benötigt, sind sowohl in der Anwendung als auch in den zugrunde liegenden Systemen entsprechende Vorkehrungen zu treffen. Tabelle 2 zeigt in einer Übersicht die möglichen Werte der verschiedenen Kriterien.

	Verfügbarkeit	Disaster Recovery	Änderbarkeit	Support	Integration	Zugriff	Datenkritikalität
1	ab 99,9%	1-24 Stunden	Programmierung > 20 Tage	> 200 Benutzer	Stellt Schnittstellen generell zur Verfügung	Extern schreibend	Streng geheim
2	99,5%	1-7 Tage	Programmierung < 20 Tage	51-200 Benutzer	Stellt Schnittstellen explizit zur Verfügung	Extern lesend	geheim
3	99%	2-8 Wochen	Konfiguration	6-50 Benutzer	Nutzt Schnittstellen	Innen global	vertraulich
4	bis 99%	2-4 Monate	Installation	1-5 Benutzer	Keine Integration in die Infrastruktur	Innen lokal	öffentlich

Tabelle 2: Kriterien für die Klassifikation von Anwendungen

Die Kriterien für die Klassifikation stellen einen Rahmen für die Einordnung von Anwendungen dar. Die Zuordnung einer Anwendung zu einer Klasse erfolgt aus der Aggregation der bewerteten Kriterien (siehe Abschnitt 4.3.2). Die Gestaltung der Werte der jeweiligen Kriterien, wie sie in Tabelle 2 dargestellt sind, ist individuell für das betrachtete Unternehmen festzulegen.

Eine Anwendung, deren Verfügbarkeit bei 99,9% liegt, muss eine hohe Relevanz für das Unternehmen haben. Ansonsten lassen sich die enormen Kosten für die erforderlichen Maßnahmen zur Erfüllung der hohen Verfügbarkeit nicht darstellen. Diese Annahme liegt allen Kriterien zugrunde. Somit steigt die Relevanz einer Anwendung für das Unternehmen, je höher die Bewertung der einzelnen Kriterien ist. Dies ist die Grundlage für die Auswahl der Anwendungen in Kapitel 4 und 5, die für eine Bewertung des Industrialisierungsgrades herangezogen werden.

In den folgenden Abschnitten werden die Bedeutung der einzelnen Kriterien sowie deren Bewertung näher erläutert.

2.7.3.1 Verfügbarkeit

Während das System zur Steuerung der Produktionsanlage permanent zur Verfügung stehen muss, kann die Darstellung des Speiseplans im Intranet einen Ausfall ohne wesentliche Konsequenzen für das Geschäft verkraften. Die Notwendigkeit einer hohen Verfügbarkeit wirkt sich direkt auf die Bereitstellung entsprechender Mechanismen aus, die den Ausfall eines Systems, z.B. aufgrund eines Hardwarefehlers, auffangen. Der damit verbundene Aufwand sowie die hierdurch anfallenden Kosten stehen in Zusammenhang mit der Wichtigkeit der Anwendung für den Geschäftsprozess.

Wert	Bezeichnung	Beschreibung
1	99,9%	Verfügbarkeit der Anwendung liegt bei 99,9%. Ein Ausfall von insgesamt 8,76 Stunden im Jahr ist vertretbar.
2	99,5%	Verfügbarkeit der Anwendung liegt bei 99,5%. Ein Ausfall von insgesamt 43,8 Stunden im Jahr ist vertretbar.
3	99%	Verfügbarkeit der Anwendung liegt bei 99%. Ein Ausfall von insgesamt 87,6 Stunden im Jahr ist vertretbar.
4	bis 99%	Für die Verfügbarkeit der Anwendung sind keine spezifischen Vorkehrungen zu treffen. Ein Ausfall von mehr als 87,6 Stunden im Jahr wird akzeptiert.

Tabelle 3: Werte für das Kriterium Verfügbarkeit

Die Messung der Verfügbarkeit eines Systems richtet sich nach den für diese Anwendung zugesicherten Zeiten. Die Angabe der Verfügbarkeit erfolgt in Prozent. 99,9% Verfügbarkeit bedeutet dabei, dass ein System, welches rund um die Uhr verfügbar sein soll, im Jahr maximal 8,76 Stunden ausfallen darf (365 Tage × 24 Stunden × 0,1%). Hierbei sind die Servicezeiten, z.B. zum Einspielen einer neuen Version der Anwendung, nicht enthalten. Diese werden separat vereinbart. Um diese Verfügbarkeit sicherzustellen, sind entsprechende Vorkehrungen zu treffen. In Tabelle 3 werden die einzelnen Einstufungen aufgelistet.

Eine potentielle Maßnahme für die Sicherstellung der Verfügbarkeit ist die Installation eines redundanten Systems, das im Falle eines Ausfalls genutzt werden kann.

2.7.3.2 Wiederherstellung im Katastrophenfall

Der Katastrophenfall ist eine außergewöhnliche Situation, die selten eintritt, jedoch enorme Auswirkung auf das Unternehmen hat. Als Beispiel sei hier der Terroranschlag des 11. September 2001 genannt. In einem solchen Szenario geht es nicht um den Ausfall eines einzelnen Rechners aufgrund von Hardware-

problemen, wie es bei der Verfügbarkeit zu betrachten ist. Vielmehr ist zu berücksichtigen, dass die gesamte Infrastruktur nicht mehr verfügbar ist. Auch der Verlust von Mitarbeitern vor Ort ist zu beachten und damit der Ausfall von Personal, das die entstandenen Probleme lösen kann.

In diesem Szenario ist es wichtig, die für das Unternehmen überlebenswichtigen Anwendungen so schnell wie möglich wiederherzustellen. Der Aufwand für diese Sicherstellung kann sehr hoch sein, z.B. durch den Aufbau eines redundanten Rechenzentrums, das lediglich im Katastrophenfall genutzt wird.

Mögliche Werte für dieses Kriterium werden in der folgenden Tabelle dargestellt.

Wert	Bezeichnung	Beschreibung
1	1-24 Stunden	Die überlebensnotwendigen Anwendungen sind innerhalb von 24 Stunden wieder verfügbar.
2	1-7 Tage	Um die wesentlichen Geschäftsprozesse wieder gut unterstützen zu können, sind die Anwendungen dieser Kategorie innerhalb von 7 Tagen wiederherzustellen.
3	2-8 Wochen	Nachdem die wesentlichen Geschäftsprozesse wieder verfügbar sind, werden die unterstützenden Anwendungen innerhalb von 8 Wochen wieder zur Verfügung gestellt.
4	2-4 Monate	Die restlichen Anwendungen werden im Verlaufe von bis zu 4 Monaten wiederhergestellt

Tabelle 4: Werte für das Kriterium Katastrophenfall

Eine weitere Unterscheidung könnte ein maximaler Zeitraum sein, in dem der Verlust von Daten akzeptiert wird.

2.7.3.3 Änderbarkeit

Anwendungen unterliegen einem Lebenszyklus, der mit der initialen Entwicklung beginnt und mit der Abschaltung endet. Zwischen diesen Eckpunkten werden in der Regel an diesen Anwendungen mehrfach Änderungen vorgenommen.

Für jede Anwendung ist zu klären, in welcher Form Änderungen durchzuführen sind und welche Voraussetzungen, z.B. bzgl. der Infrastruktur oder der Kenntnisse der Entwickler, gegeben sein müssen.

Wert	Bezeichnung	Beschreibung
1	Programmierung > 20 Tage	Die Programmierung hat einen großen Umfang und stellt besondere Anforderungen an die Infrastruktur, die Kenntnisse der Entwickler, aber auch der Administratoren der Systeme. Eine gute Dokumentation ist wesentlich für die Wartung der Anwendung.
2	Programmierung < 20 Tage	Für Änderungen an der Anwendung ist eine Programmierung nötig. Die entsprechende Entwicklungsinfrastruktur sowie die benötigten Kenntnisse in den eingesetzten Technologien und Systemen sind vorzuhalten.
3	Konfiguration	Die Anwendung wird an das Unternehmen angepasst. Jedoch beschränkt sich die Änderung auf Konfiguration, die keine Programmierung benötigt bzw. keine Änderung der Funktionalität zum Ziel hat.
4	Installation	Die Anwendung wird lediglich installiert. Änderungen bestehen in der Konfiguration von Parametern.

Tabelle 5: Werte für das Kriterium Änderbarkeit

Die Relevanz der Anwendung ergibt sich bei diesem Kriterium aus dem Umfang der Programmierung. Je mehr Aufwand bei der Erstellung der Anwendung investiert wird, umso höher ist der Aufwand für die Wartung. Die hierdurch entstehenden Kosten müssen in einem sinnvollen Verhältnis zum Nutzen bzw. der Relevanz der Anwendung stehen.

2.7.3.4 Support

Im laufenden Betrieb einer Anwendung entstehen immer wieder Anfragen zur Bedienung der Anwendung. Aber auch das Melden von Fehlern oder Änderungswünsche der Funktionalität sind für jede Anwendung zu organisieren. Bei einer großen Anzahl von Nutzern einer Anwendung ist es sinnvoll, diese Anfragen durch eine Support-Organisation zu koordinieren. Insbesondere oft wiederkehrende Anfragen können so effizient bearbeitet werden.

Die Entwickler sowie der Anwendungsverantwortliche werden hierdurch von diesen Aufgaben befreit und können sich verstärkt den Aufgaben im Lebenszyklus der Anwendung widmen.

Mit einer zunehmenden Anzahl an Anwendern steigt die Relevanz dieser Anwendung im Unternehmen

Wert	Bezeichnung	Beschreibung
1	>200 Benutzer	Der Support ist in den Lebenszyklus der Anwendung eingebunden. Er ist in der Funktionalität geschult und kann die meisten Anfragen selbstständig bearbeiten.
2	51-200 Benutzer	Der Support ist über die wesentlichen Funktionen informiert und kann einzelne Anfragen bearbeiten.
3	6–50 Benutzer	Der Support ist über die Anwendung informiert und kennt die hierfür zuständigen Ansprechpartner.
4	1-5 Benutzer	Kleiner Benutzerkreis, der direkt vom Entwickler betreut wird. Der Support hat nicht zwingend Kenntnis von der Anwendung.

Tabelle 6: Werte für das Kriterium Support

2.7.3.5 Integration

Die Integration von Anwendungen in die bestehende IT-Landschaft kann unterschiedliche Ausprägungen haben. Nutzt die Anwendung selbst bestimmte Funktionalitäten von anderen Anwendungen oder Systemen, sind entsprechende Abstimmungen bzgl. der Nutzung notwendig. Änderungen an der Anwendung selbst haben jedoch keine Auswirkung auf die eingebundenen Systeme.

Bei der Bereitstellung von Schnittstellen, die von verschiedenen Anwendungen genutzt werden, sind entsprechende Vorkehrungen zu treffen, um Auswirkungen bei Änderungen oder Ausfällen auf die angebundenen Anwendungen zu vermeiden.

Wert	Bezeichnung	Beschreibung
1	Stellt allgemein Schnittstellen zur Verfügung	Die Anwendung stellt Schnittstellen zur Verfügung, die ohne weitere Absprache mit dem Anwendungsverantwortlichen genutzt werden können.
2	Stellt Schnittstellen zur Verfügung	Die Anwendung stellt für andere Anwendungen Schnittstellen zur Verfügung. Die Nutzung der Schnittstelle ist jedoch explizit zu vereinbaren.
3	Nutzt Schnittstellen	Die Anwendung nutzt Schnittstellen zu anderen Anwendungen oder Systemen.
4	Nutzt die Infrastruktur	Die Anwendung nutzt bis auf die Netzwerkinfrastruktur keine anderen Systeme.

Tabelle 7: Werte für das Kriterium Integration

Je nach Grad der Integration in die IT-Landschaft sind somit Anforderungen mit dem entsprechenden Aufwand zu erfüllen. Je höher diese Integration gefordert wird, umso relevanter ist diese Anwendung für das Unternehmen.

2.7.3.6 Zugriff auf die Anwendung

Die zunehmende Automatisierung von Prozessen erfordert beispielsweise die Eingabe von Daten direkt durch den Kunden. Anwendungen sind in diesem Fall auch außerhalb der Unternehmensgrenzen nutzbar und somit potentiellen Angreifern als Zugriffspunkte ausgesetzt.

Wert	Bezeichnung	Beschreibung
1	extern schreibend	Die Anwendung ist außerhalb des Unternehmens zugreifbar. Der Anwender kann Daten schreiben (z.B. Extranet: Austausch von Dokumenten zwischen Unternehmen).
2	extern lesend	Die Anwendung ist außerhalb des Unternehmens zugreifbar. Der Anwender kann jedoch nur lesend zugreifen (z.B. Internetauftritt).
3	intern global	Die Anwendung wird global genutzt, jedoch nur innerhalb der Unternehmensgrenzen (lesend und schreibend).
4	intern lokal	Die Anwendung hat einen stark eingegrenzten Nutzerkreis und wird nur lokal genutzt (lesend und schreibend).

Tabelle 8: Werte für das Kriterium Zugriff auf die Anwendung

Grundsätzlich gelten für Anwendungen, für die ein schreibender Zugriff auch von außerhalb des Firmennetzes erfolgen kann, erhöhte Sicherheitsvorkehrungen. Hierdurch soll die Gefahr potentieller Angriffe auf die Systeme des Unternehmens auf ein Minimum reduziert werden. Ist die Anwendung nur intern, aber global zugreifbar, sind die Anforderungen weniger hoch. Dennoch sind auch in diesem Fall, je nach Kritikalität der Daten, entsprechende Sicherheitsvorkehrungen zu treffen. Die Art des Zugriffes fordert daher verschiedene Maßnahmen und Voraussetzungen.

Aufgrund der zunehmenden Maßnahmen, je nach Art des Zugriffs auf die Anwendung, steigt auch die Relevanz der Anwendung für das Unternehmen. Dies rechtfertigt den zeitlichen und finanziellen Mehraufwand, der hierfür benötigt wird.

2.7.3.7 Datenkritikalität

Der veröffentlichte Geschäftsbericht, die noch geheime anstehende Akquisition eines Unternehmens, Patente, Rezepturen oder der Kantinenspeiseplan haben

jeweils einen sehr unterschiedlichen Wert für ein Unternehmen. Während einzelne Informationen für eine breite Öffentlichkeit vorgesehen sind, wird der Zugriff auf geheime Daten streng kontrolliert. Um den Wert von Informationen zu schützen, sind diese zunächst zu bewerten. Je nach Höhe der Kritikalität sind geeignete Voraussetzungen zu schaffen, die die vorgesehene Nutzung sicherstellen und Unberechtigten den Zugang versperren.

Wert	Bezeichnung	Beschreibung
1	streng geheim	Die Daten sind streng geheim und nur mit speziellen Rechten zugänglich. Das Veröffentlichen der Daten kann das Fortbestehen des Unternehmens gefährden.
2	geheim	Die Daten sind geheim. Sie dürfen nur explizit berechtigten Personen zugänglich sein. Werden die Informationen bekannt, kann dies einen deutlichen wirtschaftlichen Schaden verursachen.
3	vertraulich	Die Daten sind für den internen Gebrauch vorgesehen und dürfen nicht ohne Genehmigung an Externe vergeben werden.
4	öffentlich	Die Daten sind öffentlich zugänglich (z.B. Produktkatalog)

Tabelle 9: Werte für das Kriterium Kritikalität von Daten

Mit zunehmender Kritikalität der Daten sind erhöhte Sicherheitsvorkehrungen in den Anwendungen zu treffen. Diese Maßnahmen lassen sich ebenfalls mit einer zunehmenden Relevanz für das Unternehmen begründen.

3 Grad der Industrialisierung

Für die Umsetzung industrieller Prinzipien bietet die Softwarebranche bereits vielversprechende Ansätze in Form von Methoden an, die in dieser Arbeit analysiert werden. Diese Methoden sind dabei nicht isoliert zu betrachten. Vielmehr bestehen Abhängigkeiten zwischen ihnen, die eine stufenweise Einführung im Unternehmen nahelegen. So setzt die Wiederverwendung von Artefakten beispielsweise eine Standardisierung im Unternehmen voraus, die eine Einbindung in mehrere Anwendungen ermöglicht.

Unternehmen, in denen Softwareentwicklung stattfindet, verfügen bereits über eine spezifische Entwicklungsmethode. Diese kann unkoordiniert und chaotisch sein oder bereits über Strukturen verfügen, die eine bestimmte Reife aufweisen. Als Ausgangsbasis für eine Optimierung ist die Anwendungsentwicklung hinsichtlich des Industrialisierungsgrades zu analysieren und zu bewerten. Hierfür ist zu klären, wie der Industrialisierungsgrad ermittelt werden kann.

Kern dieser Arbeit ist die Entwicklung eines Modells zur Bestimmung des Industrialisierungsgrades der Anwendungsentwicklung. Dabei werden die in der Praxis bewährten Methoden mit den Prinzipien der Industrialisierung verknüpft. Die Abhängigkeiten der verschiedenen Methoden voneinander werden durch ein Stufenmodell abgebildet. Es schließt damit auch gleichzeitig die Reihenfolge mit ein, in der die Industrialisierung der Anwendungsentwicklung im Unternehmen eingeführt werden sollte.

Ein Unternehmen hat für jeden seiner Bereiche zu entscheiden, welcher Grad an Industrialisierung benötigt wird. Ob eine Methode im Unternehmen bereits etabliert ist, wird anhand eines Reifegrades ermittelt, der analog den Reifegradmodellen[37] CMMI oder SPICE aufgebaut ist.

Die Ermittlung des Reifegrades erfolgt auf Basis der Bewertung der Methoden zur Erstellung der verschiedenen Artefakte. Dieser Vorgang wird in Kapitel 4 erläutert.

In diesem Kapitel werden nach Einführung des Stufenmodells sowie der verschiedenen Reifegrade die einzelnen Stufen des Industrialisierungsgrades sowie die zugehörige Methode vorgestellt.

[37] Im Unterschied zu CMMI oder SPICE werden nicht die verschiedenen Entwicklungsprozesse, sondern die Methoden zur Erzeugung von Artefakten betrachtet.

3.1 Stufenmodell für den Grad der Industrialisierung

In Kapitel 2.1 wurden die verschiedenen Prinzipien der Industrialisierung sowie eine möglichen Zuordnung zu Themen der Softwarebranche dargestellt. Eine Priorisierung dieser Prinzipien oder Methoden ist in der Literatur nicht zu finden. Die Frage bleibt offen, welches der erste Schritt in Richtung der Industrialisierung ist.

Eine gleichzeitige Einführung aller Methoden ist bereits aus deren Abhängigkeiten untereinander nicht möglich. Beispielsweise ermöglicht erst die Einführung von unternehmensweiten Standards die allgemeine Wiederverwendung von Komponenten. Da die Einführung einer Methode nicht nur konzeptioneller Art ist, sondern auch den Transfer in die IT-Landschaft sowie in die IT-Organisation beinhaltet, sollten Veränderungen schrittweise eingeführt werden. Dies erhöht die Nachhaltigkeit der Methode, da es u.a. dem Entwickler ermöglicht, sich mit den Verfahren und Vorgaben zu beschäftigen und Erfahrungen zu sammeln.

Die in Abschnitt 2.1 genannten Prinzipien der Industrialisierung werden für die weitere Betrachtung in die folgenden vier Gruppen zusammengefasst:

- Standardisierung
- Wiederverwendung (enthält Modularisierung, Arbeitsteilung und Spezialisierung)
- Automatisierung
- Komposition (Verringerung der Fertigungstiefe)

Diesen Gruppen werden Methoden aus der Softwareentwicklung zugewiesen. Die schrittweise Einführung dieser Methoden erfolgt in Stufen, die in einem Stufenmodell zusammengefasst werden. Dieses Modell wird der Klasse der Gestaltungsmodelle zugeordnet (vgl. Abschnitt 2.5).

Die Abhängigkeiten der Methoden im Kontext der Anwendungsentwicklung ergeben eine Reihenfolge und ermöglichen damit eine Zuordnung zu einer Stufe. Diese Reihenfolge wird zudem untermauert durch die zunehmende Unterstützung der Faktoren der Anwendungsentwicklung.[38]

Die Abhängigkeiten und damit die Reihenfolge der Stufen ergeben sich aus den folgenden Überlegungen:

- *Standardisierung*: Diese Gruppe enthält alle Formen der Standardisierung von Elementen der Anwendungsentwicklung. Sie bildet die Grundlage für alle weiteren Methoden.

[38] Siehe Abschnitt 2.4.

- *Wiederverwendung*: Die mehrfache Nutzung von Komponenten in der Anwendungsentwicklung setzt den modularen Aufbau von Anwendungen voraus. Aufgrund dieser modularen Struktur können verschiedene Komponenten eingebunden werden, die durch spezialisierte Entwickler auf Basis von Arbeitsteilung entstanden sind. Die Standardisierung ist dabei eine wichtige Voraussetzung für das Zusammenspiel der Komponenten.
- *Automatisierung*: Die Automatisierung befreit den Entwickler von mechanischen Arbeiten. Dieser kann sich auf die Lösung des Problems fokussieren. Wichtige Voraussetzung ist dabei der modulare Aufbau von Anwendungen, deren spezialisierte Komponenten durch Arbeitsteilung entstehen und bei der automatisierten Implementierung genutzt werden. Bei der Generierung von Anwendungen werden zudem Vorlagen wiederverwendet.
- *Komposition*: Das Erstellen von Anwendungen wird im Wesentlichen durch die Komposition von Diensten in Form von eigenständigen Komponenten durchgeführt. Diese Dienste werden zunehmend durch externe Firmen zur Verfügung gestellt. Sie reduzieren damit die Fertigungstiefe im Unternehmen. Die Komposition basiert sowohl auf der Standardisierung, der Wiederverwendung von Komponenten sowie den durch Modelle automatisch erzeugten Artefakten.

In Abbildung 9 wird das Stufenmodell basierend auf den Prinzipien der Industrialisierung sowie den zugeordneten Methoden dargestellt. Die Stufen bauen aufeinander auf. Damit muss für das Erreichen einer bestimmten Stufe die Methode der darunter liegenden Stufe bereits etabliert sein. Jede Stufe entspricht dabei einem Grad der Industrialisierung. Je höher der Grad, umso fortgeschrittener ist die Industrialisierung in der Anwendungsentwicklung.

Jede Stufe enthält eine spezifische Methode, wie Artefakte zu erstellen sind, die mit zunehmender Stufe eine höhere Rationalität ermöglichen. Die Methoden werden in Unternehmen meist nicht sofort vollständig, sondern schrittweise umgesetzt. Daher wird für jede Stufe ein Reifegrad ermittelt, der Auskunft über den Einsatz der Methode gibt (siehe Abschnitt 3.2). Das Erreichen der nächsten Stufe bedingt somit das Erreichen eines bestimmten Reifegrades der betrachteten Stufe.

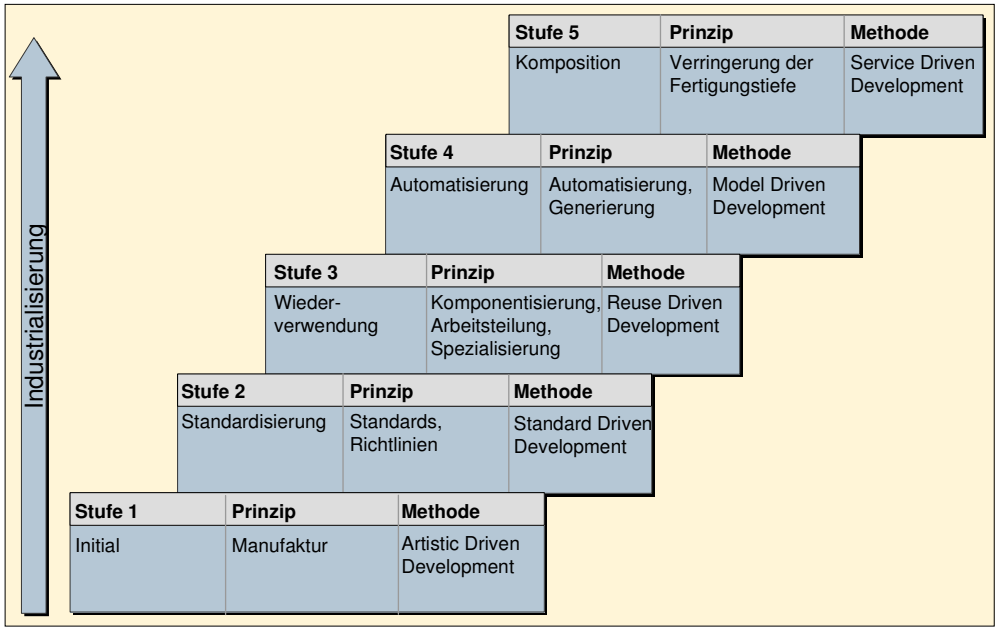

Abbildung 9: Stufenmodell der Industrialisierung

Wie in Abbildung 9 dargestellt, werden die folgenden Methoden den Prinzipien der Industrialisierung zugeordnet:

1. Initial: Artistic Driven Development

   Rasche Umsetzung von Anwendungen ohne Berücksichtigung von Standards und Richtlinien. Es sind keine Vorgaben vorhanden. Der Entwickler realisiert Anwendungen nach bestem Wissen und Gewissen.

2. Standardisierung: Standard Driven Development

   Ausrichtung der Implementierung von Anwendungen anhand der gültigen Standards im Unternehmen. Für jedes Analyseelement existiert ein Standard, der in der Entwicklung einzuhalten ist.

3. Wiederverwendung: Reuse Driven Development

   Die Entwicklung von Anwendungen basiert auf der Wiederverwendung von zentral koordinierten Komponenten. Der Aufbau der Anwendungen selbst folgt dabei den Prinzipien der Komponentenorientierung[39]. Die Komponenten werden Bestandteil der Anwendung.

4. Automatisierung: Model Driven Development

   Automatisierung von wiederkehrenden Aufgaben durch Generierung der benötigten Artefakte auf Basis von Modellen und Generatoren zur Befreiung des Entwicklers von mechanischen Arbeiten.

[39] Siehe Abschnitt 3.6.1

5. Komposition: Service Driven Development

   Das Zusammenstellen von existierenden Komponenten zu neuen Anwendungen (Composite Applications) bildet die fünfte und höchste Stufe. Es erhöht die flexible Gestaltung von Geschäftsprozessen und ermöglicht eine weitere Reduzierung der Fertigungstiefe im Unternehmen. Die Komponenten sind eigenständig und nicht Bestandteil der Anwendung.

Die Beschreibung der einzelnen Stufen erfolgt ab Kapitel 3.4. Die Frage nach der Zuordnung einer Methode zu einer Stufe ergibt sich neben der Betrachtung der Abhängigkeiten auch aus der Analyse der Faktoren der Anwendungsentwicklung (siehe Abschnitt 2.4). Ein wesentliches Element ist dabei die Abnahme der manuellen Erstellung von Quelltext, wie in Abbildung 10 dargestellt. Eine Verringerung der manuellen Tätigkeiten wirkt sich positiv auf die Reduzierung von Zeit und Kosten sowie auf die Erhöhung der Qualität aus. Die Begründungen hierzu liefern die Erläuterungen der einzelnen Stufen.

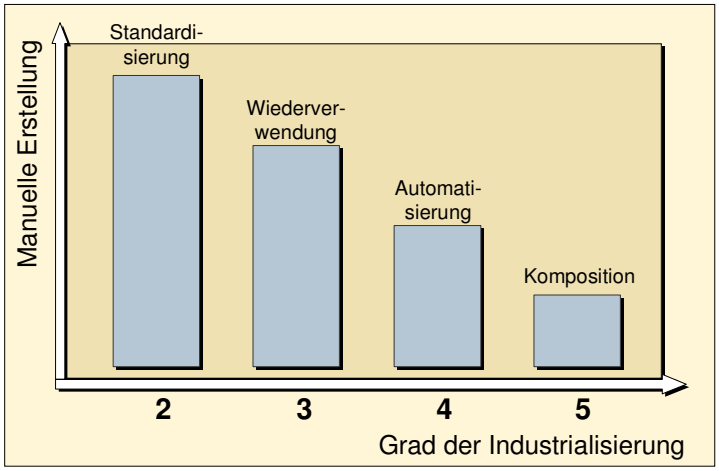

Abbildung 10: Abnahme der manuellen Erstellung von Quelltext

Der Anwendungsentwickler kann sich mit zunehmendem Industrialisierungsgrad verstärkt der Lösung der Aufgabenstellung widmen und ist weniger mit routinemäßigen Implementierungsthemen belastet.

Die einzelnen Methoden unterstützen in den höheren Stufen verstärkt die Flexibilisierung bei der Unterstützung der Geschäftsprozesse. Die Strukturierung in Komponenten und die anschließende Komposition in unterschiedlichen Szenarien ermöglichen ein schnelles Reagieren des Unternehmens auf veränderte Marktbedingungen (siehe Abbildung 11).

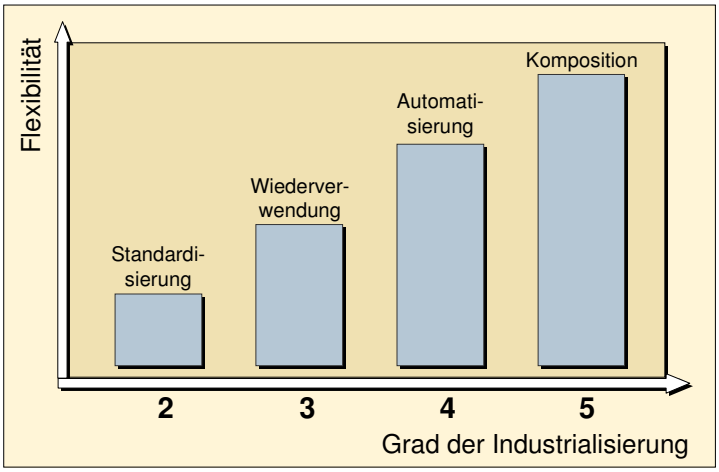

Abbildung 11: Zunahme an Flexibilität

3.2 Reife der eingesetzten Methoden

Das Stufenmodell legt fest, in welcher Stufe welche Methoden einzusetzen sind. In dieser Form gibt es jedoch noch keine Auskunft darüber, inwieweit die jeweiligen Methoden in einem Unternehmen angewendet werden. Sofern z.B. Standards definiert sind, aber nicht zum Einsatz kommen, ist der Nutzen dieser Standards für eine Organisation nicht gegeben.

Für die Bestimmung des Industrialisierungsgrades ist es daher wesentlich, den Reifegrad der eingesetzten Methoden der jeweiligen Stufe zu ermitteln. Reifegrade werden in der Informatik insbesondere für Prozesse ermittelt. Häufig werden in diesem Zusammenhang die Modelle CMMI (siehe [Kneu06]) und SPICE (siehe [HDHM06]) genannt, die Prozesse im Rahmen der Anwendungsentwicklung bewerten. Neben den Vorgaben der Prozesse werden dort auch Kriterien und Verfahren zu deren Überprüfung bereitgestellt. Diese Modelle finden bereits breite Anwendung in der Praxis und werden daher nicht weiter ausgeführt. Sie können ergänzend zu den Ausführungen dieser Arbeit angewandt werden.

3.2.1 Reifegrade

Die Bestimmung des Reifegrades der Methoden ist angelehnt an CMMI. Es werden analog fünf aufeinander aufbauende Stufen verwendet. Ein bestimmter Reifegrad kann nur erreicht werden, wenn die darunter liegenden Stufen erfüllt sind. Die Stufen werden anhand bestimmter Kriterien bewertet (siehe Abschnitt 3.2.2).

Das Reifegradmodell wurde für den Einsatz in dieser Arbeit angepasst. Statt der Prozesse werden Methoden auf ihre Reife überprüft. Die Namen der Stufen sowie deren Bedeutung wurden ebenfalls verändert. Ferner wurden die Kriterien an den Methoden ausgerichtet.

Für jeden Grad der Industrialisierung werden die folgenden Reifegrade betrachtet:

- *Initial*: Es sind keine Strukturen und Ansätze vorhanden.
- *Wiederholbar*: Die Methode wird vereinzelt angewandt, sie ist aber im Unternehmen nicht abgestimmt.
- *Definiert*: Die Methode ist allgemein definiert und kann angewendet werden. Sie ist jedoch nicht verpflichtend.
- *Eingeführt*: Die Methode wurde im Unternehmen eingeführt und wird bei jeder Anwendungsentwicklung angewandt.
- *Optimiert*: Rückmeldungen zum Einsatz der Methode werden systematisch erfasst. Mögliche Optimierungen aus diesen Rückmeldungen werden umgesetzt.

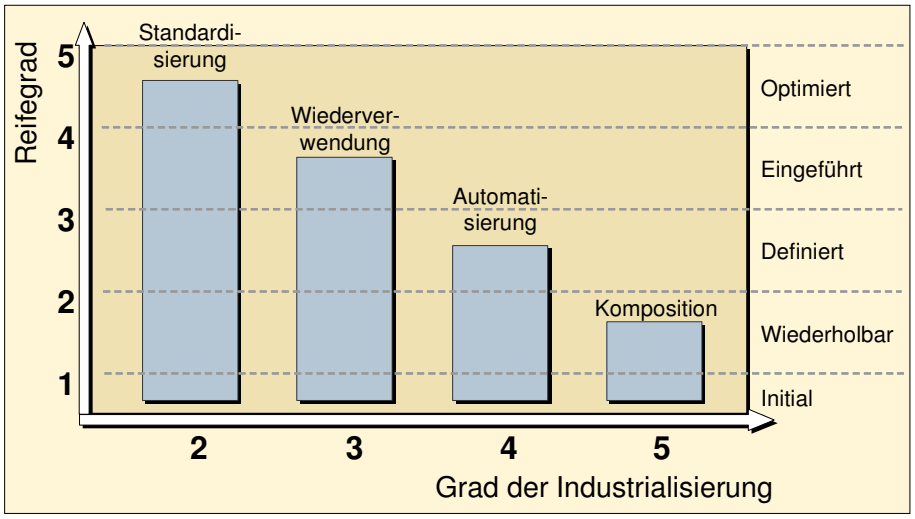

Abbildung 12: Reifegrade pro Grad der Industrialisierung

Abbildung 12 zeigt eine beispielhafte Grafik einer Bewertung, in der die Reifegrade jeder Stufe eingezeichnet werden. Allgemein kann ein bestimmter Grad an Industrialisierung nur erreicht werden, wenn die zugehörige Methode bereits in der Entwicklung verpflichtend zum Einsatz kommt und damit den Reifegrad 4 „eingeführt“ hat. Beispielsweise kann eine Wiederverwendung nur erfolgen, wenn sich die verschiedenen Komponenten an den gültigen Standards orientieren und damit eine Einbindung in unterschiedlichen Anwendungen ermöglicht wird. Die einzelnen Reifegrade werden im Abschnitt 3.2.3 detaillierter erläutert.

3.2.2 Kriterien für die Beurteilung des Reifegrades

Ein bestimmter Grad an Industrialisierung wird nur erreicht, wenn die zugehörige Methode im Unternehmen eingeführt ist. Die Beurteilung, ob die Methode eingeführt ist, erfolgt anhand der folgenden Kriterien:

- *Vollständigkeit:* Für die Nutzung einer Methode ist definiert, wie diese im Unternehmen eingesetzt wird. Vollständig bedeutet dabei, dass für die Erstellung aller in Abschnitt 2.3.2 festgelegten Artefakte der Anwendungsarchitektur sowie des Entwicklungsumfeldes das Vorgehen spezifiziert ist.

  Zudem sind für alle Elemente der Entwicklungsinfrastruktur, der Laufzeitinfrastruktur, aber auch der Gremien und Prozesse die Voraussetzungen für das Erreichen dieser Stufe definiert.

- *Konsistenz:* Bei der Entwicklung einer Anwendung sind alle Artefakte einer Anwendung nach derselben Methode erstellt und widerspruchsfrei. Hierbei wird zudem überprüft, ob die erstellten Artefakte die Vorgaben der Methode erfüllen.

- *Etabliertheit:* Die definierte Methode ist in der Organisation allen an der Entwicklung beteiligten Personen bekannt. Die Implementierung aller Anwendungen folgt den Regeln und Vorgaben der Methode. Dies betrifft die Erstellung der Artefakte, der Verfügbarkeit der Entwicklungs- und Laufzeitinfrastruktur, aber auch der Etablierung der vorgesehenen Gremien sowie der Durchführung der definierten Prozesse.

- *Aktualität:* Die Artefakte einer Anwendung sind auf einem aktuellen Stand. Beispielsweise spiegeln die Dokumentationen sowie die Tests den aktuellen Stand der Anwendung wider.

3.2.3 Spezifikation der Reifegrade mittels Kriterien

Die Reife einer Methode im Unternehmen wird auf Basis der vorgestellten Kriterien bestimmt. Die Beurteilung der verschiedenen Reifegrade erfolgt anhand des folgenden Schemas (siehe auch Tabelle 10):

- *Initial:* In der ersten Stufe wird die Methode des zugehörigen Industrialisierungsgrades nicht verwendet. Die Methode ist nicht bekannt und damit nicht etabliert. Die Art und Weise, wie und welche Artefakte bzgl. dieser Methode erstellt werden, ist abhängig von der individuellen Arbeitsweise des Entwicklers. Die Konsistenz der Artefakte untereinander ist nicht erkennbar. Es sind keine Strukturen und Ansätze vorhanden.

- *Wiederholbar:* Erste Ansätze zur Erstellung von Artefakten auf Basis der vorgegebenen Methode sind vorhanden und können wiederholbar einge-

setzt werden. Es gibt jedoch keine unternehmensweit abgestimmten Vorgaben für den Einsatz der Methode. In den verschiedenen Implementierungen werden die Methoden unterschiedlich umgesetzt.

Gremien und Prozesse sind noch nicht vorhanden. Die Vollständigkeit der Methode richtet sich nach dem individuellen Bedarf.

Die Konsistenz sowie die Aktualität der Artefakte bzgl. der eingesetzten Methode sind abhängig von den aktuellen Anforderungen der Anwendung.

- *Definiert:* Die Methode ist vollständig definiert. Für jedes spezifizierte Artefakt sind entsprechende Vorgaben und Standards vorhanden. Diese Vorgaben sind für alle Entwickler zugänglich und können genutzt werden. Gremien und Prozesse sowie die Entwicklungs- und Laufzeitinfrastruktur sind definiert. Die Voraussetzungen für den Einsatz der Methode sind vorhanden.

  Die Vorgaben der Methoden werden jedoch noch nicht in allen Entwicklungen eingehalten, da sie in dieser Stufe noch nicht verpflichtend sind. Ebenso werden die Konsistenz sowie die Aktualität der Artefakte noch nicht überprüft.

- *Eingeführt:* Die festgelegte Methode, aber auch die Gremien und Prozesse sowie die Infrastrukturen sind allen an der Entwicklung beteiligten Personen bekannt und werden genutzt. Die Entwicklungen basieren auf den Vorgaben der Methode. Die Artefakte einer Anwendung werden konsistent bzgl. der Methode erstellt und haben einen aktuellen Stand.

- *Optimiert:* Die definierte Methode wird regelmäßig auf ihre Aktualität bzgl. des Einsatzes im Unternehmen überprüft. Sofern die bestehenden Verfahren nicht mehr effizient eingesetzt werden können oder neue Verfahren entwickelt wurden, sind diese zu prüfen und in die Vorgaben einzubauen.

  Die Rückmeldungen von Personen, die an der Entwicklung von Anwendungen beteiligt sind, wird systematisch erfasst und ausgewertet. Durch ein abgestimmtes Verfahren wird diese Rückmeldung in die Methode eingearbeitet und somit schrittweise optimiert.

Dieses Schema wird in der Tabelle 10 zusammengefasst. Für jeden Reifegrad wird die Ausprägung der jeweiligen Kriterien beschrieben. Es ist ein allgemeiner Rahmen, an dem sich die Bewertung der Methode ausrichtet.

	Initial	Wiederholbar	Definiert	Eingeführt	Optimiert
Vollständigkeit	Methode ist nicht vorhanden.	Teile der Methode sind beschrieben und können genutzt werden.	Die Methode ist vollständig definiert und kann genutzt werden.	-	Feedback zur Methode wird systematisch erfasst und eingearbeitet.
Konsistenz	Konsistenz ist nicht vorhanden.	Artefakte werden teilweise in der Methode erstellt. Es gibt jedoch noch keine Vorgaben.	Erste Artefakte basieren auf den Vorgaben der Methode.	Die erstellten Artefakte basieren auf den Vorgaben der Methode und sind konsistent zueinander.	Überprüfung der Konsistenzen.
Etabliertheit	Die Methode ist im Unternehmen nicht bekannt.	Die Methode ist in Teilen bekannt und wird ad hoc verwendet.	Die Methode ist in der Organisation bekannt, und wird bereits in Teilen angewandt.	Methode wird in der Organisation verpflichtend eingesetzt. Die Einhaltung wird durch ein zentrales Gremium überprüft.	Feedback wird systematisch erfasst und ausgewertet.
Aktualität	Es ist nicht klar, welche Artefakte in welcher Form vorhanden sein sollen.	Die Aktualität der Artefakte ist in Teilen gegeben.	Es ist definiert, in welcher Form die Aktualisierung stattzufinden hat.	Die Artefakte einer Anwendung sind aktuell und synchron mit der Anwendung.	Feedback wird systematisch eingearbeitet.

Tabelle 10: Erläuterung der Reifegrade anhand von Kriterien

3.3 Erläuterungen zum Stufenmodell

In den kommenden Abschnitten werden die einzelnen Stufen des Industrialisierungsgrades beschrieben. Der Aufbau der Beschreibung richtet sich dabei jeweils nach der folgenden Struktur:

- *Beschreibung:* Eingeleitet wird die Erläuterung der Stufe mit einer allgemeinen Beschreibung der Methode sowie dem Bezug zur Industrialisierung.
- *Methode:* In diesem Abschnitt wird die Methode der jeweiligen Stufe definiert sowie analysiert, inwiefern diese Methode die verschiedenen Eigenschaften der Industrialisierung unterstützt. Ferner wird der Einfluss der Methode auf die Faktoren der Anwendungsentwicklung betrachtet.
- *Erzeugen von Artefakten auf Basis der betrachteten Methode:* In den einzelnen Stufen erfolgt das Erstellen der Artefakte nach unterschiedlichen Mechanismen. In diesem Abschnitt wird erläutert, in welchen Schritten

und nach welchen Vorgaben das Erzeugen der Artefakte erfolgt. Die Erläuterung der Schritte wird durch eine Grafik schematisch dargestellt (vgl. Abbildung 13).

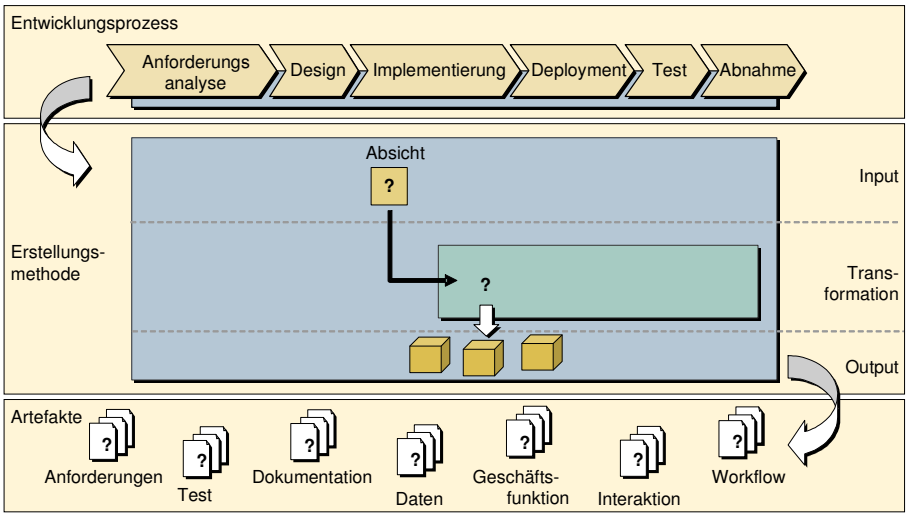

Abbildung 13: Schematische Darstellung einer Methode

Der Entwicklungsprozess enthält die Prozessschritte der Anwendungsentwicklung, in welchen die Artefakte erstellt werden. Dieser Abschnitt ist für alle Stufen des Industrialisierungsgrades gleich und wird daher nicht näher beleuchtet.[40]

Die Erstellungsmethode beschreibt die Transformation von Eingabeelementen (Inputs), wie z.B. Anforderungen an die Anwendung, zum geforderten Ergebnis (Output), wie z.B. die verschiedenen Artefakte der Anwendung.

- *Rahmenbedingungen:* Die industrielle Fertigung von Anwendungen erfordert strategische Rahmenbedingungen für den Entwicklungsprozess (siehe Abschnitt 2.2). Dies umfasst die folgenden Aufgaben:
  - Organisation der Entwicklung durch Gremien
  - Festlegung von Prinzipien und Standards auf Basis etablierter Prozesse
  - Festlegung von Vorgaben für die Entwicklungsinfrastruktur
  - Festlegung von Rahmenbedingungen der Laufzeitinfrastruktur

  Diese Aufgaben werden entsprechend der Methode für jede Stufe beschrieben.

[40] Erläuterungen zum Entwicklungsprozess: siehe Abschnitt 2.3.1.

- *Reifegrade der Methode:* Für jede Stufe werden die Reifegrade der betrachteten Methode spezifiziert. Die Beschreibung enthält die Kriterien, welche für die Bewertung der Methode benötigt werden.
- *Grenzen und Potenziale für Verbesserungen:* Neben den Möglichkeiten der beschriebenen Methoden werden auch ihre Grenzen beleuchtet. Diese Grenzen geben meist Hinweise auf die Potentiale der nächsten Stufe der Industrialisierung.

Die im Rahmen des Industrialisierungsgrades betrachteten Methoden werden anhand ihrer Prinzipien beschrieben und analysiert. Eine spezifische Unterstützung durch Werkzeuge oder Technologien wird dabei nicht untersucht. Die Automatisierung erfolgt auf Basis der modellgetriebenen Entwicklung. Ob dabei jedoch eine eigens für das Unternehmen entwickelte Modellierungssprache mit speziellen Generatoren und spezifischen Regeln entwickelt wurde oder die Werkzeuge eines Herstellers, wie z.B. SAP, verwendet werden, ist für diese Arbeit nicht relevant.

Wichtig ist die zentrale Festlegung des Unternehmens für eine spezifische Umsetzung der Methode, die für die jeweiligen Stufen einzuhalten ist. Die betrachtete Methode wird in den einzelnen Stufen daher nur vom Grundprinzip beschrieben.

3.4 Stufe 1: Initial (Handwerk)

Die initiale Stufe der Industrialisierung zeichnet sich durch einen hohen Grad an individueller Realisierung von Anwendungen aus. Es gibt kaum Abstimmungen mit anderen Bereichen des Unternehmens. Die eingesetzte Methode ist nicht spezifiziert. Sie ist abhängig vom jeweiligen Entwickler.

Anforderungen an die Entwicklung von Anwendungen werden sehr lokal bearbeitet. Eine Abstimmung mit ähnlichen Themen wird nicht in Betracht gezogen oder ist aufgrund zeitlicher Abhängigkeiten nicht gewollt.

3.4.1 Artistic Driven Development

Die initiale Stufe fordert keine Form der Industrialisierung. Die Methode dieser Stufe entspricht einem sehr pragmatischen Vorgehen:

> Individuelle Umsetzung von Anwendungen ohne Berücksichtigung von Standards und Richtlinien. Es sind keine Vorgaben vorhanden. Der Entwickler realisiert Anwendungen nach eigenen Richtlinien.

