

TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Informatik XIX

**Developing Organization-Specific Enterprise  
Architecture Management Functions Using a  
Method Base**

Sabine M. Buckl

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität  
München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr. H. Krcmar

Prüfer der Dissertation:

1. Univ.-Prof. Dr. F. Matthes
2. Prof. Dr. P. Johnson,  
KTH Royal Institute of Technology, Stockholm, Schweden

Die Dissertation wurde am 15.02.2011 bei der Technischen Universität München  
eingereicht und durch die Fakultät für Informatik am 14.04.2011 angenommen.



# Zusammenfassung

Die gegenseitige Ausrichtung von Geschäft und IT beschreibt den beständigen Wandel mit dem sich Unternehmen im Zeitalter globalisierter Märkte, sich schnell verändernder gesetzlicher Vorschriften und technologischer Innovationen konfrontiert sehen. Vor dem Hintergrund dieses Spannungsfeldes gewinnt das Management der Unternehmensarchitektur (EAM) als Instrument zur gesteuerten Weiterentwicklung des Unternehmens an Bedeutung. Dabei stellt das EAM Methoden für die Beschreibung, Analyse und Kommunikation des Ist-Zustandes, des erwünschten Soll-Zustandes sowie von Transformationsplänen der Unternehmensarchitektur bereit. Die Forschung und Praxis hat in der Vergangenheit eine Vielzahl von Entwurfstheorien, Fallstudien, Standards und praxiserprobten Lösungen über die Gestaltung einer EAM Funktion veröffentlicht. Die Anwendung dieser Ansätze ist jedoch mit Herausforderungen, wie der Ausrichtung an sich ändernden Problemstellungen, sowie der Anpassung der Funktion an den Unternehmenskontext, verbunden. Um von der bereitgestellten Wissensbasis profitieren zu können, stehen Unternehmen vor dem Dilemma, die Entwurfstheorien zu identifizieren, auszuwählen und zu verbinden, welche zu ihrem spezifischen Unternehmenskontext sowie ihren Problemstellungen passen.

Die vorliegende Forschungsarbeit stellt eine Entwurfsmethode zur Gestaltung unternehmensspezifischer EAM Funktionen vor, welche die genannten Herausforderungen adressiert. Anhand eines Aktivitätsmodells unterstützt die Entwurfsmethode Unternehmensarchitekten bei der inkrementellen Ausgestaltung einer problemadäquaten und unternehmensspezifischen EAM Funktion unter Verwendung einer Methodenbibliothek, welche praxiserprobte Lösungen in Form von Methodenbausteinen in einem Theorieverbund zusammenführt. Unter Ausnutzung der Organisation der Methodenbausteine in der Methodenbibliothek werden im Rahmen der Entwurfsmethode passende Methodenbausteine für ein Unternehmen identifiziert. Hierzu stellt die Entwurfsmethode einen Katalog von Kontextbeschreibungen zur Klassifikation des Unternehmens bereit. Zusätzlich werden Methodenbausteine anhand eines zweidimensionalen Modells von EAM Problemstellungen, welches auf der Unterscheidung von abstrakten Zielen und möglichen Anwendungsgebieten basiert, an die unternehmensspezifischen Problemstellungen angepasst. Mit Hilfe der von der Entwurfsmethode bereitgestellten Techniken zur Konfiguration, Adaption und Integration werden die Methodenbausteine zu einer umsetzbaren EAM Funktion zusammengefügt. Eine Methode zur konsistenten Weiterentwicklung der Methodenbibliothek erlaubt die Integration von neuen oder veränderten Problemstellungen, Kontexten sowie Lösungen und rundet den Beitrag der Arbeit ab.

Der Entwurf und die Entwicklung einer organisationsspezifischen EAM Funktion stellt einen komplexen und fehleranfälligen Prozess dar, welcher von einer Werkzeugunterstützung profitiert. Im Rahmen der Arbeit wird ein prototypisches Werkzeug vorgestellt, welches den Zugriff auf die Methodenbibliothek erleichtert und den Unternehmensarchitekten bei der Durchführung der Schritte der Entwurfsmethode unterstützt. Eine Evaluierung der Methodenbibliothek, der darauf aufbauenden Entwurfsmethode und der Entwurfstechniken anhand konkreter Praxisfälle schließt die Arbeit ab.



# Abstract

The mutual alignment of business and IT refers to various challenges that modern enterprises are exposed to due to globalized markets, changing legal regulations, and technological innovations. In response to these challenges enterprises aim at a strategic management of their enterprise architecture (EA) which provides a holistic model of the key elements and relationships of an enterprise and connects strategy, business, and IT. EA management is a strategic management function to describe, analyze, and communicate the current, planned, as well as the envisioned target state of the EA. Past research has produced a multitude of case studies and theories to design the EA management function. At the same time, practitioners have developed voluminous standards and textbooks that document best practices. However, enterprises have difficulties in applying these theories and practices and to adapt them to their specific problems and organizational contexts. Since context and problems change over time, a regular adaptation of the EA management function is required. As a consequence, it is difficult for an organization to benefit from the body of knowledge and to select as well as combine theories that match its specific context and problems.

In this thesis we present a method to develop organization-specific EA management functions closing the above identified gap. The development method builds on a method base, consisting of interrelated *method building blocks* (MBB) extracted from practice-proven approaches. Based on an activity framework of the EA management function the development method guides an organization in incrementally designing an EA management function by completing the activities with algorithmically identified fine-grained MBBs suitable for the given organizational context and the specific problem. The development method builds on a two-dimensional model of EA management problems distinguishing between the abstract goal and the concern describing the part of the EA to which the goal applies. The development method employs a catalog of organizational context descriptions relevant for the design of the EA management function and supplies techniques to ensure operational implementability. The development method further provides guidelines for adapting EA management functions in response to changing organizational contexts and problems. In addition, we supply an administration method to develop and evolve the method base in a consistent and coherent manner with newfound best practices.

Theory-based design and development of an EA management function is a complex and error-prone process, for which technical support is considered helpful. In this thesis, we present a prototypic web-based toolset that supports the enterprise architect in executing the development method and applying the configuration techniques. The applicability of the development method, the method base, and the corresponding techniques is evaluated in case studies from the finance industry and the public sector.



# Acknowledgment

Though my name is printed on the cover of this thesis, the word “I” does not appear within its chapters. I do this to pay tribute to the encouragement of my advisors and colleagues, and the support of my family and friends. I am indebted to many people for their long-lasting support. In the following lines some of them are gratefully acknowledged. However, I am aware of the fact that words cannot express the gratitude and respect I feel for all of them.

First and foremost, I would like to thank my supervisor Prof. Dr. Florian Matthes for his constant support, insightful suggestions, and intellectual guidance to make this thesis reality through many obstacles and difficult periods. I further want to express my sincere gratitude to Prof. Dr. Pontus Johnson for being my second referee and for our interesting discussions in the Blue Mountains (Australia) and in Stockholm.

The sebis chair has provided an excellent environment for my research. I spent many enjoyable hours with my colleagues chatting about my latest crazy ideas over a cup of coffee. Without this rich environment I doubt that many of my ideas would have come to fruition. My thanks go to Dr. Thomas Büchner, Dr. Alexander M. Ernst, Dr. Josef Lankes, Ivan Monahov, Christian Neubert, Sascha Roth, Christopher Schulz, Alexander Steinhoff, and Dr. André Wittenburg.

I would also like to thank the students who in one way or the other contributed to my research with their theses, guided research, or as students working at sebis. A special thanks goes to Thomas Dierl, who spend almost his whole student life as a motivated member of our team.

I would like to express my gratitude to all the workshop participants, interviewees, and project partners who supported my work by spending their time and sharing their views and experiences with me. Reflecting research ideas against practice greatly contributed to my work, allowed me to keep my research focused, and enabled evaluation of the results. My thanks in particular goes to Dr. Andreas Gehlert who reviewed different parts of this thesis.

This work would not have been possible without the support of my colleague and friend Christian M. Schweda, who was willing to pursue the goal of BEAMS by researching the language counterpart. You were instrumental in helping me get past all the self-doubting that inevitably crops up in the course of a Ph.D. For all of this, I thank you.

Last, but most of all, thanks to my family for their unconditional love and support. I am most grateful to my parents Erika and Manfred for their support and encouragement during my whole life, to my sister Claudia, who ensured that I take a rest by going for a ride with the horses, and my brother and his wife Dr. Katharina and Dr. Christian Buckl for reviewing this thesis. Finally, I want to give a special thanks to Timm Goldes, my ‘significant other’. You have given me the love, courage, and support to make the next transitions in my life.

Garching b. München, 14.02.2011



Sabine Buckl





<b>1</b>	<b>Introduction and Motivation</b>	<b>1</b>
1.1	Problem statement and research questions . . . . .	4
1.2	Contributions of this thesis . . . . .	8
1.3	Outline of this thesis . . . . .	11
1.4	Writing conventions . . . . .	13
<b>2</b>	<b>Research Design</b>	<b>15</b>
2.1	Research assumptions . . . . .	17
2.1.1	Ontological assumption . . . . .	18
2.1.2	Epistemological position . . . . .	19
2.1.3	Linguistic position . . . . .	23
2.2	Research outcomes . . . . .	24
2.2.1	Theorizing in design science research . . . . .	25
2.2.2	The role of patterns for theory building . . . . .	28
2.2.3	The concept of design theory nexus . . . . .	35
2.2.4	Information systems generators . . . . .	37
2.2.5	Synthesizing the research outcomes of this thesis . . . . .	38
2.3	Research method . . . . .	40
2.3.1	Problem diagnosis . . . . .	42
2.3.2	Nexus instantiation . . . . .	43
2.3.3	Application and evaluation . . . . .	45
2.4	Summary of the research design . . . . .	49
<b>3</b>	<b>Analyzing the State-of-the-Art in Designing EA Management Functions</b>	<b>53</b>
3.1	Fundamental perspectives on methods for EA management . . . . .	54
3.1.1	Modeling perspective: Model theory and the ISO Std. 42010 . . . . .	54
3.1.2	Design perspective: Simon's the sciences of the artificial . . . . .	58
3.1.3	Management perspective: The Deming wheel or Shewart cycle . . . . .	60
3.1.4	Knowledge management perspective: The cycle of Probst . . . . .	61
3.1.5	A systemic perspective: The viable systems model . . . . .	64

3.2	Framework for analyzing EA management methods . . . . .	67
3.3	Revisiting prominent to EA management . . . . .	71
3.3.1	The Zachman Framework . . . . .	74
3.3.2	Architecture of Integrated Information Systems (ARIS) . . . . .	76
3.3.3	The Generalised Enterprise Reference Architecture and Methodology (GERAM) . . . . .	78
3.3.4	Semantic Object Model approach (SOM) . . . . .	81
3.3.5	The Open Group Architecture Framework (TOGAF) . . . . .	83
3.3.6	Extended Enterprise Architecture (E2A) . . . . .	87
3.3.7	The EA management approach of MIT . . . . .	90
3.3.8	The EA management approach of TU Lisbon . . . . .	93
3.3.9	The Systemic Enterprise Architecture Methodology (SEAM) . . . . .	95
3.3.10	Archimate . . . . .	97
3.3.11	The EA management approach of KTH Stockholm . . . . .	99
3.3.12	The EA <sup>3</sup> Cube <sup>TM</sup> . . . . .	101
3.3.13	The EA management approach of Niemann . . . . .	104
3.3.14	The EA management approach of the University of St. Gallen . . . . .	106
3.3.15	Strategic IT management of Hanschke . . . . .	110
3.4	Summary and conclusion . . . . .	112
<b>4</b>	<b>Towards a Building Block-Based Method to Design EA Management Functions</b>	<b>115</b>
4.1	Characteristics of the development method . . . . .	116
4.1.1	Quality criteria of an organization-specific EA management function . . . . .	116
4.1.2	Requirements for a development method to design EA management functions . . . . .	127
4.2	Contributing theories . . . . .	134
4.2.1	The fundamentals of methods . . . . .	134
4.2.2	(Situational) method engineering . . . . .	136
4.2.3	Goal-oriented and actor-oriented modeling: the i* framework . . . . .	139
4.2.4	A pattern language for EA management . . . . .	141
4.3	BEAMS: A conceptual overview . . . . .	144
4.4	Summary . . . . .	157
<b>5</b>	<b>BEAMS: Building Blocks for EA Management Solutions</b>	<b>159</b>
5.1	Method building blocks . . . . .	161
5.1.1	Constituents of an MBB . . . . .	162
5.1.2	A notation for method building blocks . . . . .	168
5.2	Characterize situation . . . . .	173
5.2.1	Constituents of the method base . . . . .	173
5.2.2	Rules and technique to characterize a situation . . . . .	180
5.2.3	Development method . . . . .	182
5.3	Configure MBBs . . . . .	186
5.3.1	The variable concept . . . . .	186
5.3.2	Rules and techniques for configuring MBBs . . . . .	191
5.3.3	Development method . . . . .	200
5.4	Analyze EA management function . . . . .	205
5.4.1	Stakeholders and actors . . . . .	205