Definition 2: Artistic Driven Development

Die Realisierung von Anwendungen ist sehr stark von den Erfahrungen und Fertigkeiten des Entwicklers abhängig. Die Zielsetzung für diese Stufe ergibt sich aus einer kurzfristigen und möglichst kostensparenden Realisierung von Anforderungen ohne Beachtung der Qualitätsfaktoren. Diese Faktoren werden daher nicht näher analysiert. Auch die Kosten werden meist nur für die initiale Implementierung betrachtet und nicht für den Betrieb sowie die weitere Pflege der Anwendung.

Einzelne Entwickler können in dieser Stufe durchaus eine effiziente und qualitativ hochwertige Softwareentwicklung durchführen. Es bleibt jedoch, wie beim Kunsthandwerk, auf die Fertigkeit spezifischer Personen beschränkt. Keine der Eigenschaften der Industrialisierung finden Anwendung in der Entwicklung von Anwendungen im Unternehmen. Die Prinzipien werden daher auch nicht näher betrachtet.

3.4.2 Erstellung von Artefakten auf Basis des Handwerks

Für diese Stufe gibt es keine Vorgaben für den Aufbau einer Anwendung. Dem Entwickler steht die Wahl der Methode frei und er kann die Umsetzung nach seiner eigenen Vorstellung realisieren.

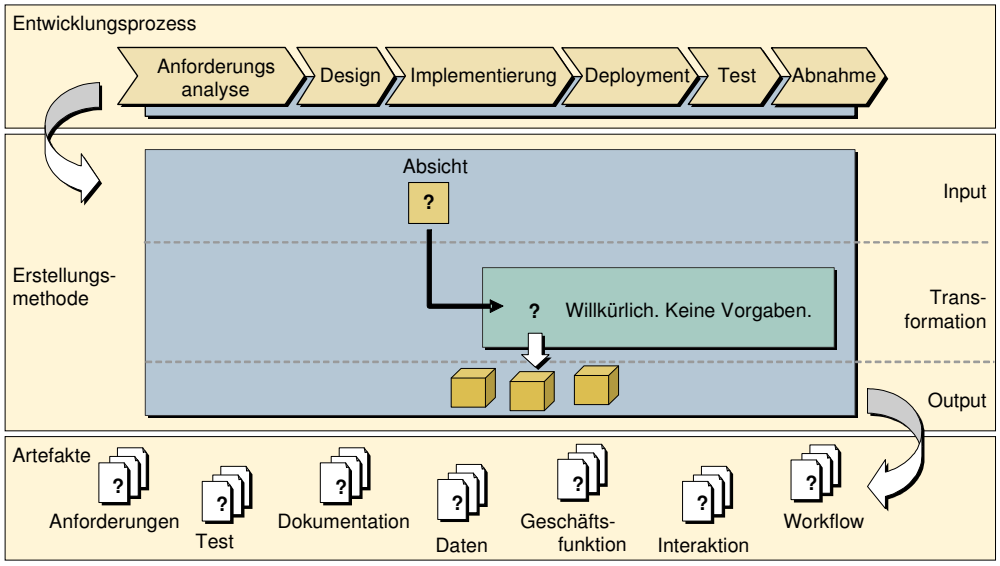

Abbildung 14: Artistic Driven Development

Dies kann bedeuten, dass keine Trennung zwischen den verschiedenen Schichten, wie sie in Abschnitt 2.3.2 vorgestellt wurden, existiert. Der Entwickler entscheidet ferner nach eigenem Ermessen, welche der Artefakte erstellt werden. Abbildung 14 stellt schematisch einen möglichen Ablauf dar.

Für die Entwicklungsmethode Artistic Driven Development erfolgt das Erstellen von Artefakten nach dem folgenden Schema:

Input

Die Anforderungen an die Anwendung (Absicht) ist der einzige Input für die Transformation. Es gibt keine Vorgaben für die Form bzw. Struktur dieses Inputs.

Transformation

Die Realisierung der Anwendung auf Basis des Inputs erfolgt individuell durch den Entwickler. Auch hier sind keine Vorgaben vorhanden.

Output

Die Art und die Form des Outputs werden individuell durch den Entwickler festgelegt. Verbindliche Vorgaben existieren nicht. Die Fragezeichen in den Artefakten der Abbildung 14 deuten an, dass sich die Art des Outputs nicht aus der Methode ergibt, sondern vom Entwickler bestimmt wird.

3.4.3 Rahmenbedingungen für das Handwerk

Für diese Stufe gibt es keine Voraussetzungen, die im Rahmen der Industrialisierung einzuhalten sind. Weder Gremien noch Prozesse stellen Vorgaben zur Verfügung. Die eingesetzte Entwicklungsinfrastruktur wird individuell durch den Entwickler bestimmt.

3.4.4 Reifegrade des Handwerkes

Die Erstellung von Anwendungen wird von der individuellen Professionalität des zuständigen Entwicklers bestimmt. Da keine Prinzipien oder Vorgaben existieren, deren Einhaltung analysiert werden kann, ist diese Stufe immer erfüllt.

3.4.5 Mögliche Einsatzgebiete

Die Anwendung von Methoden der Industrialisierung ist nicht immer von Vorteil. Es gibt Bereiche, in denen diese initiale Stufe für die Anwendungsentwicklung ausreichend ist:

- *Prototyping:* In der Analysephase der Entwicklung einer Anwendung ist es sehr hilfreich, die Anforderungen anhand eines Prototypen[41] zu ermitteln. Dieser dient lediglich zur Überprüfung der Vollständigkeit der Anforderungen und wird anschließend verworfen.
- *Notfall:* Bei Ausfall von Systemen sind schnelle Lösungen zur Behebung des Problems wichtig, um wichtige Funktionalitäten aufrechtzuerhalten.

[41] Prototyp: Vorabexemplar einer Anwendung zur Veranschaulichung von Funktionen und Eigenschaften der zu implementierenden Anwendung.

Sobald das ausgefallene System wieder verfügbar ist, wird die Zwischenlösung verworfen.

- *Einmal-Anwendungen:* Anwendungen, die nach der Erstellung nur einmalig verwendet werden. Beispiele hierfür sind die Unterstützung der Migrationen von Anwendungen oder die einmalige Erfassung von Daten. Der Fokus liegt hierbei nicht auf der Wartbarkeit, sondern auf der Funktionalität.
- *Forschung:* Für bestimmte Versuchsreihen in der Forschung kann es sinnvoll sein, spezifische und hochgradig individuelle Anwendungen zu erstellen.

3.4.6 Grenzen des Artistic Driven Developments

Das wesentliche Merkmal dieser Stufe liegt in der Freiheit des Entwicklers für die Nutzung seiner eigenen Methoden. Ausgehend von dieser Stufe liegt das größte Potential hinsichtlich der Industrialisierung in der Standardisierung. Zentrale Vorgaben in Form von Standards oder Richtlinien sowie deren Einhaltung ermöglichen eine effizientere Realisierung sowie Wartung von Anwendungen. Der Einsatz von Standards vereinfacht das Verständnis der Konstruktion der Anwendung. Dies wird insbesondere bei der Fehlerbehebung oder der Erweiterung durch zusätzliche Entwickler relevant.

3.5 Stufe 2: Standardisierung

Die Verständigung auf Standards hat eine lange Tradition. Im Handel ist die Festlegung auf bestimmte Maße, wie z.B. Länge (Meter) oder Gewicht (Kilo), eine wichtige Voraussetzung für die Bestimmung eines Wertes [Müll03].

Die Standardisierung im Rahmen der Industrialisierung war nach [Müll03] vor allem technisch geprägt. Der Fokus lag auf der Vereinheitlichung von Werks- und Betriebsstoffen. Die Existenz sowie die Einhaltung von Standards ermöglichte eine schnellere, effizientere und sicherere Produktion [Maid05].

Der erste Schritt in Richtung Industrialisierung bei der Anwendungsentwicklung ist daher geprägt von der Erstellung sowie der Nutzung von Standards. Sie sind Voraussetzung für eine effiziente Arbeitsteilung sowie für die Mechanisierung bzw. Automation von Prozessen [Adam01]. Was unter einem Standard zu verstehen ist, hat [MüHe04] aus den Beschreibungen von mehreren Autoren wie folgt zusammengefasst.

„Grundsätzlich können Standards als Institution aufgefasst werden, die systematisch darauf abzielt, mittels formalen oder informalen Regeln entweder das individuelle, soziale Verhalten oder die Eigenschaften von Produkten, Methoden und Dienstleistungen zu spezifizieren bzw. zu beschreiben. Allgemein versteht man

unter Standardisierung eine Vereinheitlichung von Gütern, von Produktionsmethoden oder anderem, letztlich von Objekten; dabei kann sich die Standardisierung auf eine Vielzahl an Bereichen erstrecken. Ein hohes Maß an Generalisierbarkeit und Wiederholbarkeit des Standardisierungsobjektes gelten als Voraussetzungen für die Entwicklung von Standards. Standards sind also eine Spezifikation, die eine breite Verwendung finden."

Im Kontext der Anwendungsentwicklung wird ein Standard wie folgt definiert:

> Standards sind akzeptierte, verpflichtende und angewandte Bestimmungen oder Normen für Technologien, Plattformen oder Infrastrukturkomponenten. Sie haben das Ziel, grundlegende Eigenschaften sicherzustellen.

Definition 3: Standard

Solche Bestimmungen oder Normen werden über einen mehrstufigen Prozess zu einem Standard. Sie sind zunächst arbeitsspezifische Vorgaben, die ein breiteres Interesse finden. Durch einen Selektionsprozess im Markt kristallisieren sich solche Bestimmungen heraus, die das größte Potential auf einen Standard aufweisen. Diese werden durch ein anerkanntes Gremium genormt. Von einem Standard spricht man, wenn diese Norm eine weite Verbreitung gefunden hat (vgl. [Müll03]).

Damit ergibt sich für die Standardisierung die folgende Definition:

> Die Standardisierung ist der Vorgang zur Festlegung und Sicherstellung von grundlegenden Eigenschaften einer Technologie, Plattform oder Infrastrukturkomponente. Standards entwickeln sich aus arbeitsspezifischen Vorgaben bzw. Bestimmungen, die aufgrund eines breiten Interesses durch ein Gremium genormt werden.

Definition 4: Standardisierung

Insbesondere in der Informatik wird nach [Müll03] erwartet, dass Standards durch ein anerkanntes Gremium nach klaren Vorgaben entwickelt und gepflegt werden. Erfolgt dies nicht, wird der Standard als „proprietär" bezeichnet. Dies gilt nach [Dumk03] auch für die Standardisierung im Bereich der Anwendungsentwicklung.

Als Beispiel für einen gültigen Standard sei die Unified Modeling Language (UML) genannt. Sie ist eine Sprache für die Modellierung von Software bzw. allgemein von Systemen. Die Vereinheitlichung der Terminologie sowie die Standardisierung der Notation im Rahmen der UML erleichtert die Verständigung zwischen den Beteiligten der Anwendungsentwicklung. Sie entstand in den 1990er Jahren als Reaktion auf die vielen verschiedenen Ansätze der Modellie-

rung von Software. Aufgrund des breiten Interesses an dieser Sprache wurde sie an die Object Management Group (OMG)[42] übergeben und in diesem Gremium genormt. Nach [GrBB02] findet diese Sprache mittlerweile breite Unterstützung bei den Werkzeug-Herstellern zur Modellierung von Systemen.

Nach [BuKö98] schaffen Standards Kompatibilität, indem sie die extensionalen Ausprägungen der Struktur sowie das Verhalten von Objekten spezifizieren.

Die ökonomische Bedeutung von Standards wird in der Betriebs- und Volkswirtschaftslehre im Zusammenhang mit Netzeffekten diskutiert (siehe [MüHe04] oder [WeKö03]). Der Netzeffekt (auch Netzwerkeffekt) beschreibt, dass der Nutzen an einem Standard oder Netzwerk wächst, wenn dessen Nutzerzahl größer wird. Das Interesse weiterer Personen wird durch den erhöhten Nutzen geweckt, so dass dieser durch eine Erhöhung der Nutzerzahl weiter gesteigert wird. Ein Telefon oder Faxgerät ist beispielsweise erst sinnvoll, wenn viele Personen über solche Geräte verfügen. Der Nutzen von Standards wächst somit, je breiter dieser Standard genutzt wird.

Der Einsatz von Standards kann jedoch auch zu Nachteilen durch Verlust von Individualität führen (siehe [BuKö98]). Dies ist aber nicht zwingend. Beispielsweise erfolgt die Produktion von Fahrzeugen der Automobilindustrie auf der Basis von Standards. Trotzdem kann das einzelne Auto durch den Kunden in einem gewissen Rahmen individuell zusammengestellt werden [HüBa08].

Auch das Internet basiert auf einem Standard, dessen Nutzung die Bereitstellung einer Vielfalt an heterogenen Informationen ermöglicht.

Standards lassen sich nach [Wey99] in vier Gruppen unterscheiden:

- *Nicht geförderte Standards*: Der Standard setzt sich am Markt ohne weitere Organisation durch. Kein Unternehmen hat Verfügungsrechte.
- *Geförderte Standards*: Standard wird durch ein oder mehrere Unternehmen am Markt durchgesetzt. Nur diese Unternehmen besitzen Verfügungsrechte.
- *Freiwillige Standards*: Diese Standards sind das Resultat der Arbeit von Verbänden nach dem Konsensprinzip.
- *Staatlich verordnete Normen*

Für die Entwicklung von Anwendungen sind insbesondere solche Standards interessant, die am Markt zur Verfügung stehen, aber auch solche, die innerhalb

[42] Object Management Group (OMG): Konsortium, das sich mit der Entwicklung von Standards für die herstellerunabhängige, objektorientierte Programmierung beschäftigt (siehe http://www.omg.org).

eines Unternehmens aufgestellt wurden. Für diese Arbeit werden die Standards nach den folgenden Kriterien unterschieden:

- *Externe Standards (unternehmensübergreifend):* Standards, die durch nationale oder internationale Institutionen wie ISO[43], ANSI[44], DIN[45] etc., erstellt und kommuniziert werden. Sie dienen der Verbindlichkeit in der Zusammenarbeit zwischen Unternehmen. Beispiele für solche Standards sind HTTP[46] , SQL[47] oder UML.
- *Interne Standards (innerhalb einer Unternehmung):* Festlegungen bzgl. des Verhaltens oder der Eigenschaften von Produkten, Technologien, aber auch von Verfahren oder Konventionen innerhalb von Unternehmen. Diese Standards haben sich als technisch nützlich erwiesen und sollen in einem breiten Umfeld eingesetzt werden. Sie können bei verschiedenen Unternehmen unterschiedlich ausgeprägt sein. Wesentlich ist dabei die Festlegung des Standards durch ein internes, zentrales Gremium.

Als Vorlage für einen internen Standard können externe Standards verwendet werden. Beispielsweise existieren verschiedene Programmiersprachen, die als Standard von externen Gremien festgelegt wurden. Aus der Vielzahl an Programmiersprachen sollte ein Unternehmen diejenigen als internen Standard definieren, die in den meisten Anwendungen zum Einsatz kommen.

Für diese Stufe der Industrialisierung sind Festlegungen bzgl. Standards in den folgenden Bereichen wesentlich:

- *Nutzung von Produkten:* Die Softwarebranche bietet ein breites Spektrum an Produkten (Anwendungen, Plattformen etc.), deren Funktionalitäten sich in vielen Bereichen überschneiden können. Standardisierung bedeutet in diesem Kontext eine Konsolidierung auf wenige Produkte, welche die Anforderungen bedarfsgerecht abdecken. Durch die Reduzierung der Anzahl an eingesetzten Produkten kann die Administration effizienter gestaltet werden.
- *Einsatz von Technologien:* Anwendungen lassen sich auf Basis unterschiedlicher Technologien realisieren. Diese können zur Erstellung von Anwendungen (z.B. durch Generatoren[48]) oder aber in der Anwendung

[43] International Organisation for Standardization (ISO).

[44] American National Standards Institute (ANSI).

[45] Deutsche Industrie Norm (DIN).

[46] HTTP (Hypertext Transfer Protocol): Protokoll für den Austausch von Daten im Internet

[47] SQL (Standard Query Language): Abfragesprache für das Ermitteln oder Verändern von Daten einer Datenbank.

[48] Generatoren ermöglichen die automatisierte Erstellung von Anwendungen auf Basis von Regeln und Modellen (siehe Abschnitt 3.7.2).

selbst (z.B. als Framework[49]) eingesetzt werden. Für jede eingesetzte Technologie im Unternehmen müssen entsprechende Kenntnisse bei den Entwicklern vorhanden sein. Eine Standardisierung in diesem Bereich bedeutet die Reduzierung auf die wesentlichen Technologien.

- *Nutzung von Werkzeugen:* Die Art der Erstellung von Anwendungen kann je nach Technologie und Plattform sehr unterschiedlich gestaltet werden. Die Reduzierung von Werkzeugen auf einen Standard zur Implementierung sowie zur Verwaltung von Anwendungen ermöglicht eine bessere Unterstützung der Entwickler.
- *Festlegung von Dokumentationsformaten:* Die Entwicklung von Anwendungen, aber auch deren Verwaltung, wird begleitet von verschiedenen Dokumentationen. Neben den Anforderungen werden z.B. auch Tests, Abnahmen oder Fehler in einer bestimmten Form dokumentiert. Die Einhaltung vorgegebener Standardformate ermöglicht allen Beteiligten die effiziente Bearbeitung der verschiedenen Aufgaben.
- *Festlegung des inneren Aufbaus von Anwendungen (Architektur):* Anwendungen werden im Laufe ihres Lebenszyklus von verschiedenen Personen bearbeitet. Die Nutzung eines standardisierten Aufbaus von Anwendungen ermöglicht ein schnelles Einarbeiten in die Strukturen und Funktionen der Anwendung.

Die Formulierung eines Standards sollte die folgenden Informationen enthalten:

- Beschreibung der Technologie, Plattform etc.
- Angaben zur konformen Nutzung des Standards (Richtlinien).
- Informationen zum Betreiben des Standards, z.B. was für die Nutzung des Standards an Installationen notwendig ist.
- Informationen zur Lizenzierung bei Technologien oder Plattformen.

3.5.1 Standard Driven Development

Die Einführung und Etablierung von Standards ist der Kern der ersten Stufe der Industrialisierung in der Anwendungsentwicklung.

Wesentlich für diese Stufe sind die Existenz sowie die Nutzung der im Unternehmen festgelegten Standards. Externe Standards können hierbei als Basis bei der Festlegung der internen Standards dienen. Ferner sollte jeder Entwickler das gleiche Verständnis bzgl. der Standards der verschiedenen Artefakte besitzen.

[49] Framework: Gruppe aufeinander abgestimmter Komponenten, die durch Anpassungen gemäß definierter Schnittstellen zu einer Anwendung entwickelt werden können.

Die Definition der Methode lautet daher:

> Ausrichtung der Implementierung von Anwendungen anhand der gültigen Standards im Unternehmen. Für die Erstellung von Artefakten existieren Standards, die bei der Entwicklung von Anwendungen einzuhalten sind.

Definition 5: Standard Driven Development

Mit der Standardisierung erfolgt der erste Schritt in Richtung Industrialisierung. Durch die Einführung dieser Methode werden die Prinzipien folgendermaßen unterstützt:

- *Standardisierung*: Durch die Festlegung von Standards bei Technologien, Plattformen, aber auch bei Architekturen von Anwendungen wird die Individualität bei der Realisierung reduziert. Die Transparenz bzgl. des Aufbaus sowie der Logik der Anwendung wird erhöht.
- *Wiederverwendung*: Die Wiederverwendung findet in Form der Nutzung standardisierter Technologien und Plattformen statt. Die wiederholte Nutzung von Standards fördert das Verständnis für den Aufbau der Lösungen und ermöglicht dadurch eine effizientere Wartung sowie Erweiterung.
- *Automatisierung*: Eine direkte Unterstützung der Automatisierung ist nicht gegeben. Die Standardisierung ist jedoch Voraussetzung für den Einsatz von Generatoren zum automatischen Erzeugen der benötigten Artefakte (siehe Abschnitt 3.7.2).
- *Arbeitsteilung*: Die Nutzung von standardisierten Plattformen unterstützt die Aufteilung der Aufgaben der Leistungserstellung von Lösungen sowie der Betreuung der zugehörigen Plattform.
- *Verringerung der Fertigungstiefe*: Der Einsatz von standardisierten Plattformen reduziert die Entwicklung von technischen Funktionalitäten, die diese Plattformen bereits durch den Hersteller zur Verfügung stellen.

Die Qualität der Anwendungen wird durch die Einführung der Standardisierung verbessert. Dies ergibt sich aus der Analyse der folgenden Faktoren:

- *Funktionalität*: Die Angemessenheit bzw. die Richtigkeit der geforderten Funktionalitäten wird durch die Standardisierung nicht direkt verbessert. Jedoch stellt diese Stufe Mechanismen in Form von standardisierten Tests zur Verfügung, die eine effiziente Überprüfung der Korrektheit ermöglichen. Durch den Einsatz von Standardplattformen wird das Risiko des unberechtigten Zugriffs gemindert und damit die Sicherheit erhöht. Die Plattformen enthalten mittlerweile Funktionalitäten zur Überprüfung

der Berechtigungen für den Zugriff auf Daten. Wird in den verschiedenen Anwendungen die von dem Standard vorgeschriebene Technologie verwendet, wird die Fähigkeit zur Kommunikation mit vorgegebenen Systemen erhöht. Durch die Bereitstellung von Richtlinien sowie die Verpflichtung zu deren Einhaltung bei der Entwicklung wird die Konformität zu den anwendungsspezifischen Normen erhöht.

- *Zuverlässigkeit*: Die Reduzierung der Vielfalt von Technologien und Plattformen auf Basis einer Standardisierung ermöglicht die Konzentration auf wenige Vorgaben für die Entwicklung von Anwendungen. Diese können effizienter an die internen sowie externen Entwickler kommuniziert werden. Ferner gestaltet sich die Überprüfung der Einhaltung dieser Vorgaben einfacher, da nur noch die definierten Technologien und Plattformen zugelassen werden. Die Existenz sowie die Einhaltung dieser Vorgaben reduziert die Fehleranfälligkeit der Anwendungen.
- *Benutzbarkeit*: Die Benutzbarkeit von Anwendungen hängt von der Gestaltung der Benutzerschnittstelle ab. Dies kann nur bedingt durch Entwicklungsmethoden verbessert werden. Jedoch kann die Vereinheitlichung der Nutzung von Anwendungen die Bedienbarkeit erhöhen und den Aufwand für das Erlernen reduzieren. Das SAP R/3-System enthält beispielsweise viele Anwendungen in unterschiedlichen fachlichen Bereichen. Durch die standardisierte Oberfläche sowie die Bedienung finden sich Benutzer auch in anderen Anwendungen des R/3-Systems schnell zurecht.
- *Effizienz*: Das Zeit- sowie das Verbrauchsverhalten einer Anwendung werden bestimmt durch die konkrete Realisierung der Anwendung. Die Standardisierung liefert für dieses Merkmal ebenfalls Vorgaben und Verfahren, die auf dieses Merkmal hinweisen und spezifische Tests einfordern.
- *Änderbarkeit*: Die Diagnose sowie die Korrektur von Fehlern einer Anwendung werden durch die Nutzung von Standards, z.B. anhand einer einheitlichen Architektur bzw. der Nutzung von Standardtechnologien, vereinfacht. Je vertrauter den Entwicklern die Strukturen einer Anwendung sind, desto effizienter kann eine Fehlerkorrektur erfolgen. Dies gilt auch für die Modifikation von Anwendungen. Da sich Standards schrittweise etablieren, existieren zunehmend Erfahrungswerte, die insbesondere auch die Stabilität von Anwendungen positiv beeinflussen.
- *Übertragbarkeit*: Die Festlegung von Standards ermöglicht die flexible Nutzung von Plattformen oder Technologien. Beispielsweise hat die Fest-

legung von HTML[50] als Beschreibungssprache für Texte und HTTP[51] als Transportprotokoll das Internet zu seiner vielseitigen Nutzung verholfen. Anwendungen, die auf dem Java-Standard aufbauen, können zudem ohne weitere Modifikation der Anwendungen auf unterschiedlichen Systemen eingesetzt werden.[52]

Auf die Faktoren Zeit und Kosten wirkt sich die Standardisierung ebenfalls positiv aus:

- *Entwicklungszeit*: Die Nutzung von Standards reduziert die Einarbeitung in neue Technologien und Plattformen. Die Realisierung auf Basis einer Standardarchitektur verkürzt die Konzeption einer Anwendung, da bereits nichtfunktionale Anforderungen durch den Standard definiert sind. Die Einarbeitung neuer Entwickler erfolgt bei der Nutzung von Standards schneller, da viele Konzepte bereits bekannt sind. Bei der verteilten Entwicklung von Anwendungen sind zudem weniger Abstimmungen notwendig.
- *Kosten*: Durch die Einführung von Standards und die damit verbundene Konsolidierung wird die Vielzahl an existierenden Produkten, Technologien, Werkzeugen etc. reduziert. Der Aufbau an Wissen beschränkt sich auf die definierten Standards, die bei vielen Entwicklungen eingesetzt werden.

  Die Konsolidierung bezieht sich ebenfalls auf die Bereitstellung der Entwicklungsinfrastruktur. Es entstehen nur noch die Kosten für die definierten Umgebungen. Eine Einsparung ergibt sich zudem aus der verkürzten Zeitspanne bei der Realisierung.

3.5.2 Erstellung von Artefakten auf Basis der Standardisierung

Die Entwicklung von Anwendungen orientiert sich in dieser Stufe an der Einhaltung von Standards. Die verschiedenen Artefakte werden manuell auf Basis der Vorgaben erstellt.

In Abbildung 15 ist schematisch die Anwendungsentwicklung als Transformation der Anforderungen in die verschiedenen Artefakte einer Anwendung entlang dem Entwicklungsprozess dargestellt.

---

[50] HTML – Hypertext Meta Language: Sprache zur Formatierung von Inhalten einer Webseite.

[51] HTTP – Hypertext Transfer Protocol: Protokoll zum Austausch von Daten zwischen einem Web Client und einem Web Server.

[52] Java-Anwendungen werden unabhängig von Systemen entwickelt, auf denen sie zum Einsatz kommen. Die systemspezifischen Aspekte (Dateizugriff etc.) werden durch eine Java-Laufzeitumgebung abgedeckt, die spezifisch für jedes System installiert wird.

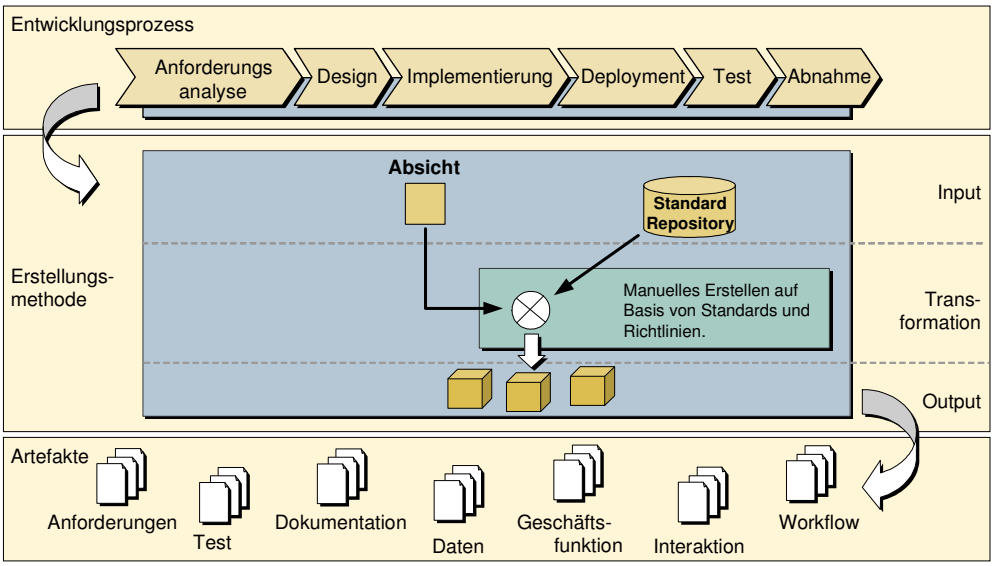

Abbildung 15: Standard Driven Development

Die Aufteilung einer Anwendung in die verschiedenen Schichten[53] ist bereits eine erste Vorgabe, die für alle weiteren Betrachtungen benötigt werden. Erst die Einführung dieses Standards ermöglicht die getrennte Analyse der zuvor eingeführten Artefakte.

Für die Entwicklungsmethode Standard Driven Development erfolgt die Erstellung von Artefakten nach dem folgenden Schema:

Input

Als Input für die Realisierung einer Anwendung werden die Anforderungen (Absicht) an die Anwendung sowie die verschiedenen Vorgaben wie Standards[54] und Richtlinien benötigt. Das Format der Anforderungen unterliegt dabei ebenfalls einer zentral vorgegebenen Struktur und Form.

Transformation

Die Transformation der Anforderungen erfolgt auf Basis der festgelegten Standards und Richtlinien. Diese sind für jeden Entwickler verfügbar. Das Erstellen der Artefakte für die verschiedenen Schichten der Anwendung erfolgt manuell. Das Vorgehen beim Erzeugen der Artefakte ist nicht näher bestimmt. Wesentlich ist lediglich die Einhaltung der Standards und Richtlinien.

Sofern die vorgegebenen Standards und Richtlinien Optionen für die Realisierung erlauben, kann sich der Entwickler im Kontext der Anwendung für eine Op-

[53] Siehe Abschnitt 2.3.2: Aufteilung der Artefakte nach verschiedenen Schichten.

[54] Standards werden über eine zentrale Ablage, dem Standard-Repository, zur Verfügung gestellt.

tion entscheiden. Beispielsweise sind in den Unternehmen meist mehrere Standards für die Gestaltung der Interaktionsschicht vorhanden.

Output

Das Ergebnis der Transformation sind die definierten Artefakte der Anwendungsarchitektur sowie des Entwicklungsumfeldes.

Aufbau und Struktur von Workflow, Interaktion, Geschäftsfunktion und Datenhaltung sind entsprechend den Richtlinien erstellt worden.

Die Artefakte im Entwicklungsumfeld dienen insbesondere der Kommunikation zwischen den verschiedenen Beteiligten der Anwendungsentwicklung. Dies sind neben dem Entwickler auch der Auftraggeber sowie der Projektleiter. Daher ist auch bei diesen Artefakten die vorgegebene Struktur wesentlich.

3.5.3 Rahmenbedingungen für die Standardisierung

Der Einsatz von Standards setzt voraus, dass diese entwickelt, akzeptiert und an die Entwickler kommuniziert wurden. Für die Analyse der Elemente der Anwendungsarchitektur sowie des Entwicklungsumfeldes sind daher verschiedene Voraussetzungen für die Definition und Etablierung von Standards nötig.

3.5.3.1 Gremium

Die Entwicklung von Anwendungen erfolgt in vielen Unternehmen in verschiedenen Abteilungen. Die Vorgehensweise unterscheidet sich dabei in der Regel anhand der Erfahrungen der jeweiligen Entwickler. Für die Etablierung von internen Standards ist daher ein abteilungs- oder themenübergreifendes Gremium notwendig. Dieses muss vom Management die Kompetenz erhalten, Standards zu definieren und einzufordern. Es ist zudem für die Kommunikation und die Bereitstellung der Standards zuständig. Folgende Themen sind durch das Gremium zu bearbeiten:

- *Standards:* Anfragen bzgl. der Standardisierung von Technologien, Plattformen etc. nimmt dieses Gremium entgegen und entscheidet, ob ein Standard entstehen soll. Das Gremium steuert den Prozess der Standardisierung.
- *Referenzarchitekturen:* Der grundsätzliche Aufbau (Referenz) von Anwendungen sowie die Prinzipien, die bei der Entwicklung zu beachten sind, wird über dieses Gremium festgelegt.
- *Entwicklungs- und Laufzeitinfrastrukturen:* Welche Infrastrukturen im Unternehmen aufgebaut und genutzt werden, ist durch ein Gremium festzulegen. Dies beinhaltet auch die Festlegung der Elemente der Infrastrukturen, wie z.B. ein Konfigurationsmanagement oder die Werkzeuge für die Anwendungsentwicklung.

3.5.3.2 Prozesse

Für die Akzeptanz von Standards bei Entwicklern sowie die Anwendungsverantwortlichen ist die Transparenz bzgl. der Erstellung dieser Standards zu gewährleisten. Der Prozess zur Definition sowie zur Änderung von Standards muss für alle Beteiligten nachvollziehbar sein. Wer welchen Standard aus welchem Grund eingebracht hat, sollte einfach zu ermitteln sein. Folgende Prozesse sind im Unternehmen zu etablieren:

- *Evaluierung:* Neue Technologien und Plattformen werden über diesen Prozess näher betrachtet und bzgl. der Eignung für Anforderungen aus dem Unternehmen analysiert. Dieser Prozess geht einer Standardisierung voraus.
- *Standardisierung:* Plattformen, Technologien sowie Infrastrukturen, die für den breiten Einsatz in einem Unternehmen betrachtet werden, sind über einen transparenten Prozess zu standardisieren. Hierbei werden die Richtlinien zur Nutzung des Standards sowie Voraussetzungen für den Einsatz beschrieben. Ferner werden diese Informationen allen an der Entwicklung beteiligten Mitarbeitern zur Verfügung gestellt. Der Prozess der Standardisierung schließt auch die Festlegung der Entwicklungs- und Laufzeitinfrastruktur mit ein.

3.5.3.3 Entwicklungsinfrastruktur

Die Standardisierung der Entwicklungsinfrastruktur ist eine wesentliche Voraussetzung für diesen Grad der Industrialisierung. Die Festlegung der Elemente der Entwicklungsinfrastruktur sowie die durchgängige Nutzung dieser Elemente bei der Anwendungsentwicklung sind ein wesentlicher Beitrag zur Erreichung der Ziele dieser Stufe. Die verschiedenen Technologien und Plattformen erfordern teilweise unterschiedliche Entwicklungsinfrastrukturen. Für jede dieser Infrastrukturen sind jedoch die folgenden Elemente festzulegen:

- *Standards und Richtlinien:* Die Standards im Rahmen der Entwicklungsinfrastruktur beziehen sich auf Festlegungen von Technologien und Plattformen, die bei der Realisierung von Anwendungen eingesetzt werden können. Ferner sind darunter die Vorgaben bei der Programmierung sowie dem Aufbau einer Anwendung (Referenzarchitektur) zu finden.

  Für die Einhaltung von Standards müssen diese zunächst bei allen Entwicklern bekannt und verfügbar sein. Dies kann z.B. über eine zentrale Ablage erfolgen. Sie sollten regelmäßig aktualisiert und kommuniziert werden.

- *Entwicklungswerkzeuge:* Für die Erstellung der Artefakte sind die Werkzeuge für die betrachtete Entwicklungsinfrastruktur zu bestimmen. Dies ermöglicht die reibungslose Zusammenarbeit zwischen verschiedenen

Entwicklern. Ferner werden Eigenheiten spezieller Werkzeuge, die Probleme bei der produktiven Nutzung der Anwendung verursachen können, frühzeitig unterbunden.

- *Konfigurationsmanagement:* Anwendungen werden über ihren Lebenszyklus mehrfach erweitert und angepasst. Somit existieren mehrere Versionen dieser Anwendung. Aber auch die jeweils zugehörigen Anforderungen, Tests und Dokumentationen sind konsistent zu halten. Diese Elemente sind daher über ein Konfigurationsmanagement zu verwalten. Wesentlicher Aspekt ist hierbei die Verwaltung von Quelltexten.
- *Zentraler Build Service*[55]: Insbesondere bei der verteilten Entwicklung von Anwendungen ist es wichtig, die spezifischen Einstellungen von Entwicklern auszuschließen. Ein zentraler Build ermöglicht das unabhängige Erstellen einer Anwendung aus den entwickelten Quelltexten. Dieser Service ist daher in jeder Entwicklungsumgebung zentral zur Verfügung zu stellen.[56] Die Art des Builds wird durch die eingesetzte Technologie bestimmt.
- *Transport und Deployment:* Für eine eindeutige Nachvollziehbarkeit der Entwicklungen im Lebenszyklus einer Anwendung ist der geordnete Transport einer Anwendung einzurichten. Dieser erfolgt von einem Entwicklungssystem über ein Testsystem in ein produktives System. Damit kann jederzeit festgestellt werden, welcher Mitarbeiter wann welche Änderungen auf dem produktiven System eingespielt hat. Dies ist insbesondere für eine potentielle Fehlersuche relevant.

3.5.3.4 Laufzeitinfrastruktur

Für die Laufzeitinfrastruktur sind die möglichen Plattformen und Systeme festzulegen, auf welchen eine Anwendung zum Einsatz kommen kann. Ferner werden Dienste der Infrastruktur festgelegt, auf die Anwendungen zur Laufzeit zugreifen können. Oft wird beispielsweise in Unternehmen eine zentrale Plattform für die Verwaltung von Anwenderdaten, wie Kennung und Passwort, zur Verfügung gestellt. Hierdurch wird vermieden, dass für jede Anwendung ein eigenes Passwort benötigt wird.

Der Aufbau einer Laufzeitinfrastruktur hängt vom Bedarf des Unternehmens ab. Wesentlich ist die Bereitstellung einer produktiven Umgebung, in der die Anwen-

---

[55] Siehe Abschnitt 2.3.4.

[56] Es gibt Technologien und Plattformen, bei denen dieser Service implizit vorhanden ist, wie z.B. dem SAP R/3-System. Der Service ist hier nicht explizit zur Verfügung zu stellen.

dung zum Einsatz kommt. In welcher Ausprägung die Laufzeitinfrastruktur in dieser Stufe gestaltet wird, wird durch das Gremium aus Abschnitt 3.5.3.1 bestimmt. Es werden somit an dieser Stelle keine weiteren Voraussetzungen definiert.

3.5.4 Reifegrade der Standardisierung

Die Stufe Standardisierung ist erreicht, wenn die Methode eingeführt ist sowie die Rahmenbedingungen erfüllt sind. Welcher Reifegrad bzgl. der Standardisierung im Unternehmen vorliegt, wird anhand der folgenden Kriterien ermittelt:

- *Initial:* Standards sind im Unternehmen nicht vorhanden. Die Entwicklungen finden individuell ohne Vorgabe statt. Es gibt keine Gremien oder Prozesse, die Standards festlegen. Die Entwicklungs- und die Laufzeitinfrastruktur werden durch den Entwickler bestimmt.
- *Wiederholbar:* Im Rahmen von Projekten oder Abteilungen werden erste Standards für die Realisierung von Anwendungen festgelegt und angewandt. Diese stellen jedoch noch keine Vorgabe für das ganze Unternehmen dar. Zudem wird nur ein Teilbereich für die Festlegung von Standards betrachtet. Die Durchgängigkeit und Aktualität der Artefakte bzgl. der Vorgaben ist nicht gewährleistet.
- *Definiert:* Es existieren ein zentrales Gremium sowie Prozesse zur Festlegung von Standards. Die Standards für die Entwicklung von Anwendungen, inklusive der Entwicklungs- und Laufzeitinfrastruktur, sind vollständig vorhanden. Erste Anwendungen wurden auf Basis der Standards realisiert. Jedoch gelten die Standards noch nicht als verbindlich für jede Anwendung. Die Aktualität der Artefakte bzgl. der Vorgaben, aber auch die Durchgängigkeit in den verschiedenen Anwendungen ist nicht gewährleistet.
- *Eingeführt:* Jede Anwendungsentwicklung im Unternehmen findet auf Basis der durch ein zentrales Gremium definierten Standards statt. Die Artefakte werden durchgängig anhand der vorhandenen Standards implementiert. Die Aktualität der verschiedenen Artefakte ist gewährleistet.
- *Optimiert:* Aus den verschiedenen Projekten der Anwendungsentwicklung werden systematisch Rückmeldungen zu Standards erfasst und durch ein zentrales Gremium ausgewertet. Sofern eine Rückmeldung den bestehenden Standard sinnvoll erweitert oder optimiert, wird sie in den Standard aufgenommen. Dies ermöglicht eine kontinuierliche Verbesserung der definierten Standards.

3.5.5 Grenzen und Potentiale für Verbesserungen

Die Standardisierung ist die erste wichtige Stufe bei der Industrialisierung der Anwendungsentwicklung. Der Schwerpunkt liegt dabei auf der Vereinheitlichung von Artefakten sowie der unterstützenden Elemente. In dieser Stufe erfolgt die Realisierung der Anwendung jedoch noch vollständig manuell.

Durch die Festlegung auf eine Architektur aus mehreren Schichten wird in dieser Stufe bereits eine grundsätzliche Aufteilung der Anwendung gefordert. Die Gestaltung der Strukturen innerhalb der Schichten bleibt jedoch dem Entwickler überlassen. Sofern gleiche Funktionalitäten in mehreren Anwendungen gefordert sind, werden diese auch mehrfach entwickelt. Die einmalige Realisierung einer Funktion, welche in mehreren Anwendungen eingesetzt werden kann, ist noch nicht vorgesehen.

Hierzu sind weitere Strukturen sowie eine Verwaltung notwendig, die in der nächsten Stufe beschrieben werden.

3.6 Stufe 3: Wiederverwendung

Neben der Standardisierung ist die Arbeitsteilung ein zentrales Element der Industrialisierung [GrLO07]. Wesentliches Konzept der Arbeitsteilung ist die Zergliederung der unterschiedlichen Tätigkeiten für die Herstellung von Gütern. Zunächst wurden hierbei die planerischen (Konstruktion von Gütern) von den durchführenden Arbeiten (Produktion von Gütern) getrennt [Kilg86]. Auch bei den Arbeiten zur Produktion von Gütern wurde, z.B. durch Taylor, die Arbeitsteilung weiter verfeinert [Wede05].

Analog erfolgte in der Softwarebranche eine Arbeitsteilung in die Spezifikation sowie in die Entwicklung einer Anwendung [GrLO07]. Diese Aufteilung der Aufgaben förderte eine Spezialisierung, die unterschiedliche Kenntnisse voraussetzt. Nach [GrLO07] erfolgt mit der Komponentenorientierung eine weitere Arbeitsteilung, indem gleichwertige Teile (Komponenten) eines Ganzen (Anwendung) getrennt voneinander realisiert werden.

Eine solche Strukturierung ermöglicht die Entwicklung von Komponenten, die in mehreren Anwendungen wiederverwendet werden können. Im Gegensatz zu vielen anderen Branchen unterscheidet sich jedoch die Softwarebranche unter anderem durch eine geringe Nutzung von Halbfabrikaten bzw. vorgefertigten Komponenten. Die meisten Produkte werden neu konzipiert und sind daher mit wiederkehrenden, ähnlichen Qualitätsproblemen behaftet ([PoRS00], [Lim94]). Eine Steigerung der Produktivität und der Qualität sowie eine Verkürzung der Entwicklungszeit können demnach durch die systematische Wiederverwendung von Softwarekomponenten erfolgen.

Die bei der Standardisierung noch nicht geforderte Wiederverwendung von Lösungen gemeinsamer Teilprobleme liefert das Potential für die Erhöhung der Industrialisierung in dieser Stufe. Als Vorteile der Wiederverwendung werden in [Andr04] die folgenden Punkte genannt:

- *Beschleunigung des Time-to-Market:* Durch die Nutzung bestehender Komponenten kann die darin enthaltene Funktionalität ohne weitere Entwicklung verwendet werden. Die Zeit für die Fertigstellung der Anwendung wird dadurch verkürzt.
- *Reduzierung der Fehlerquote:* Die Programmierung von Anforderungen ist durch den Einsatz bereits getesteter Komponenten nicht mehr nötig. Potentielle Fehlerquellen aufgrund einer manuellen Entwicklung werden vermieden.
- *Reduzierung der Wartungskosten:* Wiederverwendung ermöglicht die zentrale Pflege von Teilen (Komponenten) der Anwendung. Änderungen an zentralen Komponenten sind somit nicht in jeder Anwendung separat durchzuführen.
- *Reduzierung der Gesamtkosten:* Der Größenvorteil (Vorteil der Massenproduktion), der durch die Nutzung mehrerer identischer Instanzen für die Lösung des gleichen Problems entsteht, reduziert die Kosten für die Einzellösung.

In [EzMT98] wird Wiederverwendung von Software wie folgt definiert:

> „Software reuse is the systematic practise of developing software from a stock of building blocks, so that similarities in requirements and/or architecture between applications can be exploited to achieve substantial benefit in productivity and business performance.“

Definition 6: Wiederverwendung

Die Nutzung der Wiederverwendung beschränkt sich nicht auf die Erstellung von Software. Vielmehr können auch die Artefakte der unterstützenden Elemente, wie z.B. Anforderungen oder Dokumentation, so strukturiert werden, dass sie in verschiedenen Anwendungen einsetzbar sind oder zumindest als Vorlage dienen.

3.6.1 Komponentenorientierung

Die Basis der Wiederverwendung in Definition 6 sind die „building blocks“, auch als Komponenten bezeichnet. In Analogie zur Automobilindustrie besteht das Ziel der komponentenorientierten Entwicklung in der Erstellung neuer Anwen-

dungen durch die Kombination bestehender Komponenten [PfWi07]. Sie stehen dabei über einen Katalog zur Verfügung und werden so entworfen, dass sie in verschiedenen potentiellen Anwendungen wiederverwendet werden können [GrLO07].

Im Bereich der Softwareentwicklung wird in der Literatur die Komponente unterschiedlich definiert. In dieser Arbeit wird die Definition nach [Andr04] verwendet:

> „Eine Softwarekomponente ist ein eigenständiges Artefakt eines Softwaresystems, welches über spezifisches Wissen verfügt und gemäß ihrer Spezifikation über eine oder mehrere Schnittstellen mit anderen Softwarekomponenten und -systemen kommunizieren kann. Das Wissen einer Softwarekomponente repräsentiert ein Konzept eines Geschäftsfeldes. Eine Komponente kann verpackt und unter Berücksichtigung eines Komponentenmodells als autonome, wiederverwendbare Einheit verteilt werden."

Definition 7: Softwarekomponente

Nach [GrSh06] liegt die Kernidee der Softwarekomponente in der Trennung von Spezifikation und Implementierung. Die Spezifikation beschreibt dabei, welche Funktionalität die Komponente zur Verfügung stellt und wie sie sich durch die Nutzung von Anwendungen verhält. Die Kenntnis über die (innere) Implementierung ist dabei nicht notwendig.

Die Aufteilung von Anwendungen anhand von Komponenten hat eine Reihe von Vorteilen. Sie ermöglicht die Reduzierung der Komplexität, indem durch klare Einheiten, die Komponenten, nach dem Prinzip „Teile und Herrsche" Funktionalitäten entwickelt werden können. Dieses Prinzip sollte grundsätzlich auch in der Stufe Standarisierung genutzt werden, ist dort jedoch noch nicht gefordert.

In [Andr04] werden die folgenden zusätzlichen Vorteile von Komponenten genannt:

Komponenten

- trennen die Zuständigkeiten (vgl. Arbeitsteilung)
- sind einfach einsetzbar und kombinierbar
- sind einfach wiederverwendbar
- fördern eine schnelle Anwendungsentwicklung
- sind einfach austauschbar

Die Nutzung von Komponenten setzt ein Rahmenwerk, eine Komponenten-Architektur, voraus. Sie sind dabei in einer bestimmten Form zu konzipieren, um in der betrachteten Architektur einsetzbar zu sein. Im Bereich Java benötigt z.B.

eine EJB-Komponente[57] ein Architekturumfeld, das gemäß der JEE-Spezifikation[58] aufgebaut ist.

Für den Einsatz von Komponenten ist somit eine geeignete Architektur notwendig, die nach [Andr04] die folgenden Voraussetzungen zu erfüllen hat:

- Die Kommunikation zwischen den Komponenten ist auf Systemebene innerhalb eines Unternehmens als auch unternehmensübergreifend möglich.
- Die Komponenten können in flexiblen Schichtenarchitekturen angeordnet werden.
- Eine einfache Wiederverwendung von Komponenten wird ermöglicht.
- Änderungen und Erweiterungen von Komponenten sind auf einfache Art möglich.
- Zuständigkeiten sind klar definiert.
- Bestehende Komponenten können einfach integriert werden.

In der Praxis haben sich verschiedene Architekturen etabliert, zu denen unter anderem die folgenden Beispiele gehören:

- JEE - Java Enterprise Edition ([Bien07])
- CORBA - Common Object Request Broker Architecture ([OrHE98])
- Microsoft .Net ([BBMP06])

Die Granularität von Komponenten spielt eine wichtige Rolle im Rahmen der Wiederverwendung [PfWi07]. Sind die Komponenten zu groß, besteht die Gefahr, dass der implementierte Kontext nur in sehr wenigen Anwendungen einsetzbar ist. Sind sie zu klein, könnte der Aufwand für die Pflege und die Koordination dieser Komponenten die Vorteile aufheben. Eine allgemeine Richtlinie zur Bestimmung der optimalen Granularität existiert jedoch nicht. Dies ist sehr vom Unternehmen sowie dem jeweiligen Einsatz abhängig.

Komponenten können einen technischen, aber auch einen fachlichen Kontext aufweisen. Application Server[59], z.B. auf Basis der JEE-Spezifikation, bieten Komponenten zur Wiederverwendung mit einem rein technischen Kontext an. Hierzu gehören z.B. der Zugriff auf Datenbanken, die Verwaltung von Transakti-

[57] EJB (Enterprise Java Bean). Komponentenmodell in der Java-Technologie für die Realisierung von Geschäftsfunktionen (vgl. [Bien07]).

[58] JEE (Java Platform, Enterprise Edition): Spezifikation einer Softwarearchitektur für die transaktionsbasierte Ausführung von in Java programmierten Anwendungen (vgl. [Bien07]).

[59] Software, die als Ablaufumgebung für Anwendungen spezielle Dienste zur Verfügung stellt (Transaktionen, Zugriff auf Datenbanken etc.)

onen oder die Authentifizierung[60] von Anwendern. Im Fall der Authentifizierung hat die Anwendung lediglich zu signalisieren, dass eine Anmeldung des Anwenders benötigt wird. Die Durchführung der Anmeldung erfolgt vollständig durch den zugrunde liegenden Application Server.

In Abschnitt 2.3.2.1 wurde die Architektur einer Anwendung in verschiedene horizontale Schichten strukturiert. Dies dient der Trennung von Zuständigkeiten, wie z.B. der Darstellung von Daten auf der Benutzeroberfläche oder dem Bereitstellen von Daten durch Zugriff auf eine Datenbank.

Für jede dieser Schichten existieren auf dem Softwaremarkt technische Komponenten, die, unabhängig vom fachlichen Kontext, in mehreren Anwendungen[61] einsetzbar sind. Als Beispiel sei eine im Java-Umfeld für die Interaktionsschicht häufig genutzte Komponente, das Struts-Framework[62], erwähnt. Es basiert auf dem MVC-Pattern[63], bei dem der Entwickler lediglich die Benutzeroberfläche sowie den Zugriff auf die Schicht der Geschäftsfunktionen zu implementieren hat. Die Koordination, wann welcher Teil der Oberfläche auf Basis welcher Aktion des Anwenders dargestellt werden soll, übernimmt das Framework. Solche Komponenten existieren ebenfalls in anderen Technologien, wie z.B. dem .Net Framework.

Gleichen sich fachliche Anforderungen in verschiedenen Anwendungen, liegt auch hier die Überlegung nahe, diese Funktionalität in eine Komponente auszulagern und den verschiedenen Anwendungen zur Verfügung zu stellen.

Eine wiederkehrende Funktionalität ist beispielsweise die Bereitstellung von Personendaten, wie z.B. Telefonnummer oder E-Mail-Adresse. Die Besonderheit bei diesem Beispiel ist die allgemeine Nutzbarkeit dieser Komponente. Sie hat zwar einen fachlichen Kontext, sie kann jedoch in sehr unterschiedlichen Szenarien, wie z.B. im Vertrieb oder in der Produktion, eingesetzt werden.

Die Entscheidung für eine bestimmte Technologie zur Realisierung von fachlichen Komponenten schränkt dessen Nutzung in verschiedenen Anwendungen ein. Wird die Ermittlung der Personendaten als EJB-Komponente im Java-Umfeld realisiert, ist die Nutzung in Anwendungen auf Basis von Microsoft .Net nicht möglich. Dieser Aspekt ist bei der Planung der Wiederverwendung zu berücksichtigen.

---

[60] Überprüfung der Identität eines Anwenders z.B. auf Basis von Benutzername und Passwort.

[61] Anwendungen im gleichen technischen Kontext. Java-Komponenten können vorwiegend in Java-Anwendungen zum Einsatz kommen.

[62] Open Source Projekt der Apache Software Foundation (http://struts.apache.org/).

[63] MVC-Pattern (Model-View-Controler): Architekturmuster zur Strukturierung der Softwareentwicklung in der Oberflächenprogrammierung.

3.6.2 Arten der Wiederverwendung

Die Wiederverwendung von Komponenten kann nach unterschiedlichen Kriterien erfolgen. In [Andr04] werden die folgenden Arten differenziert:

- *Black Box Reuse*: nur das Äußere (die Schnittstellen) der Komponente ist sichtbar. Die Logik ergibt sich aus der Beschreibung der Schnittstellen. Der Vorteil liegt in der Änderbarkeit der inneren Logik, die keine Auswirkung auf die Nutzung hat. Die Komponente kann nicht modifiziert werden.
- *Glass Box Reuse*: Der innere Aufbau ist sichtbar. Dadurch kann ein Verständnis für die innere Logik aufgebaut werden, die für die Entwicklung hilfreich sein kann. Es sollte jedoch vermieden werden, sich von dieser Logik abhängig zu machen. Die Komponente kann nicht modifiziert werden.
- *White Box Reuse*: Bei Komponenten dieser Art ist ebenfalls der innere Aufbau bekannt. Zudem können sowohl Schnittstellen als auch die innere Logik verändert werden.

Welche Art der Wiederverwendung zum Einsatz kommt, ist wesentlich von den Architekturprinzipien des Unternehmens sowie dem spezifischen Kontext der Anwendung abhängig.

3.6.3 Systematische Wiederverwendung

Die Strukturierung einer Anwendung in verschiedene Komponenten ist für die Sicherstellung einer häufigen Wiederverwendung dieser Komponenten nicht ausreichend. Um die beschriebenen Vorteile der Wiederverwendung zu realisieren, bedarf es eines systematischen Ansatzes. Dieser hat für verschiedene Bereiche die gemeinsamen Unterprobleme zu identifizieren und eine einheitliche Sammlung von Artefakten hervorzubringen [GrSh06].

Wiederverwendung ist somit nicht nur rein technisch zu betrachten. Vielmehr muss die Mehrfachnutzung von Komponenten systematisch geplant und durchgeführt werden. Die Systematik ist nach [PoRS00] wesentliche Voraussetzung für die Steigerung der Produktivität, der Qualität und der Verkürzung der Entwicklungszeit.

Die systematische Wiederverwendung wird nach [GrSh06] über sogenannte Programmfamilien ermöglicht. Diese Familien stellen jeweils einen Kontext zur Verfügung, in dem die Probleme, die bei allen Anwendungen der Familie gleich sind, einmalig gelöst und damit mehrfach genutzt werden können [Parn76].

In [MeAp07] werden solche Familien, die explizit für die Wiederverwendung gemeinsamer Artefakte entworfen werden, als Software-Produktlinien bezeichnet. Sie reduzieren die Kosten für Entwicklung und Wartung und verbessern die Zuverlässigkeit der Anwendungen. Dies begründet sich aus einem stabilen Kern an

Funktionen, der in verschiedenen Anwendungen zum Einsatz kommt und dadurch in unterschiedlichen Konstellationen getestet wurde. Diese Funktionen werden somit explizit für die Wiederverwendung realisiert.

Bei der Entwicklung von Anwendungen auf Basis von Software-Produktlinien wird zwischen einer Domänenentwicklung und der eigentlichen Anwendungsentwicklung unterschieden [Kutt07] (siehe Abbildung 16).

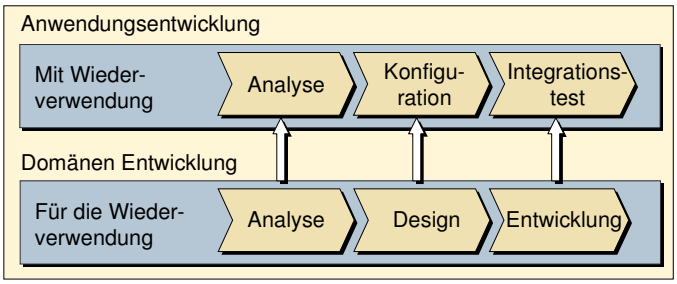

Abbildung 16: Systematische Wiederverwendung

Die Domänenentwicklung vereint alle Artefakte, welche in den verschiedenen Anwendungen der Familie wiederverwendet werden können. Die Auswahl dieser Artefakte erfolgt in der Domänenanalyse. Dabei werden die Gemeinsamkeiten und Unterschiede der zu realisierenden Anwendungen untersucht. Über das Design und die anschließende Entwicklung in diesem Bereich werden die wiederverwendbaren Komponenten zur Verfügung gestellt.

Eine Domäne charakterisiert hierbei ein Themenfeld oder einen Problembereich mit gemeinsamen Merkmalen [GrSH06]. Sie kann grundsätzlich mit der Beschreibung der Domäne in Abschnitt 2.6 verglichen werden. Die Abstraktionsebene kann jedoch variieren. Während in Abschnitt 2.6 eine Einteilung auf Unternehmensebene erfolgt, kann eine Domäne im Rahmen der Software-Produktlinie spezifischer für die konkrete Umsetzung von Anforderungen ausgelegt sein. Wie diese Domänen verwendet werden, ist durch das jeweilige Unternehmen festzulegen und zu kommunizieren.

In der Anwendungsentwicklung der Software-Produktlinie wird die Anwendung auf Basis der wiederverwendbaren Komponenten der Domäne implementiert [Kutt07]. Hierbei wird in einem ersten Schritt analysiert, welche der Anforderungen an die Anwendung durch die Domäneimplementierung bereits abgedeckt sind. Für die bereits nutzbaren Funktionalitäten werden die Freiheitsgrade der vorhandenen Domänenimplementierung gebunden, d.h. bzgl. der Anforderungen konfiguriert. Für die restlichen Anforderungen ist die Funktionalität zu entwickeln. Im letzten Schritt wird das System getestet.

3.6.4 Reuse Driven Development

Die wesentliche Steigerung der Industrialisierung bei der Anwendungsentwicklung liegt in dieser Stufe auf der Wiederverwendung von Lösungen gemeinsamer Problemstellungen. Die folgende Methode wird für diese Stufe festgelegt:

> Die Entwicklung von Anwendungen basiert auf der Wiederverwendung von zentral koordinierten Komponenten. Der Aufbau der Anwendungen selbst folgt dabei den Prinzipien der Komponentenorientierung.

Definition 8: Reuse Driven Development

Neben technischen können auch fachliche Komponenten wiederverwendet werden. Sie werden Bestandteil der zu realisierenden Anwendung. Änderungen an der originären Komponente haben zunächst keine Auswirkung auf die nutzende Anwendung. Hierzu ist die Komponente erneut einzubinden und eventuelle Änderungen sind zu berücksichtigen.

Die Standardisierung ist eine wesentliche Voraussetzung für die Wiederverwendung. Nur die Verständigung bei den Technologien, Plattformen etc. auf einen Standard ermöglicht die Einbindung und damit die Nutzung zentral organisierter Komponenten.

Durch die Einführung dieser Methode werden die verschiedenen Prinzipien der Industrialisierung folgendermaßen unterstützt:

- *Standardisierung*: Die zentrale Verfügbarkeit von Komponenten ermöglicht eine Standardisierung von implementierten Funktionalitäten. Dies erweitert den Bereich der Standardisierung.
- *Wiederverwendung*: Die Wiederverwendung von Komponenten ist der zentrale Fokus dieser Stufe. Die allgemeine Verfügbarkeit von Komponenten ermöglicht den Einsatz einer einmalig implementierten Funktionalität in mehreren Anwendungen.
- *Automatisierung*: Eine Automatisierung erfolgt noch nicht, da die Anwendung noch manuell erstellt wird. Der komponentenorientierte Aufbau ist jedoch eine wesentliche Voraussetzung für die Automatisierung.
- *Arbeitsteilung*: Die Strukturierung von Anwendungen in Komponenten ermöglicht die Arbeitsteilung bei der Realisierung. So kann beispielsweise die Interaktionsschicht von Anwendungen durch Designspezialisten erstellt werden, während die Geschäftsfunktionen durch Prozessspezialisten bearbeitet werden. Ferner wird ein Teil der Anwendung, die wiederverwendbaren Komponenten, durch eine zentrale Instanz gepflegt.

- *Verringerung der Fertigungstiefe*: Diese Methode ermöglicht die Fertigung von Komponenten außerhalb der Firma. Sofern sich diese Komponenten an die Vorgaben des Unternehmens halten, können sie von Softwarelieferanten zugekauft werden.

Der Einsatz der Wiederverwendung von Komponenten in der Anwendungsentwicklung steigert die Produktivität der Anwendungsentwicklung. Im Einzelnen hat die Wiederverwendung auf die verschiedenen Qualitätsfaktoren folgenden Einfluss:

- *Funktionalität*: Wird die benötigte Funktionalität durch Wiederverwendung bereits existierender Komponenten bereitgestellt, können die Angemessenheit sowie die Richtigkeit bereits im Vorfeld der Entwicklung überprüft werden. Die Vorgaben für die Einhaltung von Sicherheitskriterien für die betrachtete Funktionalität werden zentral implementiert und getestet. Fehler bei der Implementierung in den verschiedenen Anwendungen werden dadurch vermieden. Die Kommunikation zwischen den Komponenten ist eine wichtige Grundlage für die Funktionalität der Anwendung. Dies schließt die Kommunikation mit vorgegebenen Systemen mit ein.
- *Zuverlässigkeit*: Für die wiederverwendeten Komponenten einer Anwendung gibt es bereits Aussagen zur Reife und Fehlertoleranz. Diese werden an zentraler Stelle optimiert. Die Analyse der Zuverlässigkeit reduziert sich daher auf die für die Anwendung spezifischen Entwicklungen sowie die Einbindung der Komponenten.
- *Benutzbarkeit*: Werden Komponenten in der Benutzerschnittstelle einer Anwendung wiederverwendet, so sind dem Anwender die Konzepte der Interaktion mit dieser Komponente bereits bekannt. Hierdurch erhöht sich die Bedienbarkeit, und der Aufwand für das Erlernen der Anwendung wird reduziert.
- *Effizienz*: Das Zeit- und das Verbrauchsverhalten der wiederverwendbaren Komponenten werden an zentraler Stelle optimiert. Der Entwickler hat sich daher lediglich um die für die Anwendung spezifischen Entwicklungen zu kümmern.
- *Änderbarkeit:* Die Nutzung von Komponenten fordert einen klar strukturierten Aufbau von Anwendungen sowie eindeutige Schnittstellen für den Aufruf der Komponenten. Aufgrund dieser klaren Struktur können Fehlerquellen leichter lokalisiert werden. Ferner ist die Einarbeitung von neuen Mitarbeitern einfacher. Die Wartung der Komponenten erfolgt an einer zentralen Stelle. Sofern die Komponente mehrfach in Anwendungen zum Einsatz gekommen ist, wurde dort bereits die Stabilität betrachtet.

- *Übertragbarkeit:* Aufgrund einer klaren Struktur der Anwendung sowie der definierten Schnittstelle der Komponenten können Teile der Anwendung ohne größeren Aufwand durch neue Komponenten ausgetauscht werden. So kann flexibel auf Funktionsänderungen, aber auch Umgebungsänderungen reagiert werden.

Auf die Faktoren Zeit und Kosten hat die Wiederverwendung folgenden Einfluss:

- *Entwicklungszeit:* Die Nutzung bereits vorhandener Komponenten reduziert den zeitlichen Aufwand für die Implementierung der geforderten Funktionalität. Neben der reinen Erstellung der Komponenten reduziert[64] sich auch der Testaufwand sowie der Aufwand für die Beseitigung von Fehlern.
- *Kosten*: Durch den Einsatz existierender Komponenten werden die Kosten für die Erstellung der gleichen Funktionalität eingespart.

  Der Aufwand für die Wartung der betrachteten Funktionalität konzentriert sich an einer zentralen Stelle und ist nicht in jeder Anwendung nötig.

3.6.5 Erstellung von Artefakten auf Basis der Wiederverwendung

Die Implementierung von Anwendungen in dieser Stufe erfolgt nicht ausschließlich durch Wiederverwendung. In der Designphase der Realisierung werden die geforderten Funktionalitäten in einzelne Komponenten aufgeteilt.

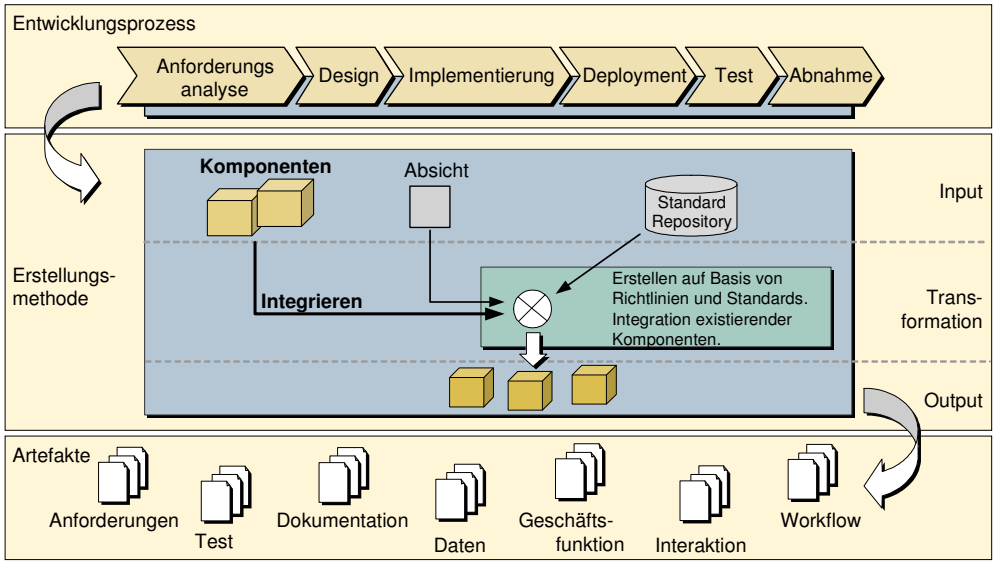

Abbildung 17: Reuse Driven Development

[64] Der Aufwand für Tests entfällt nicht ganz, da die Einbindung der Komponente ebenfalls getestet werden muss. Ferner ist zu überprüfen, ob die Komponente den geforderten Funktionalitäten entspricht.

Anschließend wird die Verfügbarkeit bereits existierender Komponenten ermittelt, die die geforderten Funktionalitäten abdecken. Diese können sowohl technischer wie fachlicher Herkunft sein. Wesentlich ist dabei die Einbindung von Instanzen dieser Komponenten in der zu realisierenden Anwendung. Abbildung 17 stellt die Methode schematisch dar.

Komponenten werden sowohl als ausführbare Einheiten (Black Box Reuse) als auch als Quelltext (Whitebox Reuse) zur Verfügung gestellt. Für die Entwicklungsmethode Reuse Driven Development erfolgt die Erstellung von Artefakten nach dem folgenden Schema:

Input

Zusätzlich zu den Anforderungen an die Anwendung sowie die Standards und Richtlinien aus der Stufe der Standardisierung werden dem Entwickler bereits existierende Komponenten zur Verfügung gestellt. Die Vorgaben (Richtlinien) bzgl. der Strukturierung von Anwendungen wurden um Komponentenorientierung erweitert. Die Komponenten sowie die zugehörige Beschreibung sind zentral für den Entwickler verfügbar.

Transformation

Der Entwickler analysiert die Anforderungen und erstellt ein Design der Anwendung unter Beachtung eines komponentenorientierten Aufbaus. Für die technischen und fachlichen Funktionalitäten wird überprüft, welche der bereits existierenden Komponenten in der Anwendung wiederverwendet werden können.

Wie bei der Standardisierung erfolgt in dieser Stufe die Implementierung der Anwendung manuell auf Basis der existierenden Standards. Jedoch werden nur die Teile der Anwendung programmiert, die nicht durch bereits existierende Komponenten abgedeckt werden können. Die manuell erstellten Teile werden anschließend mit den vorhandenen Komponenten in einer Anwendung integriert.

Dieses Vorgehen erfolgt auch für die Artefakte des Entwicklungsumfeldes. Die Testfälle der Komponenten, die Dokumentation sowie die Anforderungen werden als Bestandteile aufgenommen und manuell eingearbeitet.

Output

Das Ergebnis der Transformation ist eine Anwendung, die aus bereits existierenden Komponenten sowie Eigenentwicklung besteht. Dieser Output kann selbst wieder als Komponente gestaltet werden, die in anderen Anwendungen zum Einsatz kommen kann.

Neben den Softwareartefakten werden in dieser Stufe auch die Artefakte des Entwicklungsumfeldes wiederverwendet. Die Testfälle, die Anforderungen sowie die Beschreibungen der eingesetzten Komponenten werden hierfür genutzt.

3.6.6 Rahmenbedingungen für die Wiederverwendung

Der Einsatz sowie die Nutzung von Komponenten setzen verschiedene Rahmenbedingungen voraus. Erst die zentrale Organisation von Komponenten ermöglicht deren Verfügbarkeit für die verschiedenen Anwendungen. Es ist somit zu klären, wer im Unternehmen ein entsprechendes Portfolio an Komponenten verwaltet und welche Prozesse hierzu benötigt werden. Ferner ist eine Entwicklungsinfrastruktur zu etablieren, in der Komponenten realisiert werden können. Für den effizienten Einsatz, aber auch für die Erstellung von Komponenten sind die Entwickler zu schulen.

3.6.6.1 Gremium

Das für die Stufe Wiederverwendung benötigte Gremium hat die Aufgabe der Koordination des Portfolios an Komponenten sowie die Spezifikation der verschiedenen Vorgaben. Die Koordination basiert auf Prozessen, die durch dieses Gremium gesteuert werden (siehe Abschnitt 3.6.6.2). Zur Vermeidung von Redundanzen wird darauf geachtet, dass sich die Funktionalitäten der Komponenten möglichst wenig überschneiden. Die Vorgaben für Komponenten beinhalten Richtlinien sowohl für das Erstellen als auch für die Nutzung von Komponenten.

Das Gremium stellt zudem sicher, dass für jede Komponente ein Verantwortlicher festgelegt wird, der die Betreuung übernimmt. Hierzu gehört neben der Bearbeitung von Fehlermeldungen auch die Weiterentwicklung. Ferner ist sicherzustellen, dass die Komponente selbst sowie die zugehörigen Informationen zur Nutzung dieser Komponente für alle Entwickler zur Verfügung stehen. Welche Informationen hierfür notwendig sind, wird ebenfalls durch das Gremium bestimmt.

Die Planung der Weiterentwicklung von Komponenten ist Aufgabe der Komponentenverantwortlichen. Jedoch sind diese Erweiterungen mit dem Gremium im Rahmen des Komponentenportfolios abzustimmen.

3.6.6.2 Prozesse

Die Prozesse im Kontext der Wiederverwendung haben die Aufgabe, die verschiedenen Komponenten für die Entwickler zur Verfügung zu stellen. Sie werden durch das Gremium koordiniert. Im Einzelnen sind dies die folgenden Prozesse:

- *Beantragung neuer Komponenten*: Werden gleiche Anforderungen in mehreren Anwendungen identifiziert, so wird geprüft, ob sich die Realisierung einer neuen Komponente lohnt. In diesem Prozess kann jeder Entwickler Komponenten, die noch erstellt werden müssen oder aber bereits existieren, zur Aufnahme in das Portfolio beantragen.

- *Bereitstellung von Komponenten:* Nachdem eine Komponente erstellt wurde und sie den Kriterien des Gremiums entspricht, wird über diesen Prozess die Bereitstellung für alle Entwickler koordiniert. Hierbei wird insbesondere darauf geachtet, dass alle benötigten Informationen über diese Komponente zur Verfügung stehen.
- *Entfernen von Komponenten aus dem Portfolio:* Sofern eine Komponente aus dem Portfolio entfernt werden soll, ist zu prüfen, welche Auswirkungen dies auf die verschiedenen Anwendungen beinhaltet. Die Komponenten sind direkt in den Anwendungen integriert, daher sind keine unmittelbaren Aktionen nötig. Dennoch ist für alle Anwendungen zu klären, wie die Funktionalität der Komponente ersetzt werden soll.

3.6.6.3 Entwicklungsinfrastruktur

Gegenüber der Standardisierung ist die Entwicklungsinfrastruktur in dieser Stufe um die Verwaltung von Komponenten zu erweitern. Dies kann sehr unterschiedlich organisiert werden, z.B. durch einen Katalog. Wichtig ist hierbei, dass allen Entwicklern die Informationen über die Komponenten sowie die Komponente selbst zur Verfügung gestellt werden.

- *Standards und Richtlinien:* Die Vorgaben für die Erstellung und Nutzung von Komponenten sind über entsprechende Richtlinien zur Verfügung zu stellen. Für den in dieser Stufe geforderten komponentenorientierten Aufbau einer Anwendung ist eine entsprechende Referenzarchitektur zu beschreiben.
- *Entwicklungswerkzeuge:* Die Entwicklungswerkzeuge unterstützen die Erstellung sowie die Einbindung der Komponenten in die Anwendung. Dies kann beispielsweise durch die Integration eines zentralen Komponentenkatalogs erfolgen. Sie haben ferner die Überprüfungen der geforderten Richtlinien bzgl. eines modularen Aufbaus sicherzustellen.
- *Konfigurationsmanagement:* Die Verwaltung der verschiedenen Versionen von Komponenten ist über das in der Standardisierung vorgegebene Konfigurationsmanagement zu organisieren. Nicht jede neue Version einer Komponente wird sofort in alle zu nutzenden Anwendungen integriert. Vielmehr wird für jede Anwendung separat geprüft, ob und wann diese neue Version der Komponente integriert wird. Meist wird dies mit der Realisierung neuer Anforderungen gebündelt. Dies bedeutet jedoch, dass jede eingesetzte Version der Komponente verfügbar sein muss.

  Das Konfigurationsmanagement sollte zudem über die Möglichkeit verfügen, die verschiedenen Komponenten mit einer Beschreibung und den vorhandenen Versionen zu verwalten.

- *Zentraler Build Service:* Gegenüber der Standardisierung sind an diesen Service keine weiteren Voraussetzungen zu etablieren. Die einzelnen Elemente der Anwendung werden durch den Entwickler manuell zusammengestellt und an den Build Service übergeben.
- *Transport und Deployment:* Die Anwendung wird analog der Standardisierung über den Build Service erstellt. Der Transport sowie das Deployment unterscheiden sich gegenüber der letzten Stufe nicht.

3.6.6.4 Laufzeitinfrastruktur

Die Komponenten werden bei der Entwicklung vollständig in die Anwendung integriert. Für die Laufzeitinfrastruktur ergibt sich somit kein Unterschied zur Standardisierung. Es werden daher keine weiteren Voraussetzungen an diese Stufe gestellt.

3.6.7 Reifegrade der Wiederverwendung

Die Bereitstellung von Komponenten garantiert nicht die breite Nutzung dieser Komponenten in den Anwendungen. Die Betrachtung des Reifegrades gibt Aufschluss über die Etablierung der Wiederverwendung im Unternehmen.

Die einzelnen Stufen des Reifegrades der Wiederverwendung werden folgendermaßen definiert:

- *Initial:* Die Prinzipien der Wiederverwendung sind im Unternehmen noch nicht bekannt. Es existieren keine Komponenten, die in verschiedenen Anwendungen eingesetzt werden könnten. Es gibt weder Gremien noch Prozesse für eine zentrale Koordination von Komponenten.
- *Wiederholbar:* Teilweise sind die Prinzipien der Wiederverwendung bekannt. Vereinzelt nutzen Entwickler Komponenten, z.B. technische Komponenten aus dem Open-Source-Bereich, in ihren Anwendungen. In diesem Zusammenhang gibt es erste Gruppierungen, die Komponenten definieren und für spezielle Projekte oder Abteilungen bereitstellen. Ein allgemeines Verfahren sowie Vorgaben für die Erstellung, Verwaltung und Nutzung von Komponenten sind jedoch noch nicht vorhanden. Auch die Pflege der Komponenten ist noch nicht geregelt.

  Diese Art der Wiederverwendung findet gelegentlich und undokumentiert statt. Im Unternehmen wurden keine Maßnahmen getroffen, diese Art der Wiederverwendung zu systematisieren.
- *Definiert:* Das Gremium für die Koordination der Wiederverwendung wurde eingerichtet und die Aufgaben definiert. Es erstellt die Richtlinien für die unternehmensweite Nutzung und Verwaltung von Komponenten. Die benötigten Prozesse wurden definiert und können durchgeführt werden.

Es existiert ein erstes Portfolio von Komponenten, die bereits in Anwendungen genutzt werden können.

Die Entwicklungsinfrastruktur bietet die nötige Unterstützung für die Verwaltung von Komponenten. Die Nutzung von Komponenten wird durch einen angepassten Entwicklungsprozess ermöglicht.

Erste Entwickler strukturieren ihre Anwendung nach den Vorgaben und nutzen die bereitgestellten Komponenten. Die Strukturierung sowie die Nutzung der Komponenten sind jedoch noch nicht für alle Entwicklungen verbindlich.

- *Eingeführt:* Die Strukturierung von Anwendungen anhand der Richtlinien sowie der Einsatz der verfügbaren Komponenten erfolgt in allen Implementierungen. Die Anwendungen sind komponentenorientiert aufgebaut und ermöglichen die einfache Integration von Komponenten. Alle Entwickler verfügen über das benötigte Wissen, die Komponenten in ihre Anwendungsentwicklung einzubinden. Die erstellten Artefakte sind konsistent zur Methode.
- *Optimiert:* Rückmeldungen zu den bereitgestellten Komponenten werden systematisch erfasst und an den Verantwortlichen zur weiteren Bearbeitung weitergeleitet.

  Erweiterungen an Komponenten oder die Erstellung von neuen Komponenten werden durch einen zentralen Prozess koordiniert und über das Gremium mit dem Portfolio abgeglichen.

  Die Nutzung der Komponenten wird durch das Gremium anhand von Kennzahlen beobachtet. Komponenten, deren Wiederverwendung gering ist, werden aus dem Portfolio entfernt, wenn der Aufwand für die Pflege zu hoch ist.

Hinweis: Für die Erhöhung des Reifegrades bzgl. der Wiederverwendung wird in [PoRS00] ein Prozess beschrieben, welche Maßnahmen hierfür ergriffen werden können.

3.6.8 Grenzen und Potentiale für Verbesserungen

Anwendungen, die sich im Wesentlichen auf die vorgestellte Wiederverwendung stützen, haben eine hohe Abhängigkeit zu den integrierten Komponenten. Jede Anwendung kann eine spezifische Version der eingesetzten Komponenten verwenden und auf Neuerungen verzichten. Sofern jedoch bei diesen Versionen Fehler auftreten oder Erweiterungen umgesetzt werden, sind entsprechend viele Versionen dieser Komponenten zu verwalten. Daher sollte die Aktualisierung der eingesetzten Komponenten bei der Planung zukünftiger Versionen der Anwendung berücksichtigt werden.

Die Realisierung der Anwendungen erfolgt nach wie vor manuell. Zwar wird ein Teil der Entwicklung durch die Nutzung von Komponenten eingespart, dennoch werden viele Fehler durch unvorsichtige manuelle Arbeiten hervorgerufen. Diese Fehler entstehen meist an Stellen, die keinen Bezug zur fachlichen Anforderung aufweisen und daher nicht im Fokus stehen. Die Entlastung des Entwicklers von solchen Tätigkeiten durch die Nutzung der Automatisierung ist somit der nächste Schritt der Industrialisierung in der Anwendungsentwicklung.

3.7 Stufe 4: Automatisierung

Die manuelle Erstellung von Software ist ein aufwändiges und fehleranfälliges Verfahren zur Realisierung von Anwendungen. In [KrSc06] wird dies als ein Grund genannt, warum Softwareentwicklung langsam und teuer ist. Die auf diese Weise realisierten Anwendungen enthalten meist ernsthafte Fehler und sind dadurch in ihrer Brauchbarkeit, Zuverlässigkeit, Performance und Sicherheit beeinträchtigt.

Die manuelle Fertigung von Gütern in der Fertigungswirtschaft wurde in der Vergangenheit sukzessive durch eine Automatisierung optimiert. Sobald erkannt wurde, dass Arbeit erleichtert und produktiver durchgeführt werden kann, wurden entsprechende Geräte und Technologien erfunden. So konnte beispielsweise das Weben von Stoffen durch die Einführung von Webstühlen deutlich vereinfacht und damit effektiver gestaltet werden.

Die Automatisierung war immer ein wesentlicher Bestandteil und Treiber der Industrialisierung. Durch die automatisierten Abläufe können Produkte deutlich preiswerter und schneller erzeugt werden. Nach [HüBa08] erlaubte die Einführung der Automatisierung in der Automobilindustrie eine höhere Präzision und ein insgesamt gesteigertes Qualitätsniveau.

Allgemein wird Automatisierung wie folgt definiert:

> Automatisierung ist die mit Hilfe von Maschinen realisierte Übertragung von Arbeit vom Menschen auf Automaten.

Definition 9: Automatisierung

Seit Beginn des Software-Engineerings (1968) wird auch in der Informatik kontinuierlich die Vereinfachung der Anwendungsentwicklung fokussiert. Grundlage sind hierfür zwei Ansätze: Anhebung des Abstraktionslevels sowie die Automatisierung des Entwicklungsprozesses [KTVo07]. Der Level der Abstraktion in der Softwareentwicklung sollte dabei möglichst nah an der Problembeschreibung liegen und die technischen Details verbergen.

In der Softwareentwicklung gibt es mehrere Möglichkeiten der Automatisierung. Zu den weit verbreiteten Methoden zählen die folgenden Bereiche [GrSh06]:

- *Quelltext-Generierung:* Eine automatische Generierung erfolgt bereits bei der Übersetzung von Quelltexten einer Programmiersprache, wie z.B. Java oder C. Die darin spezifizierten Anweisungen werden in ausführbare Maschinenbefehle transformiert. Grundsätzlich erfolgt die Generierung von Quelltexten auf Basis von Metadaten, die entweder als Modell vorliegen oder in einer anderen Form gespeichert sind, z.B. einem Visual Assembly (siehe nächsten Punkt). Für die sinnvolle Transformation benötigen die eingesetzten Generatoren das zugehörige Verständnis der Zielarchitektur bzw. der zugrunde liegenden Plattform.
- *Visual Assembly:* „Visual Assembly ist eine Methode, um Quelltext und Metadaten durch Manipulation grafischer Formen in einem Werkzeug zu erstellen“ [GrSh06]. Diese Form der Automatisierung wird besonders für die Erstellung von Benutzeroberflächen eingesetzt, bei der der Anwendungsentwickler die Elemente der Oberfläche in einem geeigneten Editor positioniert. Der zugehörige Compiler wandelt die Metainformationen des visuellen Assemblys anschließend in entsprechenden Quelltext um. Ein weiteres Anwendungsgebiet ist die visuelle Komposition von Prozessen.
- *Rapid Application Development (RAD):* Als Erweiterung des Visual Assemblys werden beim RAD neben den grafischen Elementen auch weitere Konstrukte einer Anwendung automatisch generiert. Dies beinhaltet beispielsweise projektspezifische Einstellungen oder den Zugriff auf die Datenhaltung. RAD stützt sich auf das Prototyping[65], bei dem ein vorzeigbares Ergebnis so schnell wie möglich dem Auftraggeber vorgestellt werden soll. Aufgrund der hierdurch erfolgten zeitnahen Rückmeldung wird die Anwendung in mehreren Iterationen erstellt.
- *Round-Trip Engineering (RTE):* Der durch Generatoren erzeugte Quelltext, z.B. aus Modellen, liegt meist in einer für den Entwickler lesbaren Programmiersprache vor. Damit hat der Entwickler die Möglichkeit, den erzeugten Quelltext für weitere spezifische Anforderungen zu modifizieren. Hierdurch sind jedoch der Quelltext und das Modell nicht mehr synchron. Eine erneute Generierung überschreibt die durchgeführten Änderungen. Das RTE stellt sicher, dass Änderungen an dem erzeugten Quelltext automatisch in das Modell übernommen werden.

Die Gründe für die Automatisierung in der Anwendungsentwicklung werden bei [Klar06] mit den folgenden Argumenten beschrieben:

[65] Erstellung von Musterbauteilen auf Basis von Konstruktionsdaten.

- *Qualität erhöhen:* Durch die automatische Erzeugung von Quelltext werden Flüchtigkeitsfehler vermieden, die charakteristisch für die manuelle Entwicklung sind. Sofern Fehler in der Generierung selbst enthalten sind, werden diese schneller erkannt, da der Quelltext mehrfach falsch erzeugt wird.
- *Personal effektiv einsetzen:* Die Generierung von Quelltexten aus Modellen erlaubt dem Anwendungsentwickler, von den Implementierungsdetails einer Technologie zu abstrahieren. Er kann sich auf die fachliche Lösung des Problems bzw. der Anforderung konzentrieren. Wiederkehrende, mechanische Aufgaben werden hierdurch vermieden.
- *Schnellere „Time to Application“:* Sofern Generatoren für die Realisierung von Anwendungen zur Verfügung stehen, können auch Varianten dieser Anwendungen schnell erzeugt werden. Ferner können aufgrund der gesteigerten Qualität durch die Generierung von Quelltexten Fehler schneller behandelt werden. Flüchtigkeitsfehler und damit die Suche nach den dadurch verursachten Problemen werden reduziert.
- *Kosten senken:* Die Bereitstellung von Generatoren sowie die Schulung der Entwickler sind zunächst mit Kosten verbunden. Insgesamt werden jedoch die Kosten aufgrund der höheren Qualität und Produktivität gesenkt.

Nach [GrSh06] ist die wichtigste Form der Automatisierung in der Softwareentwicklung die Generierung von Implementierungen aus Spezifikationen. Dabei legt eine Spezifikation fest, was getan werden soll. Die Implementierung realisiert die Vorgaben der Spezifikation.

Spezifikationen werden unter Verwendung von Modellen in eine Form gebracht, die von Werkzeugen interpretiert werden können und so die Automatisierung ermöglicht.

3.7.1 Modellierung als wesentliches Element der Automatisierung

Die Realisierung von Anforderungen in Form von Anwendungen ist ein kreativer Prozess, dessen Wertschöpfung in der effizienten Unterstützung von Geschäftsprozessen liegt. Der Entwickler sollte sich daher hauptsächlich mit der Lösungsfindung sowie der möglichst effizienten Realisierung beschäftigen.

Die Forderung nach der Abstraktion von technischen Themen aus dem letzten Abschnitt erfolgt durch die Modellierung des Problems bzw. der Lösung. Bei der Modellierung wird die Lösung zunächst unabhängig von der technischen Plattform beschrieben. Das hierbei verwendete Modell entspricht einem Beschreibungsmodell (vgl. Abschnitt 2.5).

In [Andr04] wird ein Modell im Kontext der Automatisierung wie folgt definiert:

> „Ein Modell ist eine vereinfachte Abbildung der Realität. Ein Modell kann als Muster für das zu entwickelnde System verstanden werden. Ein Modell kann zum einen sehr detaillierte Ansichten und zum anderen Übersichtsansichten umfassen, die einen Gesamteindruck des zu entwickelnden Systems vermitteln können."

Definition 10: Modell im Kontext der Automatisierung

Modelle verbergen bestimmte Informationen (z.B. technische Details), um verschiedene Aspekte einer Lösung einfacher beschreiben zu können [GrSh06]. Je nach Verwendung liefern Modelle unterschiedliche Sichten auf die Struktur oder auch das Verhalten eines Systems.

Nach [Andr04] hat die Modellierung die folgenden Zielsetzungen:

- Visualisierung des zu entwickelnden Systems
- Betrachtung der Struktur und des Verhaltens eines Systems
- Bereitstellung einer Schablone für die Entwicklung eines Systems
- Dokumentation des Entwicklungspfades

Der Schwerpunkt der Modellierung liegt in der Beschreibung der zu realisierenden Anwendung. Jedoch können auch die Artefakte des Entwicklungsumfeldes, die für die Analyse des Industrialisierungsgrades relevant sind, als Modell abgebildet werden.

Die Notation von Modellen erfolgt meist auf der Basis von grafischen Visualisierungen. Insbesondere wird die Unified Modeling Language (UML) als Standard in vielen Modellierungen verwendet. „Die UML ist eine Sprache, um Artefakte eines Softwaresystems zu visualisieren, spezifizieren, konstruieren und dokumentieren" [Andr04]. Die grafische Darstellung fördert das leichte Verständnis sowie den Umgang mit Modellen.

Nach [GrSh06] ist es jedoch wichtig, die durch ein Modell erfassten Informationen von dessen Visualisierung zu unterscheiden. Ein Modell kann auch in einer reinen Textform erfasst und verarbeitet werden.

Die formale Beschreibung eines Systems oder generell von Artefakten in Form von Modellen ist die Grundlage für die automatisierte Entwicklung von Anwendungen. Aufgrund der vorgegebenen Strukturen (Syntax) von Modellen ist die Anwendung von Regeln zur Erzeugung von Quelltext durch einen Generator erst möglich.

Diese formale Beschreibung erfolgt in einer Sprache, mit der der Entwickler seine Absicht formulieren kann.

3.7.2 Generierung von Software

Das Ergebnis der Modellierung ist die Spezifikation von Anforderungen in einer für einen Automaten lesbaren Form. Die Generierung von Software basiert auf der Interpretation dieser Modelle sowie der Anwendung von Regeln zur Transformation.

Eine Transformation muss dabei nicht direkt in einer ausführbaren Anwendung resultieren. Eine Modellierung kann zunächst unabhängig von der Technologie erfolgen, um die Spezifikation auch für den Auftraggeber lesbar zu gestalten. Ferner ermöglicht diese technikneutrale Modellierung eine Abstraktion von technischen Besonderheiten. Die spätere Festlegung einer Technologie erlaubt zudem die Realisierung einer Anwendung auf verschiedenen Plattformen.

Die Generierung von Software aus einer fachlichen, technikneutralen Anforderung erfolgt somit in mehreren Schritten. Zunächst wird das fachliche Modell in ein weiteres Modell transformiert, das die technischen Eigenschaften enthält. Aus diesem technischen Modell wiederum wird der eigentliche Quelltext für die Anwendung generiert. Abbildung 18 zeigt den schematischen Ablauf einer mehrstufigen Transformation.

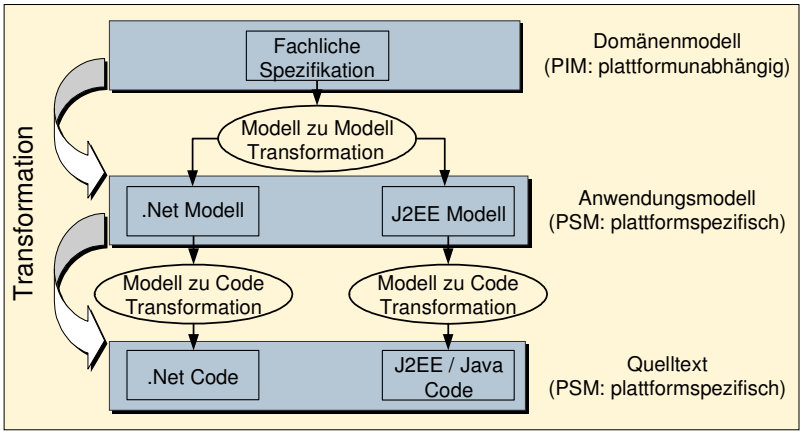

Abbildung 18: Mehrstufige Transformation bei der Generierung[66]

Diese Vorgehensweise hat die OMG[67] in der Model Driven Architecture (MDA) Spezifikation beschrieben. MDA ist ein Standard „für modellgetriebene Softwareentwicklung, der die Repräsentation von Software von der Programmcodeebene auf die Modellebene heben möchte“ [WaSi07]. Die Grundidee dabei ist, „Modelle unterschiedlicher Abstraktionsebenen zu definieren, um damit unterschiedliche Aspekte zu modellieren“. Sofern ein Modell noch keinen Bezug zu einer Plattform hat, spricht man in diesem Zusammenhang von einem plattformunabhängi-

[66] Siehe [Klar06]
[67] OMG: Object Management Group.

gen Modell (PIM: Platform Independent Model). Die Anreicherung des Modells mit Informationen zu einer konkreten Plattform führt zu einem plattformspezifischen Modell (PSM: Platform Specific Model).

Eine plattformunabhängige fachliche Modellierung ist jedoch nicht in allen Bereichen notwendig. Sofern aufgrund der Standardisierung im Unternehmen die Plattform bzw. die Technologie für die Entwicklungen von Anwendungen vorgegeben ist, wird ein von der Plattform oder der Technologie unabhängiges Modell nicht benötigt.

Die Transformation zwischen den verschiedenen Modellen sowie das Erzeugen von Implementierungen aus Modellen erfolgt durch Generatoren. Dies sind Automaten, die auf Basis von Regeln, Vorlagen sowie dem Modell das geforderte Artefakt erzeugen. Eine schematische Darstellung ist in Abbildung 19 zu sehen.

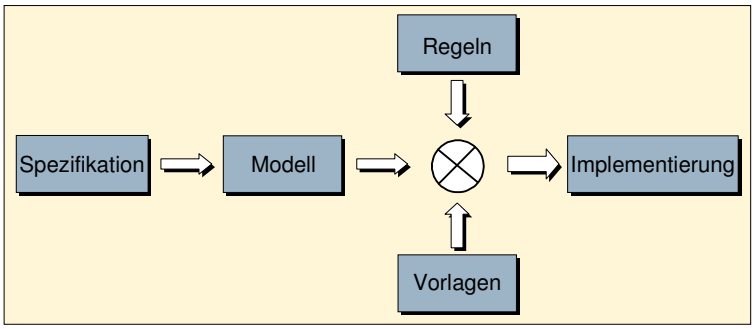

Abbildung 19: Schematische Darstellung des Generierens

Hierbei wird nicht das Design der Anwendung automatisiert, sondern die Auswahl, Kombination, Anpassung und Anwendung vordefinierter Designelemente. Mittlerweile existiert eine Vielzahl kommerzieller und freier Generatoren mit zum Teil unterschiedlichen Ansätzen zur Transformation. Für die modellgetriebene Entwicklung basieren nach [Klar06] die gängigsten Methoden auf Templates oder Frames.

Templates sind parametrisierbare Vorlagen, die Fragmente für den zu generierenden Quelltext enthalten. Es sind ferner Regeln hinterlegt, wie diese Fragmente für den Quelltext anzuordnen und auszugeben sind. Variablen in den Vorlagen ermöglichen für verschiedene Ausprägungen von Modellen, den hierfür spezifischen Quelltext zu erzeugen. Die Aufgabe des Generators ist es, das Modell auf die entsprechenden Templates anzuwenden.

Frames enthalten ebenfalls parametrisierbare Quelltextvorlagen. Die Generierung erfolgt jedoch in zwei Schritten. Zunächst werden die Frames als Objekte instanziiert und die Variablen an die Werte des Modells gebunden. Aufgrund der möglichen Verschachtelung von Frames entsteht ein Baum. Die Generierung erfolgt im zweiten Schritt durch Erzeugen von Quelltext anhand der Struktur des erzeugten Baumes.