---

5.4.2	Techniques for ensuring organizational implementability . . . . .	206
5.4.3	Development method . . . . .	206
5.5	Adapt and evolve the EA management function . . . . .	209
5.6	Developing and maintaining the method base . . . . .	212
5.7	Summary . . . . .	223
<b>6</b>	<b>Developing a Prototypic Toolset: The BEAMS Configurator</b>	<b>227</b>
6.1	Use cases of the BEAMS configurator . . . . .	228
6.2	Design of the BEAMS configurator . . . . .	232
6.3	Prototypic implementation of the BEAMS configurator . . . . .	234
6.3.1	Browse and administrate the method base . . . . .	234
6.3.2	Develop the EA management function . . . . .	236
6.3.3	Exporting the EA management function . . . . .	239
6.4	Summary and conclusion . . . . .	239
<b>7</b>	<b>Evaluating and Applying BEAMS</b>	<b>241</b>
7.1	Theoretic evaluation against the characteristics . . . . .	242
7.2	Comparing BEAMS with the state-of-the-art . . . . .	244
7.2.1	Revisiting a fictitious case from literature . . . . .	245
7.2.2	Combining BEAMS with other approaches . . . . .	247
7.3	Observational case studies . . . . .	247
7.3.1	A case from the financial industry . . . . .	249
7.3.2	A case from the public sector . . . . .	254
7.3.3	Findings from the case studies . . . . .	257
7.4	Critical reflection of the evaluation . . . . .	258
<b>8</b>	<b>Critical Reflection and Outlook</b>	<b>261</b>
8.1	Summary of results . . . . .	261
8.2	Critical reflection and future research . . . . .	263
<b>A</b>	<b>An excerpt from BEAMS</b>	<b>265</b>
A.1	Method building blocks . . . . .	265
A.2	Catalog of organizational contexts . . . . .	276
A.3	Catalog of participants . . . . .	279
<b>B</b>	<b>A case study from the public sector</b>	<b>280</b>
B.1	Selected and configured MBBs . . . . .	280
	<b>Bibliography</b>	<b>283</b>



---

## List of Figures

---

1.1	High-level structure of the EA . . . . .	3
1.2	Outline of the thesis . . . . .	11
2.1	Research design according to Becker et al. [Be03, page 5] . . . . .	16
2.2	Epistemological, ontological, and linguistic framework adapted from Becker et al. [Be03] and Becker and Niehaves [BN07] . . . . .	19
2.3	Elements of theorizing in design science research [NCP91, Ge09] . . . . .	25
2.4	Components of a design theory according to Walls et al. in [WWES92] . . . . .	27
2.5	Components of a design theory according to Gregor and Jones in [GJ07] . . . . .	29
2.6	Levels of abstractions of research outcomes according to [VK04] . . . . .	30
2.7	Components of a design theory according to Buckl et al. [BMS10k] . . . . .	34
2.8	Components of a design theory nexus according to Pries-Heje and Baskerville in [PHB08] . . . . .	36
2.9	The research activity framework of this thesis (based on Venable’s activity framework [Ve06a, Ve06b]) . . . . .	41
2.10	The epistemological, ontological, and linguistic assumptions underlying this thesis	50
3.1	Excerpt from the conceptual model of the ISO 42010 [In07] . . . . .	57
3.2	Conceptual framework for EA design . . . . .	59
3.3	The PDCA cycle according to Deming in [De82, page 88] . . . . .	60
3.4	The KM cycle of Probst (cf. [Pr98]) . . . . .	62
3.5	Applying a viable system perspective to EA management . . . . .	66
3.6	Two dimensional schema behind the Zachman Framework (Source: <a href="http://zachmanframeworkassociates.com/index.php/">http://zachmanframeworkassociates.com/index.php/</a> ) . . . . .	75
3.7	ARIS house . . . . .	77
3.8	The components of the GERAM framework [IF99, page 5] . . . . .	79
3.9	V-Model (method model) of the SOM approach [FS95, page 9] . . . . .	82
3.10	The architecture development method of TOGAF [Th09a, page 54] . . . . .	85
3.11	The enterprise architecture program cycle according to Schekkerman in [Sc08a, page 38] . . . . .	87

List of Figures

---

3.12	The foundation for execution according to Ross et al. [RWR06, page 10]	90
3.13	The unification core diagram according to Ross et al. [RWR06, page 54]	91
3.14	Basic development method of SEAM according to Wegmann et al. [We08]	96
3.15	The ArchiMate architecture framework according to Jonkers et al. [Jo03]	98
3.16	The normative EA management process of Johnson and Ekstedt [JE07]	100
3.17	The EA <sup>3</sup> cube description framework [Be05, page 40]	102
3.18	The EA Cycle of Niemann [Ni06c]	104
3.19	The information model of Niemann [Ni06c]	105
3.20	Essential layers of an EA [WF06]	107
3.21	EA process model of Hafner and Winter [HW08]	108
3.22	EA framework “Best practice enterprise architecture” of Hanschke [Ha10, page 66]	110
3.23	IT landscape planning process according to Hanschke [Ha10, page 158]	111
4.1	Fundamental elements of a method according to Gutzwiller [Gu94, page 13]	135
4.2	The process of situational method engineering according to [Ha97]	138
4.3	Design theory nexus instantiation for EA management functions	145
4.4	Interlinking MBBs and LBBs via different types of variables	148
4.5	Transforming enterprises: the interplay between models and methods for EA management	150
4.6	Current, planned, and target state of the EA	150
4.7	Method framework detailing the main activities of an EA management function	152
5.1	The main contributions of BEAMS: method base, development, and administration method	160
5.2	Constituents of an MBB—detailed method description	163
5.3	Constituents of an MBB—task, forces, and consequences	164
5.4	Constituents of an MBB—participants and tasks	165
5.5	Constituents of an MBB—participants and viewpoints	167
5.6	Conceptual model of an MBB	168
5.7	A method description of the EA management function	172
5.8	BEAMS: interrelating context, problem, and MBB	173
5.9	Constituents of an MBB—tasks and information model	177
5.10	Constituents of an MBB—pre-conditions and post-conditions	178
5.11	Integrating MBBs with other enterprise-level management functions	180
5.12	The MBB constituents and their environmental factors as interlinked by the BEAMS method base	181
5.13	Development method: characterize situation	183
5.14	Information model variable	188
5.15	Input and output information models of tasks	190
5.16	Constituents of an MBB—trigger and trigger variables	191
5.17	The conceptual model of BEAMS	192
5.18	Development method: configure EA management function	200
5.19	Constituents of an MBB—Stakeholders and actors	206
5.20	Development method: analyze EA management function	207
5.21	Administration method—developing and maintaining the method base	213
5.22	MBB describe by interview	222
5.23	Activity diagram illustrating the development method	224

---

6.1	Use case diagram illustrating the development of an organization-specific EA management function . . . . .	230
6.2	Use case illustrating the administration of the method base . . . . .	231
6.3	BEAMS configurator: editor for modeling MBBs . . . . .	235
6.4	Method base: a wiki page describing MBBs . . . . .	236
6.5	BEAMS configurator: dashboard of the fictitious example . . . . .	237
6.6	Not customized method resulting from MBB integration . . . . .	238
6.7	BEAMS configurator: results from applying the assessment technique . . . . .	238
7.1	Combining BEAMS with TOGAF's ADM . . . . .	248
7.2	Case study FI: excerpt from the method description . . . . .	252
7.3	Case study FI: stakeholder-actor-dependencies . . . . .	253
7.4	Case study PS: excerpt from the method description . . . . .	256
7.5	Case study PS: stakeholder-actor-dependencies . . . . .	257
A.1	MBB develop planned states of the EA . . . . .	266
A.2	MBB ensure information consistency . . . . .	268
A.3	MBB Publish architectural Description . . . . .	270
A.4	MBB Officially gratify standard conformance . . . . .	271
A.5	MBB Perform single expert evaluation . . . . .	273
A.6	MBB Multi expert evaluation . . . . .	274





---

## List of Tables

---

2.1	Existing approaches to design theorizing in IS research . . . . .	33
2.2	The contributions of this thesis and their relations to the research outcomes . .	51
2.3	The research method of this thesis and contributing works . . . . .	51
3.1	Method classification for an EA management approach . . . . .	71
3.2	Characterization of the review synthesis presented in this chapter according to Fettke in [Fe06, page 259] . . . . .	73
3.3	Method classification for the Zachman Framework . . . . .	75
3.4	Method classification for ARIS approach . . . . .	77
3.5	Method classification for GERAM . . . . .	81
3.6	Method classification for the SOM approach . . . . .	83
3.7	Method classification for TOGAF . . . . .	86
3.8	Method classification for E2A . . . . .	89
3.9	Method classification for the approach of MIT . . . . .	92
3.10	Method classification for the approach of TU Lisbon . . . . .	94
3.11	Method classification for the Systemic Enterprise Architecture Methodology . .	97
3.12	Method classification for Archimate . . . . .	99
3.13	Method classification for approach of KTH Stockholm . . . . .	101
3.14	Method classification for EA <sup>3</sup> Cube Framework . . . . .	103
3.15	Method classification for the approach of Niemann . . . . .	106
3.16	Method classification for the University of St. Gallen . . . . .	109
3.17	Method classification for Strategic IT management . . . . .	112
3.18	Summary of method classifications . . . . .	114
4.1	Quality criteria for EA management function and the originating sources . . . .	127
5.1	Fulfillment of requirements of the selected process modeling languages . . . . .	169
5.2	Graphic elements for describing EA management methods . . . . .	171
5.3	Relating activities and meta-attributes . . . . .	195
5.4	MBB describe by interview . . . . .	221

6.1	Realizing use cases of the BEAMS configurator by enterprise 2.0 services . . . .	240
7.1	Method classification for BEAMS . . . . .	245
7.2	<i>ACME Energy</i> : results from applying the assessment technique . . . . .	246
7.3	<i>ACME Energy</i> : results from applying the assessment technique (second iteration)	246
7.4	Case study FI: selected MBBs . . . . .	251
7.5	Case study FI: pre-conditions and post-conditions of the selected MBBs . . . .	252
7.6	Case study PS: pre-conditions and post-conditions of the selected MBBs . . . .	255
A.1	MBB develop planned states of the EA . . . . .	265
A.2	MBB ensure consistency . . . . .	268
A.3	MBB Ensure Consistency . . . . .	270
A.4	MBB officially gratify standard conformance . . . . .	271
A.5	MBB perform single expert evaluation . . . . .	273
A.6	MBB multi expert evaluation . . . . .	274
B.1	The selected MBBs for the case study from the public sector . . . . .	281

---

## List of Examples

---

1.1	Expository example . . . . .	14
3.1	Interests in a system . . . . .	55
4.1	The i* framework . . . . .	140
4.2	The variable concept . . . . .	147
4.3	The EA management activity framework . . . . .	153
4.4	Relationships and conditions . . . . .	154
4.5	Integrable, alternatives, and related MBBs . . . . .	155
5.1	Forces influencing task execution . . . . .	164
5.2	Participant involvement . . . . .	166
5.3	Types of viewpoints . . . . .	166
5.4	An exemplary EA management method . . . . .	172
5.5	Problems from the EA management pattern catalog . . . . .	174
5.6	Operationalizing goals . . . . .	175
5.7	Organizational context . . . . .	175
5.8	Specifying a lower bound for an information model . . . . .	176
5.9	The concept of meta-attributes . . . . .	178
5.10	Pre-conditions and post-conditions . . . . .	178
5.11	The concept of mixinMBB . . . . .	179
5.12	Applying the characterization technique . . . . .	182
5.13	Development method—select context . . . . .	184
5.14	Development method—identify problem . . . . .	185
5.15	Development method—specify information sources . . . . .	185
5.16	Information model . . . . .	187
5.17	Subsumes relationship between information models . . . . .	187
5.18	Representation vs. notation function . . . . .	189
5.19	Assessment technique . . . . .	197
5.20	Discrepancies in trigger configuration . . . . .	198
5.21	Idle method configuration . . . . .	199
5.22	Development method—select MBB . . . . .	202
5.23	Development method—configure MBB . . . . .	203
5.24	Development method—integrate MBB . . . . .	204
5.25	Development method—analyze EA management function . . . . .	207

## List of Examples

---

5.26	Development method—dependency graph . . . . .	208
5.27	Development method—organizational interventions . . . . .	209
5.28	Development method—adapt EA management function . . . . .	210
5.29	Development method—evolve EA management function . . . . .	212
5.30	Administration method—document as pattern . . . . .	215
5.31	Administration method—derive context descriptions . . . . .	216
5.32	Administration method—derive goals and problems . . . . .	217
5.33	Administration method—detail method description . . . . .	220
5.34	Administration method—decompose patterns . . . . .	221

Today's organizations<sup>1</sup> are confronted with an ever changing economic, regulatory, and technical environment that they are forced to continuously adapt to (cf. Wagter et al. [Wa05] and Ross et al. [RWR06]). Performing the necessary and beneficial adaptations is aggravated by the intricate and highly interwoven architecture of organizations. Therein, local changes to one organizational artifact, e.g. a business process or a business application, might have unforeseen global consequences and potentially detrimental impacts on related artifacts. Based on the aforementioned impact, organizational change can be distinguished into optimization (incremental change) and transformation (fundamental change) as discussed by Aier et al. [Ai09c]. Whereas support for the former change is typically provided by functional methods of business administration, e.g. human resources, distribution, or marketing, the latter requires a holistic approach to support organizational transformation [Wi08a]. A commonly accepted instrument to support and guide such transformations is a strategic management of the *enterprise architecture* (EA) which provides a holistic model of the key elements and relationships of an enterprise and connects strategy, business, and IT. The main goal of EA management is to enhance the alignment of business and IT [LLO93]. An effectively applied EA management a) reduces local maintenance costs due to increased standardization [RV08], b) increases responsiveness via reduced project duration [RV08, Th09a], c) facilitates risk management through reduced complexity and an organization-wide view on organizational changes [RWR06, Th09a], and d) enhances strategic business outcomes by increasing effectiveness of business processes, applications, etc. through standardization [Bi98].