Es existieren weitere Generierungstechniken, die jedoch für eine modellgetriebene Entwicklung nicht geeignet sind: API basierte Generatoren, Code Attribute, Inline Generierung, Code Weaving (siehe [Klar06]).

3.7.3 Model Driven Development

Wesentlicher Bestandteil der Automatisierung ist die modellgetriebene Entwicklung. Nach [GrSh06] ist dies „ein Verfahren, um die Aufgaben der Softwareentwicklung mit den in Modellen erfassten Metadaten zu automatisieren“. Für diese Stufe wird die Methode somit wie folgt festgelegt:

> Die modellgetriebene Entwicklung ist die Automatisierung von wiederkehrenden Aufgaben durch Generierung der benötigten Artefakte auf Basis von Modellen und Generatoren. Hierdurch wird der Entwickler von mechanischen Aufgaben befreit.

Definition 11: Model Driven Development

Die Modellierung ist das wesentliche Gestaltungselement dieser Stufe. Eine vollständige Modellierung ist jedoch meist nicht sinnvoll. Insbesondere bei der Gestaltung der Oberfläche sind oft manuelle Eingriffe notwendig. Bei dieser Methode wird vielmehr ein möglichst hoher, aber sinnvoller Grad an Automatisierung angestrebt.

Die Wiederverwendung ist Grundlage der Automatisierung, da insbesondere der komponentenorientierte Aufbau der Anwendung wesentlich für die Generierung der Artefakte aus dem Modell ist. Die Vorlagen und Regeln sind zudem so zu gestalten, dass sie wiederverwendet werden können. Ferner wird durch die Automatisierung die Wiederverwendung verbessert, da sich Modelle flexibler als Quelltext wiederverwenden lassen.

Durch die Einführung dieser Methode werden, aufbauend auf der Wiederverwendung, die verschiedenen Eigenschaften der Industrialisierung zusätzlich weiter unterstützt:

- *Standardisierung*: Die Vorlagen bei der Generierung enthalten lediglich die im Unternehmen standardisierten Technologien und Plattformen. Ferner sind die Architekturvorgaben bereits hinterlegt. Damit werden die Standards automatisch in die Anwendungen generiert.
- *Wiederverwendung*: Die Modellierung erlaubt die Einbindung von existierenden Komponenten bereits im Modell. Darüber hinaus können auch die Modelle selbst wiederverwendet werden. Diese Wiederverwendung findet auf einer höheren Abstraktionsebene statt.

- *Automatisierung*: Die durch Modelle getriebene Entwicklung von Anwendungen ermöglicht dem Entwickler die Ausrichtung auf den Lösungsraum der Problemstellung. Sie entlastet ihn von routinemäßigen Arbeiten. Das Erstellen der Artefakte erfolgt automatisiert auf Basis der Modelle.
- *Arbeitsteilung*: Die Konzeption der Anwendung sowie die Realisierung waren bisher eng gekoppelt. Durch die Einführung von Modellen können die Konzeption sowie die Realisierung auf verschiedene Personen aufgeteilt werden. Die Aufgabe der Konzeption ist die Modellierung der Prozessunterstützung, während der technische Entwickler die Transformation des entstandenen Modells in die Anwendung im Fokus hat.
- *Verringerung der Fertigungstiefe*: Werden Standards eingesetzt, die über das Unternehmen hinaus verwendet werden, kann das Erzeugen der Anwendung auch durch externe Dienstleister erfolgen. Ein Beispiel hierfür ist die Firma Integranova[68], die aus einem Modell unter bestimmten Voraussetzungen eine komplette Anwendung generiert.

Für die Qualitätsfaktoren der Anwendungsentwicklung hat die Etablierung der modellgetriebenen Entwicklung die folgenden Auswirkungen:

- *Funktionalität:* Aufgrund der Modellierung der geforderten Funktionalitäten kann bereits zu einem sehr frühen Zeitpunkt die Angemessenheit sowie die Richtigkeit der Funktionen überprüft werden. Die Vorlagen und Regeln für die Generierung der Anwendung enthalten bereits zu einem großen Teil die sicherheitsrelevanten Elemente der Anwendung. Diese sind meist durch den Entwickler nicht mehr erneut zu implementieren. Damit reduziert sich das Risiko, sicherheitsrelevante Themen zu übersehen.
- *Zuverlässigkeit*: Vorlagen für die Generierung werden mit besonderer Aufmerksamkeit erstellt, da sie für mehrere Anwendungen zum Einsatz kommen. Die hierfür spezialisierten Entwickler können Mechanismen integrieren, die eine erhöhte Fehlertoleranz sowie eine reduzierte Fehleranfälligkeit ermöglichen. Für diese Mechanismen werden zudem entsprechende Testszenarien generiert, die potentielle Fehlerquellen überprüfen. Die Entwickler von Anwendungen können sich somit auf die fachlichen Problemstellungen konzentrieren.
- *Benutzbarkeit*: Sofern die eingesetzte Technologie die Modellierung von Oberflächen erlaubt, kann diese bereits zu einem frühen Zeitpunkt durch die Anwender analysiert und optimiert werden. Dies fördert die Verständ-

[68] Integranova: Softwarehersteller, der auf Basis von Modellen komplette Anwendungen generiert. (http://www.integranova.de/)

lichkeit sowie die Bedienbarkeit. Änderungen an der Oberfläche aufgrund von Rückmeldungen der Anwender können einfacher in die Modellierung einfließen und damit schneller umgesetzt werden.

- *Effizienz*: Die Vorlagen für die Generierung von Anwendungen enthalten bereits die Erfahrungen bzgl. des Zeit- und Verbrauchsverhaltens anderer Anwendungen in einer vergleichbaren Umgebung. Der Entwickler kann sich somit auf die Realisierung der Prozesslogik konzentrieren. Zudem wird dieses Merkmal durch die generierten Tests automatisch überprüft.
- *Änderbarkeit:* Die Generierung der Lösung aus Modellen erfolgt nach einem vorgegebenen Schema, das vielen Entwicklern bekannt ist. Eine Wartung kann daher von mehreren Entwicklern durchgeführt werden. Änderungen an der Logik der Anwendung können mit dem Auftraggeber auf Basis der Modelle diskutiert und besprochen werden. Änderungen lassen sich somit schneller realisieren. Aufgrund der Generierung des Quelltextes werden Fehler der Entwickler für routinemäßige Arbeiten vermieden. Durch die mehrfache Nutzung der Generierungsvorlagen und Regeln sind diese bereits getestet.
- *Übertragbarkeit:* Auf Basis eines Modells können unterschiedliche Realisierungen generiert werden. Eine Anwendung, die bspw. für eine Java-Plattform realisiert wurde, kann mit entsprechend geänderten Generierungsvorschriften[69] auch für eine .Net-Plattform erstellt werden.

  Änderungen im Prozessablauf können auf Basis eines Modells besser erörtert werden. Zudem ermöglicht die Generierung eines veränderten modellierten Prozessablaufs eine schnellere und damit flexiblere Realisierung der Anwendung.

Auf die Faktoren Zeit und Kosten hat die Automatisierung die folgenden Auswirkungen:

- *Entwicklungszeit*: Die Entwicklung der Anwendung reduziert sich auf die Bearbeitung der fachlichen Lösung. Hierbei stehen die Konzeption sowie die Modellierung im Vordergrund. Durch die Fokussierung auf die eigentliche Problemstellung und die Entlastung von routinemäßigen Arbeiten kann der Entwickler die Anwendung schneller realisieren.
- *Kosten*: Die Erstellung von Quelltext entsteht direkt aus den Modellen des Entwicklers. Die meist sehr zeitaufwendigen und fehleranfälligen manuellen Tätigkeiten entfallen an vielen Stellen. Die Generierung reduziert somit die Bearbeitungszeit und damit die Kosten für die Erstellung.

[69] Regeln und Vorlagen, die spezifisch für eine Technologie oder Plattform ausgelegt sind.

3.7.4 Erstellung von Artefakten auf Basis der Automatisierung

Die Modellierung der Absichten bzw. Anforderungen sowie die Generierung der verschiedenen Artefakte auf Basis des Modells sind die wesentlichen Methoden dieser Stufe. Abbildung 20 veranschaulicht den Zusammenhang.

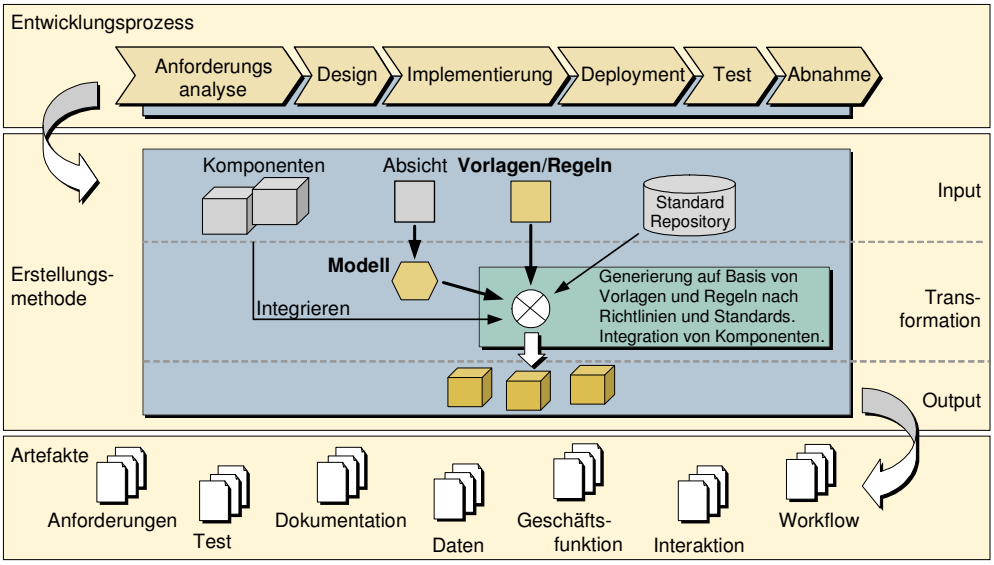

Abbildung 20: Model Driven Development

Die Erstellung der Artefakte erfolgt bei dieser Methode nach dem folgenden Schema:

Input

Aus den vorherigen Stufen stehen dem Entwickler aus der Standardisierung die verschiedenen Standards und Richtlinien, aus der Wiederverwendung die Komponenten sowie die Vorschrift zu einem komponentenorientierten Aufbau der Anwendung zur Verfügung. In dieser Stufe werden neben den Anforderungen zusätzlich Vorlagen und Regeln für die Generierung der Anwendung zur Verfügung gestellt. Diese Vorlagen enthalten bereits die Standards und Richtlinien zur Programmierung sowie die zu integrierenden Komponenten. Der Entwickler kann sich somit auf die Modellierung der Anwendung konzentrieren.

Transformation

Die Absichten bzw. Anforderungen an die Anwendung werden zunächst unabhängig von einer Plattform modelliert. Im zweiten Schritt wird aus den modellierten Absichten, aufgrund der Entscheidung für eine Technologie sowie eine Plattform, das plattformspezifische Modell generiert. Die benötigten Artefakte werden

anschließend aus diesem Modell anhand der spezifischen Generierungsvorlagen und Regeln unter Einsatz von Generatoren erzeugt.[70]

In den Vorlagen sind die Komponenten aus der Stufe Wiederverwendung bereits eingearbeitet. In einem weiteren Schritt der Transformation wird die Anwendung aus den generierten Quelltexten automatisch gebaut. Die Vorlagen können für die betrachtete Anwendung durch manuelle Programmierung spezifisch erweitert werden. Neben der Generierung der Anwendung werden auch die Testfälle sowie die Dokumentation aus den Modellen generiert.

Output

Als Ergebnis der Transformation stehen das Modell, die aus dem Modell generierten Artefakte sowie die ausführbare Anwendung zur Verfügung. Die generierten Artefakte enthalten sowohl die Elemente der Anwendung als auch die Testfälle und Dokumentationen.

3.7.5 Rahmenbedingungen für die Automatisierung

Die Generierung von Anwendungen auf Basis von Modellen erfordert ein koordiniertes Vorgehen sowohl bei der Einführung der Methode als auch im täglichen Gebrauch. Während bei der Wiederverwendung der herkömmliche manuelle Erstellungsprozess genutzt wurde, erfordern die Modellierung sowie die Generierung die Schulung der neuen Verfahren.

Die folgenden Voraussetzungen sind für die Einführung der Automatisierung wesentlich.

3.7.5.1 Gremium

Es gibt eine Vielzahl an Verfahren und Werkzeugen zur Generierung von Artefakten. Das Gremium hat die Aufgabe, das für das Unternehmen sinnvolle Verfahren sowie die hierzu notwendigen Werkzeuge festzulegen. Wesentlich ist für diese Stufe, dass auf Basis eines Modells durch einen Generator, unter Verwendung von entsprechenden Vorlagen und Regeln, die benötigten Artefakte erstellt werden können.

Folgende Festlegungen sind in diesem Zusammenhang durchzuführen:

- Modellierungssprache für die Beschreibung der Anwendung sowie eine Richtlinie für die Modellierung
- Werkzeuge für die Modellierung sowie die Generierung der Artefakte
- Richtlinien, Verfahren sowie Vorgehen bei der Generierung der Artefakte

[70] Die Werkzeuge und Generatoren werden in dieser Arbeit nicht festgelegt. Dies ist durch das jeweilige Unternehmen zu spezifizieren.

Als Input für die Generierung dienen Vorlagen und Regeln. Hierfür ist durch das Gremium ein Verfahren zu definieren, wie die Vorlagen und Regeln im Unternehmen verwaltet und gewartet werden.

3.7.5.2 Prozesse

Nach der Einführung der Methode im Unternehmen ist sicherzustellen, dass die benötigten Werkzeuge, Richtlinien etc. allen Entwicklern zur Verfügung stehen. Veränderungen, die sich z.B. durch neue Anforderungen im Unternehmen, aber auch durch Hersteller aufgrund von Modifikationen von Werkzeugen ergeben, sind durch die folgenden Prozesse zu koordinieren:

- *Evaluierung:* Die Softwarebranche bietet verschiedene Ansätze für die modellgetriebene Entwicklung. Im Rahmen eines Evaluierungsprozesses werden potentielle neue Verfahren auf die Einsatzfähigkeit im Unternehmen überprüft.
- *Standardisierung:* Für den einheitlichen Einsatz der modellgetriebenen Entwicklung ist sicherzustellen, dass nur die vom Gremium freigegebenen Modellierungssprachen und Werkzeuge zur Modellierung und Generierung eingesetzt werden. Hierzu wird ein Prozess zur Standardisierung dieser Elemente aufgesetzt, der durch das Gremium koordiniert wird.
- *Koordination von Vorlagen und Regeln:* Die zentrale Bereitstellung von Vorlagen und Regeln erfordert ein koordiniertes Verfahren zur Verwaltung sowie zur Erweiterung. Der hierfür notwendige Prozess wird durch das Gremium gesteuert und stellt sicher, dass die Entwickler auf die aktuellen Vorlagen und Regeln Zugriff haben.

3.7.5.3 Entwicklungsinfrastruktur

Die Entwicklungsinfrastruktur für die Automatisierung erweitert die Anforderungen aus der Wiederverwendung. Für verschiedene Technologien oder Plattformen kann die zugehörige Entwicklungsinfrastruktur unterschiedlich aufgebaut werden. Dies wird durch das Gremium geregelt. Für jeden Entwickler sind über die Entwicklungsinfrastruktur die folgenden Elemente zur Verfügung zu stellen:

- *Standards und Richtlinien:* Die Standards und Richtlinien aus der Wiederverwendung werden um die Vorgaben für die Generierung von Artefakten aus Modellen erweitert. Dies beinhaltet einerseits die Art und Weise der Modellierung sowie das Vorgehen bei der Generierung. Eine Richtlinie zur Modellierung legt Namenskonventionen, verbindliche Modellstrukturen und Standarddiagramme fest.
- *Entwicklungswerkzeuge:* Im Rahmen der modellgetriebenen Entwicklung sind Werkzeuge sowohl für die Modellierung als auch für die Generierung notwendig. Welche Werkzeuge hierbei zum Einsatz kommen, wird durch

das Gremium festgelegt. Viele Hersteller von integrierten Entwicklungsumgebungen (IDE) bieten Erweiterungen ihrer Werkzeuge für die Modellierung und Generierung von Software an. Art und Umfang der Unterstützung variieren jedoch sehr.

- *Konfigurationsmanagement:* Die Vorlagen und Regeln für die Generierung der Artefakte ist über das Konfigurationsmanagement zu verwalten. Es können anwendungsspezifische Ausprägungen dieser Vorlagen erstellt werden, die zusammen mit den Artefakten der Anwendung zu verwalten sind. Dies kann z.B. bei der Oberflächengestaltung der Fall sein, wenn spezifische Anforderungen an das Nutzerverhalten gestellt werden. Der Entwickler hat in diesem Fall Änderungen an den zentralen Vorlagen und Regeln in den anwendungsspezifischen Vorlagen manuell einzupflegen.
- *Zentraler Build Service:* Das zentrale Erstellen von Anwendungen durch den Build Service ist um die Generierung der Artefakte zu erweitern. Der Build Service muss einerseits das zu transformierende Modell interpretieren können und anschließend Zugriff auf die Vorlagen und Regeln haben, um die Artefakte erzeugen zu können.
- *Transport und Deployment:* Der Transport und das Deployment verändern sich gegenüber der Wiederverwendung nicht, da auch hier die komplette Anwendung durch den zentralen Build Service bereitgestellt wird. Die Anwendung wird anschließend über die Systeme Entwicklung und Test nach Produktion transportiert und installiert.

3.7.5.4 Laufzeitinfrastruktur

Die modellgetriebene Softwareentwicklung richtet den Fokus auf die Erstellung der Anwendung. An die Laufzeitumgebung werden daher keine weiteren Voraussetzungen geknüpft.

3.7.6 Reifegrade der Automatisierung

Durch die Automatisierung der Anwendungsentwicklung verändert sich die Realisierung von Anwendungen. Das Lösen von der Entwicklung in gewohnten Technologien hin zur technikneutralen Modellierung einer Anwendung bedarf einer kontinuierlichen Kommunikation und Schulung, aber auch Kontrolle der aufgestellten Vorgaben. Für die verschiedenen Reifegrade dieser Stufe gelten die folgenden Kriterien:

- *Initial:* Die Modellierung und Generierung von Anwendungen wird im Unternehmen nicht eingesetzt. Es gibt keine Vorgaben oder Verfahren für diese Methode. Weder das entsprechende Gremium noch die hierfür benötigten Prozesse sind vorhanden.

- *Wiederholbar:* Vereinzelt wird in der Anwendungsentwicklung die Generierung von Software durchgeführt. Die Nutzung ist jedoch abhängig vom jeweiligen Entwickler und wird nur für bestimmte Teile der Anwendung verwendet. Erste Gruppierungen definieren für bestimmte Projekte oder Abteilungen Modellierungssprachen und Werkzeuge. Diese Festlegungen haben jedoch keinen unternehmensweiten, verbindlichen Charakter. Ein zentrales Gremium ist nicht vorhanden. Die benötigten Prozesse sowie Vorgaben sind nicht definiert.
- *Definiert:* Ein zentrales Gremium sowie die Prozesse im Rahmen der Automatisierung sind definiert und eingerichtet. Das Gremium übernimmt die geforderten Aufgaben und koordiniert die Prozesse.

  Die Modellierungssprache sowie die Werkzeuge zur Modellierung und Generierung sind definiert. Die Vorgaben für die Modellierung und Generierung von Artefakten wurden erstellt. Den Entwicklern stehen die Werkzeuge und Vorgaben zentral zur Verfügung.

  Erste Anwendungen werden auf Basis der definierten Modellierung und Generierung erstellt. Es ist jedoch noch nicht verpflichtend im Unternehmen eingeführt.
- *Eingeführt:* Die Methode wurde erfolgreich im Unternehmen eingeführt. Die Entwickler haben die Methode verstanden und wenden sie an. Die Anwendungsentwicklung basiert auf der Modellierung der Anwendung sowie der Generierung der verschiedenen Artefakte und richtet sich nach den Vorgaben dieser Stufe.

  Die Konsistenz sowie die Aktualität der verschiedenen Artefakte sind gewährleistet. Hierzu gehört beispielsweise, dass das Modell den aktuellen Stand der Anwendung wiedergibt und nicht am Modell vorbei eine Entwicklung stattfindet. Ferner entsprechen Test und Dokumentation dem Stand der Anwendung.
- *Optimiert:* Das Verfahren der Modellierung und anschließender Generierung wird regelmäßig überprüft. Rückmeldungen, die das Verfahren optimieren, werden systematisch erfasst und bearbeitet. Ferner werden neue Entwicklungen im Bereich der Modellierung und Generierung auf dem Softwaremarkt beobachtet und bei Bedarf analysiert.

3.7.7 Grenzen und Potentiale für weitere Verbesserungen

Die Modellierung von Artefakten einer Anwendung sollte sinnvoll eingesetzt werden. Bei der Modellierung von komplexen Algorithmen sollte zumindest hinterfragt werden, ob der hierfür notwendige Aufwand gerechtfertigt ist. Auch bei be-

sonders kritischen, z.B. sicherheitskritischen, Algorithmen ist zu hinterfragen, ob dieser allgemeinverständlich modelliert werden soll.

Die Generierung von Artefakten auf Basis von Modellen unter Einsatz von Vorlagen und Regeln ermöglicht eine effiziente Realisierung von Anwendungen. Zwar werden auch gleiche Funktionalitäten durch Komponenten eingebunden, jedoch erfolgt dies separat für jede Anwendung. Eine Komponente hat damit so viele Instanzen, wie es Anwendungen gibt, die sie einbinden. Eine Änderung an einer Komponente, die so schnell wie möglich in alle Anwendungen ausgebracht werden muss, verursacht einen hohen Aufwand. Dies kann insbesondere bei sicherheitsrelevanten Themen der Fall sein.

Diese Problematik könnte durch die zentrale Bereitstellung von Komponenten zur Laufzeit gelöst werden. Die Komponenten werden dabei nicht während der Entwicklung in die Anwendung integriert, sondern zur Laufzeit von der Anwendung aufgerufen. Änderungen an der Komponente erfolgen in diesem Fall lediglich an der zentralen Instanz.

Automatisierung bedeutet auch, dass in kürzerer Zeit mehr Anwendungen entstehen können, als ursprünglich geplant wurde. Der Aufwand für die manuelle Realisierung von Anwendungen hat in der Vergangenheit verhindert, dass Anwendungen erstellt wurden, die dem Unternehmen keinen sichtbaren Mehrwert aufzeigen. Ferner wurde bei jeder Entwicklung untersucht, ob bereits eine ähnliche Anwendung existiert.

Wenn der Aufwand für die Erstellung von Anwendungen durch die Automatisierung deutlich reduziert wird, besteht das Risiko, dass diese Prüfungen nicht mehr stattfinden werden. Die Problematik hierbei liegt im Betrieb und der Betreuung der Anwendungen, deren Kosten durch zu viele Anwendungen unnötig steigen. Der Aufbau bzw. die Bestandteile der IT-Landschaft sind daher durch ein entsprechendes Gremium zu koordinieren.

3.8 Stufe 5: Komposition

Unternehmen, deren Anwendungsentwicklung bereits durch die Automatisierung geprägt ist, haben bereits einen hohen Grad an Industrialisierung erreicht.

Es werden Standardtechnologien und -plattformen eingesetzt. Die modular aufgebauten Anwendungen werden auf Basis eines Modells konzipiert und unter Verwendung von Vorlagen und Regeln generiert. Der Entwickler wird immer weniger mit Aufgaben konfrontiert, die nicht originär zur Erfüllung der Anforderungen zählen. Er kann sich somit auf die fachliche Lösung konzentrieren. Die Abstraktion der Konzeption durch die Modellierung ermöglicht es zudem, intensiver mit dem Auftraggeber über die Gestaltung der Lösung zu diskutieren und somit Veränderungen frühzeitig zu erkennen und einzubringen.

Ohne weitere Steuerung in der Erstellung von Anwendungen bzw. die Bebauung der IT-Landschaft kann dieses Vorgehen jedoch erhebliche Probleme verursachen. Gibt es im Unternehmen keine klare Vorstellung, welche Geschäftsprozesse durch welche Anwendungen unterstützt werden, besteht das Risiko, dass gleiche Anforderungen durch mehrere unterschiedliche Anwendungen realisiert werden. Dies tritt auf, wenn ein Unternehmen in mehrere Geschäftsbereiche aufgeteilt ist, in denen jeweils unabhängig von anderen Bereichen Anwendungen entwickelt werden.

Die einzelne Entwicklung wird durch die Automatisierung effizienter. Der Aufwand für die Pflege von redundanten Anwendungen in einer komplexen IT-Landschaft steigt jedoch. Ferner können die Kosten aufgrund unnötiger Lizenzforderungen von Infrastrukturkomponenten ebenfalls steigen. Die Vorteile der effizienten Erstellung von Anwendungen werden durch die unkoordinierte Bebauung der IT-Landschaft aufgehoben.

Ein weiteres Risiko ist die unkontrollierte Nutzung von Schnittstellen verfügbarer Anwendungen. In vielen Unternehmen gibt es zentrale Systeme, die über Schnittstellen sowohl Informationen zur Verfügung stellen als auch Anfragen oder Aufträge an andere Anwendungen senden. Diese Systeme enthalten beispielsweise zentrale Prozesse wie die Verwaltung von Kundenaufträgen oder Produktinformationen. Sofern Anwendungen direkt auf diese Systeme zugreifen, entsteht mit der Zeit ein Geflecht von Abhängigkeiten, das schwer durchschaubar ist. Fällt ein System aus oder wird in einem System eine neue Version eingespielt, sind die Auswirkungen auf die Unterstützung der verschiedenen Geschäftsprozesse nicht transparent.

Ein Schwerpunkt der fünften Stufe der Industrialisierung in der Anwendungsentwicklung ist daher die koordinierte Planung der IT-Bebauung sowie die Entflechtung von Abhängigkeiten zwischen Anwendungen.

Die zentrale Bereitstellung von Funktionalitäten in Form von eigenständigen Diensten ist ein weiterer Aspekt, der in dieser Stufe betrachtet wird. In Stufe drei wurden für die Wiederverwendung die Komponenten noch in die Anwendung direkt integriert. Sofern jedoch die Funktionalitäten dieser Komponenten nicht mehr dezentral in Anwendungen integriert, sondern zentral zur Verfügung gestellt werden, hat der Anwendungsentwickler diese nur noch zur Laufzeit aufzurufen. Der Betrieb sowie die Wartung dieser Komponente erfolgt in diesem Fall zentral. Anpassungen an der zentralen Funktionalität können somit effizient bearbeitet werden, da an den Anwendungen keine Änderungen durchzuführen sind.

Die Strukturierung von Funktionalitäten in Komponenten und deren Bereitstellung in Form von Diensten ermöglicht die Komposition neuer Anwendungen auf

Basis dieser Dienste. Funktionalitäten einer Anwendung werden hierdurch für andere Anwendungen nutzbar.

Diese Themen werden in dem Konzept der serviceorientierten Architektur (SOA)[71] adressiert. Die Bedeutung von SOA ist bislang nicht eindeutig definiert[72], dennoch gibt es bereits Literatur zu diesem Thema, welche die grundsätzlichen Prinzipien beschreibt (siehe u.a. [Josu08], [Oasi06], [HaLi07], [Erl08]). Insbesondere die Hersteller von Anwendungsplattformen nutzen dieses Konzept zum Verkauf der eigenen Produkte. Dabei wird insbesondere der Fokus auf die Technik gelegt, wie Anwendungen über technische Services miteinander kommunizieren ([EHHJ08]). Beraterfirmen nutzen dieses Konzept eher zur fachlichen Strukturierung des Unternehmens und klammern die technische Realisierung zunächst aus. Nur wenige Hersteller kombinieren die fachlichen und technischen Konzepte von SOA in ihren Produkten. Die SAP stellt beispielsweise mit der Enterprise Service Architecture (ESA) ein Konzept bereit, wie eine serviceorientierte Architektur auf Basis ihrer NetWeaver Produktplattform gestaltet werden kann. Dabei wird das Konzept der Web-Service-basierten Entwicklung, Verwaltung und Komposition um betriebswirtschaftliche Elemente ergänzt [HaLi07].

Für Unternehmen, die Anwendungen zur Unterstützung ihrer Geschäftsprozesse benötigen, ist es schwer, zwischen den verschiedenen Fronten ein abgestimmtes Konzept zu identifizieren, das ihre IT-Landschaft weiter optimiert. Aus diesem Grund haben sich Unternehmen gruppiert, um ein besseres Verständnis von SOA aufzubauen, unabhängig von Herstellern und Beratern. Beispiel einer solchen Gruppierung ist das SOA Innovation Lab[73].

Für diese Stufe des Industrialisierungsgrades sind der Service sowie die Serviceorientierung die zentralen Elemente. Im nächsten Abschnitt werden die Konzepte und Prinzipien der Serviceorientierung in der Anwendungsentwicklung genauer erläutert.

3.8.1 Serviceorientierung

Die Ausrichtung der Entwicklung von Anwendungen auf Services ist derzeit ein viel diskutiertes Thema. Die grundsätzlichen Konzepte der serviceorientierten Architektur sind bereits seit 1996 in einem Gartner-Report veröffentlicht worden [Gart96]. Der eigentliche Durchbruch dieses Konzeptes gelang jedoch erst durch die allgemeine Verfügbarkeit von Web-Services auf mittlerweile fast allen Plattformen und Technologien. Mit Web-Services wird nach [Josu08] „eine Sammlung von Standards bezeichnet, die die Interoperabilität bei systemübergreifenden

[71] Die Beschreibung von SOA erfolgt in Abschnitt 3.8.1.2.

[72] Es gibt derzeit kein öffentlich anerkanntes Gremium, das SOA eindeutig spezifiziert.

[73] Unabhängiger Verein, dessen Mitglieder aus Anwenderunternehmen bestehen (siehe http://www.soa-lab.de).

Aufrufen sicherstellen sollen“. Web-Services sind eine mögliche technische Realisierung von Services. Sie dürfen jedoch nicht mit SOA gleichgesetzt werden [Gart03].

Ein wesentlicher Aspekt bei der Serviceorientierung ist die Möglichkeit, vorhandene Fachlichkeit in Form von Services zu jedem Zeitpunkt in einem Prozess zu integrieren. Dabei kann dieser Service sowohl innerhalb als auch außerhalb des Unternehmens zur Verfügung gestellt werden.

3.8.1.1 Service

Derzeit gibt es kein anerkanntes Standardisierungsgremium, das die Eigenschaften eines Services festlegt und diesen Begriff einheitlich und verbindlich definiert. Vielmehr gibt es eine Reihe von Definitionen, die aus den unterschiedlichsten Gründen motiviert sind.

Unter anderem hat die OASIS[74] in ihrem „Reference Model for Service Oriented Architecture“ [Oasi06] verschiedene Definitionen im Kontext von SOA aufgestellt. Ein Service wird dort folgendermaßen beschrieben: „A service is a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description. A service is provided by an entity – the service provider – for use by others, but the eventual consumers of the service may not be known to the service provider and may demonstrate uses of the service beyond the scope originally conceived by the provider.“

Eine etwas kürzere Definition von [Josu08], die in ähnlicher Form auch in [HaLi07] zu finden ist, wird in dieser Arbeit verwendet:

> Ein Service ist eine IT-Repräsentation von fachlicher Funktionalität, die durch eine (wohldefinierte) Schnittstelle beschrieben wird. Services sollten außerdem in sich abgeschlossen sein (autark sein und für sich selbst stehen).

Definition 12: Service

Der Service sollte dabei die folgenden Eigenschaften aufweisen:

- *Abgeschlossenheit:* Zur Minimierung von Abhängigkeiten sind Services unabhängig von anderen Services und Systemzuständen zu gestalten.
- *Grobgranularität*: Die Granularität der Services muss mit Bedacht gewählt werden. Eine zu feine Granularität bietet zwar eine sehr flexible Gestal-

[74] OASIS: Organization for the Advancement of Structured Information Standards: eine Non-Profit-Organisation für die Entwicklung, Zusammenführung und Adaption von offenen Standards. (http://www.oasis-open.org)

tung von Prozessen, verursacht aber einen enormen Verwaltungsaufwand, der zudem die Komplexität erhöht. Eine konkrete Definition, welche Granularität ein Service haben sollte, ist in der Literatur nicht zu finden.

- *Sichtbarkeit und Auffindbarkeit*: Zur Nutzung eines Services muss dieser den Entwicklern bekannt sein sowie produktiv zur Verfügung stehen.
- *Zustandslosigkeit*: Für die einfache Handhabung von Failover-Over- oder Load-Balancing-Mechanismen sollten Services zustandslos sein. Dies bedeutet, dass zwischen zwei aufeinanderfolgenden Aufrufen eines Services kein Zustand gehalten wird und damit keine Abhängigkeit existiert.
- *Idempotenz*: Der identische, mehrfache Aufruf eines Services darf zu keinem anderen Ergebnis führen als der erste Aufruf. Dies ist insbesondere wichtig, wenn bei fehlender Rückmeldung des Services dieser erneut mit denselben Parametern aufgerufen wird.
- *Wiederverwendbarkeit*: Jede Funktionalität sollte nur einmal implementiert werden. Der Service muss daher von verschiedenen Seiten (Technologien, Plattformen, etc.) aufrufbar sein.
- *Komponierbarkeit*: Jeder Service kann durch einen anderen Service aufgerufen werden. Hierdurch können größere Services in Teilschritte unterteilt werden oder zu höherwertigen Services aggregiert werden.

Die Fähigkeit zur Komposition von Services ermöglicht die Orchestrierung, also die Kombination mehrerer Services zu einem neuen Service. In der Literatur wurden Services daher kategorisiert. [Johu08] schlägt die folgende Unterteilung vor:

- *Basis Services*: Services dieser Kategorie stellen fachliche Basisfunktionalitäten zur Verfügung, bei denen eine weitere Aufteilung wenig sinnvoll ist.
- *Composed Services*: Die Komposition von Basis Services wird als Composed Service bezeichnet. Hierbei werden die Funktionalitäten der verschiedenen Services zu einem neuen Service zusammengestellt. Dieser kann wiederum auch Composed Services enthalten. Dieser Service ist zustandslos und hat eine kurze Laufzeit.
- *Prozess Services*: Im Gegensatz zu den Composed Services repräsentieren die Prozess Services ganze Workflows oder Prozesse. Dabei handelt es sich um eine eher lang laufende Folge von Aktivitäten, die meist auch mit einem Zustand behaftet ist.

Zusätzlich zu der Realisierung der fachlichen Funktionalität, die in der vierten Stufe aus Modellen generiert wurde, ist für die Nutzung sowie die Bereitstellung

von Services weitere Programmierung notwendig. Hierzu zählen die Bereitstellung von Schnittstellen des Serviceanbieters sowie der Aufruf von Services aus Anwendungen oder anderen Services.

Diese Artefakte haben nach [Josu07] für die unterschiedlichen Services meist den gleichen Aufbau und eignen sich daher ebenfalls für die Generierung auf Basis einer Modellierung von Services. Die Automatisierung spielt in dieser Stufe somit eine wichtige Rolle, um den Entwickler von routinemäßigen Aufgaben zu befreien und dadurch Flüchtigkeitsfehler zu vermeiden.

3.8.1.2 Serviceorientierte Architektur

Die Komposition von Services setzt eine Architektur voraus, in der Services kombiniert werden können und die es Anwendungen ermöglicht, verschiedene Services zur Laufzeit einzubinden. Eine solche serviceorientierte Architektur muss nach [HaLi07] folgende Prinzipien beinhalten:

- *Dekomposition und Kapselung:* Geschäftsprozesse können in kleinere autonome Einheiten, die Services, zerlegt werden. Sie stellen somit die gekapselte Realisierung von Funktionalitäten oder Teilprozessen dar.
- *Abstraktion*: Die Definition eines Services erfolgt auf Basis der Schnittstellenbeschreibung. Die Art und Weise, wie dieser Service im Detail implementiert wurde, ist für den Servicenehmer nicht relevant.
- *Offenheit und Standardisierung*: Für die Austauschbarkeit der Services sollten für die Implementierung der Interaktion mit diesem Service die offenen allgemein akzeptierten Standards verwendet werden (z.B. WSDL[75] oder SOAP[76]).
- *Wiederverwendbarkeit*: Verwendung derselben Implementierung gekapselter Funktionalitäten in unterschiedlichen Anwendungen bzw. Geschäftsprozessen.
- *Lose Koppelung*: Autonomie der miteinander verbundenen Services zur Vermeidung von Abhängigkeiten. Lose gekoppelte Services lassen sich leicht austauschen bzw. flexibel miteinander kombinieren.
- *Komposition*: Verkettung von wiederverwendbaren Services zu Teilprozessen oder durchgängigen Prozessketten. Aufgrund der Eigenschaften der Services können geänderte Anforderungen an Prozesse oder Anwendungen flexibel und schnell neu zusammengestellt werden.

---

[75] WSDL - Web Services Description Language: Eine auf XML basierende Sprache für die standardisierte Definition von Webdiensten.

[76] SOAP - Simple Object Access Protocol: Netzwerkprotokoll für den Zugriff auf Systeme sowie den Austausch von Daten.

Nach [GrSh06] zerlegt eine serviceorientierte Architektur (SOA) Anwendungen in grobkörnige Servicekomponenten, deren Interaktionen bidirektional und asynchron erfolgen können. SOA ist jedoch keine Technologie oder Plattform, die käuflich erworben werden kann. Es ist vielmehr ein Konzept oder Paradigma, das existierende Konzepte und Praktiken für bestimmte Anforderungen in einen gemeinsamen Kontext bringt.

Für die Definition einer SOA wird die Beschreibung von [Josu07] übernommen:

> SOA ist ein Architekturparadigma für den Umgang mit Geschäftsprozessen, die über große Landschaften mit existierenden und neuen heterogenen Systemen verteilt werden, wobei die Systeme unterschiedliche Eigentümer haben.

Definition 13: Serviceorientierte Architektur

3.8.1.3 Enterprise Service Bus

Eine der entscheidenden Eigenschaften einer SOA ist die lose Koppelung von Services. Sie ermöglicht die flexible Komposition von Services und damit die effiziente Wiederverwendung von Funktionalität. Zur Vermeidung von Abhängigkeiten zwischen der Anwendung, bzw. allgemein dem Servicenutzer, und dem Service wird ein separater technischer Vermittler benötigt. Auf diese Weise müssen weder der Servicenehmer noch der Service voneinander Kenntnis haben.

Dieser Vermittler ist ein wesentlicher Bestandteil einer serviceorientierten Infrastruktur und wird als Enterprise Service Bus (ESB) bezeichnet. Es ist ein technischer Dienst, der von verschiedenen Softwareherstellern zur Verfügung gestellt wird. Nach [Josu07] ist ein ESB dafür verantwortlich, dass Nutzer die von Anbietern bereitgestellten Services aufrufen können. Für diesen Zweck sind verschiedene Aufgaben durch den ESB zu erfüllen:

- Aufbau einer Verbindung zwischen Servicenehmer und -anbieter.
- Transformation von Daten zwischen Nutzer und Anbieter für den Fall unterschiedlicher Beschreibung von Daten.
- Einhaltung der Sicherheitsrichtlinien beim Aufruf von Services.
- Möglichkeit zur Überwachung und Protokollierung der Abläufe sowie der Fehleranalyse.

3.8.1.4 Gestaltung der Anwendungslandschaften

Nach [EHHJ08] stellt die Serviceorientierung eine Methode zur Strukturierung von Anwendungslandschaften. Sie ermöglicht einen Übergang von der rein technischen Orientierung hin zu einer fachlich orientierten Sichtweise der IT. Die

Serviceorientierung geht somit über die technische Betrachtung der Anwendungsentwicklung hinaus und ist ein wichtiger Bestandteil der IT-Strategie für die zukünftige Ausrichtung der Anwendungslandschaft.

Der Brückenschlag zwischen der IT und den Anforderungen der Fachbereiche ist ein Thema, das insbesondere im Rahmen des IT-Architektur-Managements behandelt wird. SOA kann hierzu einen wichtigen Beitrag liefern (siehe auch [EHHJ08]). Die Analyse der Geschäftsprozesse sowie das Vorgehen zur Abbildung der Prozesse auf IT-Anwendungen ist jedoch nicht Thema dieser Arbeit. Daher wird auch auf das Thema Business-Prozess-Management (BPM) nicht näher eingegangen. Weitere Informationen hierzu sind in [Josu08] oder [SRMC09] zu finden. Der Fokus in dieser Stufe liegt auf der Bereitstellung, der Komposition sowie der Nutzung von Services auf einer technischen Ebene unter Einsatz der Automatisierung.

Eingangs wurde erwähnt, dass das Risiko der Automatisierung eine potentielle Überladung der IT-Landschaft mit zum Teil redundanten Anwendungen und Zuständigkeiten ist. Aufgrund der zentralen Bereitstellung von Services ist es daher wesentlich, eine klare Vorstellung der Struktur der IT-Anwendungslandschaft sowie eine eindeutige Zuordnung von Verantwortlichkeiten für spezifische Services zu haben.

Eine solche Struktur wird durch die Bildung von Domänen erreicht. In Abschnitt 2.6 wurden Domänen eingeführt, die Komponenten nach fachlichen Gesichtspunkten gruppieren. Die Komponenten dieser Definition spezifizieren dabei die Services in einer SOA. An dieser Stelle seien nochmals die Eigenschaften der Domänen erwähnt:

- Vollständige und redundanzfreie Strukturierung der Fachlichkeit auf logischer Ebene.
- Gezielte Zuweisung von Anforderungen, Daten und IT-Systemen zur Unterstützung übergreifender Steuerungsprozesse des Managements.
- Kapselung fachlich eng gekoppelter Funktionen und Geschäftsobjekte sowie Bereitstellung von Leistungen in Form von Services.

Die Vertreter der verschiedenen Geschäftsbereiche haben zusammen mit den IT-Architekten eine Aufteilung der Geschäftsfunktionalitäten auf die Domänen zu erstellten. Ein Beispiel einer Domäne wurde in Abschnitt 2.6 erläutert. Für die Domänen bzw. Subdomänen ist zusätzlich ein Verantwortlicher zu benennen, der die Bebauung dieser Domäne überwacht.

Bei der Realisierung von Anwendungen in dieser Stufe ist somit sicherzustellen, dass sowohl Services als auch Anwendungen genau einer Domäne zugeordnet werden. Hierdurch wird eine unkoordinierte Bebauung der IT-Landschaft vermieden.

3.8.2 Service Driven Development

Die fünfte Stufe der Industrialisierung basiert auf den Prinzipien der Serviceorientierung. Anwendungen werden dabei nicht mehr als eigenständiges Konstrukt betrachtet, sondern setzen sich aus verschiedenen, bereits existierenden Services zusammen. Der Entwickler hat somit die Aufgabe, die Unterstützung der Geschäftsprozesse auf Basis der bestehenden Services zu gestalten.

Die Steigerung der Industrialisierung erfolgt in dieser Stufe durch die Einbindung bzw. die Komposition von Services zu einer Anwendung. Die Methode für die Komposition ist das Service Driven Development:

> Die Entwicklung von Anwendungen besteht im Wesentlichen aus der Komposition bereits existierender Services. Diese werden erst zur Laufzeit in die Anwendung eingebunden. Sofern eine Funktionalität noch nicht verfügbar ist, ist ein entsprechender Service zu erstellen.

Definition 14: Service Driven Development

In dieser Stufe stehen die verschiedenen Services bereits als Modelle zur Verfügung. Die Komposition durch den Entwickler erfolgt daher auf Basis einer Modellierung der Anwendung. Aus diesem Modell wird anschließend der entsprechende Quelltext für die Anwendung generiert, der insbesondere den jeweiligen Aufruf der integrierten Services enthält. Die Stufe „Automatisierung“ ist daher eine wichtige Voraussetzung für diesen Industrialisierungsgrad.

Die Serviceorientierung und damit die Möglichkeit, Anwendungen zu komponieren, stellt in dieser Arbeit die höchste Stufe der Industrialisierung in der Anwendungsentwicklung dar. Die verschiedenen Prinzipien werden dabei folgendermaßen unterstützt:

- *Standardisierung*: Die Festlegung der Domänen in der Anwendungslandschaft sowie die eindeutige Zuordnung von Anwendungen zu einer Domäne ermöglicht eine klare und transparente Strukturierung der Fachlichkeit. Die Standardisierung bezieht sich in dieser Stufe somit nicht nur auf Technologien oder Plattformen, sondern wird um Fachlichkeit in Form von Services erweitert. Diese Fachlichkeit wird damit nicht mehrfach in unterschiedlichen Formen realisiert, sondern steht den Entwicklern zentral zur Verfügung.
- *Automatisierung*: Eine weitere Steigerung der Automatisierung erfolgt in dieser Stufe durch die Möglichkeit der Komposition von Services. Bereits auf den bestehenden Modellen der Services können neue Anwendungen komponiert und anschließend generiert werden.

- *Wiederverwendung*: Die bisherige Wiederverwendung erfolgte auf Basis der Integration von Komponenten, oder allgemein Artefakten, in die Anwendung. Die Wiederverwendung dieser Stufe integriert nicht die Artefakte, sondern lediglich die Funktionalität. Der Entwickler ist dabei nur für die Einbindung, nicht aber für den Service selbst zuständig.
- *Arbeitsteilung*: Eine Arbeitsteilung in dieser Stufe erfolgt sowohl in der Entwicklung als auch im Betrieb. Die Entwicklung verteilt sich auf die Bereitstellung von Services sowie die Komposition von Anwendungen. Diese Aufteilung ist auch charakteristisch für den getrennten Betrieb von Services und Anwendungen.
- *Verringerung der Fertigungstiefe*: Aufgrund der Eigenständigkeit können auch externe Dienstleister Services anbieten. Diese Services können dabei innerhalb oder außerhalb der Unternehmensgrenzen bereitgestellt werden. Auf diese Weise können Funktionalitäten ausgelagert und somit die Fertigungstiefe reduziert werden.

Durch die Einbindung verfügbarer Services zur Laufzeit in eine Anwendung werden die Aufgaben zur Qualitätssicherung an diese Services delegiert. Die Sicherstellung der Qualität reduziert sich damit auf die verbleibenden Entwicklungen in der Anwendung. Für die einzelnen Qualitätsfaktoren bedeutet dies:

- *Funktionalität:* Die Funktionalität der Services ist bekannt. Die Angemessenheit sowie die Richtigkeit werden bereits bei der Auswahl der Services bestimmt. Die geforderten Sicherheitsaspekte sind in dem Service bereits realisiert und getestet. Der Entwickler ist lediglich für die Komposition der Services zuständig und hat dabei die übergreifenden Sicherheitsaspekte zu beachten.
- *Zuverlässigkeit:* Die Fehlertoleranz sowie die Fehleranfälligkeit für die durch Services bereitgestellten Funktionalitäten werden an zentraler Stelle optimiert und überwacht. Die Sicherstellung der Zuverlässigkeit reduziert sich damit auf die in der Anwendung zu entwickelnden Funktionalitäten sowie die Einbindung der Services.
- *Benutzbarkeit:* Sofern Services eine Benutzeroberfläche anbieten, die in Anwendungen integriert wird, kann die Benutzbarkeit durch ein einheitliches Konzept bei der Bedienung der Anwendung erhöht werden. Während bei der Automatisierung noch der Entwickler für die Gestaltung der Oberflächen zuständig war, werden diese bei den Services bereits geliefert. Bei der Freigabe der Services ist dabei jedoch darauf zu achten, dass die Bedienung der Oberflächen den Vorgaben entspricht. Ferner muss sich der Entwickler ebenfalls an diesen Vorgaben orientieren.

- *Effizienz:* Das Zeit- sowie das Verbrauchsverhalten der eingebundenen Services werden an einer zentralen Stelle optimiert. Es liegt nicht mehr in der Zuständigkeit des Entwicklers. Betrachtet werden nur noch die in der Anwendung verbleibenden Teile sowie die Integration der Services.
- *Änderbarkeit:* Die Wartung der Services erfolgt an zentraler Stelle und ist damit keine Aufgabe der Anwendung. Solange sich die Schnittstellen eines Services nicht ändern, kann eine Wartung an diesem Service jederzeit ausgeführt werden und steht sofort für alle nutzenden Anwendungen zur Verfügung. Problemfälle, die durch Services verursacht werden, können schneller lokalisiert werden, da sie jeweils über ein eigenes Fehlermanagement verfügen. Der Service befindet sich bereits im Einsatz und hat die Stabilität bereits unter Beweis gestellt. Der Entwickler hat sich bzgl. der Anforderungen, die durch den Service abgedeckt werden, nicht mehr um die Stabilität zu kümmern.
- *Übertragbarkeit:* Services werden zur Laufzeit von Anwendungen eingebunden. Sie können daher durch andere Services ausgetauscht werden, ohne die Anwendung wesentlich zu modifizieren. Die Einhaltung von Standardtechnologien für den Aufruf eines Services ermöglicht deren Nutzung über Technologiegrenzen hinweg. Aufgrund der losen Koppelung der Anwendung mit dem Service ist zudem eine schnellere Reaktionsfähigkeit auf Änderungen der Funktionalitäten möglich.

Auf die Faktoren Zeit und Kosten hat die Nutzung von eigenständigen Services die folgenden Auswirkungen:

- *Entwicklungszeit*: Sofern die geforderte Funktionalität durch einen zentralen Service abgedeckt werden kann, ist eine eigene Programmierung nicht notwendig. In diesem Fall reduziert sich die Entwicklung auf die Einbindung des zentralen Services in die Anwendung.
- *Kosten*: Die Kosten der Anwendungsentwicklung bei der Komposition werden aufgrund der reduzierten Entwicklungszeit ebenfalls gesenkt. Durch die eigenständige Pflege der Services fallen die Kosten für das Betreiben des Services nicht bei der Entwicklung der verschiedenen Anwendungen an, sondern werden an zentraler Stelle optimiert.

3.8.3 Erstellen von Artefakten auf Basis der Komposition

Services stellen Funktionalitäten zur Verfügung, die von Anwendungen zur Laufzeit eingebunden werden. Vor der Realisierung einer Anwendung ist daher zunächst eine Analyse notwendig, welche der geforderten Funktionalitäten durch bereits vorhandene Services abgedeckt werden können.

Abbildung 21 stellt schematisch den Ablauf der Erstellung von Artefakten auf Basis der Serviceorientierung dar. Die Grafik sieht zunächst der Darstellung aus Stufe vier, der Automatisierung, recht ähnlich. Der wesentliche Unterschied ist jedoch, dass nur noch die Bestandteile einer Anwendung erstellt werden, die im Unternehmen noch nicht als Service zur Verfügung stehen. Existierende Services werden zur Laufzeit eingebunden und der Anwendung zur Verfügung gestellt. Für diese Services sind keine Artefakte zu erstellen.

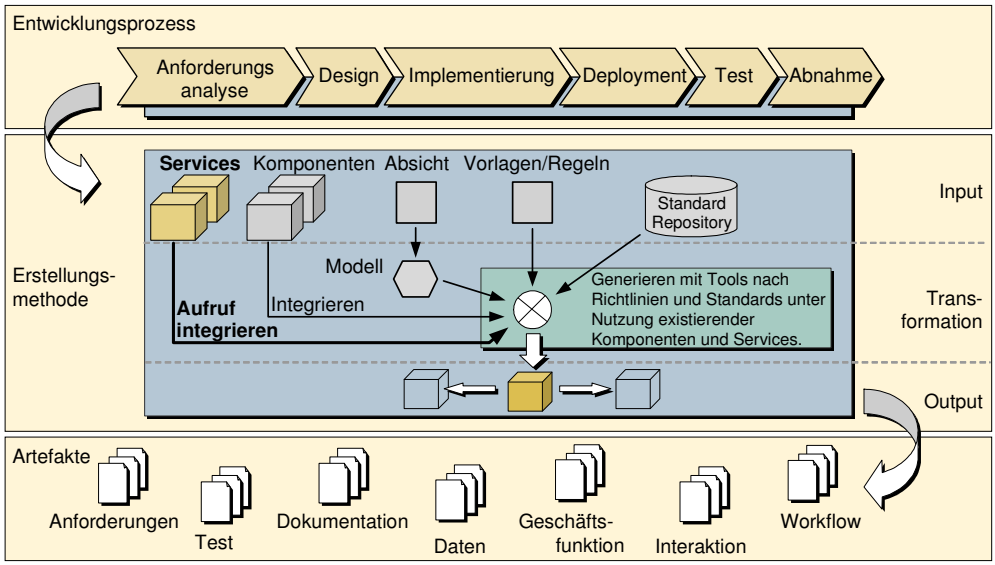

Abbildung 21: Service Driven Development

Die Erstellung der Artefakte auf Basis des Service Driven Developments basiert auf folgendem Schema:

Input

Neben den Anforderungen an die Anwendung stehen dem Entwickler die existierenden Services, z.B. mittels eines Kataloges, zur Verfügung. Dies können interne aber auch externe, von anderen Unternehmen bereitgestellte, Services sein.

Ferner hat der Entwickler Zugriff auf die benötigten Standards und Richtlinien der Entwicklung auf Basis der Komposition. Die Vorlagen und Regeln für die automatische Erzeugung der Anwendung wurden für die Komposition angepasst und stehen ebenfalls über eine zentrale Plattform zur Verfügung.

Komponenten sollten weitestgehend durch Services ersetzt werden. Sofern es dennoch sinnvoll ist, bestimmte Komponenten zu verwalten, können diese ebenfalls in die Generierung der Anwendung eingebunden werden.

Transformation

Auf Basis der Anforderungen wird analysiert, welche der vorhandenen Services dem Unternehmen zur Verfügung stehen. Aufbauend auf der Automatisierung werden in der Modellierung der Anwendung die verfügbaren Services integriert.

Funktionalitäten, die weder durch bestehende Services noch durch Komponenten abgedeckt werden können, werden auf Basis der Automatisierung für diese Anwendung erstellt. Bei der Generierung werden die für die Anwendung spezifischen Artefakte sowie die Aufrufe der Services erstellt und zu einer Anwendung komponiert. Analog der Automatisierung wird die Anwendung aus den generierten Quelltexten gebaut.

Die Artefakte des Entwicklungsumfeldes der Services, wie z.B. Tests oder Dokumentation, werden ebenfalls eingebunden. Für Entwicklungen, die spezifisch für die Anwendung anfallen, werden diese Artefakte analog der Automatisierung erstellt.

Output

Ergebnis der Transformation ist eine Anwendung, deren Funktionalität zu einem großen Teil aus eigenständigen Service besteht, die zur Laufzeit eingebunden werden.

Für die spezifischen Entwicklungen der Anwendung stehen die modellierten bzw. generierten Artefakte des Entwicklungsumfeldes zur Verfügung. Die bereits existierenden Testfälle sowie die Dokumentationen der Services werden referenziert.

3.8.4 Rahmenbedingungen für die Komposition

Die Besonderheit in dieser Stufe der Industrialisierung ist die Einbindung von Funktionalität, deren Ausführung durch eine zentrale Serviceinstanz koordiniert wird. Modifikationen an diesen Services haben daher direkten Einfluss auf die Anwendungen. Zwar wurden im Rahmen der Wiederverwendung ebenfalls Funktionalitäten eingebunden, die entsprechenden Komponenten wurden dabei jedoch vollständig in die Anwendung integriert. Zentrale Änderungen an diesen Komponenten hatten somit keine unmittelbaren Auswirkungen auf die Anwendung.

Die Einführung der Komposition erfordert daher die folgenden Voraussetzungen für den erfolgreichen Einsatz des Service Driven Developments.

3.8.4.1 Gremium

Die Einführung sowie der Betrieb der Komposition erfordern zusätzlich zu den Aufgaben der Automatisierung weitere Regelungen, die durch ein Gremium zu koordinieren sind. Da die Services zentral Funktionalitäten für alle Anwendungen zur Verfügung stellen, ist ein Serviceportfolio zu definieren, welche der Funktio-

nalitäten durch welche Services bereitgestellt werden. Dieses Portfolio basiert auf der Struktur der Domänen des Unternehmens. In jeder Domäne wird somit festgelegt, welche Funktionalität zur Verfügung gestellt wird. Die Gestaltung der Domänenstruktur sowie die Pflege des Portfolios sind Aufgaben des Gremiums und erfolgen in Abstimmung mit Vertretern der Geschäftsbereiche.

Für die Nutzung der Services ist ferner die Entwicklungs- und Laufzeitinfrastruktur einzurichten sowie entsprechende Standards und Richtlinien für die Entwicklung auf Basis von Services zu definieren.

Die Überprüfung der Einhaltung der Vorgaben sowohl für die Bereitstellung als auch die Nutzung der Services wird durch das Gremium koordiniert.

3.8.4.2 Prozesse

Die zentrale Nutzung von Services erfordert abgestimmte Verfahren zur Bereitstellung sowie zur Nutzung von Services.

- *Beantragung von Services:* Um eine mögliche Redundanz an Funktionalitäten der Services zu vermeiden, ist in einem Prozess zu koordinieren, welche Services für die Einbindung in Anwendungen freigegeben werden. Als Entscheidungsbasis wird das Serviceportfolio auf Basis der Domänenstruktur verwendet.
- *Bereitstellung von Services:* Da Services zentral zur Verfügung gestellt werden, haben sie die vorgegebenen Eigenschaften zu erfüllen. Vor einer Freigabe zur Nutzung durch Anwendungen wird analysiert, ob die Anforderungen an einen Service erfüllt sind. Bei positiver Prüfung wird dieser freigegeben.
- *Modifikation von Services:* Die Veränderung eines Services an der Schnittstelle hat zur Folge, dass alle Anwendungen, die diesen Service nutzen, angepasst werden müssen. Um den damit verbundenen Aufwand zu reduzieren, kann der Service zwei oder drei Instanzen gleichzeitig zur Verfügung stellen.

  Die Modifikation von Services ist daher durch einen Prozess zu koordinieren, bei dem alle Anwendungen informiert werden, die den Service nutzen.

3.8.4.3 Entwicklungsinfrastruktur

Die Bereitstellung und Einbindung von Services in die Anwendung erfordert zusätzliche Elemente der Entwicklungsinfrastruktur:

- *Standards und Richtlinien:* Die Vorgaben für die Erstellung sowie die Nutzung von Services sind zu definieren und allen Entwicklern zentral zugänglich zu machen.

- *Konfigurationsmanagement:* Im Gegensatz zu den Komponenten der dritten Stufe (Wiederverwendung) werden bei der Komposition für die Anwendungen nur Informationen über den Service bereitgestellt, nicht der Service selbst. Hierzu wird ein Service-Repository[77] benötigt, welches den Werkzeugen der Entwicklungsinfrastruktur Informationen über Datentypen, Schnittstellen, Nachrichtentypen, etc. für den Aufruf des Services zur Verfügung stellt.
- *Entwicklungswerkzeuge:* Für die Nutzung von Services in Anwendungen werden Werkzeuge benötigt, die auf Basis der Informationen aus dem Service-Repository die bereitgestellte Funktionalität einbinden können. Durch die Auswahl eines Services wird über diese Werkzeuge die Logik der Übergabe von Parametern an den Service automatisch in die Anwendung integriert.
- *Zentraler Build-Service:* Der zentrale Build-Service erstellt anhand eines Bauplanes, analog der Stufe Automatisierung, die Anwendung anhand eines Modells sowie der Vorlagen und Regeln. Die Nutzung von Services erfordert dabei keine spezielle Voraussetzung. Dies gilt auch für die Bereitstellung von Services.
- *Transport und Deployment:* Für diese Stufe gibt es keine zusätzlichen Anforderungen. Bei dem Deployment von Services ist darauf zu achten, dass diese sowohl installiert, als auch an dem zentralen Serviceverzeichnis (siehe Abschnitt 3.8.4.4) registriert werden. Ferner ist zu prüfen, wie viele Instanzen des Services bereits zur Verfügung stehen und ob mit dem Deployment einer neuen Version eine ältere Version abgeschaltet werden muss.

3.8.4.4 Laufzeitinfrastruktur

Services sind eigenständige Dienste, die unabhängig von den Anwendungen, die sie aufrufen, betrieben werden. Hierfür ist eine Laufzeitinfrastruktur nötig, die einerseits die Services zur Verfügung stellt, andererseits die Einbindung in die Anwendung ermöglicht.

- *Serviceverzeichnis (auch Service-Registry genannt):* Das Serviceverzeichnis verfügt über Funktionen zur Veröffentlichung, Klassifizierung sowie zum Auffinden von Services. Es gibt Auskunft darüber, welche Services in der IT-Landschaft aufrufbar sind. Dadurch wird die dynamische Einbindung von Services zur Laufzeit ermöglicht.

[77] Service-Repository: Zentrale Ablage von Informationen über Services in Form von Modellen. Diese Informationen werden benötigt, um Services in eine Anwendung während der Entwicklung einzubinden.

- *Enterprise Service Bus:* Die lose Koppelung zwischen Anwendungen ermöglicht eine flexible Gestaltung von Geschäftsprozessen. Die technische Umsetzung dieser losen Kopplung erfolgt auf Basis eines Enterprise Services Bus. Dieser stellt die Verbindung zwischen Anwendungen und Services zur Verfügung, ohne dass diese direkt miteinander verbunden sind.

3.8.5 Reifegrade der Komposition

Die Komposition stellt die höchsten Anforderungen an die Anwendungsentwicklung im Rahmen der Industrialisierung. Die effiziente Nutzung von Services, auf die der Entwickler keinen Einfluss hat, setzt die Etablierung der Infrastrukturen, Verfahren sowie Vorschriften voraus.

- *Initial:* Die Methode ist im Unternehmen nicht bekannt. Das Gremium sowie die benötigten Prozesse sind nicht definiert. Die Entwicklungsinfrastruktur bietet keine Unterstützung für diese Stufe.
- *Wiederholbar:* Die Methode ist wenigen Entwicklern bekannt. Vereinzelt werden Services auf Basis von Komponenten zur Verfügung gestellt, die von verschiedenen Anwendungen genutzt werden können. Es gibt jedoch noch keine Systematik hinter der Nutzung von Services. Die Bereitstellung der Services sowie die Nutzung basieren nicht auf Standards oder Richtlinien. Die genutzte Infrastruktur ist sehr individuell auf den aktuellen Bedarf ausgelegt. Ein zentrales Gremium sowie die benötigten Prozesse sind nicht zentral definiert.
- *Definiert:* Ein Gremium für die Koordination der Komposition wurde eingerichtet und die Aufgaben sowie Zuständigkeiten definiert. Die für die Komposition notwendigen Prozesse sind festgelegt und können genutzt werden. Die Standards und Richtlinien wurden durch das Gremium bestimmt und können in der Anwendungsentwicklung verwendet werden.

  Die Entwicklungs- und Laufzeitinfrastruktur wurde auf Basis der Vorgaben eingerichtet und kann genutzt werden. Erste Services wurden erstellt und werden über das Portfolio verwaltet. Teilweise werden die Services in Anwendungen eingesetzt. Es gibt keine Verpflichtung für alle Anwendungen, die Strukturen und Vorgaben der Komposition einzusetzen.
- *Eingeführt:* Die Methode ist allen Entwicklern bekannt. Die Anwendungsentwicklung basiert auf der Serviceorientierung und richtet sich nach den Standards und Richtlinien der Komposition. Alle Services werden zentral über ein Portfolio definiert und verwaltet. Die Anwendungen nutzen die über die zentrale Infrastruktur bereitgestellten Services, um die geforderten Funktionalitäten abzubilden.

Die Prozesse zur Bereitstellung und Verwaltung der Services werden genutzt. Die Konsistenz sowie die Aktualität der verschiedenen Artefakte sind gewährleistet. Dokumentation und Test entsprechen dem Stand der Anwendung.

- *Optimiert:* Rückmeldungen zu Services, aber auch zu Entwicklungs- und Laufzeitinfrastruktur werden systematisch erfasst und bearbeitet. Die Optimierung auf Basis dieser Rückmeldung wird durch das Gremium koordiniert. Das Portfolio wird regelmäßig analysiert. Ist die Nutzung von Services zu gering, werden sie aus dem Portfolio entfernt und stehen nicht mehr zur Verfügung[78].

3.8.6 Grenzen der Industrialisierung

Die Serviceorientierung stellt in diesem Modell die höchste Stufe der Industrialisierung in der Anwendungsentwicklung dar. Die Grenzen der Industrialisierung wurden bereits im Abschnitt 2.2.1 beschrieben.

Der Aufwand, der initial für die Etablierung einer Serviceorientierung investiert werden muss, ist nicht für alle Bereiche eines Unternehmens gleichermaßen gerechtfertigt. Es ist daher zu klären, wie hoch der Sollgrad der Industrialisierung in den verschiedenen Bereichen (Domänen) sein sollte.

Die Bestimmung eines Sollgrades pro Domäne ist ein wichtiger Bestandteil der Ermittlung des Industrialisierungsgrades. Dies wird im nächsten Kapitel näher beleuchtet.

[78] Der Service wird nicht gelöscht, sondern lediglich aus dem Serviceportfolio entfernt. Die Betreuung des Services wird eingestellt, da der Pflegeaufwand im Vergleich zum Nutzen zu hoch ist. Er sollte damit nicht mehr eingebunden werden.

4 Ermittlung des Industrialisierungsgrades

Voraussetzung für eine Optimierung der Anwendungsentwicklung ist die Kenntnis über den derzeitigen Zustand. Das Stufenmodell aus Kapitel 3 bildet einen Rahmen für die Ermittlung des aktuellen Industrialisierungsgrades. Eine weitere Voraussetzung für eine Optimierung ist die Bestimmung eines Sollindustrialisierungsgrades. Dieser gibt vor, welcher Grad an Industrialisierung erreicht werden soll. Sofern sich dieser vom aktuellen Industrialisierungsgrad unterscheidet, sind entsprechende Maßnahmen einzuleiten.

In diesem Kapitel werden die Schritte erläutert, die zu einer Bewertung bzw. Einstufung der Anwendungsentwicklung führen. Dies beinhaltet auch die Bestimmung des Sollindustrialisierungsgrades. Eine regelmäßige Durchführung dieser Ermittlung ermöglicht eine Steuerung der Optimierung. Dabei wird analysiert, ob die eingeleiteten Maßnahmen eine Verbesserung bewirkt haben.

Die Anforderungen an die Anwendungsentwicklung sind nicht für alle Bereiche des Unternehmens gleich. Dies wurde bereits in Abschnitt 2.6 beschrieben. Dort wurde eine Einteilung des Unternehmens in Domänen vorgeschlagen. Domänen ermöglichen eine redundanzfreie Strukturierung der Fachlichkeit des Unternehmens auf logischer Ebene. Die Fachlichkeit in den Domänen ist in den meisten Unternehmen umfangreich. Daher werden in jeder Domäne weitere Domänen, die sogenannten Subdomänen, identifiziert. Diese Hierarchie kann beliebig fortgesetzt werden. Dabei ist wesentlich, dass eine für das Unternehmen sinnvolle Struktur erstellt wird. Die Anwendungen werden den untersten Strukturen zugeordnet.

In dieser Arbeit wird die Betrachtung auf zwei Ebenen begrenzt, die Domänen und die Subdomänen. Für diese Bereiche ist jeweils der Industrialisierungsgrad zu bestimmen. Anwendungen werden den Subdomänen zugeordnet.

Die Bewertung einer Domäne erfolgt in mehreren Schritten (siehe Abbildung 22 und Abbildung 24). Sofern das Unternehmen noch nicht über eine Domänenlandschaft verfügt, sind im ersten Schritt die verschiedenen Domänen anhand des in Abschnitt 2.6 beschriebenen Verfahrens zu identifizieren[79].

Die Anwendungen einer Subdomäne basieren auf Technologien, Plattformen, Infrastrukturen und werden unter Berücksichtigung von Standards und Richtlinien erstellt. Diese Strukturelemente und Vorgaben werden meist für das ganze

[79] Die Identifizierung von Domänen ist meist ein langwieriger Prozess und bedarf der intensiven Kommunikation mit allen Beteiligten. Wie in Abschnitt 2.6 beschrieben, wird in dieser Arbeit auf diesen Vorgang nicht näher eingegangen.

Unternehmen festgelegt und sind damit domänenübergreifend gültig. Dies gilt insbesondere für Firmen, deren IT zentral organisiert ist[80].

In einem zweiten Schritt ist somit domänenübergreifend zu klären, welchen Grad an Industrialisierung die vorhandenen Entwicklungs-[81] und Laufzeitinfrastrukturen unterstützen. Darin enthalten ist die Bewertung der benötigten Gremien und Prozesse. Sie sind die Grundlage für die Bewertung der Infrastrukturen. Eine Entwicklungsinfrastruktur kann beispielsweise nicht standardisiert sein, wenn es kein Gremium gibt, das eine Standardisierung durchführt.

Die Entwicklungs- und Laufzeitinfrastrukturen stellen die Werkzeuge, Richtlinien, Technologien, Plattformen etc. zur Verfügung, die für die Umsetzung der jeweiligen Methode benötigt werden. Eine Subdomäne kann daher maximal jenen Grad erreichen, der durch die Infrastrukturen, welche in der Subdomäne zum Einsatz kommen, unterstützt wird.

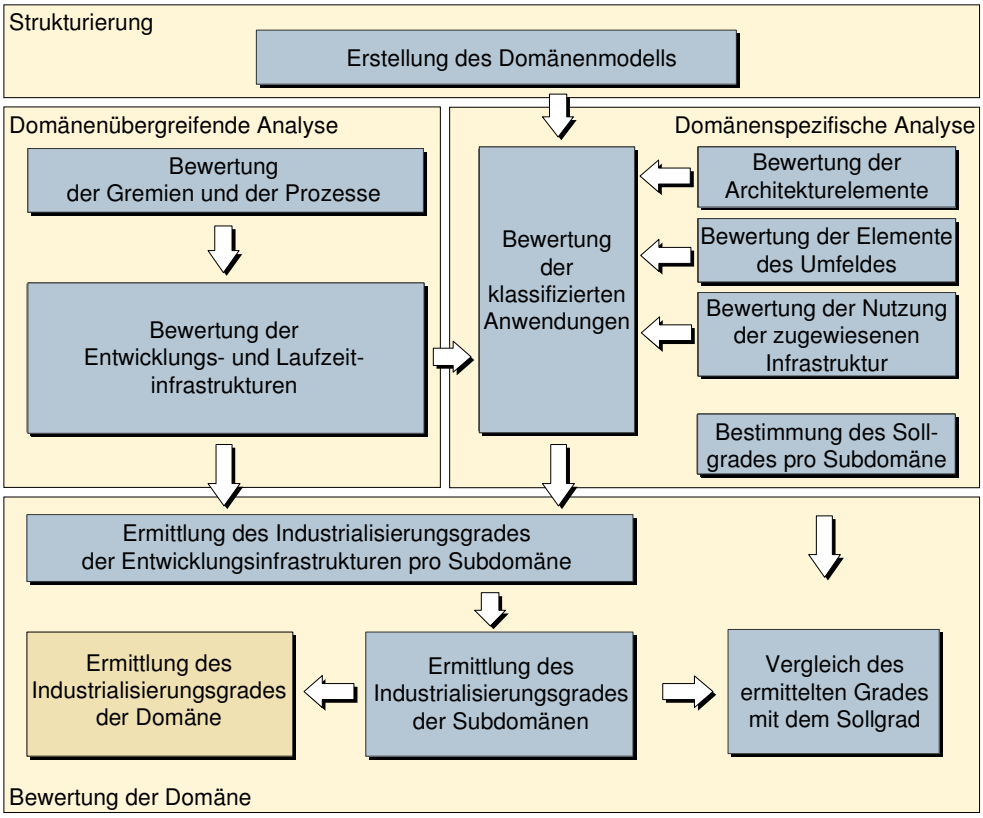

Abbildung 22: Schritte zur Ermittlung des Industrialisierungsgrades

[80] In Konzernen mit vielen verschiedenen Bereichen und dezentraler IT können durchaus unterschiedliche Vorgaben existieren. Das Vorgehen ist auch hier anwendbar, jedoch sind verschiedene zusätzliche Schritte notwendig, die im Rahmen dieser Arbeit nicht näher betrachtet werden.

[81] Unterschiedliche Technologien benötigen meist eigene Entwicklungsinfrastrukturen mit entsprechenden Vorgaben und Prozessen. Die Umsetzung der Methoden kann pro Technologie potentiell unterschiedlich erfolgen.

Die für eine Domäne spezifischen Untersuchungen werden im dritten Schritt durchgeführt. Hierbei wird nach der Erfassung der Anwendungen zunächst bestimmt, welcher Sollgrad für die Subdomänen benötigt wird.

Die anschließende Analyse der Anwendungen in der betrachteten Subdomäne erfolgt auf Basis der verschiedenen Artefakte. Es wird ermittelt, welche der Methoden bei der Erstellung der Artefakte verwendet wurde. Für die Entwicklung von Anwendungen stellt die zugehörige Infrastruktur die Voraussetzungen zur Verfügung. Der Reifegrad dieser Infrastruktur ist daher ein wichtiger Aspekt für die Bewertung der Anwendungen. Ferner wird die Nutzung dieser Infrastruktur näher beleuchtet.

Die Ermittlung des Industrialisierungsgrades erfolgt im vierten Schritt aufgrund der Bewertung der Domäne bzw. Subdomäne. Die Ergebnisse der domänenspezifischen sowie der domänenübergreifenden Analyse liefern hierzu unterschiedliche Beiträge. Während bei den Entwicklungsinfrastrukturen die Vollständigkeit ein wichtiges Kriterium ist, wird bei den Anwendungen analysiert, inwiefern die Vorgaben in die Realisierung übernommen wurden und damit im Unternehmen etabliert sind. Grundlage dieser Analysen bilden die in Abschnitt 3.2.2 vorgestellten Reifegradkriterien. Hierbei werden die folgenden Aspekte überprüft:

- Vollständigkeit

  Domänenübergreifend: Verfügbarkeit der Entwicklungsmethode für alle Artefakte. Definition bzw. Beschreibung der benötigten Gremien und Prozesse, der Vorgaben sowie der vorgesehenen Entwicklungs- und Laufzeitinfrastrukturen.

  Domänenspezifisch: Vollständigkeit der Artefakte der betrachteten Anwendungen bzgl. der definierten Methode auf Basis der vorgesehenen Entwicklungsinfrastruktur.

- Etabliertheit

  Domänenübergreifend: Überprüfung der aktiven Gestaltung der Infrastrukturen durch Gremien und Prozesse. Die Etablierung von Infrastrukturen ergibt sich jedoch aus der Nutzung in der Anwendungsentwicklung. Dieser Aspekt kann für Infrastrukturen daher nicht separat, sondern nur in Zusammenhang mit den domänenspezifischen Realisierungen von Anwendungen untersucht werden.

  Domänenspezifisch: Einhaltung der Vorgaben bei der Entwicklung von Anwendungen einer Subdomäne sowie der Nutzung der vorgegebenen Entwicklungs- und Laufzeitinfrastrukturen.

- Konsistenz

  Domänenübergreifend: Einheitliche Vorgaben und gleiche Verfügbarkeit für alle Entwickler. Konformität der Vorgaben zur Entwicklungs- und Laufzeitinfrastruktur.

  Domänenspezifisch: Einheitliche Umsetzung der vorgegebenen Methode in den Artefakten der Anwendungen einer Subdomäne. Die Artefakte einer Anwendung sind widerspruchsfrei zueinander.

- Aktualität

  Domänenübergreifend: Aktualität der verfügbaren Vorgaben für die verschiedenen Methoden.

  Domänenspezifisch: Aktualität aller Artefakte einer Anwendung.

Die Anwendungen einer Subdomäne können in verschiedenen Technologien realisiert sein und basieren daher auf unterschiedlichen Entwicklungsinfrastrukturen. Die unterste Ebene der Ermittlung des Reifegrades für die einzelnen Stufen findet daher pro Entwicklungsinfrastruktur statt. Der Industrialisierungsgrad einer Subdomäne ergibt sich aus der Zusammenfassung der Analyseergebnisse der enthaltenen Infrastrukturen. Die Bewertungen der einer Domäne zugehörigen Subdomänen werden anschließend zum Industrialisierungsgrad der Domäne aggregiert.

Für die Ermittlung des Industrialisierungsgrades einer Domäne sind somit die folgenden Schritte notwendig (Detailschritte in Abbildung 24):

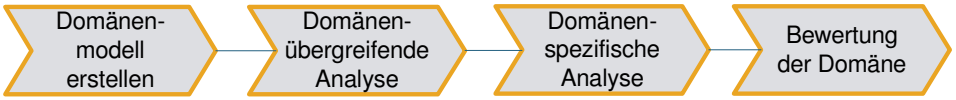

Abbildung 23: Ermittlung des Industrialisierungsgrades

Schritt 1: Domänenmodell erstellen

- Erstellung des Domänenmodells, sofern noch nicht vorhanden.

Schritt 2: Domänenübergreifende Analyse

- Identifizierung der im Unternehmen vorhandenen Gremien und Bewertung der wahrgenommenen Aufgaben.
- Identifizierung und Bewertung der vorhandenen Prozesse zur Etablierung der Infrastrukturen.
- Identifizierung und Bewertung der existierenden Entwicklungsinfrastrukturen, welche die Grundlagen für die Anwendungsentwicklung bereitstellen.
- Identifizierung und Bewertung der Infrastrukturen, die durch die Anwendungen zur Laufzeit genutzt werden.

Schritt 3: Domänenspezifische Analyse

- Erfassung von Anwendungen des Unternehmens in den einzelnen Subdomänen.
- Klassifikation der Anwendungen bzgl. der Relevanz für das Unternehmen.
- Bestimmung des Sollindustrialisierungsgrades pro Subdomäne.
- Ermittlung der Entwicklungsinfrastrukturen pro Subdomäne.
- Bewertung der Anwendungen pro Entwicklungsinfrastruktur in jeder Subdomäne.

Schritt 4: Bewertung der Domäne

- Bewertung der Gruppierungen der Anwendungen nach Entwicklungsinfrastruktur pro Subdomäne.
- Bewertung der Subdomänen pro Domäne.
- Vergleich des ermittelten Grades mit dem Sollgrad pro Subdomäne.
- Definition von Maßnahmen bei Differenzen des ermittelten Grades mit dem Sollgrad.
- Bewertung der Domäne (optional).

In Abbildung 24 werden diese Schritte in Form einer ereignisgesteuerten Prozesskette (EPK) dargestellt. EPKs sind eine verbreitete grafische Notation von Geschäftsprozessen. Sie stellen eine Abfolge von Funktionen dar, die durch Ereignisse ausgelöst und gesteuert werden [LeWS08]. Funktionen bezeichnen einen Vorgang, eine Aktion oder eine Aufgabe. Ereignisse sind Zustände, die vor oder nach einer Funktion auftreten. Sie lösen eine oder mehrere Funktionen aus. Konnektoren legen fest, wie Ereignisse und Funktionen miteinander verknüpft werden[82].

Die Bewertung der Domäne in Schritt vier ist optional. Da die in der Domäne enthaltenen Subdomänen sehr unterschiedliche Anforderungen an die Anwendungsentwicklung haben können, ist die Aggregation der Grade nicht immer sinnvoll. Dies wird in Abschnitt 5.3.3 an einem Beispiel verdeutlicht. Es ist daher bei jeder Domäne zu prüfen, ob eine Bewertung durchgeführt werden soll.

[82] EPK ist Bestandteil von ARIS (Architektur integrierter Informationssysteme), dem Konzept zur Beschreibung von Organisationen. Eine detaillierte Beschreibung von EPK und ARIS ist in [LeWS08] zu finden.

In den folgenden Abschnitten werden die einzelnen Schritte detaillierter beschrieben. In Kapitel 5 erfolgt eine beispielhafte Bewertung eines Unternehmens anhand dieser Schritte.

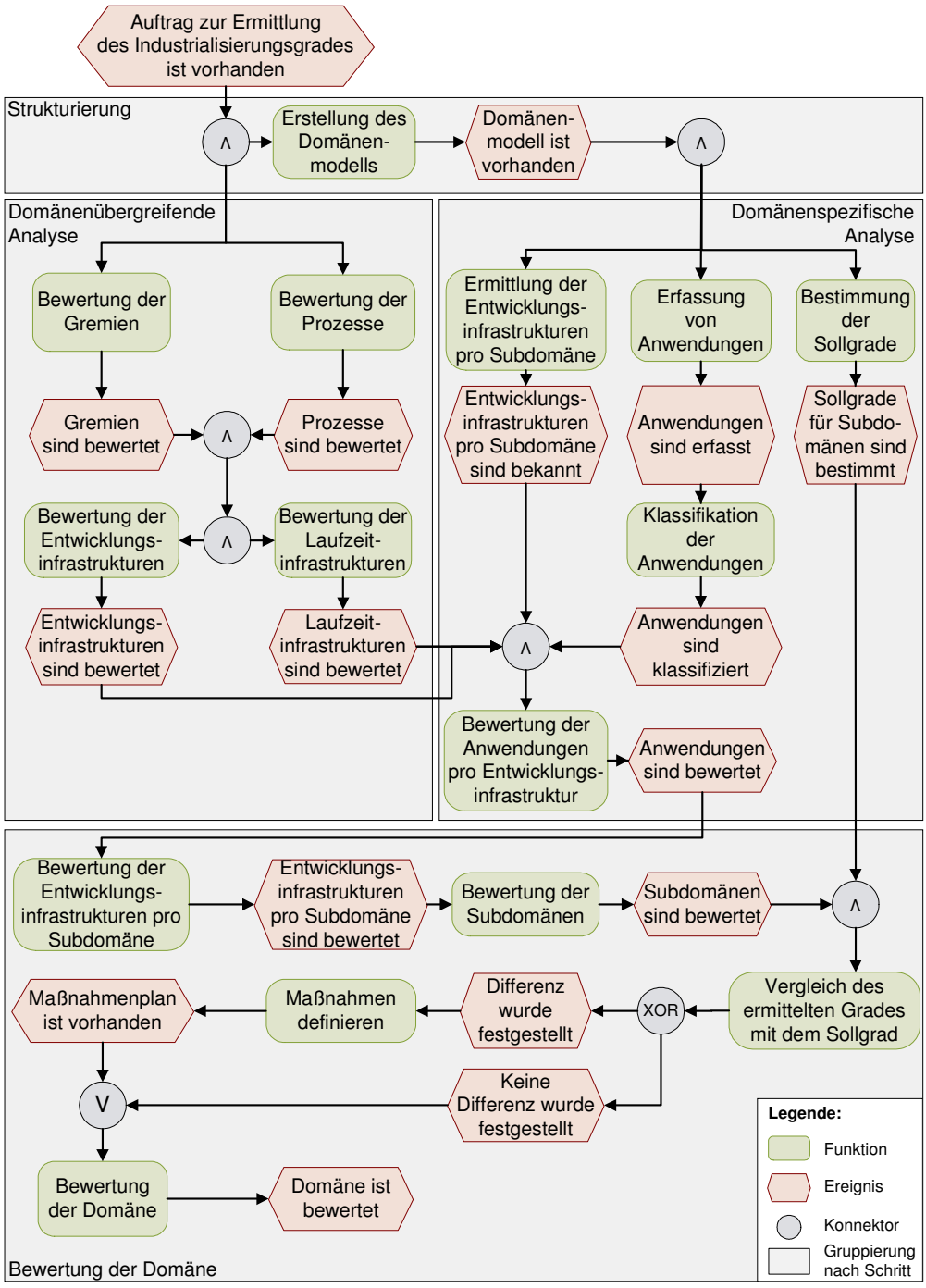

Abbildung 24: Detailschritte zur Ermittlung des Industrialisierungsgrades

4.1 Domänenmodell

Den Ordnungsrahmen für die Strukturierung von Anwendungen bilden in dieser Arbeit die bereits vorgestellten Domänen. Sie gruppieren die Komponenten einer Anwendungslandschaft nach fachlichen Gesichtspunkten.

Sofern ein Unternehmen noch über keinen entsprechenden Ordnungsrahmen verfügt, erfolgt im ersten Schritt die Erstellung eines Domänenmodells. In [EHHJ08] wird hierzu eine Methode zum Entwurf von Domänen vorgestellt. Wesentliche Informationen werden hierfür aus den Geschäftsdimensionen und deren Ausprägungen, den Geschäftsservices sowie den Geschäftsobjekten, entnommen. Als weitere Informationsquellen dienen beispielsweise die Prozesslandkarte[83] oder die Aufbauorganisation des Unternehmens.

Für die meisten Branchen existieren bereits spezifische Domänenmodelle. Jedes Unternehmen hat jedoch individuelle Anforderungen an eine solche Struktur, so dass das Domänenmodell spezifisch zu erstellen ist. Auf eine detaillierte Beschreibung zur Herleitung von Domänen wird in dieser Arbeit verzichtet. Weitere Informationen für die Erstellung eines Domänenmodells können [EHHJ08] entnommen werden.

Das Finden von geeigneten Domänen bedarf einer intensiven Kommunikation mit den Geschäftsbereichen. Es gibt dabei keine richtigen oder falschen, sondern lediglich an den Bedarf des Unternehmens ausgerichtete Domänen. Für das konkrete Beispiel in Kapitel 5 wird für ein Unternehmen ein spezifisches Domänenmodell anhand der erwähnten Schritte erstellt.

4.2 Domänenübergreifende Analyse

Die isolierte Analyse von Anwendungen einer Subdomäne ist nicht ausreichend für die Ermittlung des Industrialisierungsgrads. Eine wesentliche Voraussetzung für die Zuordnung von Anwendungen zu einer Stufe des vorgestellten Stufenmodells ist die Konformität zur zugehörigen Entwicklungsinfrastruktur. Diese Infrastruktur wird jedoch meist[84] nicht spezifisch für eine Subdomäne bereitgestellt, sondern steht Anwendungen aus verschiedenen Subdomänen zur Verfügung. Zur Vermeidung von Mehrfachanalysen der gleichen Infrastrukturen werden die-

[83] In einer Prozesslandkarte werden die Kern-, Steuer- und Unterstützungsprozesse des Unternehmens verwaltet und meist in einer Grafik visualisiert.

[84] Für Spezialanwendungen, die eine gesonderte Entwicklungsumgebung benötigen und auf eine Domäne begrenzt sind, gilt diese Verallgemeinerung nicht. Dieser spezielle Fall wird jedoch nicht näher in dieser Arbeit betrachtet, da er keine Auswirkung auf das Vorgehen hat.

se einmalig domänenübergreifend betrachtet. Hierbei werden insbesondere die folgenden Elemente untersucht:

- *Gremien*: Die Festlegung von Standards oder Verfahren zur Erstellung von Artefakten hat zentral über dafür zuständige Gremien zu erfolgen. Sie legen die Rahmenbedingungen für die Entwicklung von Anwendungen fest.
- *Prozesse*: Die transparente Gestaltung der Vorgaben sowie Verfahren zur Anwendungsentwicklung erfolgt über zentral organisierte bzw. koordinierte Prozesse.
- *Entwicklungsinfrastruktur*: Die Implementierung von Anwendungen hat im Rahmen von vorhandenen Entwicklungsinfrastrukturen zu erfolgen. Sie enthalten u.a. Werkzeuge, Richtlinien und Verfahren zur Erstellung, aber auch zur Verwaltung von Artefakten.
- *Laufzeitinfrastruktur*: Die Bereitstellung von Diensten, die von Anwendungen zur Laufzeit genutzt werden, wird durch die Laufzeitinfrastruktur definiert.

Mit zunehmenden Grad der Industrialisierung wird der Anspruch an diese Elemente für die Anwendungsentwicklung größer.

Der vollständige Reifegrad für die jeweilige Stufe des Industrialisierungsgrades kann in diesem Schritt für die Infrastrukturen nicht separat bestimmt werden. Die Verfügbarkeit dieser Elemente liefert noch keine Aussage über deren Nutzung durch die Anwendungsentwicklung. Es ist jedoch für jede Infrastruktur in den verschiedenen Stufen zu prüfen, ob der Reifegrad „definiert“ erfüllt ist. Dabei wird insbesondere analysiert, ob die für die jeweilige Stufe benötigten Vorgaben vorhanden sind.

4.2.1 Bewertung von Gremien

Für die Etablierung von Methoden im Unternehmen sind die hierzu benötigten Verfahren durch entsprechende Gremien zu koordinieren. Mit steigendem Industrialisierungsgrad erfordern die Methoden vermehrt zentrale Vorgaben, die durch Gremien abgestimmt werden. So bedarf beispielsweise die Standardisierung einer Technologie der Autorität eines zentral anerkannten Gremiums. Die Serviceorientierung benötigt zudem ein zentral koordiniertes Portfolio an Services. Aufgaben und Zielsetzungen der benötigten Gremien wurden in den Beschreibungen der einzelnen Stufen in Kapitel 3 erläutert.

In diesem Schritt werden die im Rahmen der Anwendungsentwicklung benötigten Gremien ermittelt und deren Aufgabenbereich analysiert. Für jeden Industrialisierungsgrad wird geprüft, ob die benötigten Gremien vorhanden sind und die vorgegebenen Aufgaben erfüllen.

Für eine Bewertung werden die nachstehenden Kriterien überprüft:

- *Vollständigkeit*: In jeder Stufe werden Gremien und deren Aufgaben definiert. Bei der Analyse wird bewertet, ob die vorgegebenen Gremien vorhanden und die entsprechenden Aufgaben festgelegt sind.
- *Konsistenz:* Sofern Gremien vorhanden sind, wird bei diesem Kriterium geprüft, ob die vorgegebenen Aufgaben den Aktivitäten des Gremiums entsprechen.
- *Aktualität:* Die Methoden und damit die Aufgaben von Gremien können sich ändern. Bei diesem Kriterium ist daher zu prüfen, ob die beschriebenen Aufgaben dem aktuellen Bedarf entsprechen.
- *Etabliertheit:* Wichtige Aufgabe der Gremien ist die Koordination bei der Gestaltung der Elemente der Infrastrukturen. Es wird hierbei zunächst geprüft, ob das geforderte Gremium existiert und aktiv ist. Ferner wird analysiert, ob Entscheidungen des Gremiums direkten Einfluss auf die Elemente der Infrastrukturen haben.

4.2.2 Bewertung von Prozessen

Die Einführung von Methoden basiert auf der koordinierten Durchführung von Prozessen. Die Anforderungen an die benötigten Prozesse der jeweiligen Stufen wurden in Kapitel 3 beschrieben.

Bei der Analyse in diesem Schritt werden die eingeführten Prozesse identifiziert. Für jeden Industrialisierungsgrad wird anschließend geprüft, inwiefern die Prozesse den Anforderungen der betrachteten Stufe entsprechen.

Für die Bewertung der Prozesse werden die folgenden Kriterien betrachtet:

- *Vollständigkeit*: In jeder Stufe wurden die geforderten Prozesse beschrieben. Bei der Analyse wird geprüft, ob alle Prozesse im Unternehmen definiert sind und durchgeführt werden.
- *Konsistenz*: Bei den vorhandenen Prozessen wird überprüft, ob die durchgeführten Prozessschritte den Beschreibungen entsprechen.
- *Aktualität*: Veränderungen an den Methoden können auch Auswirkungen auf die Prozesse haben. Bei diesem Kriterium wird geprüft, ob die definierten Prozessschritte noch den aktuellen Anforderungen entsprechen.
- *Etabliertheit*: Kern dieser Untersuchung ist die Prüfung, ob die geforderten Prozesse lediglich definiert sind oder ob sie „gelebt“ werden. Dies bedeutet, dass die Durchführung der durch das Gremium koordinierten Prozesse direkten Einfluss auf die Gestaltung der Infrastrukturen hat.

4.2.3 Bewertung von Entwicklungsinfrastrukturen

Die Infrastruktur für die Entwicklung von Anwendungen hat wesentlichen Einfluss auf den Industrialisierungsgrad. Sofern die geforderten Methoden der einzelnen Stufen zentral definiert und unterstützt werden, ist die Voraussetzung für einen hohen Grad gegeben. Eine unzureichende Infrastruktur führt jedoch zu einem geringen Grad.

Für jede Entwicklungsinfrastruktur sind die in Abschnitt 2.3.4 beschriebenen Komponenten zu analysieren. Hierbei wird für jede Stufe des Industrialisierungsgrades betrachtet, in welcher Ausprägung die geforderte Methode unterstützt wird. Ferner ist zu klären, ob diese Methode durch ein etabliertes Gremium anhand der vorgegebenen Prozesse definiert wurde.

Ziel der Analyse ist die Klärung, welcher Grad an Industrialisierung prinzipiell unterstützt werden kann. Ob die einer Subdomäne zugeordneten Anwendungen das Potenzial der Infrastrukturen ausschöpfen, ist Gegenstand der domänenspezifischen Analyse.

Voraussetzung für den Reifegrad „definiert“ sind die Gremien und Prozesse. Die Standards und Verfahren innerhalb der Entwicklungsinfrastruktur sind durch ein entsprechendes Gremium auf Basis der beschriebenen Prozesse zu koordinieren.

Für die Bewertung werden die folgenden Kriterien betrachtet:

- *Vollständigkeit*: In jeder Stufe werden die benötigten Elemente (Werkzeuge, Vorgaben, Verfahren etc.) für die Entwicklung von Anwendungen beschrieben. In diesem Schritt wird die Verfügbarkeit dieser Elemente überprüft. Voraussetzung für die Bewertung ist die Festlegung der geforderten Elemente durch das zuständige Gremium.
- *Konsistenz*: Bei diesem Kriterium wird analysiert, ob die Elemente der Infrastruktur widerspruchsfrei zu den durch ein Gremium definierten Vorgaben sind.
- *Aktualität*: Durch die kontinuierliche Verbesserung von Verfahren, Technologien oder Plattformen durch die Hersteller ist bei diesem Kriterium zu prüfen, ob die vorhandenen Elemente den aktuellen Anforderungen entsprechen.

Keine Aussagen sind für das folgende Kriterium in diesem Schritt möglich:

- *Etabliertheit*: Die Verfügbarkeit einer vollständig definierten Entwicklungsinfrastruktur enthält keine Aussagen zur Nutzung der Elemente dieser Infrastruktur durch die Anwendungsentwicklung.

Die fehlende Aussage zur Etabliertheit ermöglicht in diesem Schritt nur die Bestimmung eines Reifegrades, der maximal „definiert“ sein kann.

4.2.4 Bewertung von Laufzeitinfrastrukturen

Benötigte Dienste oder Funktionalitäten, die eine Laufzeitinfrastruktur bereits zur Verfügung stellt, müssen nicht mehr durch die Anwendung realisiert werden. Je effizienter die vorhandenen Dienste genutzt werden können, desto höher ist der Grad an Industrialisierung.

In diesem Schritt der Analyse wird für jede Stufe betrachtet, inwiefern die geforderten Methoden durch die Laufzeitinfrastruktur unterstützt werden:

- *Vollständigkeit*: Die verschiedenen Elemente der Laufzeitinfrastruktur werden durch das zuständige Gremium vorgegeben. Bei der Bewertung wird überprüft, ob alle geforderten Elemente verfügbar sind.
- *Konsistenz:* Bei den vorhandenen Elementen der Laufzeitinfrastruktur wird die Konformität zu den durch ein Gremium koordinierten Vorgaben geprüft.
- *Aktualität*: Bei der Betrachtung dieses Kriteriums wird geprüft, ob die bereitgestellten Elemente auf einem aktuellen Stand sind.