Research on the topic of EA management dates back to the late eighties, when Zachman published the first version of his framework for *information systems* (IS) architecture [Za87]. Since that time the number of publications in the area of EA management has been increasing steadily (see e.g. the surveys of Langenberg and Wegmann [LW04a] or our survey in [My11]).

---

<sup>1</sup>The terms organization and enterprise are used synonymously in the remainder of the thesis as generic terms covering enterprises, public organizations, companies, etc.

Driven by the demand from practice, a multitude of approaches to EA management has been developed by practitioners (cf. [La05, Ni06c, Ke07, Sc08a, Ha10]), researchers (cf. [Sc01, Fr02, We03, WF06, Ai09c]), tool vendors (cf. [ME02, Ma08]), public institutions (cf. [NA07, De09a]), and standardization bodies (cf. [Th09a]). These approaches usually distinguish the following activities of the EA management function: documenting current states, developing planned and target states of the EA, communicating and enforcing EA plans and principles, and analyzing and evaluating architectural states. Although most approaches agree on the aforementioned activities, they greatly differ regarding the scope, reach, and focus of the proposed methods and models. While some approaches are limited in scope and, for instance, restrict the ‘width’ of the EA management function to the design of the IT architecture only (cf. Keller [Ke07]), others take a more holistic perspective supplementarily taking strategic and business aspects into account [FS05]. Similarly, the chosen focus of the proposed methods differs: Typically a focus on languages and visualizations for EA descriptions (cf. [La05]) and a focus on methods and procedures for developing architectures (cf. [Th09a]) can be distinguished.

The missing common conception in the area is also reflected by the existing plurality of definitions for EA or EA management. In 2008 Schönherr [Sc08b] conducted an extensive analysis of 126 publications in the context of EA management concerning the used definitions for **EA**. While some publications used the term EA in a ‘descriptive’ way, referring only to the management subject, i.e., the architecture of the organization (cf. Johnson and Ekstedt in [JE07]), others also used the term in a ‘normative’ way, without distinguishing between the management subject and the management process (cf. Bernus et al. in [BNS03]). We stick to the former understanding of the term, presuming that architecture is an inherent property of each organization. This understanding is backed by Rehtin’s statement that “every system and organization has an architecture” [Re99, page 175] no matter, if this architecture is documented or not. Correspondingly, we define EA in accordance with the ISO Standard 42010 [In07, page 3] as

**Definition: Enterprise architecture (EA)**

EA is the fundamental conception of the organization in its environment, embodied in its elements, their relationships to each other and to its environment, and the principles guiding its design and evolution.

The EA encompasses elements and their relationships to each other. Thereby, the question whether an element or relationship should be considered a part of the EA can according to Aier et al. [Ai08b] be answered by the criteria of width, depth, and pragmatism. Figure 1.1 shows a high-level structure of the EA. The layers mirror the overall make-up of the organization ranging from **business & organization layer** via the **application & information layer** to the **infrastructure & data layer**. The cross-cutting aspects, **visions & goals**, **strategies & projects**, and **principles & standards**, represent elements which are not part of, but exert influence on any of the elements organized in the layers [Ma08]. Additional abstractions are incorporated in Figure 1.1. These abstractions encapsulate functionalities of the underlying layer or cross-cutting aspects, i.e., **business capabilities**, **business services**, **infrastructure services**, for the different layers and **questions & key performance indicators (KPIs)** for the cross-cutting aspect.

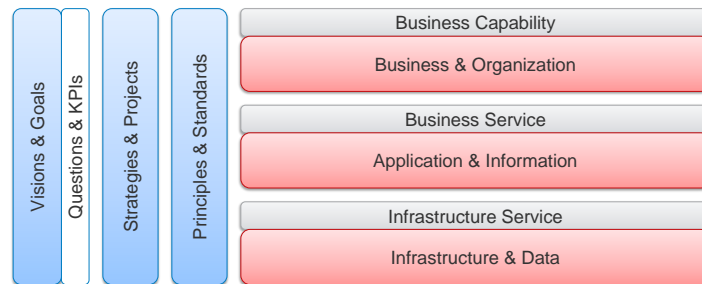


Figure 1.1.: High-level structure of the EA

Besides the different elements grouped in the layers, the relationships between the elements are important. These relationships reflect the alignment between business and IT and between organizational and IS aspects. According to Hevner et al. business IT alignment is “central to the IS discipline” [He04, page 78]. The interplay between business and IT is especially important as IT is seen as an enabler for business and organizational aspects. Therefore, Orlikowski and Barley encourage researchers to take organization studies into account during their research in IT and vice versa [OB01]. In 2008 Schönherr identified an interesting misconception during his literature analysis [Sc08b]: 60 percent (72 out of 126 publications) concerning the topic EA management address only one layer with most approaches dealing with organizational aspects, lacking the holistic perspective which would be expected in the context of EA management. The plurality regarding the scope of existing EA management approaches can be ascribed to the diversity in that area. Therefore, researchers call for developing a common terminology for EA management [Sc08b, SW09].

The diversity in the context of EA research is further investigated by Schelp and Winter [SW09]. Based on the work of Becker et al. [Be03] and the work of Hevner et al. [He04], they develop an analysis framework and discuss the contributions of seven major academic groups in the field of EA management. Schelp and Winter present two main constituents of an EA management approach—a *meta-model* or **language**, which is used to describe the EA, and a *procedure model* or **method**, which specifies how the EA is managed, i.e., documented, maintained, and planned. As a result of the survey, Schelp and Winter conclude that most of the research groups constitute local language communities with a self-developed language and method [SW09]. The appearance of isolated solutions can be ascribed to the organization-specificity of the EA management function. On the one hand the goals that an organization pursues with an EA management function differ widely (cf. [Bu06, ARW08b, Bu08b]) and on the other hand the organizational context in which the EA management function has to be integrated must be reflected by the used approach. To provide standardization for the field, a method to design EA management functions needs to be developed that enables customization of the management function based on the specific problems and organizational context settings of an organization. The **development method** for such organization-specific EA management functions must on the one hand correspond to practitioners’ demands to be applicable in their specific situations and on the other hand reflect the demands from academia to lever the standards regarding both the quality of the language community and the quality of the design research in the field. The following section further discusses the identified “business need” [He04, page 79], details the problem statement, and poses the research questions addressed by this thesis.

## 1.1. Problem statement and research questions

The mutual alignment of business and IT, which is central to the IS discipline [He04, page 78], goes beyond a mere provider role of the IT in which IT resorts to solely fulfilling business requirements. In contrast, IT should take an enabler role, proactively seeking to increase flexibility and adaptability to foster the agility of the overall organization. This two-fold role of IT well illustrates the core of business IT alignment, which could have also been described conversely from a business perspective. In consequence, mutual alignment is a goal best approached from both perspectives, a business and an IT perspective, and is hence not the focus of the management functions for business or IT management alone [OB01]. This calls for a **management function** with an embracing management subject spanning both business- and IT-related concepts, but most preferably also accounting for cross-cutting aspects, such as strategies and projects (see above the definition of EA). The latter is especially necessary as a managed evolution of the organization inevitably connects to the strategies as drivers of organizational change and the projects as its vehicles [MWF08]. According to that understanding, we provide a preliminary definition of EA management to support an intuitive understanding of the term, which will be further discussed and refined in Section 4.3.

### **Preliminary definition: EA management**

EA management is a continuous management function seeking to improve the alignment of business and IT and to guide the managed evolution of an organization. Based on a holistic perspective on the organization the EA management function is concerned with the management, i.e., the documentation, analysis, planning, and enactment, of the EA.

The above definition emphasizes the fact that EA management is a continuous effort. If information about the EA, for instance, is only gathered once and not maintained for following projects, the expenditures and investments will not amortize [Ai09a]. Typical challenges for enterprises in the development of an EA management function are a) designing a comprehensive and not fragmented management function, b) the integration of the EA management function with other enterprise-level management functions, like project portfolio management, demand management, or strategies and goals management, and c) the adaptation of the EA management function to the organizational context [RV08]. The development of a suitable EA management function addressing the aforementioned challenges and the establishment as well as integration thereof in the organizational context is a difficult task. An organization willing to introduce such management function can choose between

- a **greenfield approach**, in which the EA management function is developed from scratch,
- a **customization of existing approaches**, like The Zachmann Framework [Za09], *The Open Group Architecture Framework* (TOGAF) [Th09a], or approaches developed by practitioners as well as researchers (cf. [Wa05, RWR06, Sc08a]), and
- an **integration approach**, which supports reusing existing best practices from different sources (cf. [Bu07d, KW07, Er10]).



The advantages and disadvantages of these approaches as discussed by Aier et al. [Ai08a], Buckl et al. [Bu07d, Bu08c], and Ernst [Er10] are summarized in the following.

While the greenfield approach comprises the advantage of developing an EA management function, which reflects the specific needs of an organization, it does not consider related initiatives already existing inside or outside the organization. Lacking an actual starting point for EA management, organizations using a greenfield approach tend to collect requirements from all potential EA stakeholders resulting in an all-embracing, giant EA management information model, which precludes a low cost-benefit ratio of the initiative. Furthermore, own methods for performing EA management, i.e., for collecting and maintaining information, for developing target states of the EA, or for communicating and enacting architectural principles have to be developed such that each greenfield approach is ‘reinventing the wheel’ over and over again.

To avoid typical pitfalls during a start from scratch, existing approaches can be reused and customized to the specific needs of an organization. However, to decide which framework should be used, expert knowledge about the different approaches is required. In addition, the adaptation and customization of frameworks bears the risks of “destabilization of existing terminologies”, “missing stakeholder acceptance”, “misusing of concepts and methods”, and “too detailed models” [Ai08a, page 559]. While the last risk is closely related to the challenges of the greenfield approach, the other risks typically arise during customization. Misuse of concepts occurs, if the intended application scenarios of the concepts are not fully understood or if they do not directly correspond to the demands of the organization. Similarly, a forced terminology with an external origin typically suffers from acceptance problems, especially, if an organization-specific terminology is already in use. Furthermore, EA management frameworks like Zachman or TOGAF are usually either too abstract and, therefore, not implementable or too extensive to be used in practice. Although the importance of adaptation to organization-specific needs and contexts is mentioned in most approaches, no guidelines or methods on how to perform this task are given.

To overcome the shortcomings of typical EA management frameworks, integration approaches leverage best practice knowledge from different sources. By combining **building blocks** extracted from prevalent sources, e.g. EA management frameworks, scientific literature, or best practices from industry, they support flexibility and allow incremental extension of the resulting EA management function. Based on the demands of the associated organization building blocks representing method and/or language fragments for EA management are selected and integrated into an organization-specific EA management function, which can be further extended and adapted by the integration of additional building blocks. However, information on the integration of building blocks is scarce in literature as prevalent approaches are typically not designed to be used in combination. Experience and knowledge on how to integrate building blocks from different sources is usually only available as ‘tacit knowledge’ of EA management experts thus leading to a costly and time-consuming integration process, if no experts are available. An example of an integration approach to EA management, which especially addresses integration with other approaches is the *enterprise architecture management pattern catalog* (EAMPC) published by sebis<sup>2</sup>. The EAMPC is a collection of best practice building blocks, so-called patterns, dedicated to the area of EA management. Accounting for

---

<sup>2</sup>sebis is the abbreviation for the Chair for *software engineering for business information systems* (sebis) of the Technische Universität München ([www.matthes.in.tum.de](http://www.matthes.in.tum.de)).

the aspect of integration, we illustrate how to complement TOGAF with EA management patterns in [Bu09c]. While the collection of patterns in the EAMPC delineates best-practice knowledge, the EAMPC does not provide an overall picture of the EA management function, leaving organizations willing to introduce such a management function alone in answering questions like ‘When is my organization-specific EA management function complete?’ ‘How can I ensure that my integration is correct?’ or ‘How does my organizational context affect the integration?’. In addition, the integration of patterns, or building blocks in general from different sources yields the risk of terminological inconsistencies. Terminological issues can thus be expected due to the different origins of the building blocks and due to terminological disruptions between the terminology established in the corresponding organization and the terminologies as employed in the building blocks.