Domänenübergreifend kann zu folgendem Kriterium keine Aussage getroffen werden:

- *Etabliertheit*: Die Nutzung der verfügbaren Laufzeitinfrastruktur lässt sich erst durch die Betrachtung der verschiedenen Anwendungen analysieren.

Der maximale Reifegrad ist bei der Betrachtung der Laufzeitinfrastruktur ebenfalls, aufgrund der fehlenden Information zur Nutzung durch die Anwendungen, „definiert“. Voraussetzung für diesen Reifegrad ist die Existenz der benötigten Gremien und Prozesse.

In den Stufen Wiederverwendung und Automatisierung werden insbesondere Methoden zur Realisierung von Anwendungen beschrieben und gefordert. Das Ergebnis ist dabei eine Anwendung, die keine zusätzlichen Anforderungen an die Laufzeitinfrastruktur bzgl. der Wiederverwendung oder Automatisierung stellt. In diesen Stufen wird der Reifegrad daher jeweils, ohne weitere Analyse, auf „definiert“ gesetzt.

4.3 Domänenspezifische Analyse

Die hier betrachtete Analyse erfolgt auf Basis der Anwendungen der verschiedenen Subdomänen. Hierzu sind zunächst die relevanten Anwendungen zu erfassen. Die Relevanz ergibt sich aus einer Klassifizierung verschiedener Aspekte der Anwendung (siehe Abschnitt 4.3.2.).

Die einzelnen Subdomänen eines Unternehmens haben unterschiedliche Anforderungen hinsichtlich der Unterstützung durch die Anwendungsentwicklung. Für

jede Subdomäne ist daher zunächst zu klären, welche Stufe der Industrialisierung angemessen ist. Zielgerichtete Maßnahmen können somit bei Abweichen des Analyseergebnisses von dem Sollgrad eingeleitet werden.

Bestandteil der Analyse der ermittelten Anwendungen ist die eingesetzte Methode zur Erstellung der Artefakte sowie die konforme Nutzung der zugeordneten Entwicklungsinfrastruktur. Die eingesetzten Infrastrukturen sind daher vorab zu ermitteln. Die Analyse dieser Infrastrukturen ist bereits in der domänenübergreifenden Analyse erfolgt.

Diese für die domänenspezifische Analyse benötigten Schritte werden in den folgenden Abschnitten vertieft.

4.3.1 Erfassung der Anwendungen des Unternehmens

Die Basis für die domänenspezifische Analyse sind die Anwendungen der Subdomänen. Unternehmen, deren IT-Prozesse nach ITIL[85] strukturiert sind, verfügen bereits über eine entsprechende Anwendungsdatenbank. In diesem Fall ist lediglich zu prüfen, ob die für die Analyse erforderlichen Attribute der Anwendungen gepflegt sind. Sofern diese Daten noch nicht vorliegen, ist eine Erfassung notwendig.

Wie eine solche Erfassung erfolgt, ist abhängig von der Organisationsstruktur und für jedes Unternehmen individuell durchzuführen. Sie wird daher in dieser Arbeit nicht weiter spezifiziert. Meist erfolgt eine solche Erfassung pro Abteilung. Da die Domänen meist nicht die Organisationsstruktur des Unternehmens abbilden, sollten in diesem Fall zunächst alle Anwendungen ermittelt werden. Bei der Identifizierung der Anwendungen ist die Zuordnung zu einer Subdomäne zu berücksichtigen. Eine konkrete Erfassung könnte dabei die folgende Struktur haben:

- *Klärung des Anwendungsbegriffes:* Für die einheitliche Sicht auf den Umfang einer Anwendung ist initial zu klären, was unter einer Anwendung zu verstehen ist.
- *Identifizierung von Anwendungen der Abteilung:* Aufgrund der meist unterschiedlichen Auffassung des Anwendungsbegriffes ist es wichtig, die für eine Analyse relevanten Anwendungen zu identifizieren. Die Kategorisierung aus Abschnitt 2.7.1 liefert hierzu wichtige Unterscheidungskriterien. Bei der Erfassung der Anwendungen werden diese den verschiedenen Kategorien zugeordnet.

[85] ITIL: IT Infrastructure Library (ITIL) ist eine prozessorientierte Sammlung von Best Practices für die Planung, Überwachung und Steuerung von IT-Leistungen. Weitere Informationen hierzu sind in [ZaBP05] enthalten. Insbesondere für das Change Management ist hierfür eine Datenbasis aufzubauen, die alle Anwendungen enthält.

Neben der Eigenentwicklung werden Anwendungen auch gekauft und für den Einsatz kaum verändert. Diese Kaufsoftware spielt für die Analyse des Industrialisierungsgrades der Anwendungsentwicklung keine Rolle. Daher sind nur solche Anwendungen zu betrachten, die überwiegend spezifisch für das Unternehmen entwickelt wurden.

- *Beschreibung der Attribute:* Für jede Anwendung werden, zusammen mit dem entsprechenden Verantwortlichen, die Werte für die geforderten Attribute erfasst.
- *Freigabe der Anwendungsliste:* Die Liste der identifizierten Anwendungen sowie der hinterlegten Attributwerte wird durch den Verantwortlichen freigegeben.

In der Erfassung der Anwendungen sollten u.a. die folgenden Attribute enthalten sein:

- *Name der Anwendung*
- *Plattform*: Sofern die Anwendung auf einer Plattform aufsetzt (z.B. einem SAP R/3-System), ist diese anzugeben.
- *Technologie*: Auswahl der Technologie, mit der die Anwendung realisiert wurde.
- *Entwicklungsanteil:* Bestimmung des Anteils der Entwicklung bei der betrachteten Anwendung (sofern die Anwendung oder Teile davon gekauft und verändert wurden).
- *Entwicklungsumgebung*: Zuordnung der Entwicklungsinfrastruktur, in welcher die Anwendung realisiert wurde.
- *Subdomäne:* Angabe der Subdomäne, die der Anwendung zugeordnet wurde.
- *Anzahl Nutzer:* Angabe der Anzahl an potentiellen Nutzern der Anwendung.
- *Kriterien für die Klassifikation*: Für die Klassifikation der Anwendung sind die vorgegebenen Kriterien zu bewerten (s. Abschnitt 2.7.3).

Die Liste kann um weitere unternehmensspezifische Attribute ergänzt werden.

Für eine Wiederholung der Bestimmung des Industrialisierungsgrades sollte ein Verfahren festgelegt werden, in welchem die Daten der bestehenden Anwendungen aktuell gehalten werden. Ferner sollten neue Anwendungen möglichst automatisiert in die Erfassung mit aufgenommen werden.

Da die Modifikation oder die Neuerstellung von Anwendungen meist im Rahmen von Projekten durchgeführt wird, sollte die Aktualisierung der Anwendungsdaten im Projektmanagement als zwingende Aktion festgeschrieben werden.

Es ist zudem hilfreich, diese Anwendungsdaten zentral in einer Datenbank zu hinterlegen, um Auswertungen durchführen zu können.

4.3.2 Klassifikation von Anwendungen einer Subdomäne

Der Begriff „Anwendung“ kann sehr unterschiedlich ausgelegt werden. Die Relevanz der einzelnen Anwendungen für das Unternehmen kann daher variieren. Vor der Bewertung einer Subdomäne ist daher festzulegen, welche der erfassten Anwendungen für eine Analyse in Frage kommen.

In Abschnitt 2.7 wurden vier Klassen sowie die Kriterien für die Einstufung einer Anwendung erläutert. Diese sind in der Abbildung 25 dargestellt. Für jede Anwendung ist diese Klassifizierung anhand der Werte durchzuführen.

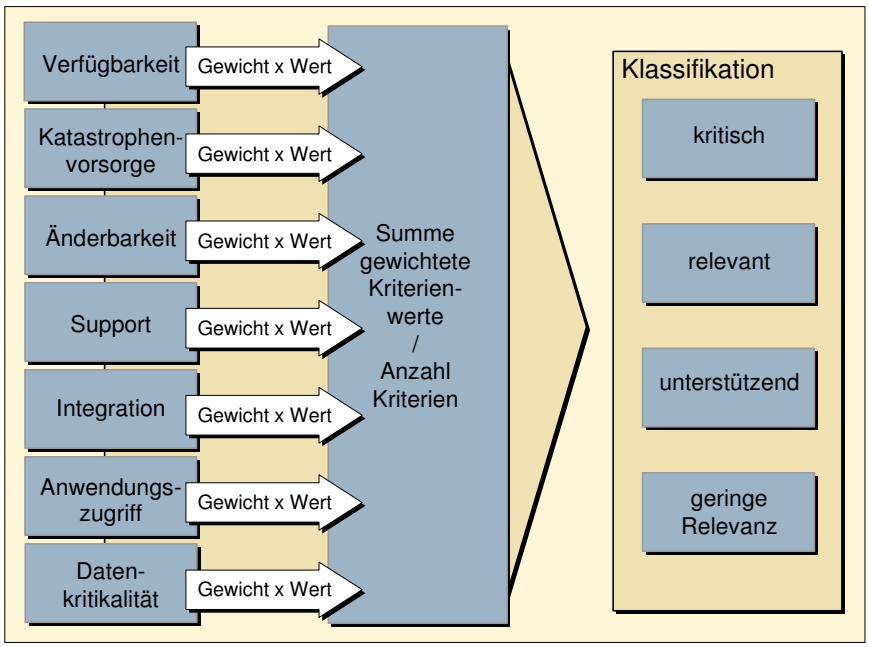

Abbildung 25: Klassifikation von Anwendungen

Wesentlich für die Analyse sind insbesondere die Anwendungen der kritischen und relevanten Klassen. Sofern für das Unternehmen wichtig, kann auch die Klasse der unterstützenden Anwendungen mit in die Analyse aufgenommen werden. Anwendungen mit geringer Relevanz sind von der Betrachtung ausgeschlossen, da sie keinen wesentlichen Beitrag zur Unterstützung der Geschäftsprozesse liefern.

Wie in Kapitel 2 beschrieben, ist für jede Anwendung pro Kriterium ein Wert zwischen 1 und 4 zu vergeben. Der Wert 1 ist dabei der höchste, der Wert 4 der niedrigste. Dieser Wert wird mit dem Gewicht des Kriteriums multipliziert. Sofern für ein Unternehmen bestimmte Kriterien eine höhere Bedeutung haben, kann die Gewichtung entsprechend verteilt werden.

Es wird bewusst auf eine allgemeine Festlegung der Gewichte verzichtet, da dies ein Gestaltungselement des Unternehmens ist. Für diese Arbeit wird jedes Kriterium gleich gewichtet.

Die Einstufung in eine Klasse erfolgt durch die Bildung des arithmetischen Mittels der gewichteten Kriterienwerte:

$$k = \frac{1}{n} \sum_{i=1}^{n} x_i y_i$$

k: Klassifikation der Anwendung
n: Anzahl Kriterien
x_i: Wert des Kriteriums i
y_i: Gewicht des Kriteriums i

Die Summe der Gewichte y_i ergibt 1. Die Zugehörigkeit zu einer Klasse ergibt sich aus den folgenden Intervallen:

- Kritisch: [1 , 1.5]
- Relevant:]1.5 , 2.5]
- Unterstützend:]2.5 , 3.5]
- Geringe Relevanz:]3.5 , 4]

4.3.3 Bestimmung des Sollindustrialisierungsgrades

Vor einer Bewertung der Anwendungen und Infrastrukturen ist der Sollindustrialisierungsgrad der betrachteten Subdomäne zu bestimmen. Eine mögliche Differenz zu dem durch die Analyse ermittelten Grad bildet die Grundlage für Maßnahmen zur Optimierung der Anwendungsentwicklung. Bei der Bestimmung des Sollgrades sind die Zielsetzungen und mögliche Nachteile, die im Kapitel 3 für jede Stufe der Industrialisierung beschrieben sind, zu beachten.

Der Sollindustrialisierungsgrad kann auf verschiedenen Ebenen bestimmt werden. Auf der Ebene des Unternehmens ist diese Festlegung, wie bereits geschildert, wenig sinnvoll, da die einzelnen Domänen unterschiedliche Anforderungen an die Entwicklung stellen. Weitere mögliche Ebenen sind die Domänen, die Subdomänen oder die in den Subdomänen enthaltenen Entwicklungsinfrastrukturen.

Die Entscheidung für eine Ebene kann für einzelne Unternehmen unterschiedlich ausfallen. Für diese Arbeit wird die Subdomäne als Ebene gewählt. Die Festlegung des Sollgrades auf Domänenebene ignoriert die eventuell sehr unterschiedlichen Anforderungen der Subdomänen an die Industrialisierung. Auf der Ebene der Entwicklungsinfrastrukturen stehen die technischen Aspekte zu sehr im Vordergrund. Ferner ist diese Ebene zu wenig spezifisch für die Eigenschaften einer Subdomäne.

Für jede Subdomäne werden in diesem Schritt die Prinzipien der verschiedenen Stufen der Industrialisierung mit den Anforderungen an die Anwendungsentwicklung in der Subdomäne abgeglichen und somit der benötigte Sollgrad festgelegt.

Sofern für das Unternehmen von Interesse, kann aufbauend auf dem gewählten Sollgrad der Subdomänen auch für die gesamte Domäne der Sollgrad festgelegt werden.

4.3.4 Ermittlung der Entwicklungsinfrastrukturen pro Subdomäne

Die Anwendungen der verschiedenen Subdomänen werden meist in unterschiedlichen Technologien und damit in verschiedenen Entwicklungsinfrastrukturen realisiert. Welche der in der domänenübergreifenden Analyse identifizierten Infrastrukturen für die betrachteten Subdomänen zur Verfügung stehen, wird in diesem Schritt ermittelt.

Bei der Erfassung der Anwendungen erfolgt bereits eine Zuweisung zu einer existierenden Entwicklungsinfrastruktur sowie einer Subdomäne. Für die Ermittlung genügt es daher, die Anwendungen einer Subdomäne nach den verschiedenen Entwicklungsinfrastrukturen zu gruppieren.

Anwendungen können maximal den Grad erreichen, der auch von der zugehörigen Infrastruktur unterstützt wird. Sofern die verschiedenen Elemente der Entwicklungsinfrastruktur für einen bestimmten Grad nicht definiert sind, ist auch die Industrialisierung der Anwendungsentwicklung für diesen Grad nicht möglich.

4.3.5 Bewertung der Anwendungen pro Entwicklungsinfrastruktur

Für jede ermittelte Entwicklungsinfrastruktur einer Subdomäne werden die zugeordneten Anwendungen analysiert. Die Bewertung erfolgt auf Basis der Artefakte der Anwendungsarchitektur sowie des Entwicklungsumfeldes. Ferner wird die Nutzung der zugewiesenen Entwicklungs- und Laufzeitinfrastruktur betrachtet. Dabei werden lediglich Anwendungen berücksichtigt, deren Relevanz zumindest „unterstützend“[86] ist. Es wird für jede Stufe überprüft, ob die geforderte Methode angewandt wurde.

Die Bewertung der Anwendungen erfolgt in den einzelnen Stufen u.a. anhand der folgenden Fragen:

- *Standardisierung*: Wurden bei der Realisierung der verschiedenen Artefakte die Standards eingehalten (Technologien, Referenzarchitekturen, Vorgaben, Plattformen etc.)? Werden die vorgegebenen Infrastrukturen verwendet?

[86] Je nach Zielsetzung des Unternehmens kann es auch Sinn machen, nur die ersten beiden Kategorien (kritisch und relevant) zu betrachten.

- *Wiederverwendung*: Ist eine Aufteilung der Anwendung in Komponenten erkennbar? Werden bereits existierende Komponenten wieder verwendet? Werden die Möglichkeiten der Entwicklungsinfrastruktur genutzt?
- *Automatisierung*: Ist die Generierung ein wesentliches Element zur Erstellung der benötigten Artefakte? Basiert die Erstellung auf den in der zugewiesenen Infrastruktur vorgegebenen Verfahren?
- *Komposition*: Entspricht die Architektur der Anwendung einer Serviceorientierung? Sind die einzelnen Artefakte entsprechend einer Serviceorientierung aufgebaut? Werden die vorgegebenen Infrastrukturen benutzt?

Eine detaillierte Beschreibung der Kriterien zur Erstellung der Artefakte wurde in Kapitel 3 für jeden Grad spezifiziert. In Tabelle 11 werden die Ergebnisse der Analyse der Artefakte sowie der Nutzung der Infrastrukturen einer Anwendung für jede Stufe eingetragen. Die Werte geben dabei die Konformität zu den jeweiligen Vorgaben in Prozent an.

Anwendung	Workflow	Interaktion	Business	Data	Tests	Dokumente	Entwicklungs-infrastruktur	Laufzeit-infrastruktur
Standardisierung								
Wiederverwendung								
Automatisierung								
Komposition								

Tabelle 11: Tabelle zur Bewertung einer Anwendung anhand der Elemente

Die Einstufungen der Anwendung aufgrund der Analyse der Artefakte folgen dabei dem folgenden Schema[87]:

- Nicht vorhanden (Wert: 0): Die Methode wurde bei kaum einem der Artefakte angewandt (weniger als 25%).
- In Ansätzen vorhanden (Wert: 25): Die Methode wurde bei mindestens 25% der betrachteten Artefakte angewandt.
- Teilweise vorhanden (Wert: 50): Bei mindestens 50% der analysierten Artefakte wurde die betrachtete Methode angewandt.

[87] Die prozentuale Aufteilung der Werte orientiert sich an einem ähnlichen Schema, welches in [DuGa06] vorgestellt wird. Dies kann jedoch je nach Bedarf im Unternehmen angepasst werden.

- Überwiegend vorhanden (Wert: 75): Die meisten Artefakte, mindestens 75%, wurden auf Basis der betrachteten Methoden erstellt. Mit dieser Stufe gilt die Methode als angewandt.
- Vollständig vorhanden (Wert: 100): Alle Elemente wurden mit der betrachteten Methode erstellt.

Für die Bewertung, ob die Artefakte der verschiedenen Schichten einer Anwendung die Anforderungen an die betrachtete Stufe erfüllen, wird der geringste Wert in der entsprechenden Zeile der Tabelle 11 betrachtet. Eine Methode gilt als genutzt oder vorhanden, sofern sie für die meisten Artefakte der jeweiligen Schicht eingesetzt wird und damit die Einstufung „überwiegend vorhanden“ (75) erhält.

Die Bewertung der einzelnen Anwendungen kann nur im Zusammenhang mit den zugeordneten Infrastrukturen erfolgen. Der Reifegrad „eingeführt“ in jeder Stufe des Industrialisierungsgrades wird nur erreicht, wenn die in der Entwicklungsinfrastruktur definierten Methoden und Verfahren zum Einsatz kommen. Die Vorgaben zur Nutzung der Infrastrukturen sind daher zu 100% zu erfüllen.

Die Anwendung erfüllt damit die Voraussetzungen für eine Stufe, wenn in Tabelle 11 die Spalten der Artefakte der jeweiligen Stufe mindestens den Wert 75 und die Spalten der Infrastrukturen 100 enthalten.

Der Grad an Industrialisierung kann nicht anhand einer einzelnen Anwendung ermittelt werden. Hierzu sind alle Anwendungen einer der Subdomäne zugeordneten Entwicklungsinfrastruktur zu betrachten (siehe nächster Abschnitt).

Hinweis: Gremien und Prozesse werden in diesem Schritt nicht mehr betrachtet. Diese Elemente sind wesentlicher Input für den Reifegrad der Infrastrukturen.

4.4 Bewertung der Domäne

Die Ergebnisse der domänenübergreifenden und domänenspezifischen Bewertungen werden in diesem Schritt zusammengefasst. In Abbildung 26 sind die verschiedenen Elemente, die Einfluss auf die Bestimmung des Grades für eine Domäne haben, dargestellt.

Die Ermittlung des Grades findet dabei zunächst für die verschiedenen Entwicklungsinfrastrukturen der betrachteten Subdomäne statt. Die Infrastruktur mit der niedrigsten Bewertung liefert dabei den Grad für die Subdomäne. Diese Bewertung wird für alle Subdomänen durchgeführt. Die Subdomäne mit dem niedrigsten Grad legt wiederum den Grad für die zugehörige Domäne fest, sofern eine solche Bewertung sinnvoll ist.

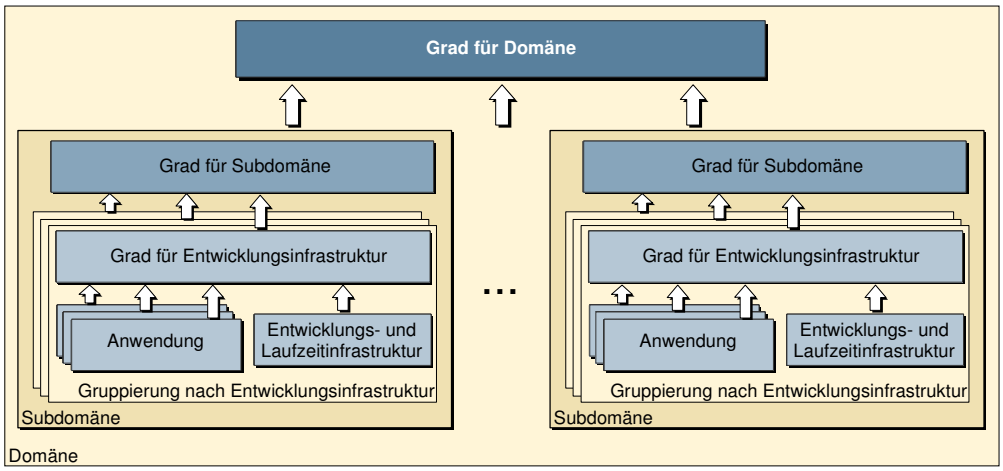

Abbildung 26: Einflussfaktoren für die Bestimmung des Grades pro Domäne

Die folgenden Abschnitte beschreiben die stufenweise Ermittlung des Grades.

4.4.1 Bewertung der Gruppierung nach Entwicklungsinfrastrukturen

Die unterste Stufe der Ermittlung des Industrialisierungsgrades erfolgt anhand der Anwendungen, die einer Entwicklungsinfrastruktur der betrachteten Subdomäne zugeordnet sind.

Bei dieser Bewertung werden auch die Entwicklungsinfrastruktur selbst sowie die Laufzeitinfrastruktur betrachtet. Zwar wurden die Anwendungen bereits bzgl. der Nutzung der Infrastrukturen analysiert. Die Nutzung gibt jedoch noch keine Auskunft darüber, ob diese Infrastrukturen auch über den benötigten Reifegrad verfügen.

Der hier betrachtete Reifegrad für jede Stufe wird durch Zusammenführung der domänenspezifischen und domänenübergreifenden Analysen ermittelt. Hierbei gilt folgendes Schema:

- *Initial*: Die Methode ist weder in den realisierten Anwendungen noch in den Elementen der Entwicklungsinfrastrukturen erkennbar.
- *Wiederholbar*: Verschiedene Anwendungen wurden mit der betrachteten Methode realisiert. Es gibt jedoch keine definierten Entwicklungsinfrastrukturen, die Werkzeuge für betrachtete Methoden zur Verfügung stellt. Somit ist die Umsetzung auf individueller Basis erfolgt. Ebenfalls steht keine Laufzeitinfrastruktur speziell für den betrachteten Grad zentral zur Verfügung.
- *Definiert*: Die Methode sowie die zugehörigen Vorgaben wurden von einem zentralen Gremium anhand der vorgegebenen Prozesse definiert bzw. dokumentiert. Die entsprechenden Entwicklungs- und Laufzeitinfra-

strukturen stehen für alle Anwendungen zur Verfügung. Es wird jedoch nur vereinzelt in der Entwicklung von Anwendungen genutzt (weniger als 75% der Anwendungen).

- *Eingeführt*: Die definierten und kommunizierten Methoden sowie die hierzu notwendigen Infrastrukturen werden von den meisten Anwendungen genutzt (mindestens 75%).
- *Optimiert*: Die Methode sowie die Umsetzung in den Infrastrukturen werden regelmäßig überprüft und an neue Anforderungen angepasst.

Für jede Stufe des Industrialisierungsgrades wird die Reife der Anwendungsentwicklung anhand der Entwicklungsinfrastruktur und der darin enthaltenen Anwendungen ermittelt. In der Tabelle 12 werden hierzu die Reifegrade aus der domänenübergreifenden und domänenspezifischen Analyse zusammengeführt.

Entwicklungsinfrastruktur	Anwendungen	Entwicklungs-infrastruktur	Laufzeit-infrastruktur
Standardisierung			
Wiederverwendung			
Automatisierung			
Komposition			

Tabelle 12: Reifegradermittlung pro Entwicklungsinfrastruktur

Die Spalte „Anwendungen“ in Tabelle 12 enthält den Anteil der Anwendungen (in %), die die Voraussetzungen der jeweiligen Stufe erfüllen. Die Spalten der Infrastrukturen enthalten den jeweiligen Reifegrad.

Eine Stufe gilt als erreicht, wenn mindestens 75% der Anwendungen die Voraussetzungen der Stufe erfüllen und die Entwicklungs- und Laufzeitinfrastruktur jeweils mit dem Reifegrad „definiert“ bewertet wurden.

4.4.2 Bewertung der Subdomänen

Sofern in einer Subdomäne Anwendungen in verschiedenen Entwicklungsinfrastrukturen realisiert wurden, ist die Bewertung von Abschnitt 4.4.1 für die infrastrukturspezifischen Gruppierungen der Subdomäne durchzuführen.

Die Subdomäne erreicht einen Industrialisierungsgrad, wenn alle infrastrukturspezifischen Gruppierungen mindestens mit dieser Stufe bewertet werden. Die Gruppierung mit der niedrigsten Bewertung legt somit den Grad der Subdomäne fest.

4.4.3 Bewertung der Domänen

Für die Domäne wird der Industrialisierungsgrad analog der Subdomäne identifiziert. Zunächst sind alle Subdomänen der betrachteten Domäne zu bewerten. Der Grad für die Domäne richtet sich anschließend nach der Subdomäne mit dem geringsten Industrialisierungsgrad.

Domänen umspannen meist einen großen Bereich im Unternehmen. Zwischen den enthaltenen Subdomänen kann es große Unterschiede bzgl. der Anforderungen an die Anwendungsentwicklung geben. Die Aussagekraft der Industrialisierung für eine ganze Domäne sollte daher überprüft werden. Diese Fragestellung wird in Abschnitt 5.4 anhand eines konkreten Beispiels verdeutlicht.

4.5 Vergleich des ermittelten Grades mit dem Sollgrad

Im abschließenden Schritt erfolgt der Vergleich des ermittelten Grades pro Subdomäne mit dem zuvor bestimmten Sollgrad. Stimmt die Festlegung des Sollgrades mit der Analyse überein, erfüllt die Anwendungsentwicklung die Anforderungen. Somit sind keine weiteren Aktionen notwendig.

Sofern die Analyse einen geringeren als den vorgegebenen Sollgrad ermittelt, ist die Ursache hierfür zu untersuchen. Zunächst sollte nochmals hinterfragt werden, ob der geforderte Sollgrad tatsächlich benötigt wird. Wenn dies der Fall ist, sollten die beiden folgenden Bereiche untersucht werden:

- Infrastruktur
- Entwicklung der Anwendung

Sofern die Infrastruktur nicht den Anforderungen entspricht, ist zu klären, mit welchen Initiativen die Infrastruktur optimiert werden kann. Hierzu sind die verschiedenen Elemente der Infrastruktur näher zu beleuchten.

Bietet die Infrastruktur die nötigen Voraussetzungen, ist zu klären, warum in der Entwicklung der Anwendungen die Potentiale nicht ausgeschöpft bzw. die Vorgaben nicht eingehalten werden. Dies kann vielfältige Gründe haben, auf die in dieser Arbeit nicht näher eingegangen wird.

Für den Fall, dass die Analyse einen zu hohen Wert ermittelt hat, ist zu überlegen, ob der Aufwand für die Anwendungsentwicklung zu hoch angesetzt ist. Für die Anforderungen könnte das Unternehmen in der betrachteten Subdomäne auch mit einer geringeren Industrialisierung auskommen.

4.6 Weitere Anmerkungen

Die einmalige Erfassung des Grades dient der Aufnahme des Ist-Zustandes. Es ist hilfreich, mit diesem Wert die zukünftigen Ziele der Anwendungsentwicklung

festzulegen. Die Ermittlung des Grades ist somit ein kontinuierlicher Prozess. Hierzu sind Mechanismen aufzubauen, die eine regelmäßige Ermittlung des Grades in den verschiedenen Subdomänen erlauben.

Der Industrialisierungsgrad enthält Aussagen über den Zustand der Anwendungsentwicklung. Dies sollte dem IT-Management jederzeit zur Verfügung stehen.

Mit der Balanced Scorecard kann ein Unternehmen die Roadmap für die verschiedenen Domänen festlegen. Eine Balanced Scorecard enthält Ziele, Kennzahlen, Zielwerte und Maßnahmen. Dies kann verbunden werden mit der Identifizierung und Planung des Grades.

Der Vergleich eines Unternehmens mit anderen in Form eines Benchmarks ist eine wichtige Analyse für die Wettbewerbsfähigkeit. Der Grad der Industrialisierung in der Anwendungsentwicklung ist jedoch nur bedingt dafür nutzbar, da die Einschätzungen und die Zielsetzungen zwischen den Unternehmen sehr unterschiedlich sein können.

Die Ziele des Unternehmens sind wesentlich für die Festlegung des Sollindustrialisierungsgrades. Daher sollte für jeden Bereich aus den Zielen der benötigte Grad bestimmt werden.

5 Fallstudie

Der Ablauf zur Ermittlung des Industrialisierungsgrades in der Anwendungsentwicklung für die Domänen eines Unternehmens wurde in Kapitel 4 beschrieben. Im Rahmen einer Fallstudie wird die Anwendbarkeit des Stufenmodells sowie der Ermittlung des Grades in der Praxis untersucht.

Die beispielhafte Durchführung der Bewertung einer Subdomäne wird anhand eines Fertigungsunternehmens durchgeführt. Das Unternehmen hat weltweit 20 Produktionsstandorte in Europa, Amerika und Asien. Es ist in sechs Geschäftsbereiche sowie zehn Zentralbereiche aufgeteilt.

Die IT ist innerhalb des Unternehmens zentral organisiert und zuständig für die effiziente und kostengünstige IT-Unterstützung der Geschäftsprozesse. Die Entwicklung von Software legt somit den Schwerpunkt auf die technische Umsetzung von konkreten Anforderungen aus den Geschäfts- und Zentralbereichen. Dabei wird dem Einsatz von Standards in Technologien, Plattformen und Infrastrukturen ein hoher Wert zugeordnet.

Die Durchführung der Bewertung erfolgt anhand des im letzten Kapitel vorgestellten Vorgehens. Die Analyse wird jedoch nicht für jede Domäne bzw. Subdomäne und aller darin enthaltenen Anwendungen dokumentiert. Für die Fallstudie werden die einzelnen Schritte der Ermittlung in Beispielen vorgestellt. Die Dokumentation der Untersuchung sämtlicher Anwendungen in den verschiedenen Bereichen wäre zu umfangreich. Die im Folgenden vorgestellten Schritte sind jedoch auf die nicht dokumentierten Bereiche direkt übertragbar.

Zur Wahrung der Geheimhaltung unternehmensspezifischer Informationen werden in den kommenden Abschnitten die vorgestellten Strukturen und Werte anonymisiert bzw. verändert. Die Aussagen bzgl. der Anwendbarkeit des Stufenmodells sowie der Bewertung der Anwendungsentwicklung bleiben jedoch gültig.

5.1 Domänenmodell

Das Unternehmen ist in verschiedene Geschäftsbereiche aufgeteilt. Gemeinsame Prozesse werden durch zentrale Bereiche für das gesamte Unternehmen weltweit zur Verfügung gestellt.

Eine Strukturierung in Domänen ist in dieser Form bislang noch nicht durchgeführt worden. Die Analyse der verschiedenen Bereiche sowie der vorhandenen Prozesslandkarten, Anwendungen etc. resultierte in einem abgestimmten Domänenmodell (s. Abbildung 27).

Wie in Abschnitt 2.6 beschrieben, gibt es für die Herleitung eines Domänenmodells keine festen Regeln, sondern lediglich Hinweise bzw. mögliche Verfahren. Wesentlich ist, dass im Unternehmen ein Mehrwert in dieser Aufteilung gesehen wird und Projekte oder allgemein Vorhaben anhand dieses Modells organisiert werden können. Alternative Gruppierungen sind möglich.

Als Grundlage für die Bestimmung der Domänen wurde die im Unternehmen vorhandene Prozesslandkarte verwendet. Die darin enthaltenen Prozesse wurden auf der ersten und zweiten Ebene analysiert. Ferner wurde die Organisationsstruktur des Unternehmens auf weitere Hinweise für eine Domänenbildung überprüft. Die folgende Übersicht stellt die Aufteilung des Unternehmens in die verschiedenen Domänen dar.

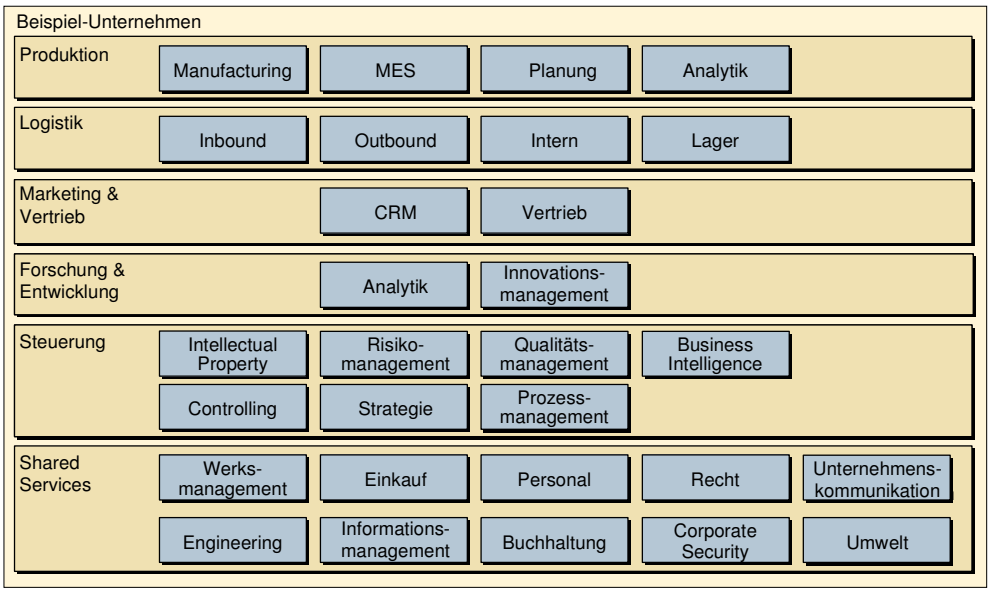

Abbildung 27: Domänenmodell des betrachteten Unternehmens

Die Domänen sowie die darin enthaltenen Subdomänen werden in den folgenden Abschnitten kurz erläutert. Die Fallstudie wird anhand einer konkreten Domäne, der Produktion, durchgeführt. Die Analyse der restlichen Domänen erfolgt analog.

5.1.1 Produktion

Die Produktion von Gütern stellt das Kerngeschäft des Unternehmens dar. Produziert wird eine Vielzahl von unterschiedlichen Produkten in den sechs Geschäftsbereichen, die als Halb- sowie als Fertigprodukte verkauft werden.

Die Aufgaben bzw. Anwendungen der Produktion werden in die folgenden Subdomänen unterteilt:

- *Manufacturing*: Anwendungen, die unmittelbar bei der Herstellung von Produkten eingesetzt werden.
- *Analytik:* Analytische Anwendungen unterstützen die Produktion durch die Analyse von Messwerten. Dies dient der Sicherung der Qualität des aktuellen Herstellungsprozesses, aber auch der Optimierung der Auslastung.
- *MES – Manufacturing Execution System:* Die Anwendungen dieser Subdomäne stellen die Anbindung der produktionsnahen Systeme mit einem Auftragssystem, z.B. einem ERP-System[88], sicher. Neben der Übermittlung und Koordination der Aufträge an die produktionsnahen Systeme erfolgt darüber auch die Rückmeldung über die Fertigstellung eines Auftrages.
- *Planung*: Anwendungen, die für die Planung der Produktion von Gütern eingesetzt werden. Die Einplanung der Produktion basiert dabei zum Teil auf direkten Kundenaufträgen und zum Teil auf einer regelmäßigen Planung des zu erwartenden Absatzes.

5.1.2 Logistik

Da es sich um ein produzierendes Unternehmen handelt, spielt die Logistik eine sehr wichtige Rolle. Rohstoffe, Hilfsmittel, Halb- und Fertigprodukte sind entsprechend den Aufträgen zu koordinieren.

Die Logistik wird hier in die folgenden Bereiche unterteilt:

- *Inbound:* Alle Warenbewegungen in das Unternehmen. Dies sind insbesondere die Lieferungen der Rohstoffe, die das Unternehmen verarbeitet.
- *Outbound:* Organisation des Transports von Waren, die das Unternehmen verlassen. Hierzu gehören insbesondere die Produkte, die das Unternehmen erzeugt.
- *Intern:* Die Herstellung der verschiedenen Güter erstreckt sich meist über mehrere Stationen. Die Warenbewegungen zwischen diesen Stationen werden durch die interne Logistik gesteuert.
- *Lager:* Anwendungen zur Steuerung und Koordination des Warenlagers. Darin werden sowohl Produkte als auch Zwischenprodukte gelagert.

[88] ERP – Enterprise Resource Planning: System zur Planung des effizienten Einsatzes von Ressourcen für den betrieblichen Ablauf.

5.1.3 Marketing & Vertrieb

Die Koordination von Aufträgen sowie Aktivitäten zur Kundenbindung sind die zentralen Aufgaben dieser Domäne. Sie unterteilt sich in die beiden folgenden Subdomänen:

- *CRM – Customer Relationship Management*: Die Maßnahmen zur Kundenbindung umfassen vorbereitende Aktivitäten, z.B. über Kampagnen, sowie eine Nachbereitung im Anschluss an den Verkauf. Diese Maßnahmen, zu denen auch alle Markting-Aktivitäten zählen, werden durch Anwendungen dieser Subdomäne unterstützt.
- *Vertrieb:* Die Subdomäne Vertrieb umfasst alle Anwendungen, die direkt in den Verkaufsprozess eingebunden sind.

5.1.4 Forschung und Entwicklung

Das Unternehmen investiert einen hohen Anteil des Umsatzes in die Erforschung neuer Verfahren und Produkte. Dieser Bereich hat speziell in der Analytik Überschneidungen mit der Produktion. Jedoch sind die Verfahren sehr unterschiedlich.

- *Analytik:* Anwendungen zur Analyse der Verfahren sowie der Qualitätseigenschaften der betrachteten Stoffe.
- *Innovationsmanagement:* Die Erforschung von neuen Stoffen und Verfahren werden durch die Anwendungen in diesem Bereich detailliert dokumentiert.

5.1.5 Steuerung

Die Steuerung des Unternehmens erfolgt auf unterschiedliche Weise. Unterstützt wird diese Aufgabe durch die Funktionalitäten der folgenden Subdomänen:

- *Intellectual Property:* Das Wissen des Unternehmens, das sich vom Wettbewerb unterscheidet, ist besonders zu schützen.
- *Risikomanagement:* Anwendungen zur Analyse der bestehenden Risiken im oder für das Unternehmen.
- *Qualitätsmanagement:* Neben der Analytik werden weitere allgemeine qualitätssichernde Maßnahmen durchgeführt. Dies betrifft insbesondere die verschiedenen bereichsübergreifenden Verfahren.
- *Business Intelligence:* Für die Steuerung des Unternehmens sind aktuelle Kennzahlen über die verschiedenen Bereiche wesentlich. Diese werden durch die Anwendungen in diesem Bereich zur Verfügung gestellt.

- *Controlling:* Das Bilden von Kennzahlen sowie deren Analyse und die daraus resultierenden Maßnahmen werden durch entsprechende Anwendungen unterstützt.
- *Strategie:* Dieser Bereich umfasst alle Anwendungen, die für die Gestaltung der strategischen Ausrichtung des Unternehmens benötigt werden.
- *Prozessmanagement:* Für die effiziente Durchführung von bereichsübergreifenden Prozessen ist die Transparenz bzgl. der Aufgaben und Zuständigkeiten notwendig. Anwendungen in dieser Subdomäne liefern einen Überblick über die vorhandenen Prozesse.

5.1.6 Shared Services

Die unterstützenden Funktionen des Unternehmens für die verschiedenen Geschäfts- und Zentralbereiche werden in dieser Domäne gebündelt. Die Shared Services stellen Dienstleistungen für die anderen Domänen zur Verfügung.

- *Einkauf:* Im Einkauf werden alle Leistungen zur Beschaffung von Waren und Dienstleistungen koordiniert und durchgeführt.
- *Unternehmenskommunikation:* Die verschiedenen Dienste für die interne und externe Kommunikation des Unternehmens werden hier zur Verfügung gestellt.
- *Engineering:* Dieser Bereich enthält Anwendungen für die Planung sowie die Durchführung des Neubaus von Anlagen oder Betrieben. Darin enthalten ist auch die Instandhaltung der existierenden Anlagen und Betriebe.
- *Buchhaltung:* Diese Subdomäne enthält die Anwendungen für die Durchführung von Finanzdienstleistungen des Unternehmens für die verschiedenen Geschäfts- und Zentralbereiche.
- *Corporate Security:* Die Sicherheit der verschiedenen Betriebe des Unternehmens wird durch Corporate Security koordiniert. Die Anwendungen dieser Subdomäne unterstützen die erforderlichen Maßnahmen z.B. bzgl. des Zugangs zum Werk oder der Sicherheit an den verschiedenen Anlagen.
- *Personalmanagement:* Alle Dienste im Bezug auf die Mitarbeiter des Unternehmens werden in dieser Subdomäne gebündelt.
- *Recht:* Für die rechtliche Überprüfung von Vereinbarungen mit Kunden, Lieferanten, Patente, etc. werden alle juristischen Aufgaben gebündelt und mit Anwendungen unterstützt.
- *Informationsmanagement:* Anwendungen aus dieser Subdomäne unterstützen die verschiedenen Leistungen zur Verwaltung von Informationen

des Unternehmens. Darin sind auch die IT-Leistungen für das Unternehmen enthalten.

- *Werksmanagement:* Die Organisation der verschiedenen Werke des Unternehmens umfasst mehrere Aufgaben, die durch die Anwendungen in diesem Bereich unterstützt werden.
- *Umwelt:* Als produzierendes Unternehmen unterliegt es strengen gesetzlichen Umweltauflagen, die über Anwendungen dieser Subdomäne koordiniert werden.

5.2 Domänenübergreifende Analyse

Entwicklungsinfrastrukturen sowie die Gremien und Prozesse werden in diesem Unternehmen über verschiedenen Domänen hinweg genutzt. Sie geben die Art der Anwendungsentwicklung vor. Dieser Abschnitt beschreibt die Analyse dieser Strukturen.

5.2.1 Bewertung der vorhandenen Gremien

Im Rahmen der Anwendungsentwicklung sind in diesem Unternehmen die folgenden Gremien vorhanden:

- *Architektur-Board:* Die Aufgabe des Architektur-Boards ist die Koordination der verschiedenen Architekturen der IT-Anwendungslandschaft sowie der Sicherstellung der Einhaltung der für die Architektur relevanten Vorgaben. Neben der Abstimmung der Architekturen der verschiedenen Bereiche werden über dieses Gremium auch Kontrollen der Einhaltung der Vorgaben durchgeführt.

  Das Board trifft sich in regelmäßigen Abständen. Es setzt sich zusammen aus Vertretern aller für die Architektur relevanten Bereiche der IT. Hierzu gehören die Anwendungsentwicklung[89], Security[90], Client[91] sowie Infrastruktur[92].
- *Technologie-Board:* Dieses Gremium ist zuständig für die Festlegung und Koordination von Standards bei Technologien und Plattformen, aber auch für die hierzu benötigten Infrastrukturen. Dieses Gremium trifft sich in re-

[89] Unterschiedliche Bereiche der IT, die nach Themen wie Kundenmanagement oder Lieferantenmanagement gegliedert sind.

[90] Organisationseinheit, die sich mit Vorgaben und Überprüfung von sicherheitsrelevanten Themen in der IT befasst.

[91] Organisationseinheit, die sich zentral um alle Themen der Endgeräte wie Desktop, Notebook, PDA, etc. kümmert.

[92] Organisationseinheit für die Bereitstellung der Leistungen des Rechenzentrums.

gelmäßigen Abständen und enthält jeweils einen Vertreter aus den Bereichen Anwendungsentwicklung, Security, Client und Infrastruktur (Rechenzentrum sowie Architektur). Dieses Board ist eine Teilmenge des Architektur-Boards mit dem speziellen Fokus auf der Standardisierung von Technologien.

Die Analyse dieser Gremien für die einzelnen Stufen des Industrialisierungsgrades ergab die folgende Bewertung:

- Standardisierung: eingeführt

  Die Strukturen wie Technologien, Plattformen und Infrastrukturen werden zentral über die Gremien Architektur- und Technologie-Board definiert und standardisiert. Darin sind neben der Festlegung der Standards auch die Vorgaben für eine Referenzarchitektur sowie die Entwicklungs- und Laufzeitinfrastruktur enthalten. Die Mitglieder dieses Gremiums decken die für die Architektur relevanten Bereiche ab, so dass die Kommunikation der festgelegten Vorgaben wie Standards in die Anwendungsentwicklung sichergestellt ist.

  Die Gremien, die für den Grad Standardisierung gefordert werden, sind in diesem Unternehmen definiert und etabliert. Die Beschlüsse der Gremien fließen direkt in die Realisierung von Anwendungen ein. Für diesen Grad werden daher die Gremien als „eingeführt“ festgelegt.

- Wiederverwendung: wiederholbar

  Die systematische Koordination von Komponenten, die in mehreren Anwendungen zum Einsatz kommen könnten, erfolgt durch keines der bekannten Gremien. Sporadisch werden in den verschiedenen Gremien Synergieeffekte diskutiert, die durch die gemeinsame Nutzung von Komponenten, z.B. im Portal Umfeld, erzielt werden können. Dies ist jedoch zurzeit nicht in den Aufgaben der Gremien definiert.

- Automatisierung: wiederholbar

  Die Automatisierung bei der Erstellung von Artefakten erfolgt projektspezifisch. Die hierzu notwendigen Vorgaben könnten in den vorhandenen Gremien systematisch festgelegt werden. Dies ist jedoch zurzeit nicht vorgesehen. Daher werden nur sporadisch Vorschläge für eine Automatisierung diskutiert. Verbindliche Vorgaben werden nicht verabschiedet. Der Reifegrad wird daher mit „wiederholbar“ bewertet.

- Komposition: initial

  Die Komposition von Anwendungen im Sinne einer Serviceorientierung ist im Unternehmen noch nicht verankert. Die vorhandenen Gremien haben daher zurzeit noch keine Aufgaben definiert, die eine zentrale Koordinati-

on bzw. Festlegung von Richtlinien zur Komposition von Anwendungen vorsieht. Der Reifegrad wird somit mit „initial“ bewertet.

Für die Standardisierung sind die benötigten Gremien definiert und vorhanden. Sie sind zudem etabliert, da sie Einfluss auf die Infrastrukturen und damit der Anwendungsentwicklung im Unternehmen ausüben.