Summarizing the above considerations, establishing an EA management function often requires consulting. This may be caused by the all-embracing nature of EA management incorporating multiple areas in an organization, e.g. business functions and IT artifacts. The above discussion on the different approaches to develop organization-specific EA management functions nevertheless reveals the following characteristics influencing and framing the design problem addressed in this thesis:

- An EA management function must be suitable to address the organization-specific EA management problems by providing appropriate methods.
- An EA management function must account for the specific context of the associated organization, i.e. must ensure organizational implementability.
- An EA management function must be configurable, adaptable, and incrementally extendable to correspond to the maturity of the associated organization.
- An EA management function should ground in best-practice approaches and re-use well-established solutions.

In consequence, an EA management function should be tailored to the specific organization to be both effective and efficient, i.e., to deliver the expected objectives at low overall investments. This is even more important as an overly extensive management function can hamper and delay necessary adaptations of the architecture, and reduce the overall flexibility of the organization. From this observation, the research objective of this thesis can be derived.

**Research objective:** Develop a method for designing and re-designing organization-specific EA management functions based on best-practice knowledge documented in current EA management approaches.

Research in the area of EA management is typically conducted in close cooperation between researchers and practitioners. While this on the one hand ensures relevance of the achieved research results and fosters validation, the research agenda is typically influenced by the partnering organization’s pace and goals, and may hence be obliged to deliver research results early, which impedes the development of comprehensive theoretical underpinnings. This poses a special challenge to the researcher who has to employ a research method, which on the one hand is appropriate for the situation at hand and on the other hand ensures rigor of the achieved results. In line with Aier et al. [Ai09c] and van der Raadt and van Fliet [RV08], we opt for understanding an EA management function as a design artifact. Developing a method

for designing organization-specific EA management functions can therefore be understood as developing design theories. The development of an appropriate (research) method accounting for the aspects of rigor and relevance leads to our first research question.

**Research question 1:** How does a method for building and linking design theories in close interaction with an organization (willing to apply these theories) look like?

In line with the idea of competing theories as proposed by Pries-Heje and Baskerville in [PHB08], it seems sensible that a research method taking into account the specificities of the research topic can be developed that builds on the existing knowledge base of prevalent EA management approaches. Thereby, especially the integration approach seems to be promising as it supports re-usability of existing best practices and thus enables consideration of the organizational constraints in which a solution can be applied. Existing integration approaches however lack an overall picture of the EA management function as well as guidance and methods on the configuration and integration of solutions from different sources. Based on the integration idea and patterns as modular reusable best practices, the concept of **building blocks** seems to be promising to support on the one hand a uniform structure for describing the solutions and on the other hand flexibility to match the situation at hand by building on the basic principles of integration and customization. Existing best practices have thus to be identified, formalized, and made available in a reusable and uniform manner, i.e., as building blocks. Preparing the design and development of the building blocks, we revisit existing approaches to develop EA management functions and explore the incorporated understanding of the associated management activities, thus answering the second research question of this thesis.

**Research question 2:** Which approaches describe which activities constituting the EA management function; which organization-specific configuration techniques are supplied by prevalent approaches?

Backed by existing work on EA management functions, e.g. by Hafner and Winter [HW08] and van der Raadt and van Vliet [RV08], it seems sensible to discuss the typical activities of the EA management function separately. In addition, the interrelations and dependencies between the activities should be detailed to facilitate their integration into a holistic function. In Chapter 3 the state-of-the-art is analyzed using a systematic framework. Thereby, the similarities and differences of existing approaches are revisited using different perspectives, i.e., a model-centric, a design science, a management, a knowledge management, and a systemic perspective. As a result, a framework of activities and possible configuration aspects is presented. Complementing the theoretic discussion, a critical reflection of existing approaches from the different perspectives is performed. Under the assumption that reusable method building blocks and configuration points can be derived from a literature analysis, the third research question arises.

**Research question 3:** Which requirements does the EA management domain entail with respect to the configuration and integration of method building blocks for the development of an EA management function?

Under the premise that commonly accepted typical activities of the EA management function can be identified, a general method to develop organization-specific EA management functions by integrating and configuring existing best practices has to be devised. The **development**

**method** builds on an organized library of method building blocks, called **method base**. Method building blocks from the method base are selected, configured, and integrated by the users of the development method to design organization-specific EA management functions. Prevalent best practices have to be organized and translated into a uniform structure. A documentation method for business processes and a similar documentation method for applications, for instance, are abstracted to one method building block for documenting EA-related information. Moreover, a modeling language suitable for describing method building blocks has to be developed, which provides sufficient expressive power and formality to allow configuration, adaptation, and consistency checking of an integrated EA management function. The need to identify the relevant constructs (syntax and semantics) of a method building block as well as a graphical representation (notation) results in the subsequent research question.

***Research question 4:*** How does a conceptual model and a respective graphical notation representing the foundation for the method base that supports configuration, adaptation, and integration of method building blocks look like?

Based on the assumption that a respective conceptual model for describing method building blocks can be derived the application and administration of the building blocks requires further investigation. To guide the users in the development of an organization-specific EA management function, the **development method** defines steps, guidelines as well as techniques to apply the method base, i.e., select, configure, and integrate appropriate method building blocks based on the organization-specific problems and organizational contexts. Due to the increasing importance of the EA management topic in practice and the changing goals of EA management endeavors, the method base needs to account for newfound and reshaped EA management problems, new organizational contexts, and novel best practices. Therefore, the **administration method** provides guidelines and techniques to maintain and evolve the method base. Thereby, both methods have to prevent the users from making configuration errors resulting in inconsistencies. To support the latter, models, languages, and tools typically prove to be helpful. This leads us to the fifth research question of the thesis.

***Research question 5:*** What methods and techniques can be used to facilitate the application and administration of the method base and how does a respective tool support look like?

In addressing the fifth research question, the theoretic discussions on concepts, techniques, and methods, are complemented by a practical toolset. The toolset does not represent an EA management tool such as the tools discussed and analyzed by us in [Ma08] but can be referred to as EA management **configurator**. The configurator presented in Chapter 6 is intended to support the development method and the administration of the method base by operationalizing the theoretic artifacts in such a way that they can beneficially be employed in developing organization-specific EA management functions.

## 1.2. Contributions of this thesis

The objective of this thesis is to contribute to the knowledge base regarding the development and establishment of organization-specific EA management functions. Therefore, a building block-based development method (**building blocks for EA management solu-**

tions—**BEAMS**) is proposed. The development method provides guidance to design and adapt coherent and consistent organization-specific EA management functions based on a method base that interlinks practice-proven method descriptions from different origins. The development method ensures consistency and provides analysis techniques to investigate organizational implementability of the resulting method description. The main contributions of the thesis are

- a **modeling language** that lays the basis for describing methods in the context of EA management and corresponding building blocks,
- a **method base**, i.e., an organized collection of method building blocks, that can be used to develop an organization-specific EA management function,
- an **administration method** for creating and maintaining the organized collection, e.g. for adding new method building blocks,
- a **building block-based development method** guiding the use of rules and techniques during the design and re-design of an organization-specific EA management function, and
- a **prototypic toolset**, which employs the modeling language and implements techniques for supporting the users during the steps of the development method to design an organization-specific EA management function.

Enabling a unified description of method building blocks the **modeling language** specifies the syntax, semantics, and notation for describing method building blocks. The syntax defines concepts as triggers, tasks, participants, consequences, or forces to describe methods related to EA management activities. Furthermore, the syntax provides concepts to specify pre-conditions and post-conditions for method building blocks such that relations to other building blocks and rules for their integration into a coherent management function can be defined. Representing the basis for configuration during the development method the **variable concept** is introduced. A variable represents a ‘placeholder’ that must be configured in the development method to reflect the specific requirements of the associated organization. The modeling language further provides a glossary describing the semantics of the introduced concepts. Complementing the triple of constituents of a modeling language, we provide a notation or *concrete syntax* for representing the concepts of the modeling language graphically that is based on the *business process model and notation* (see Section 5.1.2).

The **method base** is an organized collection that brings together best practice design prescriptions for EA management functions gathered from existing de-facto standards, scientific literature, and practitioners experience. The design prescriptions from different sources are documented as method building blocks using the syntax, semantics, and notation provided by the modeling language and linked in the method base. The method base establishes means to select the method building blocks best suited for a specific organization. The method base is organized according to the main activities of an EA management function—develop & describe, communicate & enact, analyze & evaluate to provide dedicated integration points of the method building blocks and to facilitate their selection (see Section 4.3).

The **administration method** describes steps to be taken and techniques to be employed in creating and maintaining the organized collection of method building blocks. The main

steps relate to the integration of newfound or changed elements, e.g. method building blocks, organizational contexts etc. During integration, special attention has to be paid to aspects of maintaining the interlinked method building blocks contained in the method base. The method prescribes how explicit relationships, i.e., ‘hard-coded’ links, between method building blocks can be avoided to keep the maintenance efforts low. Instead, the method provides techniques that use implicit relationships to link method building blocks via the meta-attribute concept. Further, the administration method has to account for terminological issues. Considering the various sources from which new method building blocks might emerge, glossaries for defining participants, triggers, and organizational contexts are provided.

The **building block-based development method** establishes a framing process and different configuration techniques to support the users in the development of an organization-specific EA management function. It specifies a step-wise procedure and guidelines ‘when’ to use ‘which’ technique. The method further states which information about the associated organization has to be gathered and how this information needs to be organized to provide input for the selection of method building blocks from the method base and to the configuration thereof, e.g. to determine possible design options. The configuration techniques ensure that a configuration is sound with respect to the information processed, i.e., with respect to pre-conditions and post-conditions, and ‘vivid’ with respect to the triggers and participants. Using the concept of variables the problem-independent method building blocks can be configured to specific EA management problems and the general participants can be replaced by organization-specific roles. Techniques for analyzing the organizational implementability are provided by the method that make dependencies between information consumers, i.e., stakeholders, and information suppliers, i.e., actors of EA management-related tasks transparent and provide respective enactment mechanisms. The development method depicts possible paths of evolution and maturation regarding the EA management function, as EA management represents a long-term endeavor that has to deal with varying problems to be addressed and changing organizational contexts to be accounted for. Therefore, the development method employs the configuration techniques that call on the implicit relationships between the method building blocks.

Finally, the **prototypic toolset** implementing the modeling language, the techniques, and the methods demonstrates the feasibility of the approach. Thereby, the prototypic toolset supports both the maintenance and evolution of the method base (administration method) and the application of the method base (development method). Relating it to classical EA management tools as analyzed by us in [Ma08], the prototypic toolset can be considered a **configurator** supporting the design of EA management functions that can then be employed in ‘classical’ EA management tools. In this vein, the prototypic tool is used to facilitate the evaluation of the contributions of this thesis in practical settings.

While the first four results present mainly theoretical contributions to the knowledge base, the last one, the prototypic toolset, follows a technical paradigm. This tool is further used in another application area, targeting languages for EA models [BMS10g] embedding this thesis into a larger research project, targeting the development of organization-specific methods and languages of the EA management function [BMS10d].

### 1.3. Outline of this thesis

This thesis is structured in eight chapters. The main body of the thesis is organized in three parts, **problem diagnosis**, **nexus instantiation**, and **application & evaluation**. These parts are illustrated in Figure 1.2 and build an organization structure for the single chapters of this thesis. The first chapter motivates the problem statement of this thesis, summarizes the core contributions, and outlines the course of the thesis as well as the writing conventions.

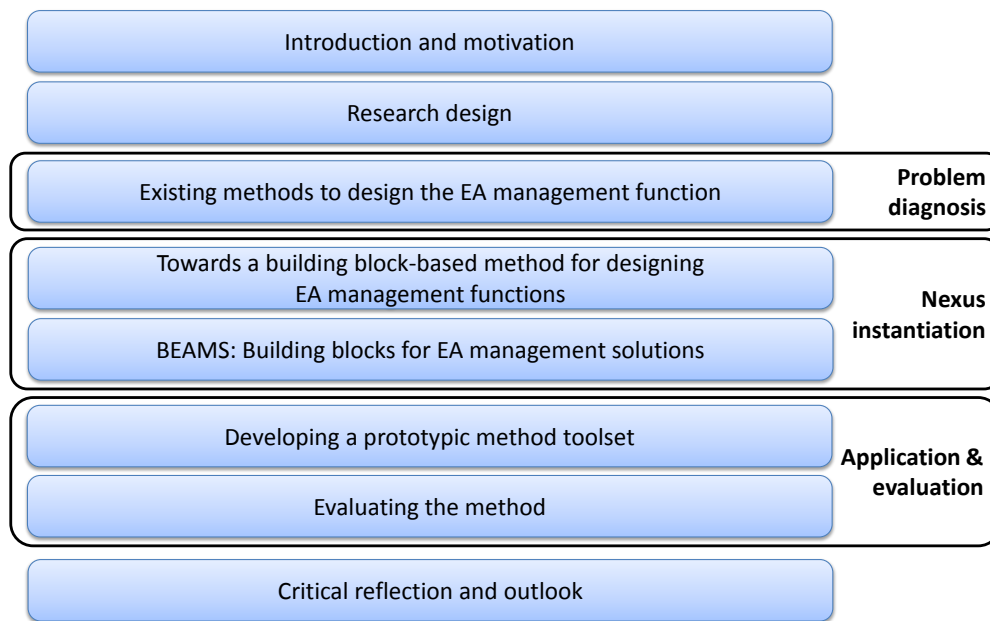


Figure 1.2.: Outline of the thesis

In Chapter 2—**research design**—the theoretical foundation for the research presented in this thesis is established. Based on the epistemological framework of Becker and Niehaves in [BN07], the epistemological, ontological, and linguistic positions that guide the course of research are explained and discussed. Following the exposition of our fundamental positions, the research perspective taken in this thesis is presented by delineating the expected research outcomes. Complementing the research design, the research method is made explicit. Our research method synthesizes the methodical framework of Venable [Ve06b], the five step method to link design theories of Pries-Heje and Baskerville [PHB08], and our approach to pattern-based theory building. The research method is further adapted based on the specificities of the research context, the epistemological perspective taken, and the intended research outcomes.

In Chapter 3—**analyzing the state-of-the-art in designing EA management functions**—prominent approaches for the design of EA management functions are revisited and assessed with a focal point on the proposed methods. Following the guidelines for literature reviews as proposed by Webster and Watson in [WW02] and a classification according to Fettke’s framework in [Fe06], 15 prominent approaches to design an EA management function are revisited. Thereto, we use a classification framework derived from a systemic and a management perspective on EA management. The literature review in particular

emphasizes on the provided methods and techniques to configure EA management functions to organizational specificities.

In Chapter 4—**towards a building block-based method to design EA management functions**—the core idea of this thesis is outlined. Thereto, we elicit quality criteria of an organization-specific EA management function. Further requirements on the meta-level, i.e., regarding the development method and the method base, are elicited from a utilitarian perspective. We investigate kernel theories for the solution domain that contribute models and techniques reusable in the context of designing organization-specific EA management functions. In response to the identified characteristics, we propose an approach called “building blocks for EA management solutions (BEAMS)” that presents the concept of method building blocks, method base, development method, and the classification framework spanned by the main activities of an EA management function. Thereby, we identify the method base with a *design theory nexus instantiation* as presented by Pries-Heje and Baskerville [PHB08] and the method building blocks with design theories.

In Chapter 5—**BEAMS: building blocks for EA management solutions**—we describe the development method that uses a method base of practice-proven building blocks to design an organization-specific EA management function. Focusing on method aspects of EA management functions, the concept of method building blocks is detailed. We introduce a language for describing method building blocks that supports selection, configuration, integration, and adaptation of method building blocks. Complementing the foundation for our development method to design organization-specific EA management functions, we discuss techniques to ensure consistency of the integrated method description and analyze the organizational implementability. The techniques are used during the development method which describes the steps how to apply the method base to design an organization-specific EA management function. The theoretic exposition of the method is complemented by an example. Finally, we describe the administration method to maintain and evolve the method base.

In Chapter 6—**developing a prototypic toolset: The BEAMS configurator**—we elicit use cases and discuss the design of tool-based support for our building block-based development method to design EA management functions. Thereto, we present use case descriptions from the perspective of a ‘user’ of the development method and associated techniques as well as the perspective of an ‘administrator’ of the method base. Based on the elicited requirements we delineate the prototypic design of a toolset supporting BEAMS and discuss issues related to the functionalities and implementation of the toolset.

In Chapter 7—**evaluating and applying BEAMS**—we evaluate our research outcome in a threefold way. Firstly, we present a theoretic evaluation of BEAMS. Thereto, we subject the theoretical underpinnings of BEAMS to an argumentative evaluation process by revisiting the requirements and general guidelines as discussed before. Secondly, we evaluate BEAMS against the state-of-the-art. Using a fictitious case derived from prevalent literature, we show that a skilled enterprise architect designs a similar EA management function as provided in the example case by using the development method of this thesis. Thirdly, we show utility of BEAMS by discussing observations made during the application of the research method in practice.

In Chapter 8—**critical reflection and outlook**—the contributions of the thesis are summarized and critically reflected. Starting with a review on the research questions, we summarize



the contributions and findings of this thesis. We further identify possible limitations of the method taken, reflect on our contributions, and derive open questions and directions of future research furthering the thesis contribution.

## 1.4. Writing conventions

Doing research in the area of EA management, the communication challenge becomes obvious in every speech, presentation, discussion, and workshop. This challenge arises from the different stakeholders and perspectives involved as well as from the multitude of terminologies used by them. Therefore, this thesis improves the overall readability and thus foster understandability and discussion of the contributions by adhering to defined writing conventions that are made explicit in the following.

To facilitate orientation for readers of this thesis, each main chapter starts with a short introduction to the motivation and objective of the according chapter and concludes with a summary of the achieved results. Furthermore, a table of contents, list of figures, and tables as well as an index and a nomenclature enhance the usability of this thesis.

New and important terms are highlighted in **this way** at their first occurrence. In the subsequent text flow the terms are not specifically highlighted anymore to allow for an uninterrupted text flow. Exceptions from that rule are only made when the usage of the term without highlighting would cause ambiguities or would result in hardly readable sentences. Important terms of this thesis are introduced using a definition environment, which is accentuated as follows:

**Definition: Definition**

A definition explicates a term and provides a common understanding of it. The term is used throughout the remainder of the thesis and contributes to the knowledge base of the research field.

Additionally, all defined terms are included in the index to ease the retrieval of the corresponding definition. Sometimes it is necessary to discuss a term prior to presenting a formal and comprehensive definition. Possible reasons are firstly that the reader might not yet have enough information, as not all terms used in the definition have yet been introduced and discussed. Secondly, the stream of consciousness and explanation might call for an intuitive but less formal understanding of the term for which later a comprehensive formal definition is provided. Preliminary definitions are presented similar to formal definitions as illustrated subsequently.

**Preliminary definition: Preliminary definition**

A preliminary definition is a statement describing the respective term in an intuitively understandable way.

Various approaches using different terminologies to describe the constituents of the EA management function exist. Whenever work from other research groups or practitioners is cited,

## 1. Introduction and Motivation

---

the original terminology as employed in the publications is used. To clarify the origin of the used terms and foster a common understanding, terms originating from foreign approaches are highlighted *in italics*.

Quotations and references to existing work and primary sources are an essential part of every academic work. Therefore, quotations are highlighted with respect to the length of the quotation in two different ways. A short quotation, e.g. a single term is highlighted using “quotation marks”, while longer quotes are organized in block quotes as exemplified below.

This is an example of a longer quote, which may consist of several sentences. A longer quote is typically organized in block quote.

The source of the quotation is given by a nearby citation, which may be complemented by the authors’ names. Whenever more than one source is given, the citations are arranged first chronologically and then alphabetically. If a literal citation is used, the source is supplemented by a page reference. In quotations and sections which are dedicated to paraphrasing the contents of a related work, the original terminology as used in that work is employed. Additions in quotations are enclosed in [brackets]. If the usage of the original terminology might cause ambiguities with the thesis’ terminology, footnotes are used to provide additional information on the respective terms. Own publications are referenced in first person ‘we’ and ‘our’. Exceptions from that rule are made, if the author group involves external parties other than members of the sebis Chair, as these may employ a different terminology as the one used throughout this paper.

To illustrate complex concepts, context, and information, examples are used throughout the thesis. Readability of the thesis and the text flow is ensured by organizing examples in a box as follows.



**Example 1.1: Expository example.** This is an example, i.e., something serving to explain or illustrate something else.



As the thesis incorporates parts dedicated to conceptual modeling, a distinction between terms referring to a modeling concept and the real object has to be established. Therefore, a term referring to the modeling concept is highlighted in CAPITALS to avoid confusion. At some points in the thesis, colloquial terms, e.g. ‘gut feel’, are used to avoid complex circumscriptions and are emphasized by single quotation marks.

Whenever job titles or other references to persons are given in the text, typically the plural or masculine form is used, which includes the feminine form (and vice versa) with no disrespect intended.

Driven by the multidisciplinary nature of IS research and the influence of many related disciplines including business administration, computer science, sociology, and psychology, a plurality of different methods, paradigms, and approaches has been developed in the IS discipline [BW96, VRG02, BK04, SSW04]. Since its establishment in the 1970s, the discipline of IS research went through a series of crises in which its identity and legitimacy was discussed in manifold ways (cf. Hirschheim and Klein [HK03] and Lyytinen and King [LK04]). Prominent topics of these crises were discussions concerning the relevance of IS research in practice (cf. Bernbasat and Zmud [BZ99]), research methods appropriate for the field of investigation (cf. Atkinson and Kühne [AK99] or Lee [Le99]), a shared body of knowledge (cf. Hirschheim and Klein [HK03]), common goals and objects of research (cf. Bernbasat and Zmud [BZ99]), as well as the theoretic core (cf. Lyytinen and King [LK04]). In contrast to these discussions stands the rising importance of IT and IS for enterprises. Whereas the international IS research community<sup>1</sup> focuses on behavioral research (cf. Vessey et al. [VRG02] and Coplien and Harrison [CH04a]), German IS is strongly influenced by design-oriented research [La06]. This design-orientation is reflected by a long tradition in modeling, e.g. conceptual modeling (cf. Kühn [Kü04] and Guizzardi [Gu05]), enterprise modeling (cf. Scheer [Sc01] and Frank [Fr02]), or reference modeling (cf. vom Brocke and Buddendick [BB04] and Becker et al. [BDK07]) and in validating research results based on prototypical implementations (cf. Heinrich [He05] and Lange [La06]).

According to Niehaves, the aforementioned divergence of international and German IS research alongside with the growing internationalization, raises some opportunities as well as threats for German IS research (cf. [Ni06c, page 13]). With the influence of IS research, methodological and epistemological reflectivity could gain momentum in German IS research, which is

---

<sup>1</sup>International IS research within this work is understood as the complement to the German IS community of *Wirtschaftsinformatik*. The international research is strongly influenced by Anglo-American research traditions.

often disregarded or only mentioned implicit in current publications (cf. Heinrich’s analysis in [He05] and the continuing study of Becker et al. in [Be09a]). In return, the epistemological and methodological pluralism existing in the field of German IS research could open rigid structures in international IS research. If German IS, in contrast, passively follows methodological discussions and unquestionedly adopts the quantitative behavioral research paradigm, it, according to Niehaves, might fail to profile on the basis of its competencies and potentially loose positive differentiation criteria [Ni06c].

The nature of IS as ‘in between’ discipline to its neighboring areas of management science and computer science, leads to repeating discussions about ‘what in fact is fundamental research in business and IS engineering’. In a recent edition of the *Business & Information Systems Engineering* journal, Winter [Wi09a] took up this discussion by presenting opinions from researchers from the German and Anglo-American community. Therein, the discussants emphasized the importance of multidisciplinary for German IS research and advocate for an explicit pluralism of theories including theories from reference disciplines like organizational and system theory [Wi09a]. To ensure that this rich tapestry of different methodological approaches is used correctly, coherent, and beneficial for the body of knowledge in the IS discipline, researchers, according to Becker in [Be03, page 3] and Niehaves in [Ni06b, page 11], should explicate the **research design** of their work to allow others to comprehend and evaluate the achieved results. According to Becker et al. choosing an appropriate *research method* is the central question of research design [Be03], which must be adequate to on the one hand achieve the desired *research outcomes* and on the other hand must be selected in accordance with the *epistemological*, *ontological*, and *linguistic* assumptions made by the researcher [Be03]. Figure 2.1 illustrates the aforementioned constituents of a research design according to Becker et al. [Be03, page 5].