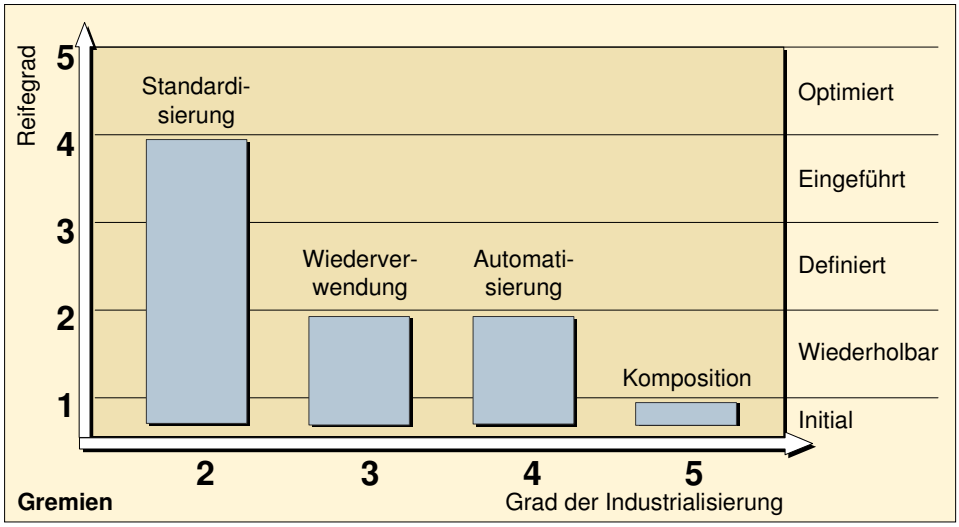

Abbildung 28: Bewertung der Gremien im Unternehmen

Für die restlichen Stufen sind die geforderten Aufgaben jedoch noch nicht definiert und damit nicht eingeführt. Daher wird für die Gremien der Grad „Standardisierung“ festgelegt.

5.2.2 Bewertung der vorhandenen Prozesse

Im Unternehmen wurden im Rahmen der Anwendungsentwicklung die folgenden domänenübergreifenden Prozesse identifiziert:

- *Standardisierung:* Standards im Bereich der Technologien, Plattformen sowie Infrastrukturkomponenten werden über einen durch das Technologie-Board koordinierten Prozess bestimmt und dokumentiert. Die Entscheidungen aus diesem Prozess werden in der Organisation kommuniziert und sind über eine zentrale Plattform verfügbar.
- *Architektur Coaching und Review:* Eine Architekturgruppe betrachtet Projekte bzw. die zu realisierenden Anwendung bzgl. der Konformität zu den vorgegeben Richtlinien. Dies erfolgt zu Beginn und zum Abschluss des Projektes. Die Gruppe wird durch einen IT-Architekten geleitet und für jedes Projekt individuell zusammengestellt. Es enthält Vertreter aus den für die Anwendung relevanten Bereichen der IT. Die Ergebnisse werden dem Architektur-Board berichtet.

- *Evaluierung:* Neue Themen oder Technologien, wie. z.B. serviceorientierte Architekturen oder Software-as-a-Service (SaaS)[93], werden über einen vom Technologie-Board koordinierten Prozess evaluiert und die Ergebnisse zentral zur Verfügung gestellt.

Weitere Prozesse sind vorhanden, jedoch findet hierzu weder eine zentrale Koordination noch Dokumentation statt. Sie werden daher nicht weiter betrachtet. Auf Basis der aufgelisteten Prozesse wurden die Reifegrade für jede Stufe der Industrialisierung ermittelt:

- Standardisierung: eingeführt

  Die Standardisierung von Komponenten erfolgt über einen definierten Prozess. Dieser wird durch das Technologie-Board koordiniert. Durch den „Architektur Coaching & Review"-Prozess wird sichergestellt, dass die Standards bei der Realisierung der Anwendungen eingehalten werden. Analog der Standardisierung wird auch die Evaluierung durch einen entsprechenden Prozess transparent koordiniert.

  Die geforderten Prozesse sind definiert und werden durchgeführt. Die Aktualität wird über die regelmäßigen Sitzungen des Technologie-Boards gewährleistet. Die Standardisierung wird daher mit „eingeführt" bewertet.

- Wiederverwendung: initial

  Es gibt keine definierten Prozesse zur Wiederverwendung von Komponenten. Sofern Komponenten definiert sind, die in anderen Anwendungen genutzt werden können, erfolgt dies auf Basis individueller Initiativen. Der Reifegrad ist daher „initial".

- Automatisierung: initial

  Für die Festlegung von Automatisierungsverfahren gibt es zurzeit keinen definierten Prozess. Projektspezifische Automatisierung erfolgt individuell. Diese Stufe wird daher mit „initial" bewertet.

- Komposition: initial

  Für die Komposition gibt es zurzeit keine Prozesse, die zentral zur Verfügung gestellt werden. Der Reifegrad wird daher mit „initial" bewertet.

Für die Standardisierung sind die erforderlichen Prozesse vorhanden und werden zentral koordiniert. Die Ergebnisse dieser Prozesse werden dokumentiert und sind für jeden transparent einsehbar. Für die restlichen Stufen ist dies nicht erkennbar.

---

[93] Geschäftsmodell, bei dem Software nicht als Produkt verkauft wird, sondern als Dienstleistung über das Internet bezogen werden kann.

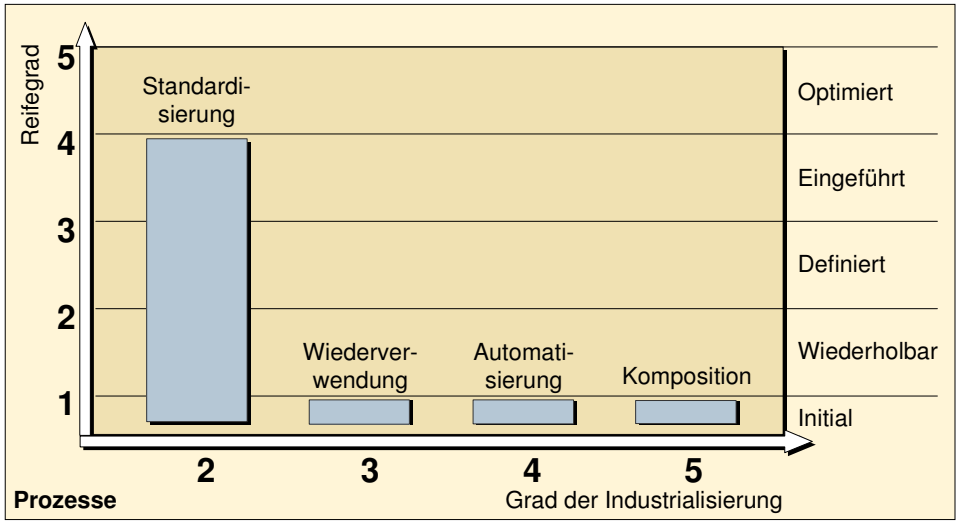

Abbildung 29: Bewertung der Prozesse im Unternehmen

Die domänenübergreifenden Prozesse im Rahmen der Anwendungsentwicklung werden daher mit der Bewertung „Standardisierung“ festgelegt.

5.2.3 Bewertung der vorhandenen Entwicklungsinfrastrukturen

Das Unternehmen nutzt aufgrund der unterschiedlichen Anforderungen in den einzelnen Domänen eine Reihe von Technologien und Plattformen. Die meisten Anwendungen werden jedoch auf Basis der beiden strategischen Plattformen und Technologien von SAP und Microsoft realisiert.

Folgende Entwicklungsinfrastrukturen werden näher betrachtet:

- SAP:
  - ABAP-Entwicklungsinfrastruktur
  - Java-Entwicklungsinfrastruktur[94]
- Microsoft:
  - SharePoint-Entwicklungsinfrastruktur
  - .Net-Entwicklungsinfrastruktur

5.2.3.1 SAP-Entwicklungsinfrastrukturen

SAP hat mit der NetWeaver[95] Produktpalette ihre bisher auf der Programmiersprache ABAP basierte Plattform um eine Java-Komponente erweitert. Hierdurch

[94] Von SAP auch als NetWeaver Development Infrastructure (NWDI) bezeichnet.

[95] Detaillierte Informationen zu SAP NetWeaver sind zu finden in [HeKa08] sowie unter http://sdn.sap.com.

können neben der klassischen ABAP-Programmierung auch Java-Anwendungen realisiert werden. Physisch besteht SAP NetWeaver aus zwei Plattformen, der ABAP-Instanz sowie der Java-Instanz, die auch hinsichtlich der Entwicklung von Anwendungen sehr verschieden ausgeprägt sind. Während die Infrastruktur für die Entwicklung von ABAP-Anwendungen zentral aufgestellt ist, sind die Elemente der Java-Entwicklungsinfrastruktur auf mehrere Systeme verteilt.

Die zentralen Geschäftsprozesse des Unternehmens werden auf Basis einer zentralen ABAP-Instanz unterstützt. In dieser Umgebung finden spezifische Entwicklungen für das Unternehmen statt, so dass hierfür eine hoch standardisierte und abgestimmte Entwicklungsinfrastruktur aufgebaut wurde.

Welche Methoden die ABAP-Entwicklungsinfrastruktur derzeit im Unternehmen unterstützt, wird aus der folgenden Analyse ersichtlich:

- Standardisierung: definiert

  Für die Realisierungen von Anwendungen existieren Vorgaben, die auf einer zentralen Dokumentationsplattform für alle Entwickler zur Verfügung stehen. Die ABAP-Instanz vereint mehrere Komponenten der Entwicklungsinfrastruktur wie Konfigurationsmanagement, Build-Service[96] sowie Deployment in einem System. Auch die Werkzeuge zur Erstellung oder Modifikation der Anwendungen sind in diesem System integriert. Anwendungen werden auf einem Entwicklungssystem implementiert und mittels eines Transportsystems über ein Testsystem auf eine produktive Umgebung auf Basis eines koordinierten Prozesses installiert.

- Wiederverwendung: wiederholbar

  Aufgrund des integrierten Systems ist es möglich, alle vorhandenen Implementierungen mehrfach zu nutzen. Eine koordinierte Abstimmung von Komponenten, die wiederverwendet werden können, ist jedoch nicht erkennbar.

- Automatisierung: initial

  Die ABAP-Instanz stellt keine durchgängigen Werkzeuge oder Methoden für die Generierung von Anwendungen auf Basis von Modellen zur Verfügung. Mit WebDynpro (vgl. [HeKa08]) sind erste Ansätze vorhanden, die jedoch noch nicht genutzt werden. Die BPM-Plattform (Business Process Management) der SAP bietet hierzu erstmals die Möglichkeit der Automatisierung. Diese kommt jedoch im Unternehmen nicht zum Einsatz.

[96] Da die Programmiersprache ABAP eine Interpretersprache ist und direkt auf dem System ausgeführt wird, ist streng genommen kein Build-Service nötigt. Für den Transport zwischen den verschiedenen Systemen ist jedoch ein Mechanismus vorhanden, der alle Änderungen bündelt und damit transparent dokumentiert.

- Komposition: initial

  SAP hat für den Bereich der Komposition insbesondere die Java-Instanz vorgesehen und dort entsprechende Mechanismen zur Verfügung gestellt. Für die ABAP-Instanz sind im Unternehmen für die Komposition noch keine Voraussetzungen geschaffen worden.

Die ABAP-Instanz der SAP zeichnet sich vor allem durch die von der SAP bereitgestellten Anwendungen aus. Im Wesentlichen werden hier Anpassungen an den ausgelieferten Anwendungen vorgenommen. Auch kleinere Anwendungen in Form von Reportings oder kleine Transaktionen werden realisiert. Größere unternehmensspezifische Entwicklungen finden jedoch nicht statt.

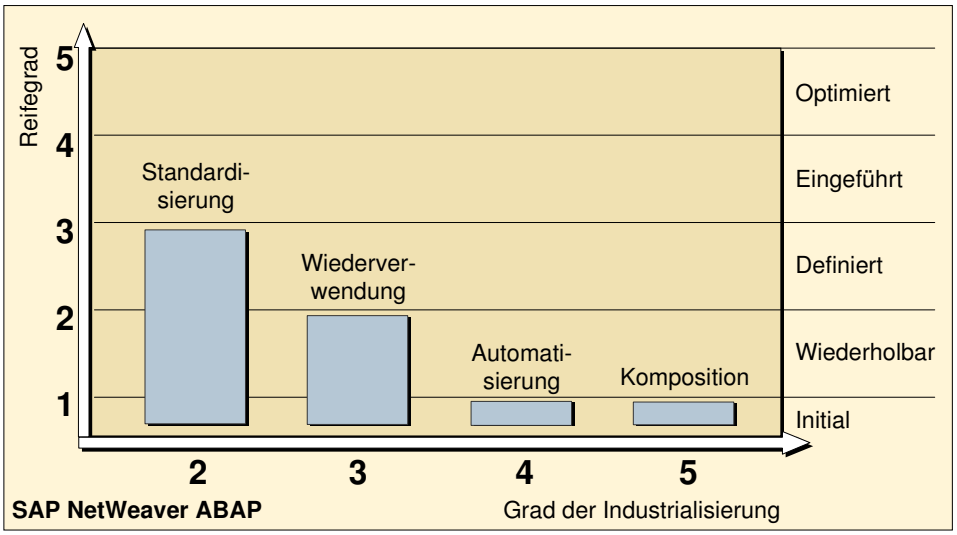

Abbildung 30: Bewertung der ABAP-Entwicklungsinfrastruktur

Aufgrund der Analyse wird für die ABAP-Entwicklungsinfrastruktur der Reifegrad „definiert“ nur für die Stufe „Standardisierung“ vergeben. Damit kann für die Anwendungsentwicklung auf dieser Basis lediglich die Stufe Standardisierung erreicht werden.

Die Java-Instanz der NetWeaver-Plattform wurde von der SAP von Grund auf neu nach den Prinzipien einer serviceorientierten Architektur konzipiert. Hierzu wurde auch die zugehörige Entwicklungsinfrastruktur neu aufgesetzt. Die Unterstützung für die einzelnen Methoden in den verschiedenen Stufen wurde dabei folgendermaßen bewertet:

- Standardisierung: definiert

  Für die Java-Programmierung existieren Vorgaben, die über eine zentrale Dokumentationsplattform zur Verfügung stehen. Die Entwicklungsinfrastruktur für die Java-Instanz wird von SAP als NetWeaver Development Infrastructure (NWDI) bezeichnet. Darin enthalten sind alle benötigten

Komponenten der Stufe Standardisierung: das Design Time Repository (DTR) ist ein Konfigurationsmanagement, der Component Build Service (CBS) stellt den zentralen Build-Service zur Verfügung. Mit dem Change Management Service (CMS) werden die verschiedenen Anwendungen von dem Entwicklungssystem über das Testsystem zum produktiven System transportiert und installiert. Das NWDI enthält somit alle benötigten Werkzeuge für die Implementierung von Anwendungen.

- Wiederverwendung: wiederholbar

  Mit dem System Landscape Directory (SLD) steht in dieser Entwicklungsinfrastruktur ein weiteres Konfigurationsmanagement an einer zentralen Stelle zur Verfügung. Damit ist die zentrale Verwaltung von Komponenten möglich, die in die Entwicklung von Anwendungen eingebunden werden können. Die lokale Entwicklungsumgebung (IDE) stellt ferner Mechanismen zur Verfügung, um wiederverwendbare Komponenten zu erstellen. Die Dokumentation hierfür ist für alle Entwickler verfügbar. Es existieren im Unternehmen derzeit jedoch keine Richtlinien zur Nutzung der Wiederverwendung. Die Wiederverwendung auf dieser Plattform findet daher nur sporadisch statt.

- Automatisierung: wiederholbar

  Mit dem Composition Environment (CE), das inhärenter Bestandteil der Infrastruktur ist, steht eine komplette Modellierungsumgebung zur Verfügung, die auf allen Schichten die Generierung von Quelltext auf Basis von Modellen ermöglicht. Eine Dokumentation hierfür steht allen Entwicklern zur Verfügung. Auch für diese Stufe stehen keine Richtlinien zur Nutzung der Automatisierung zur Verfügung. Die Nutzung ist derzeit eher sporadisch.

- Komposition: initial

  Die Komposition wird grundsätzlich von der NetWeaver-Infrastruktur zur Verfügung gestellt. Jedoch sind hierzu im Unternehmen noch keinerlei Vorgaben zur Nutzung vorhanden. Dieses Thema wurde für die Entwicklung noch nicht näher betrachtet.

Die Java-Instanz der SAP NetWeaver-Plattform bietet eine sehr gute Basis für die Industrialisierung der Anwendungsentwicklung. Sofern die Potentiale dieser Plattform genutzt werden würden, wäre damit der Grad „Komposition“ möglich. In der derzeitigen Ausprägung im Unternehmen wird der Reifegrad „definiert“ jedoch nur für die Stufe „Standardisierung“ gesetzt. Es bestehen weder für die Wiederverwendung noch für die Automatisierung Vorgaben für die Entwicklung von Anwendung auf dieser Plattform.

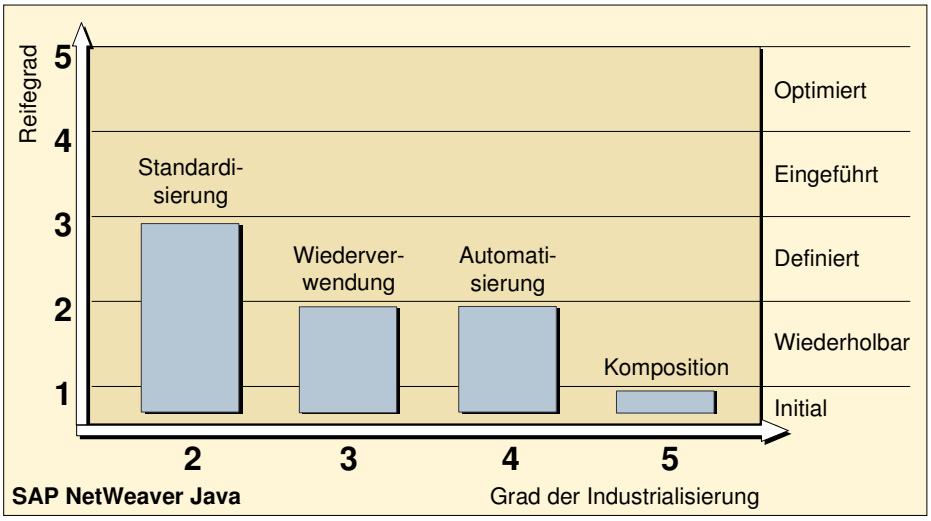

Abbildung 31: Bewertung der Java-Entwicklungsinfrastruktur

5.2.3.2 Microsoft-Entwicklungsinfrastrukturen

Die Zusammenarbeit in verschiedenen Projekten bzw. Gruppen wird in diesem Unternehmen durch die Microsoft SharePoint-Plattform[97] unterstützt. Sie bietet verschiedene Möglichkeiten, eine gemeinsame Informationsbasis in Projekten aufzubauen, diese Informationen zu verteilen, gemeinsam zu bearbeiten und zu verwalten. Ferner bietet SharePoint, neben dem reinen Austausch an Informationen, auch die Möglichkeit, weitere Funktionalitäten in diesem Kontext in Form von Anwendungen zu realisieren.

Microsoft stellt für diese Plattform verschiedene Werkzeuge für eine Anwendungsentwicklung zur Verfügung. Für diesen Zweck hat das Unternehmen eine eigene Entwicklungsinfrastruktur, die SharePoint Development Infrastructure (SPDI), aufgebaut.

Die Analyse dieser Infrastruktur ergab die folgende Bewertung:

- Standardisierung: definiert

  Die SPDI basiert auf dem Team Foundation Server (TFS)[98] von Microsoft und enthält ein zentrales Konfigurationsmanagement sowie einen zentralen Build-Service.

  Die Anwendungen werden auf einem Entwicklungsrechner implementiert und über ein Testsystem auf eine zentral administrierte produktive Um-

[97] Informationen zu SharePoint sowie den Potenzialen der Plattform für die Unterstützung von Geschäftsprozessen ist in [RMH08] zu finden.

[98] Team Foundation Server (TFS): Produkt von Microsoft für die Verwaltung des Lebenszyklus von Anwendungen. Zu den Funktionen gehören u.a. die Quelltext-Verwaltung sowie die Verwaltung von Anforderungen oder Fehlern.

gebung transportiert. Die für die Entwicklung von Anwendungen benötigten Elemente der Infrastruktur werden zentral administriert und stehen allen Entwicklern zur Verfügung.

Die Vorgaben für die Entwicklung von Anwendungen auf dieser Plattform sind vorhanden und über eine zentrale Dokumentationsplattform für alle Entwickler verfügbar.

- Wiederverwendung: wiederholbar

  Die Infrastruktur unterstützt derzeit keine Mechanismen zur systematischen Wiederverwendung von Komponenten. Zwar werden vereinzelt Komponenten in verschiedenen Anwendungen eingesetzt, dies erfolgt jedoch auf Basis individueller Initiativen und Vereinbarungen.

- Automatisierung: wiederholbar

  Eine Infrastruktur für die Generierung von Quelltexten steht grundsätzlich zur Verfügung und kann nach Bedarf erweitert werden. Mit dem sogenannten SPALM[99] steht ein Generator zur Verfügung, der auf Basis einer Konfiguration Quelltext generiert. Die Vorgaben für die Entwicklung von Anwendungen auf dieser Basis sind jedoch nicht zentral koordiniert.

- Komposition: initial

  Eine Infrastruktur für die Entwicklung von Anwendungen auf Basis der Komposition ist nicht erkennbar.

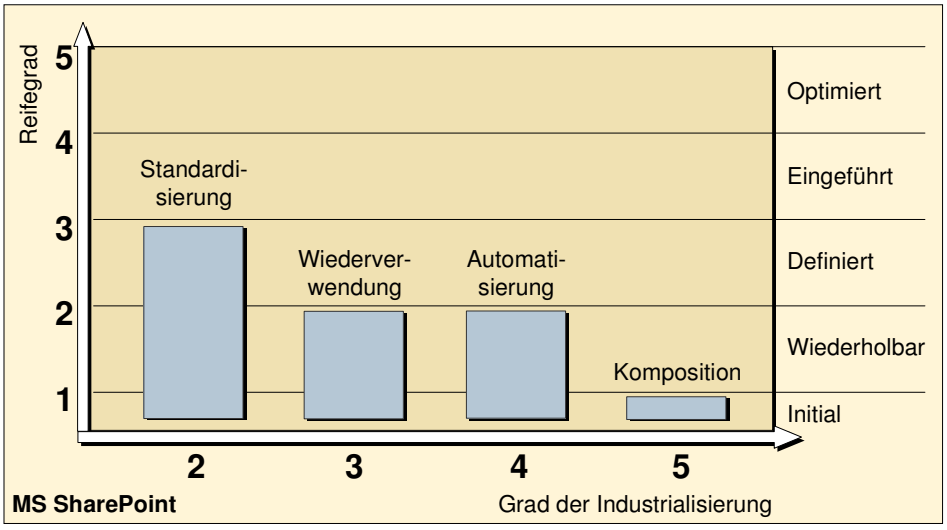

Abbildung 32: Bewertung der SharePoint Entwicklungsinfrastruktur

[99] SPALM: SharePoint Application Lifecycle Management – Ein Open-Source-Projekt der Firma Steria Mummert (http://www.codeplex.com/spalm).

Die SharePoint-Entwicklungsinfrastruktur unterstützt auf Basis dieser Analyse die Anwendungsentwicklung im Reifegrad „definiert“ lediglich für die Stufe „Standardisierung“. Diese Infrastruktur bietet zwar weitere Potentiale, die jedoch im Unternehmen nicht näher spezifiziert bzw. koordiniert werden.

Eine weitere Umgebung für Anwendungsentwicklung wird von Microsoft mit dem .Net Framework (siehe [BBMP06]) zur Verfügung gestellt. Während bei SharePoint spezielle Funktionalitäten für den effizienten Austausch an Informationen adressiert werden, bietet die .Net Umgebung einen generellen Rahmen für die Entwicklung von Anwendungen auf der Microsoft-Plattform.

Bei dem betrachteten Unternehmen wird diese Umgebung insbesondere im Bereich der Produktion eingesetzt. Viele Hersteller von Produktionsanlagen liefern ihre Produkte bereits mit einem Softwarepaket zur Steuerung der Anlage aus. Diese Softwarepakete basieren meist auf Technologien von Microsoft. Daher liegt die Adaption dieser Technologie für weitere Anwendungen in diesem Umfeld nahe.

Die Anwendungsentwicklung zeichnet sich jedoch durch eine sehr individuelle Vorgehensweise aus. Im Einzelnen liegen für die .Net-Entwicklungsinfrastruktur für die verschiedenen Grade die folgenden Bewertungen vor:

- Standardisierung: definiert

  Eine Standardisierung bzgl. der Plattformen und Technologien ist vorhanden. Vorgaben für die Realisierung von Anwendungen werden über eine zentrale Dokumentationsplattform allen Entwicklern zur Verfügung gestellt.

  Die Entwicklungsinfrastruktur wird im Wesentlichen von Microsoft vorgegeben und ist als Standard definiert. Die verschiedenen Komponenten wie das Konfigurationsmanagement werden bereitgestellt.

  Es gibt noch Potential für weitere Verbesserungen, insbesondere bzgl. der zentralen Verwaltung der Infrastruktur. Die geforderten Elemente sind jedoch vorhanden und können genutzt werden. Daher wird die Entwicklungsinfrastruktur als „definiert“ für die Stufe Standardisierung bewertet.

- Wiederverwendung: wiederholbar

  Teilweise werden Komponenten in verschiedenen Anwendungen wiederverwendet. Dies beruht jedoch auf den Initiativen einzelner Entwickler bzw. einzelner Projekte. Allgemein nutzbare Mechanismen für eine systematische Wiederverwendung im Rahmen der Entwicklung sind nicht sichtbar.

- Automatisierung: wiederholbar

  Es gibt erste Ansätze auf Basis von Konfigurationen unter Einsatz eines Generators, den Quelltext zu generieren. Diese Ansätze sind jedoch individuell geprägt und nicht in die zentrale Infrastruktur integriert. Werkzeuge zur Modellierung von Anwendungen sind nicht vorhanden.

- Komposition: initial

  Eine für die Komposition notwendige Infrastruktur ist ebenso wie die zentrale Bereitstellung und Nutzung von Funktionalitäten nicht erkennbar.

Die .Net Entwicklungsinfrastruktur ermöglicht die Entwicklung von Anwendungen in der Stufe „Standardisierung“. Darüber hinaus sind keine weiteren zentral definierten bzw. koordinierten Vorgaben oder Werkzeuge vorhanden.

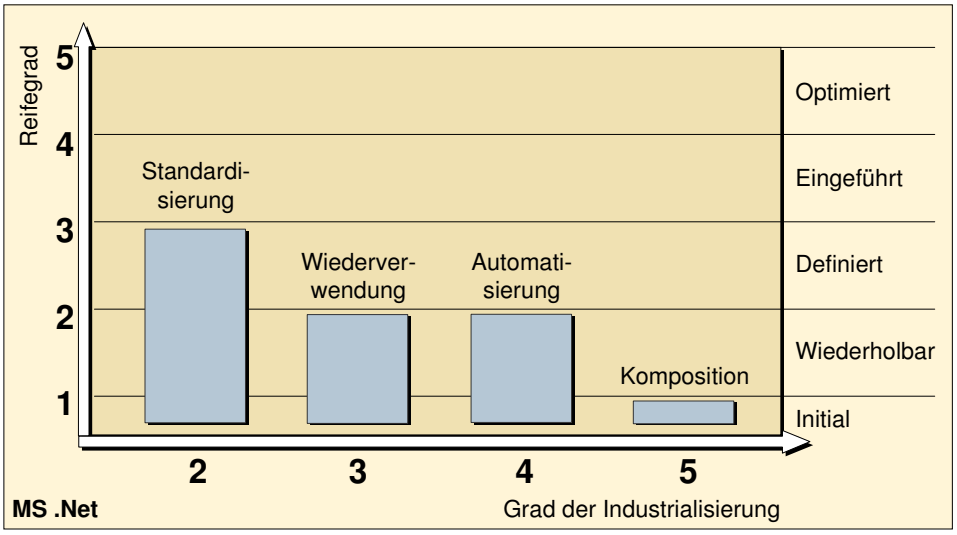

Abbildung 33: Bewertung der .Net Entwicklungsinfrastruktur

Der Reifegrad der Entwicklungsumgebung .Net ist damit lediglich bei der Standardisierung mit „definiert“ bewertet.

5.2.3.3 Zusammenfassung und weitere Plattformen

Aufgrund der Analyse des letzten Abschnittes unterstützen die betrachteten Entwicklungsinfrastrukturen die folgenden Grade mit „definiert“:

- SAP ABAP: Standardisierung
- SAP Java: Standardisierung
- Microsoft SharePoint: Standardisierung
- Microsoft .Net: Standardisierung

Speziell im Bereich Produktion gibt es weitere Entwicklungsumgebungen, die hier in komprimierter Form aufgelistet werden. Die Analyse erfolgte analog und wird an dieser Stelle nicht weiter ausgeführt.

- Delphi: initial
- Oracle: initial

5.2.4 Bewertung der vorhandenen Laufzeitinfrastruktur

Analog der Entwicklungsinfrastruktur wird bei der Laufzeitinfrastruktur ebenfalls der Schwerpunkt auf die beiden strategischen Technologien und Plattformen von SAP und Microsoft gelegt. Hierbei wurden die folgenden Bewertungen vorgenommen (gleichzeitig für SAP und Microsoft):

- Standardisierung: definiert

  Für die als Standard definierten Technologien und Plattformen existieren produktive zentral betreute Umgebungen, auf denen die implementierten Anwendungen installiert werden können.

- Wiederverwendung: definiert

  Die Wiederverwendung von Komponenten erfolgt bei der Entwicklung. Die erstellte Anwendung hat keine spezifischen Anforderungen an eine Laufzeitumgebung bzgl. der Wiederverwendung. Die Bewertung wird auf „definiert“ gesetzt, da diese Stufe aufgrund der fehlenden Anforderungen für die Laufzeitumgebung immer erfüllt ist.

- Automatisierung: definiert

  Die Automatisierung adressiert insbesondere die Generierung von Quelltext und damit den Entwicklungsprozess. An die Laufzeitinfrastruktur werden analog der Wiederverwendung keine Anforderungen gestellt. Die Bewertung wird daher ebenfalls auf „definiert“ gesetzt.

- Komposition: initial

  SAP liefert mit der NetWeaver-Plattform verschiedene Komponenten, die die Komposition von Funktionalitäten unterstützt. Die lose Kopplung zwischen den verschiedenen Elementen einer Anwendung ist damit grundsätzlich abbildbar. Diese Technologie wird jedoch in dem Unternehmen nicht eingesetzt. Hierzu fehlen die Vorgaben, wie dies in Anwendungen zu nutzen ist.

  Analog gilt dies für die Microsoft-Infrastruktur. Mit dem Biztalk-Server steht hier ebenfalls eine Plattform zur Verfügung, die die Prinzipien der Komposition unterstützt. Jedoch fehlen auch hier die konkreten Vorgaben, wie dies in den Anwendungen zu nutzen ist.

Die analysierte Laufzeitinfrastruktur bietet die notwendigen Voraussetzungen für die Stufe „Standardisierung“. Die beiden Stufen Wiederverwendung und Automatisierung haben keine spezifischen Anforderungen an die Laufzeitinfrastruktur. Die Nutzung der Komposition ist im Unternehmen noch nicht verankert. Eine entsprechende Infrastruktur ist daher nicht vorhanden.

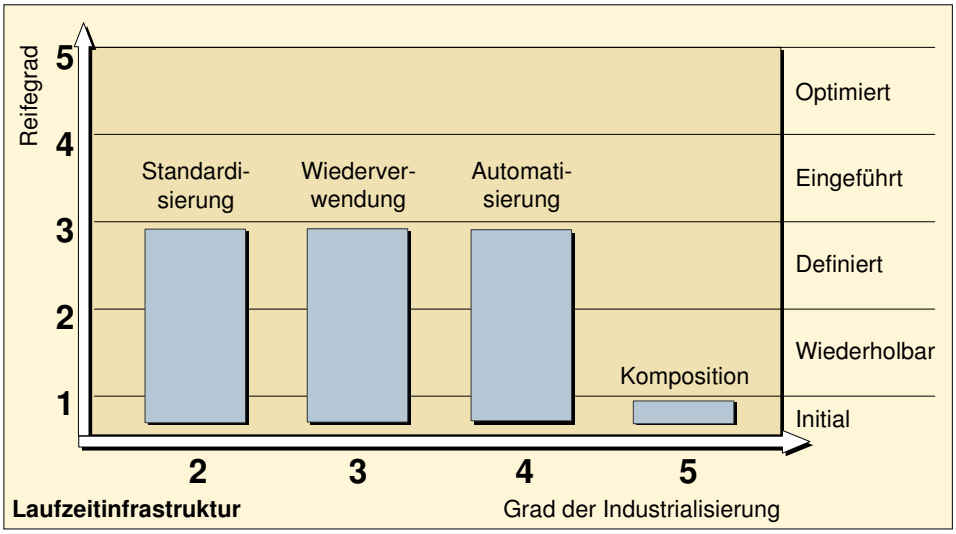

Abbildung 34: Bewertung der Laufzeitinfrastrukturen

Für die Laufzeitinfrastrukturen gilt damit bis zur Stufe Automatisierung der Reifegrad „definiert“.

5.3 Domänenspezifische Analyse

Im Rahmen der domänenspezifischen Analyse wurden die Anwendungen des Unternehmens erfasst, jeweils einer Domäne zugeordnet und bzgl. der Relevanz klassifiziert. Für jede Subdomäne wurde der Sollgrad bestimmt, der durch die Anwendungsentwicklung zu unterstützen ist.

Für die Fallstudie wird die domänenspezifische Analyse exemplarisch an der Domäne Produktion durchgeführt. Hierbei werden insbesondere die Anwendungen der Subdomäne MES betrachtet.

5.3.1 Erfassung der Anwendungen des Unternehmens

Bei der Erfassung im Unternehmen wurden insgesamt 384 Anwendungen über alle Domänen hinweg identifiziert. Tabelle 13 zeigt die Aufteilung der erfassten Anwendungen auf die verschiedenen Subdomänen.

Bei Aufsummierung der Werte aus der Tabelle ergibt sich eine Zahl von 395. Die Differenz erklärt sich aus der Mehrfachnennung von Subdomänen für spezielle Anwendungen. Für diese Anwendungen sollte geklärt werden, ob die Bereitstel-

lung der gleichen Funktionalität in verschiedenen Subdomänen sinnvoll ist. Diese Überlegungen sind jedoch nicht Gegenstand dieser Arbeit.

Domäne	Subdomäne	Anzahl App.
Produktion	Manufacturing	7
	MES	20
	Analytik	9
	Planung	5
Logistik	Inbound Logistik	9
	Interne Logistik	9
	Outbound Logistik	4
	Lager	1
Forschung & Entwicklung	Analytik	10
	Innovationsmanagement	2
Marketing & Vertrieb	CRM	23
	Vertrieb	13
Steuerung	Intellectual Property	2
	Risikomanagement	2
	Qualitätsmanagement	3
	Business Intelligence	6
	Controlling	11
	Strategie	0
	Prozessmanagement	1
Shared Services	Einkauf	44
	Unternehmenskommunikation	14
	Engineering	22
	Buchhaltung	18
	Corporate Security	14
	Personalmanagement	40
	Recht	8
	Informationsmanagement	82
	Werksmanagement	8
	Umwelt	8

Tabelle 13: Zuordnung Anwendungen zu Subdomänen

Bei der Erfassung wurden der Domäne Produktion 41 Anwendungen zugeordnet. Diese verteilen sich auf die folgenden Kategorien:

Produktion	Eigen	Kauf	Gesamt
Business	23	13	36
Klein	0	0	0
Report	3	0	3
Technisch	1	0	1
Operativ	0	0	0
Schnittstelle	1	0	1
Gesamt	28	13	41

Tabelle 14: Kategorisierung von Anwendungen der Domäne Produktion

Für die Analyse des Industrialisierungsgrades werden lediglich die Anwendungen betrachtet, die spezifisch für das Unternehmen entwickelt wurden. Die Anzahl der zu untersuchenden Anwendungen sind in der Spalte „Eigen“ aufgeführt. Insgesamt sind in der Produktion 28 Anwendungen zu analysieren. 16 Anwendungen entfallen dabei auf die Subdomäne MES.

Je nach Zielsetzung des Unternehmens ist eine weitere Einschränkung nach der Kategorie möglich. Beispielsweise könnten für den Grad der Industrialisierung lediglich die Business-Anwendungen betrachten werden. In dieser Fallstudie wird jedoch keine Einschränkung nach einer Kategorie vorgenommen.

5.3.2 Klassifikation der Anwendungen der Subdomänen

Die Anwendungen wurden bei der Erfassung bzgl. der verschiedenen Klassifikationskriterien bewertet. Die folgende Tabelle zeigt die Verteilung der erfassten Anwendungen auf die verschiedenen Kriterien der Klassifikation in der Subdomäne MES.

MES	Niedrig	Standard	Hoch	Premium
Verfügbarkeit		7	9	
Katastrophenvorsorge		8	8	
Änderbarkeit				16
Support		10	5	1
Integration		3	13	
Zugriff	14	2		
Datenkritikalität		6	2	8

Tabelle 15: Verteilung der erfassten Anwendungen auf die Klassenkriterien

Am Beispiel der Anwendung BDE (Betriebsdaten Erfassung) der Subdomäne MES wird die Einstufung in eine Klasse vorgestellt. Für diese Anwendung wurden die folgenden Werte angegeben:

BDE	Wert	numerisch
Verfügbarkeit	Standard	3
Katastrophenvorsorge	Standard	3
Änderbarkeit	Premium	1
Support	Hoch	2
Integration	Hoch	2
Zugriff	Standard	3
Datenkritikalität	Hoch	2

Tabelle 16: Beispielhafte Klassifikation einer Anwendung

Die Summe der Kriterien ergibt 16. Die Gewichtung der einzelnen Kriterien ist dabei gleich verteilt (jeweils 1/7). Nach der Formel aus Abschnitt 4.3.2 ergibt sich

die Klassifikation aus dem arithmetischen Mittel der gewichteten Kriterien. Bei dieser Anwendung liegt der Wert bei 2,29. Nach der Definition in Abschnitt 4.3.2 wird die Anwendung damit als „relevant“ klassifiziert.

Die Klassifizierung der restlichen Anwendung erfolgt analog. Für den Bereich MES ergibt sich dabei die folgende Auswertung:

MES	#Anwendungen
Kritisch	0
Relevant	11
Unterstützend	5
Geringe Relevanz	0

Tabelle 17: Klassifikation der Anwendungen der Subdomäne MES

Für die Analyse des Industrialisierungsgrades werden nur die Anwendungen untersucht, die zumindest der Klasse „unterstützend“ angehören. Ferner muss die Anwendung eine Eigenentwicklung aufweisen. Die 16 klassifizierten Anwendungen der Subdomäne MES werden nach Tabelle 17 somit für die domänenspezifische Analyse untersucht.

5.3.3 Bestimmung der Sollindustrialisierungsgrades

Der Produktionsbereich lässt sich, wie in Abschnitt 5.1.1 erläutert, in die Subdomänen MES, Manufacturing, Analytik und Produktionsplanung untergliedern. Die hierfür bereitgestellten Anwendungen decken die Anbindung an das kaufmännische System zur Auftragsverarbeitung sowie generell Planungs-, Überwachungs- und Instandhaltungsfunktionen ab.

Allen Subdomänen gemeinsam ist der Bedarf nach Standardisierung sowohl der Technologien als auch der benötigten Plattformen. Die einheitliche Integration des Auftragssystems über die Planungs- und Steuerungssysteme mit den Produktionsanlagen[100], ermöglicht die effiziente Einbindung mehrerer Betriebe oder sogar Geschäftsbereiche in den Prozess. Die Standardisierung ist somit wesentliche Voraussetzung für die kosteneffiziente Unterstützung des Produktionsprozesses.

Darüber hinaus gelten für die einzelnen Teilbereiche die folgenden Festlegungen:

[100] In diesem Kontext: Anlagen zur Erstellung der Güter des Unternehmens. Sie werden meist vom Hersteller mit vorinstallierten, spezifischen Anwendungen zur Steuerung der Anlage ausgeliefert.

- Manufacturing: Standardisierung

  Die Anwendungen im Bereich Manufacturing sind geprägt von herstellerspezifischen Anlagen. Eine Modifikation dieser Anwendungen ist ohne Verlust der Herstellergarantie nicht möglich. Sie werden daher vom restlichen Unternehmensnetz abgeschirmt, um mögliche Sicherheitslücken dieser Anwendungen lokal einzuschränken.

  Der Bedarf nach Wiederverwendung oder Automatisierung im Bereich der Anwendungsentwicklung ist nicht vorhanden. Die Komposition kommt ebenso nicht in Betracht.

- MES: Automatisierung

  Das Unternehmen ist global aufgestellt und verfügt über mehrere Produktionsstätten. Die Wiederverwendung der Systemlandschaft und der darin enthaltenen Anwendungen und Funktionen sind elementar.

  Die Prozesse im Bereich MES sind in den verschiedenen Betrieben auch innerhalb eines Geschäftsbereiches zum Teil unterschiedlich. Der grundsätzliche Verlauf ist jedoch sehr ähnlich. Es liegt daher nahe, die spezifischen Ausprägungen über eine Modellierung und damit eine Automatisierung effizienter zu gestalten.

  Die zentrale Organisation verschiedener Funktionalitäten sowie die Komposition dieser Dienste wäre ein grundsätzliches Ziel. Zurzeit sind jedoch auch die MES-Anwendungen lokal in den Betrieben gebunden. Die Komposition wird daher nicht als Ziel festgelegt.

- Analytik: Automatisierung

  In der Analytik werden unterschiedliche Systeme für die Durchführung von Messungen der Produktqualität verwendet. Die Analyseanwendungen in diesem Bereich werden jedoch nicht selbst implementiert, sondern mit den entsprechenden Anlagen mitgeliefert.

  Die Entwicklung von Anwendungen bezieht sich daher auf die Aggregation sowie der Übermittlung der Daten an zentrale Systeme. Diese sollten möglichst wiederverwendbar sein, um nicht für jede neue Anlage die gleiche Funktionalität erneut zu realisieren.

  Durch die Modellierung der Abläufe und die damit verbundene Automatisierung können diese Verfahren weiter industrialisiert werden. Man erhofft sich hierdurch Einsparungen bei der Integration neuer Anlagen.

  Die Komposition wäre auch für diesen Bereich denkbar, jedoch scheut man zurzeit den organisatorischen Aufwand, der mit der Vereinheitlichung der Services über die verschiedenen Bereiche hinweg nötig wäre.

- Produktionsplanung: Standardisierung

  Bei der Durchführung der Produktionsplanung setzt man auf Standardsoftware. Anpassungen finden lediglich im Rahmen einer Konfiguration der Anwendungen statt. Wiederverwendung und Automatisierung im Rahmen der Anwendungsentwicklung spielen in dieser Subdomäne keine Rolle. Komposition ist nicht angedacht.

Am Beispiel der Produktion wird deutlich, dass die verschiedenen Subdomänen sehr unterschiedliche Anforderungen an die Entwicklung von Anwendungen stellen. Es ist daher für die jeweilige Domäne zu prüfen, ob die Festlegung eines Industrialisierungsgrades auf Domänenebene sinnvoll ist. Allen Subdomänen gemeinsam sind die Anforderungen für die Standardisierung. Daher ist für die Domäne Produktion zumindest die Standardisierung zu realisieren.

Für die Betrachtung der einzelnen Subdomänen wird der Bereich MES exemplarisch untersucht. Die Analyse der restlichen Subdomänen erfolgt analog.

5.3.4 Ermittlung der Entwicklungsinfrastrukturen in der Produktion

Die Analyse der Anwendungen für die Domäne Produktion ergab die folgende Aufteilung auf die im Unternehmen verfügbaren Entwicklungsinfrastrukturen:

Produktion	.Net	SAP ABAP	SAP Java	Delphi	Oracle	Gesamt
MES	3			6	7	16
Analytik	3	1		1		5
Planung		1				1
Manufacturing		4	2			6
Gesamt	6	6	2	7	7	28

Tabelle 18: Zuordnung Anwendungen zu einer Entwicklungsinfrastruktur

Die Zuordnung zu den jeweiligen Infrastrukturen erfolgte anhand der Technologie, in der die Anwendung realisiert wurde. Diese Information wurde bereits bei der Erfassung der Anwendung dokumentiert.

Sofern eine Anwendung in mehreren Technologien verschiedener Infrastrukturen realisiert wurde, sind die einzelnen Teile bzw. Komponenten in den jeweiligen Infrastrukturen zu betrachten. Beispielsweise könnte für eine Anwendung die Schicht Interaktion auf einer Java-Instanz, die Geschäftslogik jedoch vollständig auf einer ABAP-Instanz mit der dafür vorgesehenen Infrastruktur realisiert werden. Die Nutzung unterschiedlicher Technologien erfordert bereits bei der Entwicklung eine klare Trennung von Zuständigkeiten. Daher können die Bestandteile auch getrennt betrachtet werden.

5.3.5 Bewertung der Anwendungen in der Subdomäne MES für .Net

Die Analyse der Anwendungen in den verschiedenen Subdomänen der Domäne Produktion wird exemplarisch für den Bereich MES vorgestellt.

Für jede Entwicklungsinfrastruktur der Subdomäne MES sind die Anwendungen zu untersuchen. Überprüft wird dabei, welche Methoden in welcher Tiefe bei der Realisierung zum Einsatz gekommen sind. Ferner wird die Nutzung der vorgegebenen Infrastruktur analysiert.

Für den Bereich .Net wurden 3 Anwendungen identifiziert (vgl. Tabelle 18). Exemplarisch wird für die Anwendung BDE die Analyse ausführlicher beschrieben. Die Betrachtung der restlichen Anwendungen erfolgt analog.

Die Anwendung BDE wird zur Erfassung von Betriebsdaten in der Produktion verwendet. Es ist eine Client-Server-Anwendung, die auf Basis des .Net Frameworks in der Sprache C# realisiert wurde.

Die Analyse der Anwendung erfolgt auf Basis der Artefakte der Anwendungsarchitektur sowie des Entwicklungsumfeldes. Ferner wird die Konformität der Nutzung der zugehörigen Entwicklungsinfrastruktur geprüft. Für jedes Element werden die einzelnen Stufen des Industrialisierungsgrades betrachtet.

Tabelle 19 stellt das Ergebnis der Analyse für BDE dar, das in den nachfolgenden Abschnitten erläutert wird.

Anwendung: BDE	Workflow	Interaktion	Business	Daten	Tests	Dokumente	Entwicklungs-infrastruktur	Laufzeit-infrastruktur
Standardisierung	-	100	100	100	100	100	100	100
Wiederverwendung	-	25	50	75	25	25	0	100
Automatisierung	-	25	50	75	0	25	0	100
Komposition	-	0	0	0	0	0	0	0

Tabelle 19: Beispielbewertung einer Anwendung

In der Anwendung sind keine Workflowkomponenten enthalten. Dieser Bereich wird daher nicht näher betrachtet.

5.3.5.1 Analyse der Artefakte der Anwendungsarchitektur

Für die Artefakte der Anwendungsarchitektur wurden die folgenden Bewertungen festgelegt.

- *Standardisierung*: Die Anwendung wurde auf Basis des .Net Frameworks von Microsoft mit der Programmiersprache C# realisiert. Beide sind als Standard definiert.

  Die Benutzeroberfläche (Interaktion) wurde mit Windows Forms anhand der gültigen Richtlinien realisiert. Die Abbildung der Geschäftslogik orientiert sich an den gültigen Standards.

  Für die Datenhaltung wurde eine SQL-Datenbank von Oracle verwendet. Der Zugriff erfolgt auf Basis von Stored-Procedures.

  Alle genannten Technologien und Plattformen sind in dem Unternehmen als Standard eingestuft. Bei der Entwicklung wurden ferner die verschiedenen Vorgaben eingehalten.