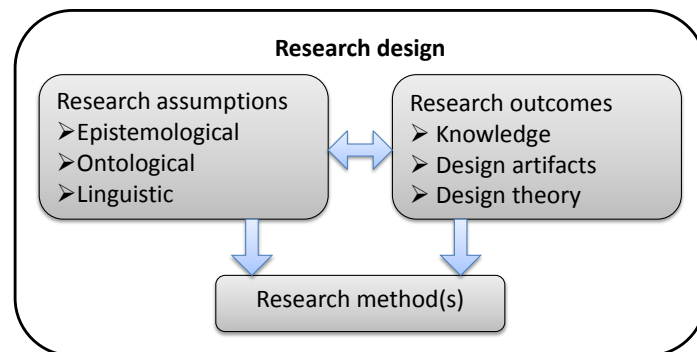


Figure 2.1.: Research design according to Becker et al. [Be03, page 5]

This chapter is organized alongside the constituents of a research design as shown in Figure 2.1. Subsequent Section 2.1 uses the epistemological framework developed by Becker and Niehaves in [BN07] to present and discuss the epistemological, ontological, and linguistic research assumptions underlying this thesis. We advocate for understanding the ‘design and establishment’ of an organization-specific EA management function as a *design process* with the organization-specific EA management function as the corresponding *design artifact*. Although the design process for the EA management function is in practice mostly carried out in an ad hoc manner and therefore not made explicit, other authors back our understanding of an

organization-specific EA management function as a design artifact (cf. Aier et al. in [Ai08b], Ernst in [Er10], or van der Raadt and van Vliet in [RV08]).

In Section 2.2 the research outcomes of this thesis are presented. Based on the aforementioned assumptions and the research context of designing EA management functions, implications targeting the research outcomes are discussed. Typically, research projects in the area of EA management are conducted in close cooperation with industry partners, thus yielding the intrinsic conflict for researchers between responding to practitioner demands and the methodological rigors required for academic contributions [Ga07]. In this thesis, the conflict is solved by research outcomes of trifold nature. Firstly, existing knowledge on designing EA management functions published in literature and best practice knowledge from practice is consolidated. Secondly, we develop methodical support for conducting the design process via precise and specific prescriptions guiding users that seek to develop and establish an organization-specific EA management function. Thirdly, we provide an IS design artifact, i.e., a tool, supporting the use of the aforementioned outcomes.

To achieve these outcomes, the notion of *design theory* initially introduced by Walls et al. [WWES92] and further refined by Gregor and Jones [GJ07] is adapted as appropriate means to formulate design prescriptions for the design of EA management functions. In response to the challenges of industry-funded research, we develop a pattern-based theory building process which enables reuse of best practice knowledge and early delivery of research results [BMS10d, BMS10k]. The plethora of application areas of EA management calls for the application of multiple design theories. In other words, the design theories represent alternative solutions for the design of an enterprise-specific EA management function, whose suitability depends on the environment and the goals pursued by the initiative. The concept of the *design theory nexus instantiation* as introduced by Pries-Heje and Baskerville in [PHB08] is used, which facilitates the selection of design theories best suited for a specific application context and problem. The nexus instantiation represents the basis of the methodical support provided for the development method of an EA management function as part of the research outcome and follows the *method-language dichotomy* of EA management as discussed in the work of Schelp and Winter in [SW09] and Aier and Schelp in [AS09].

Complementing the research design according to Becker et al. the research method is presented in Section 2.3. The research method is based on the work of Venable [Ve06b], who outlined different phases for a *general design science research method*, our method of pattern-based theory building, and the stepwise procedure for constructing a design theory nexus instantiation as presented by Pries-Heje and Baskerville [PHB08]. In putting strong emphasis on the aspect of evaluation and validation, the synthesized method incorporates the evaluation and validation techniques as devised by Verschuren and Hartog [VH05].

## 2.1. Research assumptions

As mentioned before, IS is a multidisciplinary and multinational area of research spanning a heterogeneous environment. Taking the internationalization of IS research into account, the explication of epistemological assumptions certain research results are based on becomes more and more apparent. If not made explicit, collaborative work and evaluation of research with respect to the appropriateness of the research method choice and application is aggra-

vated [Ni06b, page 2]. In addition, problems may arise, if the implicit assumptions made are not shared by international researchers, reviewers, or readers of the research results. Several endeavors to systematically analyze research paradigms have been made also taking into account the underlying epistemological, ontological, and linguistic assumptions (cf. Burrell and Morgan [BM79], Hirscheim et al. [HKL95], and Becker and Niehaves [BN07]). The epistemological framework initially presented by Becker et al. in [Be03] and further developed by Becker and Niehaves in [BN07] is the most comprehensive framework, which showed its usability in different national in international publications (cf. Recker [Re05a, Re05b]). We use the framework subsequently to make the epistemological, ontological, and linguistic positions taken in this thesis explicit. The framework consists of five questions: i) What is the object of cognition (ontology), ii) What is the relationship between cognition and the object of cognition (subject-object-relationship), iii) What is true cognition (concept of truth), iv) By what means can cognition be achieved (methodology), and v) Where does cognition originate (origin of cognition). The framework does not only provide assistance via making explicit the different assumptions underlying a research and thus facilitating assessment of research rigor, it further identifies dependencies between certain assumptions as possible combinations of ontological and epistemological positions [Be03, page 8]. Subsequently, the assumptions underlying this thesis are described alongside the framework developed by Becker and Niehaves [BN07]. Figure 2.2 provides an overview on the framework according to Becker et al. in [Be03, page 6] and Becker and Niehaves in [BN07, page 202].

### 2.1.1. Ontological assumption

The above introduced framework is grounded on the ontological assumptions made by the researcher. Ontology is the philosophical study of the nature of being, existence, or reality in general, i.e., explores the questions “what is” and “how it is” [Fo96, page 366]. A researcher thereby has to answer the question, if an objective world exists, independent from pure imagination of the subject. In the context of IS research this question can be formulated as “what is the object of our research” (cf. [Mo03]). Possible positions of researchers answering the aforementioned question range from ontological realism to ontological idealism.

**Realism:** Researchers assuming that a real world exists, independent from cognition, thought, and speech processes, take a position of ontological realism. An extreme form of realism is *materialism*, which assumes that all entities and processes are composed of matter, material forces, or physical processes [St98].

**Idealism:** Researchers adhering to ontological idealism, perceive reality as a construct, which is dependent on human consciousness. A prominent exponent of ontological idealism was Plato, who posited the primacy of spirit, mind, and language over matter, thereby negating the existence of a real world independent from human thinking [Ja01].

**Kantianism:** Trying to overcome the differences between realism and idealism, Kant emphasizes the existence of ‘things in themselves’, so-called *noumena* and ‘the appearing of those things’ to an observer, referred to as *phenomena*. Researchers adhering to kantianism believe that both noumena and phenomena exist and that our mind forces the world we perceive [Ka08]. Thus, knowledge acquirable by an observer is restricted to phenomena, while in contrast noumena are unknowable.

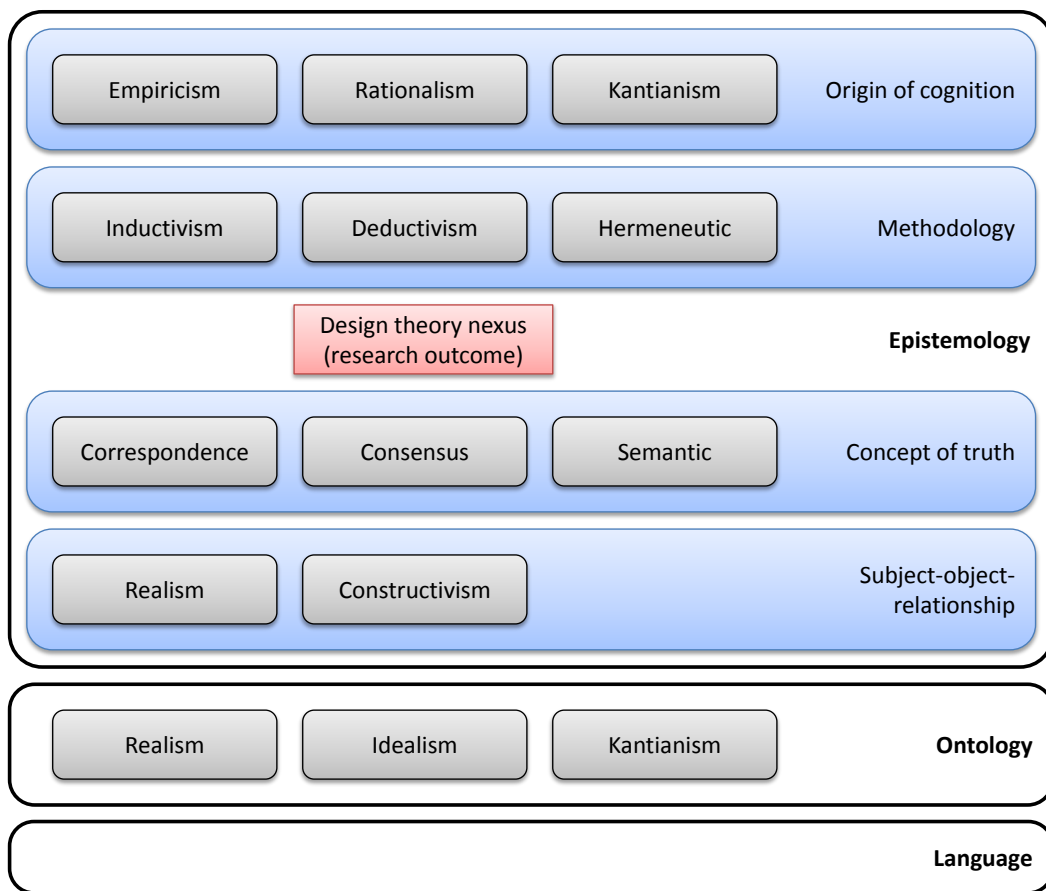


Figure 2.2.: Epistemological, ontological, and linguistic framework adapted from Becker et al. [Be03] and Becker and Niehaves [BN07]

### Position taken in this thesis

In accordance with the latter discussion, we take an ontological perspective of *kantianism* in this thesis. Taking the high-level perspective on an enterprise as presented in Figure 1.1 into account, a distinction in concepts representing noumena and phenomena can easily be made and justifies this position. To exemplify our considerations, the business & organizational layer can be used. Whereas some constituents of that layer, e.g. locations or buildings, can be assumed to be things themselves independent from human consciousness, other constituents, e.g. business processes or organizational units, represent logic constructs that depend on human consciousness. Similar considerations can be made for the other layers and cross-cutting aspects.

#### 2.1.2. Epistemological position

Debates on *philosophy of science* concerned with assumptions, foundations, methods, and implications of science and **epistemology** as a part of philosophy of science have a long history in IS research (cf. [WWES92, Sc99, Fr03]). Philosophy of science according to Niehaves [Ni06b]

seeks to answer the question how scientists should conduct research to achieve certain research aims. Epistemology as a part of philosophy of science can be understood as the science of analyzing the way human beings comprehend knowledge about what is perceived to exist (cf. [BM79]). In other words, epistemology aims at investigating how humans can achieve ‘true’ knowledge. The epistemological position we take in this thesis is made explicit subsequently, alongside the four questions as proposed by Becker et al. in [Be03] and Becker and Niehaves in [BN07].

### 2.1.2.1. What is the relationship between cognition and the object of cognition—Subject-object-relationship

This question investigates the relationship between ‘what perceives’ and ‘what is perceived’. In other words, the question is “whether entities beyond human thoughts and speech can, at least in principle, be recognized as objective” [BN07, page 203]. Two main answers to this question exist:

**Realism:** Researchers following the assumption of epistemological realism claim that objective cognition of an objective reality is possible, if suitable measures to eliminate subject-dependent distortions are provided (cf. Loose in [Lo01]). As epistemological realism assumes that objective cognition of an independent reality is possible, it relies on the position of *ontological realism*<sup>2</sup>.

**Constructivism:** Researchers following the assumption of epistemological realism believe that cognition is ‘constructed’ and therefore subjective instead of being discovered. In other words, the relationship between cognition and the object of cognition is determined by the subject [Lo87]. Thus, the subject can decide on the actual quality of the relationship and hence on the quality of cognition. Becker and Niehaves further identified different forms of constructivism, with a continuum ranging from *interpretivist constructivism* to *radical constructivism*. While researchers following the former school assume that cognition is the interpretation of an objective reality by a subject (cf. Burrell and Morgan [BM79]), the researchers following the latter school adhere to the idea that cognition is always ‘private’ because no such thing as an objective reality exists (cf. Glasersfeld [Gl89]).

### Position taken in this thesis

As already mentioned above, ontological and epistemological assumptions are interdependent (cf. the combination possibilities presented by Becker et al. in [Be03, page 8]). Our ontological assumption of kantianism excludes a position of epistemological realism. Therefore, we advocate for a position of epistemological constructivism, more precisely for an interpretivist constructivism. This choice is only indirectly determined by the existing dependency between ontology and epistemology, but in fact derived from our object of investigation—the management of EAs. The architecture of an enterprise is perceived by a multitude of different stakeholders with different backgrounds and interests in the systems (cf. [Ai08a, Bu09b, Th09a]).

---

<sup>2</sup>The combination of ontological and epistemological realism is often referred to as *positivism* in IS research (cf. [CH04b]).



The perception of the EA strongly depends on these stakeholders, which subjectively interpret the ‘reality’ and construct their ‘mental model’ of the EA.

#### 2.1.2.2. What is true cognition—Concept of truth

The meaning of the term *truth* is a central topic in epistemology. Truth may have a variety of different meanings, e.g. referring to being the case or being consistent with a fact or reality. Therefore, the assumptions made regarding the concept of truth are highly important for analyzing and validating research results. The concept of truth refers to the question how humans can achieve ‘true’ cognition. In the notion of Becker and Niehaves the question can be rephrased more intuitively to “the extent to which ‘correct’ knowledge can really be obtained and how the ‘correctness’ of knowledge has to be verified”. The continuum of possible answers to that question is given subsequently:

**Correspondence:** The correspondence theory of truth builds on the understanding that truth is a result of correspondence between two *relata*, of which the first is referred to as *statement* and the second is a *fact* in terms of an ontological realism. In evaluating the correspondence of the two *relata*, the statement can be classified as either true or false.

**Consensus:** The consensus theory of truth claims that true is whatever is agreed upon. In its elementary form, the consensus results from everyone. A more moderate form of the consensus theory of truth emphasizes on the subject-object-relationship and accordingly enables the limitation of the range of truth to a particular group (the addressees). Thus, a statement is true if it is acceptable for this group.

**Semantic:** The semantic theory of truth builds on linguistics, the precision of argumentation, and the compact instrument of modern semantics. Tarski who mainly influenced this theory, provided the differentiation of a language and its meta-language as the condition for truth, to eliminate self-referential statements and resulting logical paradoxa [Ta44].

#### Position taken in this thesis

Taking the subject of this thesis into account, the consensus theory of truth emerges as first choice. Especially the latter understanding, which limits the group of addressees, proves itself useful in the context of EA management. The stakeholders determine the group of people, which have an interest in the enterprise (cf. [In07, page 3]), and therefore are the addressees of the developed approach. In the terms of Kamlah and Lorenzen they represent a *language community*, a community which shares a common language, e.g. in terms of a conceptual model of the enterprise, thereby making mutual understanding possible or easier [KL67]. Concepts of such a language in the context of EA management are, for example, strategy, business process, organizational unit, etc, which are *agreed upon* by the stakeholders. If more technical aspects of an EA, e.g. servers, infrastructure devices but also technical measures like throughput and latency, are considered, truth is not solely be a matter of consensus but also follows the correspondence theory of truth.

### 2.1.2.3. By what means can cognition be achieved—Methodology

This question is concerned with the way “how humans perceive” [BN07, page 205] or in other words investigates the modes of how knowledge can be acquired. In contrast to research methods, like explanatory research, constructive research, empirical research, or qualitative vs. quantitative research, which describe procedures, the methodological aspects of this question refer to the methodology of perception. Different methodological positions exist.

**Inductivism:** Transferring knowledge gained from (observed or empirical) individual cases to a universal phrase or law by generalization is the way how cognition is obtained inductively [Ze08, page 33]. Thus, inductivism is typically used posteriori, especially in natural sciences in explanatory research contexts.

**Deductivism:** Deriving a statement from other statements with the help of logical conclusion, is the way how knowledge can be acquired deductively [Ze08, pages 33–35]. In contrast to inductivism, deductivism is the derivation of the individual from the universal.

**Hermeneutics:** Gaining deeper knowledge about the ‘nature’ of a text or fact requires a step-wise investigation, which is conducted in so-called *hermeneutic cycles*, and results after a number of iterations in profound understanding [Ze08, page 26]. Originating from the background of comprehending texts, hermeneutic means that reading or interpreting a text or making an observation is influenced by the background of our previous understanding of the entire and in the same way means that while reading the text or observing a fact we gain new knowledge with vice versa leads to a better understanding of the entire.

### Position taken in this thesis

As repeatedly discussed in the motivating Section 1, the design of an EA management function is highly dependent on the specific enterprise and its culture and context respectively. Furthermore, enterprises form complex systems, which can not only be regarded as systems by themselves but supplementary consist of a number of other systems and act in an environment, which can itself be regarded as a system [Ha05]. Hence the resulting complexity advises the utilization of a hermeneutic methodology investigating the area in an iterative manner. Both, deductive and inductive methods can be applied during the different hermeneutic cycles to achieve intended research outcomes. Considering, for example, the identification of typical activities of an EA management function, existing EA management functions as proposed in literature or established in enterprises can be investigated to derive generalized constituents of the EA management function. Contrastingly, the investigation of potential configuration possibilities may call for an deductive methodology, which derives potential influences on the EA management function from related disciplines like business administration.

### 2.1.2.4. Where does cognition originate—Origin of cognition

Relating to the question where our knowledge derives from, this question investigates the different sources of cognition capability. While experiences of people and sensation can on the one hand be regarded as origin of cognition, intellect can also be regarded as a source of

cognition, as an object can be intellectually conceived to result in cognition. The aforementioned different positions are further extended by the position of kantianism, which provides an intermediate position [BN07].

**Empiricism:** The school of empiricism regards experiences as the main source of cognition. Empirical knowledge is often connected to the natural sciences and experiments and therefore also referred to as *posteriori knowledge* [ACB89].

**Rationalism:** Considering intellect as main source of cognition, the school of rationalism believes that objects can become a matter of cognition through the conceptual efforts of a subject [BN07, page 205]. This non-experienced-based knowledge is also referred to as *a priori knowledge*.

**Kantianism:** Allowing both, experiences as well as intellect as valuable sources of knowledge, the school of kantianism takes a conciliating position. Thereby, Kant emphasizes the fact that none of the aforementioned positions should be preferred as objects can best be experienced and categories are best to be conceptually designed by the intellect [Ka08].

### Position taken in this thesis

Again referring to the subject of investigation—EA management—in general and the objective of this thesis—a method to design organization-specific EA management functions—in special, a kantianism position is most promising, allowing both experience and intellect as sources of cognition. On the one hand experiences have proven to be supportive in the context of EA management, see e.g. pattern-based approaches for EA management (cf. [Er10]) or empirical observations of success factors (cf. [AS09]), on the other hand intellectual imaginations, e.g. the development of an EA information modeling language (cf. [Fr02]), provide useful input. Consequently, both origins of knowledge should be taken into account in this thesis.

### 2.1.3. Linguistic position

A central task of researchers or academics is to publicize their research findings to facilitate discussion and to acquire feedback from others. Thus, research can be regarded as exchanging linguistic artifacts (papers, presentations, speech, etc.). The epistemological and ontological assumptions discussed before at least implicitly influence the linguistic position of a researcher. In [Be03], Becker et al. hence propose to make the linguistic position explicit and present a framework, which distinguishes the three functions of language, namely the *cognitive*, *expressive*, and *communicative function* detailed subsequently.

**Cognitive function:** The cognitive function is concerned with answering the question regarding the task of language during intellectual effort of a person. In other words, it introspects the role of language in mental processes. Thus, the cognitive function can be regarded as purely intrasubjective, i.e., it is concerned with the internal cognition of a person. While it is commonly accepted that thinking is done using language or images, we abstain from more sophisticated discussions as they are most likely fruitless due to the limited observation possibilities.

**Expressive function:** Answering the question how language artifacts obtain semantics, the expressive function of a language focuses on the relationship between the subject and the language artifact. Thereby, two main positions can be distinguished. Under the assumption of an epistemological position of realism explicated language artifacts inhere an objective, unambiguous semantics. Contrastingly, a epistemological position of constructivism or kantianism implies that language artifacts are interpreted by the subject and therefore have a subjective semantics.

**Communicative function:** Taking the external intrasubjective aspect of language into account, the communicative function of language is concerned with the question how language artifacts can be used to exchange information between the sender of the information, i.e., the person expressing their thoughts by using the language, and the receiver. Thereby, it is of special interest if language artifacts can ensure that the receiver comprehends the meaning intended by the sender. Especially if language artifacts are assumed to be interpreted subjectively, investigation of the communicative function is of interest.

### Position taken in this thesis

Considering EA management, linguistic aspects are of vital importance both in the application area as well as regarding research conducted in the field. In the application area, communication is accounted for as one major task and central challenge (cf. Schekkermann in [Sc06d, page 88] and Hafner and Winter in [HW05, page 8]). In addition, language takes a prominent role in EA management research, which according to Schelp and Winter in [SW09] as well as Aier and Schelp in [AS09] typically has two main constituents—a **language** for EA descriptions and a **method** for the EA management function. Based on these discussions, it is obvious that especially the positions regarding interperson aspects, i.e., the expressive function and the communicative function, should be made explicit. Following the above substantiated position of epistemological kantianism, we assume that language artifacts have a subjective semantics. This assumption is further backed by the findings of Schönherr in [Sc08b], who conducted an extensive literature analysis investigating different definitions of the term EA management. In this context, Schelp and Winter apply the term “language community” [SW08], which refers to a group of individuals that agree on a set of terms as well as on a shared understanding of the semantics of these terms. Such language communities can be found in our daily lives, e.g. people with the same profession, dialect, or interest, but can also be found in the academic world, where the active EA management research groups form language communities [SW08]. A method to design organization-specific EA management functions therefore needs to provide mechanisms, which account for the linguistic plurality.

## 2.2. Research outcomes

Research in the area of EA management is often conducted in interaction with an organization willing to practically apply the research results. On the one hand, this opens the door for “developing case studies” (cf. van Aken [Ak04, page 232]) by employing an intrinsically motivated industry partner. On the other hand, industry-funded research projects usually underly the partnering organization’s pace and are hence often forced to early delivery of results, which

aggravates the development of comprehensive theoretical underpinnings. This leads to a situation in which researchers are challenged to ensure that their research does not degenerate into “routine design” that according to Hevner et al. [He04, page 82] must be distinguished from design science. In contrast, the close cooperation can be used to contribute to design theory building e.g. via extracting case studies (cf. van Aken [Ak04] and Eisenhardt and Graebner [EG07]). Building on a figure from Gehlert et al. in [Ge09, page 442] that illustrates the twofold relation between theory and design according to Nunamaker et al. in [NCP91], we discuss how the contribution to theory building can be achieved (cf. Figure 2.3).

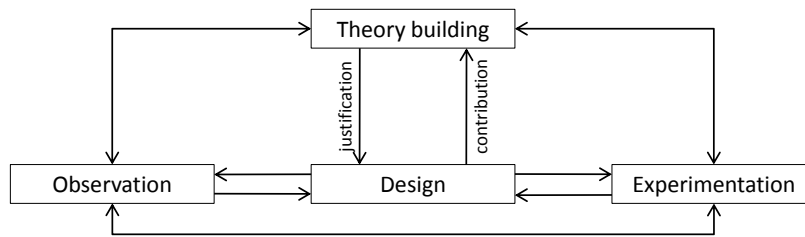


Figure 2.3.: Elements of theorizing in design science research [NCP91, Ge09]

The subsequently presented **design theory nexus instantiation for EA management** proposes a research outcome suitable for balancing relevance and rigor by relating pragmatics of industry-funded projects with the methodological rigors of scientific research. The concept of a design theory nexus is introduced in a step-wise manner starting with the notion of *design theory* as initially presented by Wall et al. [WWES92]. Based on the developed understanding of design theories, the role of *patterns* for theory building is discussed, leading to a revised understanding of the components of a design theory. Further the concept of the *design theory nexus* as proposed by Pries-Heje and Baskerville [PHB08] is introduced as means to link competing design theories. Finally, the concept of a *nexus-based IS generator* providing tool support for the evaluation of competing design theories and for constructing an IS operationalizing the design theory (theories) is discussed.

### 2.2.1. Theorizing in design science research

While the term **theory** is often used, little agreement on the components and structure of a theory can be found (cf. Gregor [Gr06]). According to Popper a theory is a set of statements of universal validity, that can be tested against observations of what occurs in the real world [Po80, page 59]. Gregor distinguishes in [Gr06] five types of theories in IS: i) theory for analyzing, ii) theory for explaining, iii) theory for predicting, iv) theory for explaining and predicting, and v) theory for design and action. We refer to the last type of theory in the remainder of the thesis as **design theories**. The notion of design theory was initially presented by Walls et al., who defined a design theory as “a prescriptive theory based on theoretical underpinnings which says how a design process can be carried out in a way which is both effective and feasible” [WWES92, page 37]. Design theories are subject to ongoing discussions in IS research (cf. Nunamaker et al. [NCP91], Markus et al. [MMG02], and Walls et al. [WWES04]). Although theories of the types i) to iv) have been opposed by March and Smith in [MS95] and Hevner in [He04] as outcomes of design research, we in line with Baskerville and Myers [BM02] as well as Winter [Wi08b] advocate for design theories as ar-

tifacts of design science research. While the purpose of a design theory is to support the achievement of goals, this goal-orientation is missing in other theories, like the explanatory or predictive theories typically originating from natural or social sciences.