- *Wiederverwendung:* Der Aufbau der Anwendung ist in verschiedene klar identifizierbare Schichten (Interaktion/Business/Daten) aufgeteilt. In den einzelnen Schichten ist die Anwendung in eigenständige Module organisiert. Die Strukturierung und Organisation der Datenhaltung und des Datenzugriffs entspricht grundsätzlich der Methode. Bei den restlichen Artefakten ist dies nur vereinzelt erkennbar. Ansätze für diese Stufe sind vorhanden, die jedoch sehr individuell geprägt sind. Es gibt keine Grundlage für eine systematische Wiederverwendung.

- *Automatisierung*: In dieser Anwendung wurde der Zugriff auf die Daten sowie die Datenhaltung selbst auf Basis von Konfigurationen durch Generatoren automatisiert erstellt. Für Erweiterungen der Anwendung existieren Vorlagen und Strukturen, die auf Basis von Konfigurationen neue Artefakte erzeugen können. Hierbei werden automatisch die Grundfunktionen Anlegen, Lesen, Ändern und Löschen für die Datenschicht generiert. Diese Methode ist für die Artefakte der restlichen Schichten nur vereinzelt erkennbar.

  Die Generierung der verschiedenen Artefakte wurde individuell durch den Entwickler erzeugt. Ein allgemeines Verfahren zum Erzeugen von Artefakten im .Net Umfeld ist noch nicht vorhanden.

- *Komposition*: Eine Komposition von Artefakten ist in dieser Anwendung nicht erkennbar.

5.3.5.2 Analyse der Artefakte des Entwicklungsumfeldes

Die Bewertung der Artefakte aus dem Entwicklungsumfeld der Anwendung basiert auf der folgenden Betrachtung:

- *Standardisierung:* Die für die Anwendung erstellten Tests entsprechen den Standardvorgaben. Die einzelnen Anwendungsszenarien wurden als

Testfälle beschrieben. Ferner wurden die durchgeführten Tests anhand der Testfälle dokumentiert.

Die Dokumentation enthält alle benötigten Bestandteile der Anwendung in dem vorgegeben Format. Die geforderten Artefakte entsprechen damit den Standardvorgaben.

Alle Artefakte sind auf einem aktuellen Stand.

- *Wiederverwendung:* Eine Komponentenorientierung der Tests sowie der Dokumentation ist aufgrund der Strukturierung der Anwendung zum Teil vorhanden. Die Ausprägung ist jedoch individuell und orientiert sich nicht an den unternehmensspezifischen Vorgaben.
- *Automatisierung:* Die Tests werden noch manuell erstellt. Es gibt erste, jedoch sehr individuelle Ansätze der Automatisierung. Teilweise wird die Dokumentation der Anwendung aus den Kommentaren der Quelldateien generiert. Auch hier ist dies jedoch sehr individuell durch den Entwickler geprägt. Die restliche Dokumentation wird manuell erstellt.
- *Komposition:* Eine Komposition der Tests sowie der Dokumentation ist nicht vorhanden.

5.3.5.3 Nutzung der vorgegebenen Entwicklungsinfrastruktur

Die Programmierung der Anwendung erfolgt auf Basis der definierten Entwicklungswerkzeuge. Die Richtlinien zur Entwicklung von .Net-Anwendungen werden eingehalten. Das zentrale Konfigurationsmanagement zur Verwaltung der Quelldateien sowie der Informationen zur aktuellen Version der Anwendung wird genutzt. Der Transport sowie das Deployment entsprechen den Standardvorgaben. Ein zentraler Build-Service ist vorhanden und wird genutzt.

Für die Wiederverwendung, Automatisierung sowie der Komposition ist die Umgebung noch nicht „definiert“. Daher kann die Entwicklung dieser Anwendung auch nicht entsprechend konform sein.

5.3.5.4 Nutzung der vorgegebenen Laufzeitinfrastruktur

Die im Rahmen von .Net definierten Standards für die Laufzeitinfrastruktur werden genutzt.

Für die Wiederverwendung und Automatisierung gibt es keine spezifischen Anforderungen an eine Laufzeitinfrastruktur.

Eine .Net Infrastruktur für die Komposition ist nicht vorhanden und kann daher nicht genutzt werden.

5.3.5.5 Einstufung der Beispiel Anwendung

Aus der Tabelle 19 kann die Bewertung der Anwendung entnommen werden. Da lediglich für die Standardisierung die Spalten der Artefakte mindestens den Wert 75 sowie die Spalten der Infrastrukturen 100 enthalten, wird für die Anwendung BDE die Stufe „Standardisierung“ zugewiesen. Bei der Wiederverwendung und Automatisierung sind insbesondere bei der Datenhaltung sowie beim Zugriff auf die Daten Ansätze vorhanden, die jedoch individuell gestaltet wurden. Komposition ist nicht vorhanden. Die Anwendung erfüllt somit die Vorgaben der Stufe Standardisierung. Die Analyse der restlichen Anwendungen der Subdomäne erfolgt analog.

5.4 Bewertung der Domäne Produktion

Nachdem die Grundlagen in den domänenübergreifenden und -spezifischen Analysen erarbeitet wurden, erfolgt die schrittweise Bewertung der Subdomänen der Domäne Produktion. Hierbei wird der Industrialisierungsgrad zunächst in jeder Subdomäne gruppiert nach den Infrastrukturen ermittelt. Das Ergebnis wird anschließend zum Industrialisierungsgrad der Subdomäne aggregiert.

Die Ermittlung erfolgt auch in diesem Abschnitt exemplarisch anhand von Beispielen. Die restlichen für den nächsten Schritt erforderlichen Ergebnisse werden lediglich angegeben und nicht weiter erläutert.

5.4.1 Bewertung der .Net Infrastruktur in der Subdomäne MES

In der Subdomäne MES gibt es für die .Net-Entwicklungsinfrastruktur zwei weitere Anwendungen. Die Analyse erfolgte analog für die Elemente in den verschiedenen Stufen. Auch diese Anwendungen orientieren sich am gültigen Standard. Erste Ansätze zur Wiederverwendung und Automatisierung sind vorhanden, jedoch sind diese durch den jeweiligen Entwickler geprägt. Komposition ist nicht vorhanden. Tabelle 20 stellt das Ergebnis für die .Net-Entwicklungsinfrastruktur dar.

.Net (MES)	Anwendungen[101] in %	Entwicklungs-infrastruktur	Laufzeit-infrastruktur
Standardisierung	100	definiert	definiert
Wiederverwendung	0	wiederholbar	definiert
Automatisierung	0	wiederholbar	definiert
Komposition	0	initial	initial

Tabelle 20: Ergebnis Analyse der Anwendungen

[101] Anteil der Anwendungen der betrachteten Infrastruktur, die aufgrund der Bewertung der Artefakte sowie der Nutzung der Infrastruktur die Vorgaben der Stufe erfüllen.

Aus der Betrachtung der Entwicklungsinfrastruktur in Abschnitt 5.2.3.2 sowie der Bewertung der Anwendungen in Abschnitt 5.3.5 ergibt sich für die .Net-Infrastruktur in der Subdomäne MES der Industrialisierungsgrad „2 – Standardisierung“ (siehe Abbildung 35).

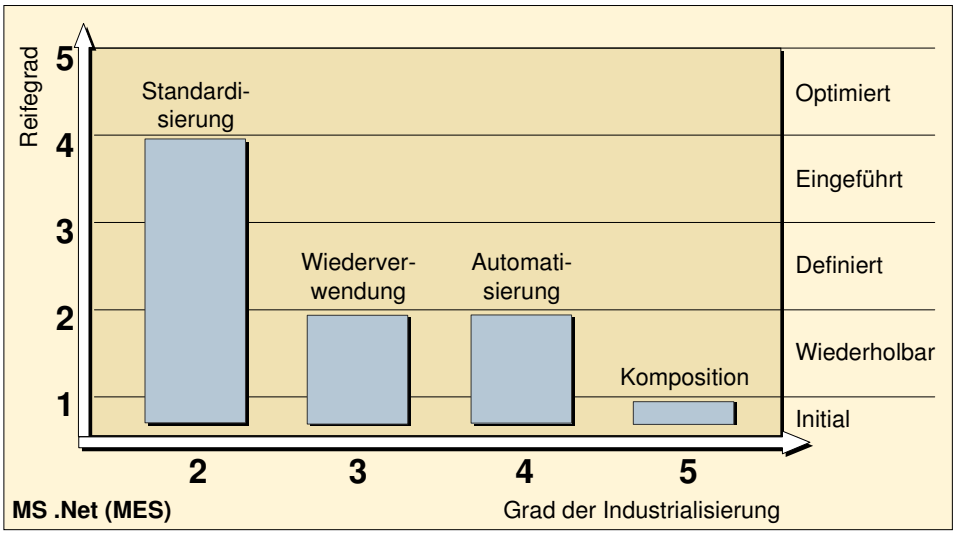

Abbildung 35: Bewertung der MES Anwendungen der .Net-Infrastruktur

Die Herleitung des Industrialisierungsgrades für den .Net-Anteil der Subdomäne MES ergibt sich aus der Zusammenfassung der bisherigen Analysen:

- Standardisierung: eingeführt

  Die Entwicklungsinfrastruktur ist bzgl. der Komponenten im Standard vollständig. Die Anwendungen richten sich überwiegend nach den Vorgaben und sind damit konsistent zur Infrastruktur. Die einzelnen Artefakte sind auf einem aktuellen Stand. Aufgrund der Einhaltung der Vorgaben wird die Stufe als „eingeführt“ bewertet. Für den Reifegrad „optimiert“ gibt es erste Ansätze hinsichtlich der Rückmeldungen zu vorgegebenen Standards sowie deren Optimierung. Dieser Prozess ist jedoch noch nicht allgemeingültig aufgesetzt.

- Wiederverwendung: wiederholbar

  Vereinzelt wird die Wiederverwendung in Anwendungen betrachtet. Dies gilt insbesondere für den strukturierten Aufbau der Anwendungen. Jedoch gibt es für dieses Umfeld noch keine abgestimmten zentralen Vorgaben. Die Entwicklungsinfrastruktur ist bzgl. dieser Methode nicht vollständig und kann somit auch nicht etabliert sein. Die Wiederverwendung hat daher den Reifegrad „wiederholbar“.

- Automatisierung: wiederholbar

  Bei den analysierten Anwendungen gibt es verschiedene Ansätze zur Automatisierung bei der Erstellung der Artefakte. Der Bedarf für diese Methode

wurde erkannt. Jedoch stellt die Entwicklungsinfrastruktur noch keine definierten Verfahren und Werkzeuge zur Verfügung, die diese Methode für alle Anwendungen systematisch unterstützen. Die Entwicklungsinfrastruktur ist damit nicht vollständig. Diese Stufe hat somit den Reifegrad „wiederholbar“.

- Komposition: initial

  Der Bedarf für die Komposition wird noch nicht gesehen. Die Entwicklungsinfrastruktur stellt weder Methoden noch Werkzeuge zur Verfügung. Auch bei der Realisierung der Anwendungen ist die Komposition nicht weiter betrachtet worden. Die Komposition hat daher den Reifegrad „initial“.

5.4.2 Bewertung der Subdomäne MES

Aus Tabelle 18 wird ersichtlich, dass sich in der Subdomäne MES weitere Anwendungen befinden. Diese verteilen sich auf die Entwicklungsinfrastrukturen Delphi und Oracle. Für beide Infrastrukturen wurden die Bewertungen analog durchgeführt. In Tabelle 21 wird das Ergebnis zusammengefasst.

Delphi wurde in dem betrachteten Unternehmen nicht als Standard definiert, daher sind auch die verwendeten Infrastrukturen nicht etabliert. Die Entwicklungen in diesem Bereich sind sehr individuell geprägt.

Die Anwendungen im Umfeld von Oracle basieren auf Oracle Forms. Diese Technologie sowie die zugehörige Entwicklungsinfrastruktur wurden in der Vergangenheit als Standard deklariert. Obwohl Oracle diese Technologie in dieser Form nicht weiter unterstützt, wird der Status „Standard“ momentan nicht geändert. Der Aufwand für eine Migration in eine andere Technologie ist derzeit zu hoch. Da die Analyse eine Ist-Betrachtung ist, wird dieser Umgebung sowie den Anwendungen noch der Industrialisierungsgrad „Standardisierung“ zugesprochen.

MES	.Net	Delphi	Oracle
Standardisierung	eingeführt	initial	eingeführt
Wiederverwendung	wiederholbar	initial	wiederholbar
Automatisierung	wiederholbar	initial	wiederholbar
Komposition	initial	initial	initial

Tabelle 21: Bewertung der Subdomäne MES

Tabelle 21 stellt die Bewertungen der in der Subdomäne MES vorhandenen Infrastrukturen dar. Der Grad der Industrialisierung richtet sich nach der Infrastruktur mit dem geringsten Wert. Aufgrund der Bewertung bei Delphi kann der Subdomäne MES daher nur der Industrialisierungsgrad „initial“ vergeben werden.

5.4.3 Vergleich des ermittelten Grades mit dem Sollgrad

Die restlichen Subdomänen der Domäne Produktion wurden nach dem gleichen Schema bewertet. Tabelle 22 fasst das Ergebnis zusammen.

Die Anwendungen der Subdomäne Analytik basieren zum Teil auf standardisierten Entwicklungsinfrastrukturen. Jedoch wurden die Vorgaben bei der Realisierung der Artefakte sowie der Nutzung der Infrastrukturen überwiegend nicht eingehalten. Die Stufe Standardisierung ist somit nicht erreicht.

Die identifizierten Anwendungen der Subdomäne Manufacturing basieren im Wesentlichen auf Basis der SAP ABAP-Infrastruktur und orientieren sich nach den vorgegebenen Standards. Wiederverwendung und Automatisierung sind in diesem Umfeld jedoch noch nicht vorgesehen.

Produktion	Istgrad	Sollgrad	Maßnahmen
MES	Initial	Automatisierung	eingeleitet
Analytik	Initial	Automatisierung	eingeleitet
Manufacturing	Standardisierung	Standardisierung	-
Planung	Standardisierung	Standardisierung	-

Tabelle 22: Vergleich des ermittelten Grades mit dem Sollgrad

In der Subdomäne Planung wird überwiegend Kaufsoftware eingesetzt, bei der keine Entwicklung stattgefunden hat. Die verbleibende Eigenentwicklung basiert auf der SAP ABAP-Infrastruktur und wurde nach den entsprechenden Vorgaben realisiert. Somit konnte auch hier der Industrialisierungsgrad „Standardisierung“ ermittelt werden. Ansätze zur Wiederverwendung oder Automatisierung sind nicht vorhanden.

Der spezifizierte Sollgrad ist für die Bereiche Planung und Manufacturing erreicht. In diesen Subdomänen werden keine Veränderungen bzgl. der Anwendungsentwicklung benötigt.

Für MES sowie die Analytik ist der geforderte Sollgrad nicht erreicht. Der Unterschied zur Vorgabe (Stufe Automatisierung) ist noch sehr groß. Für diese Subdomänen wurden erste Maßnahmen zur Etablierung der Standardisierung der Technologien sowie der Plattformen verabschiedet. Auch die einheitliche Dokumentation von Prozessen wird eingefordert.

Erste Ansätze gibt es auch im Bereich der Wiederverwendung sowie der Automatisierung. Hiervon erhofft man sich eine Steigerung der Produktivität in der Anwendungsentwicklung. Neben ersten Versuchen, Generatoren selbst zu implementieren, werden auch die Möglichkeiten von Microsoft und SAP näher betrachtet. Die Aktivitäten werden jedoch nachrangig zur Standardisierung verfolgt.

5.4.4 Bewertung der Domäne Produktion

Bei der Bestimmung des Sollindustrialisierungsgrades für die einzelnen Subdomänen wurden bereits unterschiedliche Grade festgelegt. Für die Bewertung der Domäne muss daher hinterfragt werden, welche Schlussfolgerung aus dem ermittelten Grad gezogen werden kann.

Sofern dennoch für die Domäne der Industrialisierungsgrad ermittelt werden soll, richtet sich dieser nach der Subdomäne mit der geringsten Einstufung.

Aufgrund der Bewertungen in den Subdomänen MES und Analytik wird die Domäne Produktion mit „initial“ bewertet.

6 Zusammenfassung und Ausblick

Die Steigerung der Effizienz bei der Herstellung von Gütern basiert in der Fertigungsindustrie auf der Standardisierung, der Wiederverwendung, der Automatisierung, der Arbeitsteilung sowie der Verringerung der Fertigungstiefe. Die Anwendung dieser Prinzipien steigerte in der Vergangenheit kontinuierlich die Produktivität von Unternehmen der Fertigungsindustrie.

Im Vergleich zur Fertigungsindustrie ist die Softwareentwicklung eine junge Disziplin. Aber auch in diesem Umfeld werden die industriellen Prinzipien schrittweise eingeführt.

Mit ihren Konzepten, Technologien und Plattformen bietet die Softwarebranche bereits jetzt eine Fülle an Möglichkeiten, eine Industrialisierung der Anwendungsentwicklung in den Unternehmen einzuführen. Diese Konzepte werden jedoch teilweise noch isoliert voneinander betrachtet. Die Frage, welche der Konzepte im Rahmen einer Industrialisierung in welcher Reihenfolge in einem Unternehmen eingeführt werden sollte, wird derzeit noch nicht beantwortet. Die bekannten Reifegradmodelle, wie CMMI oder SPICE, geben hierzu keine Auskunft, da sie sich auf die Prozesse der Softwareentwicklung und nicht auf die Methoden der Programmierung von Anwendungen beziehen.

In dieser Arbeit wurde ein Stufenmodell aufgebaut, in dem existierende Konzepte der Softwareentwicklung integriert wurden. Jede Stufe repräsentiert dabei ein Prinzip der Industrialisierung, das durch eine entsprechende Methode der Softwareentwicklung umgesetzt wurde. Die Stufen bauen aufeinander auf. Dies erlaubt Unternehmen eine schrittweise Verbesserung der Anwendungsentwicklung in Bezug auf die Faktoren Kosten, Zeit und Qualität. Die Qualität wird dabei in Unterfaktoren wie Änderbarkeit oder Zuverlässigkeit aufgeteilt.

Als Grundlage einer Optimierung wurde in dieser Arbeit ein Bewertungsverfahren vorgestellt. Dies erlaubt Unternehmen, eine Bestandsaufnahme der aktuellen Anwendungsentwicklung durchzuführen. Das Stufenmodell enthält Kriterien für jede Stufe, auf deren Basis überprüft werden kann, ob die entsprechende Methode bei der Softwareentwicklung im Unternehmen umgesetzt bzw. etabliert wurde. Ausgehend von dieser Bestandsaufnahme können Schritte zur Verbesserung der Anwendungsentwicklung eingeleitet werden.

Der Grad der Industrialisierung gibt Auskunft darüber, in welcher Stufe sich die Anwendungsentwicklung im Unternehmen befindet. Dabei spielt die Aufteilung des Unternehmens in verschiedene Domänen und Subdomänen eine wichtige Rolle. Nicht in jeder Subdomäne ist die volle Industrialisierung der Anwendungsentwicklung sinnvoll. Vor einer Bewertung ist daher der Sollindustrialisierungsgrad für jede Subdomäne zu bestimmen. Nach der Bewertung können anschlie-

ßend aus einer möglichen Differenz zu dem geforderten Sollindustrialisierungsgrad Maßnahmen definiert werden.

Im Rahmen einer Fallstudie wurden die Konzepte dieser Arbeit einem praktischen Test unterzogen. Sowohl das Stufenmodell als auch das Bewertungsverfahren konnten dabei erfolgreich eingesetzt werden. Sie lieferten wertvolle Hinweise auf die Optimierung der Anwendungsentwicklung in dem betrachteten Unternehmen.

An mehreren Stellen im Stufenmodell, aber auch in dem Bewertungsverfahren, wurden bewusst Freiheitsgrade eingebaut. Dies ermöglicht verschiedenen Unternehmen, spezifische Schwerpunkte zu integrieren. Der Nachteil dieser Freiheitsgrade ist jedoch die fehlende Möglichkeit, sich mit anderen Unternehmen in einem Benchmark zu vergleichen. Benchmarks werden in der IT verwendet, um Potentiale für Optimierungen zu erkennen. Ob diese Freiheitsgrade jedoch notwendig sind, könnte Gegenstand weiterer Untersuchungen sein. Hierzu müsste die gleiche Bewertung in verschiedenen Unternehmen durchgeführt und die Ergebnisse analysiert werden.

Welche Technologie oder welches Werkzeug in den einzelnen Stufen für die jeweilige Methode zum Einsatz kommt, ist ebenfalls bewusst offengelassen worden. Verschiedene Technologien und Werkzeuge können jedoch unterschiedliche Ergebnisse bei der Optimierung der Anwendungsentwicklung aufzeigen. In einer weiteren Untersuchung könnte ermittelt werden, welches die jeweils für eine Stufe geeignete Technologie bzw. das geeignete Werkzeug ist.

Die exemplarische Erfassung der Anwendungen sowie die Bewertung der Infrastrukturen und Domänen erfolgte in dieser Arbeit initial. Sofern sich das IT-Management aufgrund dieser Bewertungsergebnisse für eine Optimierung der Anwendungsentwicklung entscheidet, ist eine regelmäßige Bewertung durchzuführen. Um die Transparenz der für diese Optimierung notwendigen Maßnahmen und der einzelnen Bewertungen zu erhöhen, sollte ein Werkzeug entwickelt werden, das den Prozess der Bewertung unterstützt. Dieses Werkzeug sollte zudem die initiale Bewertung, den geforderten Sollindustrialisierungsgrad sowie den aktuellen Stand der Anwendungsentwicklung in einer Übersicht zur Verfügung stellen. Dies dokumentiert dem IT-Management den aktuellen Fortschritt. Für die Bewertung sind in diesem Werkzeug die verschiedenen Gremien, Prozesse, Domänen, Subdomänen, Infrastrukturen, Technologien und Anwendungen zu verwalten.

Das Stufenmodell enthält fünf Stufen. In der niedrigsten Stufe hat der Entwickler alle Freiheitsgrade für die Erstellung von Anwendungen. Der Auftraggeber ist in dieser Stufe bei Änderungen meist direkt von diesem Entwickler abhängig. In der höchsten Stufe kann dagegen auch der Auftraggeber Modifikationen an Prozessen durchführen. Der Auftraggeber ist in dieser Stufe jedoch im Unternehmen.

Da sich die Unternehmensgrenzen zunehmend öffnen und die Kunden vermehrt auf die internen Prozesse Einfluss haben, könnten in einer sechsten Stufe die Möglichkeiten und Methoden zur Gestaltung von Prozessen durch den Kunden untersucht werden.

Die Verfügbarkeit der Konzepte, Methoden und Vorgehensmodelle sind kein Garant für die Etablierung der Industrialisierung. Der wichtigste Faktor ist die Kooperation der Organisation und darin jeder einzelne Mitarbeiter. Sofern die Notwendigkeit der Industrialisierung in den verschiedenen Bereichen von den Entwicklern, Projektleitern und Managern, aber auch den Auftraggebern nicht erkannt wird, bleibt die Steigerung der Produktivität aus. Dieser soziale Aspekt ist nur am Rande durch die Verfügbarkeit von Gremien und Prozessen angedeutet worden. Die Steigerung des Industrialisierungsgrades von einer Stufe auf die nächste ist meist mit einem kulturellen Wandel in der Organisation verbunden. Anhand von weiteren Reifegradkriterien könnte untersucht werden, ob und wie sich dieser Wandel auf die Steigerung der Produktivität auswirkt.

Glossarverzeichnis

ANSI	American National Standards Institute US-amerikanisches Institut zur Normung industrieller Verfahrensweisen.
ARIS	Architektur integrierter Informationssysteme Konzept zur umfassenden Beschreibung von Organisationen, das die Grundlage für die Analyse und Gestaltung von Informationssystemen liefert.
ASL	Application Services Library Sammlung von Best Practices für das Management von bestehenden Softwareanwendungen.
CBS	Component Build Service Komponente aus der NetWeaver Entwicklungsinfrastruktur (NWDI). Zuständig für die zentrale Erstellung einer Anwendung.
CE	Composition Environment Entwicklungs- und Laufzeitplattform der SAP für die Komposition von Anwendungen auf Basis von Services.
CMMI	Capability Maturity Model Integration Modell zur Bewertung von Prozessen in der Anwendungsentwicklung.
CMS	Change Management Service Komponente aus der NetWeaver Entwicklungsinfrastruktur (NWDI). Über diese Komponente werden die Anwendungen von Entwicklung über Test nach Produktion transportiert.
CORBA	Common Object Request Broker Architecture Spezifikation für eine objekt-orientierte Middleware. Sie ermöglicht die Kommunikation zwischen Anwendungen, die auf unterschiedlichen Plattformen eingesetzt werden.
CRM	Customer Relationship Management Managementsystem zur Unterstützung von Kundenbindungsmaßnahmen.

DIN	Deutsches Institut für Normung Wichtigste deutsche Normenstelle. DIN-Normen bilden einen Maßstab für einwandfreies technisches Verhalten.
Domäne	Gruppierung der Anwendungslandschaft nach fachlichen Gesichtspunkten.
DTR	Design Time Repository Versionskontrollsystem zur Verwaltung von Quelltexten. Komponente aus der NetWeaver Entwicklungsinfrastruktur (NWDI).
EAI	Enterprise Application Integration Konzept zur unternehmensweiten Integration der Geschäftsfunktionen, die über verschiedene Applikationen auf unterschiedlichen Plattformen verteilt sind.
EJB	Enterprise Java Beans Standardisierte Komponenten innerhalb eines Java-EE-Servers (Java Enterprise Edition). Sie vereinfachen die Entwicklung komplexer mehrschichtiger verteilter Softwaresysteme mittels Java.
EPK	Ereignisgesteuerte Prozesskette Grafische Notation von Geschäftsprozessen. Abfolge von Funktionen, die durch Ereignisse ausgelöst und gesteuert werden. Konnektoren legen fest, wie Ereignisse und Funktionen verknüpft werden.
ERP	Enterprise Resource Planning Anwendungssoftware, um die in einem Unternehmen vorhandenen Ressourcen (Kapital, Betriebsmittel oder Personal) möglichst effizient für den betrieblichen Ablauf einzusetzen und somit die Steuerung von Geschäftsprozessen zu optimieren.
ESA	Enterprise Services Architecture Konzept des Softwareherstellers SAP, wie eine serviceorientierte Architektur auf Basis der SAP Netweaver Produktplattform gestaltet werden kann.
ESB	Enterprise Service Bus Infrastruktur einer SOA-Landschaft, die die Interoperabilität der Services ermöglicht. Kernaufgaben: Konnektivität schaffen, Daten zwischen verschiedenen Plattformen transportieren, Nachrichten weiterleiten sowie die Möglichkeit zum Überwachen, Protokollieren und zur Fehleranalyse zu schaffen.

HTML	Hypertext Markup Language
	Hypertext Meta Language: Sprache zur Formatierung von Inhalten einer Webseite.
HTTP	Hypertext Transfer Protocol
	Protokoll für den Austausch von Informationen zwischen einem Browser und einem WebServer in einem Netzwerk.
IDE	Integrated Development Environment
	Ein meist grafisches Werkzeug zur Entwicklung spezifischer Software für oder in einer bestimmten Umgebung.
ISO	Internationale Organisation für Normung
	Internationale Vereinigung von Normungsorganisationen. Erstellt internationale Normen in allen Bereichen. Ausnahme: Elektrik, Elektronik und Telekommunikation.
JEE	Java Platform, Enterprise Edition
	Umgebung für die Entwicklung und Bereitstellung mehrschichtiger webbasierter Unternehmensanwendungen. Sie besteht aus einer Reihe von Diensten, APIs und Protokollen, die die Funktionalitäten zur Entwicklung dieser Anwendungen bereitstellen.
ITIL	IT Infrastructure Library
	Sammlung von Best-Practices in einer Reihe von Publikationen, die eine mögliche Umsetzung eines IT-Service-Managements beschreiben.
KPI	Key Performance Indicator
	Kennzahlen, anhand derer der Fortschritt oder der Erfüllungsgrad hinsichtlich wichtiger Zielsetzungen oder kritischer Erfolgsfaktoren innerhalb einer Organisation gemessen und/oder ermittelt werden kann.
MDA	Model Driven Architecture
	Methode zur Implementierung von Anwendungen basierend auf Modellen und Generatoren zur Steigerung der Produktivität und Qualität bei der Softwareentwicklung.
MDSD	Model Driven Software Development
	Oberbegriff für Techniken, die aus formalen Modellen automatisiert lauffähige Software erzeugen.

MES Manufacturing Execution System

Gesamtsystem, das den Bereich zwischen einem ERP-System der Unternehmensleitebene und dem eigentlichen Fertigungs- bzw. Produktionsprozess in der Fertigungs- bzw. Automatisierungsebene abdeckt.

MSDI Microsoft Development Infrastructure

Entwicklungsinfrastruktur für die Erstellung von Anwendungen basierend auf Microsoft Technologien und Plattformen.

NWDI Netweaver Development Infrastructure

Entwicklungsinfrastruktur für die Erstellung von Anwendungen basierend auf Technologien und Plattformen der SAP NetWeaver Java-Instanz.

OASIS Organization for the Advancement of Structured Information

Standards

Internationales gemeinnütziges Konsortium, das die Entwicklung, die Zusammenführung sowie die Adaption von Open Source Software vorantreibt.

OMG Object Management Group

Internationales gemeinnütziges Konsortium, das Standards für die objektorientierte Programmierung entwickelt.

PIM Platform Independent Model

Abstrahiertes Modell der Model Driven Architecture, das die Anforderungen der Anwendung wiedergibt, jedoch keine plattformspezifischen Attribute enthält.

PNN Produktionsnahes Netz

Bereich eines Netzwerkes, das die Produktionsanlagen umfasst und vom restlichen Unternehmensnetz durch eine Firewall abgeschirmt ist.

PSM Platform Specific Model

Modell einer Anwendung im Rahmen der Model Driven Architecture angereichert mit spezifischen Eigenschaften der Zielplattform.

RAD Rapid Application Development

Vorgehensmodell zur Realisierung von Anwendungen. Zeichnet sich durch kurze Entwicklungszyklen aus, um eine rasche Rückmeldung von dem Auftraggeber zu bekommen.

RTE Round Trip Engineering

Verfahren, wie ein aus Modellen erzeugter und manuell veränderter Quelltext wieder in das Modell zurückgeführt werden kann.

RUP Rational Unified Process

Iteratives Vorgehensmodell zur Entwicklung von Software. Es basiert auf der Dokumentation von Anwendungsfällen und stellt die Architektur ins Zentrum der Planung. Der RUP ist ein kommerzielles Produkt der Firma Rational (seit 2002 Teil von IBM).

SaaS Software as a Service

Software-Distributions-Modell mit der Philosophie, Software als Dienstleistung basierend auf Internettechniken bereitzustellen, zu betreuen und zu betreiben.

SEI Software Engineering Institute

Forschungs- und Entwicklungszentrum an der Carnegie Mellon University in Pittsburgh, Pennsylvania. Forschungsschwerpunkt liegt auf komplexen, verteilten, eingebetteten und echtzeitfähigen Systemen.

SLD System Landscape Directory

Komponente aus der SAP NetWeaver Entwicklungsinfrastruktur (NWDI). Verwaltet die Komponenten in der SAP NetWeaver Systemlandschaft.

SOA Service-orientierte Architektur

Paradigma für die Strukturierung und Nutzung verteilter Funktionalität (fachlich wie technisch), die von unterschiedlichen Besitzern verantwortet wird.

SOAP Simple Object Access Protocol

Netzwerkprotokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und entfernte Aufrufe durchgeführt werden können. Wird insbesondere für den Aufruf von WebServices sowie den Transfer von Daten verwendet.

SPALM SharePoint Application Lifecycle Management

Ein für die SharePoint-Umgebung spezifisches Framework für die Handhabung des Lebenszyklus von SharePoint-Anwendungen. Enthält insbesondere einen Generator zum Erzeugen von Quelltexten. Open Source Projekt der Firma Steria Mummert.

SPICE	Software Process Improvement and Capability Determination Internationaler Standard zur Durchführung von Bewertungen von Unternehmensprozessen mit Schwerpunkt auf der Softwareentwicklung.
SPDI	SharePoint Development Infrastructure Entwicklungsinfrastruktur für die SharePoint-Umgebung. Enthält alle Komponenten bzw. Werkzeuge für die Entwicklung von SharePoint-Anwendungen.
TFS	Team Foundation Server Komponente aus der Microsoft Entwicklungsinfrastruktur (MSDI). Versionskontrollsystem zur Verwaltung von Quelltexten sowie ein System zur Verwaltung von Anforderungen.
UML	Unified Modeling Software Sprache für die Modellierung von Software und anderen Systemen. Wurde von der Object Management Group (OMG) entwickelt und standardisiert.
URL	Uniform Resource Locator (Internet-Adresse) Identifiziert und lokalisiert eine Ressource über das verwendete Netzwerkprotokoll (beispielsweise HTTP oder FTP) und den Ort (engl. location) der Ressource in Computernetzwerken.
V-Modell	Phasenorientiertes Vorgehensmodell zur Entwicklung von Software. Die Aktivitäten bei der Erstellung der Software werden Phasen zugeordnet, die sequentiell durchgeführt werden.
WSDL	Web Services Description Language Eine auf XML basierende Sprache, die der standardisierten Definition von Webdiensten dient: Definition des Webdiensts, Anweisungen zum Zugriff auf den Webdienst sowie der Speicherort dieses Webdiensts.
XML	Extensible Markup Language Eine vom World Wide Web Consortium (W3C) entwickelte flexible Programmiersprache für die Erstellung allgemeiner Informationsformate.
XP	Extreme Programming Prototypisches Vorgehensmodell zur Entwicklung von Software. Der Fokus liegt auf der raschen Erstellung von Software mit einem Minimum an flankierenden Maßnahmen (z.B. Modellierung).

Literaturverzeichnis

[Adam01] *Adam, D.:* Produktions-Management, Gabler-Verlag, Wiesbaden 2001

[Alfe05] *Alfert, K.:* Von der Manufaktur zur industriellen Softwareproduktion, in: ObjektSpektrum 3/2005, Sigs-Datacom Verlag 2005

[Andr04] *Andresen, A.*: Komponentenbasierte Softwareentwicklung mit MDA, UML 2 und XML, Hanser Verlag, 2. Auflage, 2004

[Balz01] *Balzert, H.*: Lehrbuch der Software-Technik, Bd.1: Software-Entwicklung, Spektrum Akademischer Verlag, Heidelberg 2001

[BBMP06] *Beer, W.; Birngruber, D.; Mössenböck, H.; Prähofer, H.; Wöß, A.*: Die .Net-Technologie: Grundlagen und Anwendungsprogrammierung, dpunkt-Verlag, Heidelberg 2006

[Bien07] *Bien, A.*: Java EE 5 Architekturen, entwickler.press Verlag, Paderborn 2007

[BuKö98] *Buxmann, P.; König, W.*: Das Standardisierungsproblem: Zur ökonomischen Auswahl von Standards in Informationssystemen, in: Wirtschaftsinformatik Heft 2, 1998

[DeMa97] *DeMarco, T.*: Warum ist Software so teuer? Und andere Rätsel des Informationszeitalters, Carl Hanser Verlag, München, Wien 1997

[DuGa06] *Durst, M.; Gary, J.*: Sicherstellung der Strategiekonformität in IT-Projekten, in: Multikonferenz Wirtschaftsinformatik 2006, GITO Verlag, Berlin 2006

[Dumk03] *Dumke, R.:* Software Engineering, Vieweg Verlag, Wiesbaden 2003

[BuKn02] *Bunse, C.; Knethen, A.*: Vorgehensmodelle kompakt, Spektrum Akademischer Verlag, Heidelberg, Berlin 2002

[EHHJ08] *Engles, G.; Hess, A.; Humm, B.; Juwig, O.; Lohmann, M.; Richter, J-P.; Voß, M.; Willkomm, J.*: Quasar Enterprise, Anwendungslandschaften service-orientiert gestalten, dpunkt Verlag, Heidelberg 2008

[EzMT98] *Ezran, M.; Morisio, M.; Tully, C.*: Practical Software Reuse: the essential guide, Freelife Pub., Paris 1998

[Erl08] *Erl, T.*: SOA: Entwurfsprinzipien für service-orientierte Architektur, Addison-Wesley Verlag, München 2008

[Gart96] *Gartner (Schulte, R.; Natis, Y.)*: Service Oriented Architectures, Parts 1 and 2, http://www.gartner.com/DisplayDocument?id=302868; http://www.gartner .com /DisplayDocument?id=302869

[Gart03] *Gartner (Natis, Y.)*: Service-Oriented Architecture Scenario, http://www .gartner.com/DisplayDocument?doc_cd=114358

[GrLO07] *Grollius, T.; Lonthoff, J.; Ortner, E.*: Softwareindustrialisierung durch Komponentenorientierung und Arbeitsteilung, in: HMD – Praxis der Wirtschaftsinformatik, Heft 256: IT-Industrialisierung, dpunkt Verlag, Heidelberg 2007

[GrBB02] *Grässle, P.; Baumann, H.; Baumann, P.*: UML projektorientiert, Geschäftsprozessmodellierung, IT-Systemspezifikation und Systemintegration mit der UML, Galileo Press , Bonn 2002

[GrSh06] *Greenfield, J.; Short, K.*: Software Factories – Moderne Software-Architekturen mit SOA, MDA, Patterns und agilen Methoden, mitp Verlag, Heidelberg 2006

[HeKa08] *Heilig, L.; Karch, S.*: SAP NetWeaver, 2. Auflage, Galileo Press, Bonn 2008

[HaLi07] *Hack, S.; Lindemann, M.*: Enterprise SOA einführen, Galileo Press, Bonn 2007

[HDHM06] *Hörmann,K.; Dittmann, L.; Hindel, B; Müller, M.*: SPICE in der Praxis, dpunkt Verlag, Heidelberg 2006

[Herz04] *Herzig, A.*: Über Transformationen und Patterns, in: ObjektSpektrum 1/2004, Sigs-Datacom Verlag 2004

[HBMK07] *Hess, T.; Buxmann, P.; Mann, F.; Königer, M.*: Industrialisierung der Softwarebranche: Erfahrungen deutscher Anbieter, in: Management Reports des Instituts für Wirtschaftsinformatik und Neue Medien, LMU München, München, Nr. 2/2007

[HüBa08] *Hüttenrauch, M.; Baum, M.*: Effiziente Vielfalt – die dritte Revolution in der Automobilbranche, Springer Verlag 2008

[Janß05] *Janßen, R.*: Die Psychologie des Entwicklers, in Informatik Spektrum August 2005, Springer Verlag, Heidelberg 2005

[Josu08] *Josuttis, N.*: SOA in der Praxis – System-Design für verteilte Geschäftsprozesse, dpunkt-Verlag, Heidelberg 2008

[KaSc07] *Kaufmann, T.; Schlitt, M.*: Industrielle Konzepte bei der Entwicklung und Produktion von IT-Services, in: HMD – Praxis der Wirtschaftsinformatik, Heft 256: IT-Industrialisierung, dpunkt Verlag, Heidelberg 2007

[Kell07] *Keller, W.*: IT-Unternehmensarchitektur: Von der Geschäftsstrategie zur optimalen IT-Unterstützung, dpunkt Verlag, Heidelberg 2007

[Kilg86] *Kilger, W.*: Industriebetriebslehre, Band 1, Gabler Verlag, Wiesbanden 1986

[Klar06] *Klar, M.; Klar, S.*: Einfach Generieren, Generative Programmierung verständlich und praxisnah, Hanser Verlag, München 2006

[Kneu06] *Kneuper, R.*: CMMI – Verbesserung von Softwareprozessen mit Capability Maturity Model Integration, 2. Auflage, dpunkt Verlag, Heidelberg 2006

[KrBS05] *Krafzig, D.; Banke, K.; Slama, D.*: Enterprise SOA: Service Oriented Architecture Best Practices, Verlag Prentice Hall International, 2005

[Kruc00] *Kruchten, P.*: The Rational Unified Process, an Introduction, Addison Wesley 2000

[KTVo07] *Kilian-Kehr, R.; Terzidis, O.; Voelz, D.*: Industrialisation of the Software Sector, in: Wirtschaftsinformatik 2007 Sonderheft: Der Softwarestandort im Zeichen von Industrialisierung und Globalisierung, Vieweg Verlag

[Kutt07] *Kuttruff, V.*: Realisierung von Softwareproduktlinien durch Komposition modularer Belangimplementierungen, in: Wirtschaftsinformatik Band 49 Heft 3, Vieweg Verlag, Wiesbaden 2007

[LeWS08] *Lehner, F.; Wildner, S.; Scholz, M.*: Wirtschaftsinformatik – Eine Einführung, 2. Auflage, Hanser Verlag, München 2008

[Lim94] *Lim, W.C.*: Effects of Reuse on Quality, Productivity and Economics, IEEE Software, September 1994

[LiRW02] *Lippert, M.; Roock, S.; Wolf, H.*: Software entwickeln mit eXtrem Programming, dpunkt Verlag, Heidelberg 2002

[Maid05] *Maidl, J.*: Spannungsfeld zwischen Standard und Prozessführerschaft, in: Informatik Spektrum August 2005, Springer Verlag, Heidelberg 2005

[McII68] *McIlroy, D.*: Mass Produced Software Components, in: Buxton, M.; Naur, P.; Randell, B. (Hrsg.): Software Engineering Concepts and Techniques. Report on a Conference by the NATO Scientific Affairs Division, Brüssel 1968

[MeAp07] *Meister, J.; Appelrath, H.-J.*: Produktgetriebene Entwicklung von Softwareproduktlinien, in: Wirtschaftsinformatik Band 49 Heft 3, Vieweg Verlag, Wiesbaden 2007

[MüHe04] *Müller, A.; Hess, T.*: Konzepte der Standardisierung betrieblicher Anwendungssysteme, Arbeitspapiere des Instituts für Wirtschaftsinformatik und Neue Medien, LMU München, München, Nr. 2/04, Abruf am 25.03.08 http://www.wim.bwl.uni-muenchen.de/download/epub/ab_2004_02.pdf

[Müll03] *Müller, T.*: Eine Bestandsaufnahme und Bewertung nationaler und internationaler Standards im Bereich Supply Chain Execution mit Relevanz für Warehouse Management Software Systeme, GRIN Verlag 2003

[Oasi06] *Oasis*: Reference Model for Service Oriented Architecture 1.0, http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf (Abruf 01.12.2009)

[OrHE98] *Orfali, R.; Harkey, D.; Edwards, J.*: Instant CORBA – Führung durch die CORBA Welt, Addison-Wesley-Longman Verlag, Bonn 1998

[Parn76] *Parnas, D.*: On the Design and Development of Programm Families, in IEEE Transactions on Software Engineering, March 1976

[PfWi07] *Pfeiffer, D.; Winkelmann, A.*: Ansätze zur Wiederverwendung von Software im Rahmen der Software-Industrialisierung am Beispiel von Softwarekomponenten, service-orientierte Architekturen und modellgetriebenen Architekturen, in Wirtschaftsinformatik 49 (2007) Heft 3, Vieweg Verlag, Wiesbaden 2007

[Pols07] *Pols, Remko van der*: ASL - Ein Framework für das Application Management, Van Haren Publishing 2007

[PoRS00] *Pomberger, G.; Rezagholi, M.; Stobbe, C.*: RCMM/RCAM – Modelle zur Bewertung und Verbesserung wiederverwendungsorientierter Software-Entwicklungsprozesse Handbuch für Evaluation und Evaluierungsforschung in der Wirtschaftsinformatik (Handbuch für Praxis, Lehre und Forschung); Hrsg. L.J. Heinrich, I. Häntschel, Oldenbourg Verlag (München/Wien), 2000

[Reza04] *Rezagholi, M.*: Prozess- und Technologiemanagement in der Softwareentwicklung, Ein Metrik basierter Ansatz zur Bewertung von Prozessen und Technologien, Oldenburg Wissenschaftsverlag, München 2004

[RMH08] *Reinke, H.; Monadjemi, A; Herzog, D.*: Unternehmensprozesse optimieren mit Sharepoint – Das Prozesshaus-Modell, Microsoft Press Deutschland 2008

[Rübb72] *Rübbert, R.*: Geschichte der Industrialisierung – Wirtschaft und Gesellschaft auf dem Weg in unsere Zeit, München 1972

[SAP05] *SAP*: Blueprint für das Bankengeschäft der Zukunft, Vortrag beim Bankeninfotag in Wien 2005, http://www.pbfa-ffm.com/wp-content/a4.pdf (Abruf: 01. 12. 2009)

[Sche02] *Scheucher, R.*: Strategische Geschäftsfeldanalyse, in: Das große Handbuch der Strategieinstrumente. Simon, H./Von der Gathen, A., Hg. Campus-Verlag, 2002

[sd&m07] *sd&m*: Service Orientierte Architektur, Interner Vortrag von sd&m 2007

[SiSM06] *Simon, F.; Seng, O.; Mohaupt, T.*: Code Quality Management – Technische Qualität industrieller Softwaresysteme transparent und vergleichbar gemacht, dpunkt Verlag, Heidelberg 2006

[Stan04] *The Standish Group International Inc.*: Third Quarter Research Report, Chaos-Report 2004

[Stan09] *The Standish Group International Inc.*: CHAOS Summary 2009, Auszug der Ergebnisse des Chaos-Reports 2009 über die WebSeite http://www.standishgroup.com/newsroom/chaos_2009.php (Abruf: 1.12.2009)

[Star02] *Starke, G.*: Effektive Software Architekturen, Carl Hanser Verlag, München, Wien 2002

[SRMC09] *Snabe, H.; Rosenberg, A.; Møller, C.; Scavillo, M.*: Business Process Management – the SAP Roadmap, Galileo Press Verlag, Bonn, Boston 2009

[Taub05] *Taubner, D.*: Software Industrialisierung, in: Informatik Spektrum Band 28 Heft 4 (August 2005), Springer Verlag, Heidelberg 2005

[UHSB07] *Uebernickel, F.; Hochstein, A.; Schulz, V.; Brenner, W.*: Excellence-Modell der Industrialisierung des Informationsmanagements, in: HMD – Praxis der Wirtschaftsinformatik, Heft 256: IT-Industrialisierung, dpunkt Verlag, Heidelberg 2007

[WaBK07] *Walter, S.M.; Böhmann, T.; Krcmar, H.*: Industrialisierung der IT – Grundlagen, Merkmale und Ausprägungen eines Trends, in: HMD – Praxis der Wirtschaftsinformatik, Heft 256: IT-Industrialisierung, dpunkt Verlag, Heidelberg 2007

[Wall01] *Wallmüller, E.*: Software-Qualitätsmanagement in der Praxis, Software-Qualität durch Führung und Verbesserung von Software-Prozessen, 2. Auflage, Carl Hanser Verlag, München, 2001

[WaSi07] *Wanner, G.; Siegl, S.*: Modellgetriebene Softwareentwicklung auf Basis von open-Source; in: Informatik-Spektrum 2007 Band 30 Heft 5, Springer Verlag, Berlin Heidelberg 2007

[Wede05] *Wedekind, H.*: Die Arbeitsteilung ist es!, in: Informatik-Spektrum, 28 Heft 5 (Oktober 2005), Springer Verlag 2005

[WeKö03] *Weitzel, T.; König, W.*: Computational Economics als wirtschaftsinformatischer Beitrag zu einer interdisziplinären Netzwerktheorie, in: Wirtschaftsinformatik Band 45 Heft 5, Vieweg Verlag, 2003.

[Wey99] *Wey, C.*: Marktorganisation durch Standardisierung: ein Beitrag zur neuen Institutionenökonomik des Marktes, Rainer Bohn Verlag, 1999

[ZaBP05] *Zarnekow, R; Brenner, W; Pilgram, U.*: Integriertes Informationsmanagement, Springer Verlag, Heidelberg 2005