While ‘traditional’ design science artifacts, in terms of March and Smith [MS95], provide a situated solution to a specific problem in a defined context, design theories make prescriptions how a class of problems can be solved (cf. Venable [Ve06b]). This class of problems represents the **problem domain** of the design theory, whose problems are solved using corresponding methods and concepts from the **solution domain**. In line with the argumentation of Hevner et al. [He04] and our discussion on the influence of the organizational context on the design of an organization-specific EA management function, we further introduce the **context domain** that refers to the environment in which a design theory is applied. Based on the preceding considerations, we provide a definition of the term design theory.

**Definition: Design theory**

A design theory is a prescriptive theory based on theoretical underpinnings which says how a design process can be carried out to address an arbitrary problem from the theory’s problem domain with a solution from the theory’s solution domain in a way which is both effective and feasible in the given context from the theory’s context domain.

As mentioned above, no common understanding on the components of a design theory currently exists. In [WWES92] Walls et al. propose seven components of a design theory, which can be classified as either belonging to the *design product* or the *design process* respectively. Figure 2.4 provides an adapted overview on the components and their relationships. Thereby, a color-coding is used to differentiate between the design product-related components (blue symbols) and the design process-related ones (red symbols).

The components related to the design product (highlighted in blue) are:

**Kernel theories** govern the design requirements and may be of any type according to the typization of Gregor in [Gr06], e.g. analytical, explanatory, predictive, and/or design theories, often drawn from natural or social sciences.

**Meta-requirements** describe the class of problems to which the design theory applies.

**Meta-design** represents a class of artifacts hypothesized to meet the meta-requirements.

**Testable design product hypotheses** are statements that can be used to verify whether the meta-design satisfies the meta-requirements. According to Venable in [Ve06b, page 4], the testable design hypotheses can be applied on the meta and instantiation level. The former means that the test verifies whether the meta-requirements are met by the meta-design, while the latter refers to evaluating, if an instantiated design satisfies the instantiation of the meta-requirements in a specific situated context.

Complementing the above list of design product-related components, the following constituents (highlighted in red) are concerned with the design process:

**Kernel theories** govern the design requirements and may be of any type according to the typization of Gregor in [Gr06] often drawn from natural or social sciences. These kernel theories may be different from those associated with the design product.

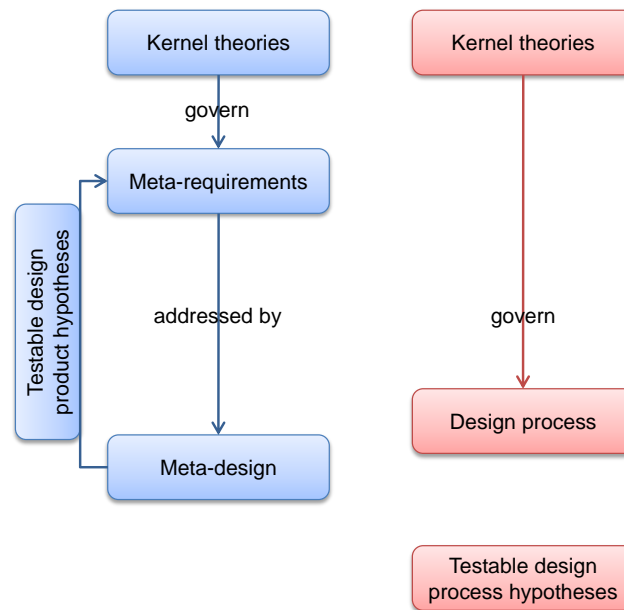


Figure 2.4.: Components of a design theory according to Walls et al. in [WWES92]

**Design process** describes procedure(s) for constructing the design products. The term *construction* should according to Venable in [Ve06b, page 4] be interpreted more abstract, applying to a broad range of activities used to develop a particular, situated instantiation of the meta-design.

**Testable design process hypotheses** are statements that can be used to verify whether or not the application of the design process will result in a proper instantiation of the meta-design.

In 2004 Walls et al. revisited their contribution from 1992 as being well-founded but identified some potentials for improvement [WWES04]. This argumentation, especially regarding the structure of a design theory, was taken up by Gregor and Jones in [GJ07], who presented an extended framework for an *information systems design theory* that targets prescriptive theories for *design artifacts*. In line with the discussion of March and Smith [MS95] the term design artifact is used in the remainder of the thesis to account for the plurality of artifacts that are created by a design method. In their critical examination they identified four issues regarding the framework proposed by Walls et al. in [WWES92] i) a lack of clarity regarding the goal of a design theory, ii) omission of the mandatory ‘units’ (constructs) and ‘system states’ (cf. Dubin [Du78]), iii) missing attention of design instantiation, and iv) a possibly unnecessary distinction between kernel theories for design processes and design artifacts. Against this background Gregor and Jones proposed eight components of a design theory, of which the former six are mandatory while the last two are optional. These eight components are introduced subsequently and mapped to the components as proposed by Walls et al. in [WWES92].

**Purpose and scope** describe what the design theory is intended for, i.e., specifies the type of artifact to which the theory applies and the scope or boundaries of the theory. It refers to the goals and meta-requirements in the terms of Walls et al.

**Constructs** are representations of the entities of interest in the theory, i.e., descriptions of physical phenomena or theoretical terms of interest. The concept of constructs is not referred to as a component of the design theory framework of Walls et al.

**Principles of form and function** represent the abstract blueprint, i.e., the principles that describe the structure and functioning of the design artifact. In terms of Walls et al. these principles represent the meta-design.

**Artifact mutability** accounts for the fact, that design artifacts in rapidly changing domains have to adapt to their changing environment. The artifact mutability represents the degree of design artifact change that is prepared by the design theory. Wall et al. do not consider this circumstance.

**Testable propositions** represent statements of truth about the design theory—testable hypotheses state, whether the design artifact matches its purpose or whether the design process correctly instantiates the design artifact. In this way, the testable propositions combine the testable design product hypotheses and the testable design process hypotheses of Walls et al.

**Justificatory knowledge** refers to the (possibly incomplete) base of explanatory knowledge on which the design theory is grounded. The justificatory knowledge summarizes the two types of kernel theories of Walls et al. used for justifying the meta-requirements or the design process, respectively.

**Principles of implementation** represents the principles for applying the theory in specific contexts, i.e., the means by which the design artifact is created. The principles of implementation refer to the design process of Walls et al.

**Expository instantiation** describes an exemplary design artifact created by using the theory that can assist in communicating and representing the theory. Furthermore, this design artifact might be helpful for exemplarily validating the theory’s applicability. Walls et al. do not explicitly account for such a concept.

Figure 2.5 shows an adapted version of the overview on components of a design theory as introduced by Walls et al. (cf. Figure 2.4). The single components are labeled according to the terminology of Gregor and Jones as discussed above. Again a color-coding is used to distinguish between design artifact-related components (blue symbols), design process-related components (red symbols), and elements resulting from the instantiation of the design theory (green symbols).

### 2.2.2. The role of patterns for theory building

Reflecting the challenges of performing research in close cooperation with an industry partner, Vaishnavi and Kuchler [VK04] coined the term *community determined output*, which delineates that the expected level of abstraction in the research outcome is determined by the community. In their publication the authors describe different levels of metaization, called *abstraction*, that are applied to get from situated implementations (solutions) to design theories. Figure 2.6 illustrates the different abstraction levels, of which in particular the intermediary level of *knowledge as operational principles* is of interest with respect to practice-driven research.



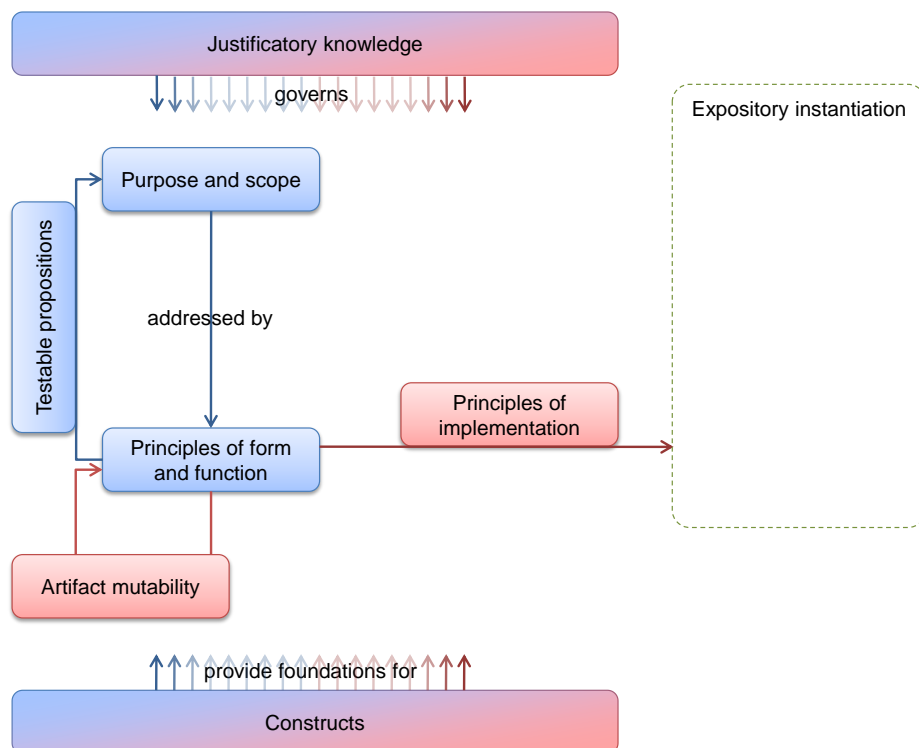


Figure 2.5.: Components of a design theory according to Gregor and Jones in [GJ07]

While we cannot expect practitioners to abstract their implementations towards ‘full-blown’ theories, knowledge sharing and communication processes within a group of practitioners leads to abstracted representations of implementations on the level of operational principles. In this sense, theory building in close cooperation with practitioners can harvest this knowledge as operational principles instead of directly developing and abstracting theories from situated implementations.

Documenting best practice solutions to recurring problems in a specific context, i.e., knowledge on operational principles in the sense of Vaishnavi and Kuchler [VK04], by so-called **patterns** is a commonly accepted way to facilitate knowledge abstraction and dissemination in design-intensive domains. The idea of patterns originates from construction and urban planning and was introduced by Alexander et al. [A172, A177, A179b], who provided the subsequent definition of a pattern [A177, page x].

#### **Definition: Pattern**

Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

Since that time, related fields have successfully adopted the idea of patterns, e.g. software engineering by Gamma et al. [Ga94], software architectures as presented by Buschmann et

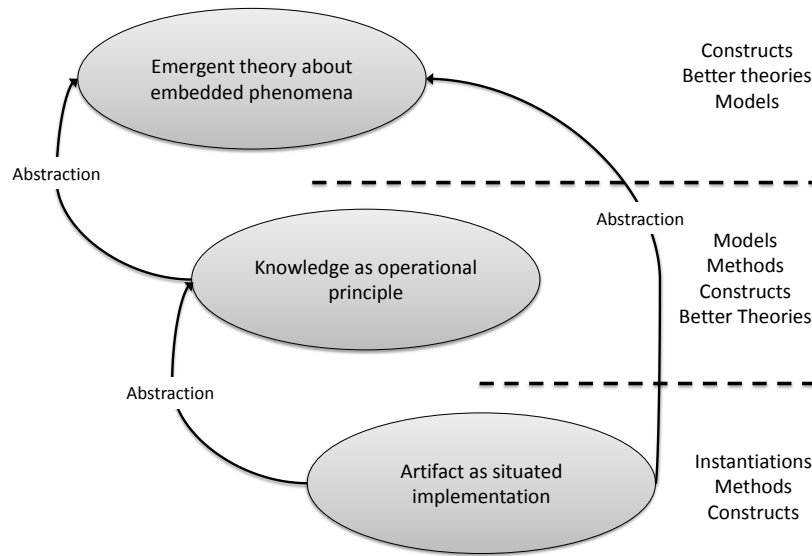


Figure 2.6.: Levels of abstractions of research outcomes according to [VK04]

al. [Bu96], or project management by DeMarco et al. [De08]. In all these disciplines patterns describe operational knowledge gained from practice, i.e., patterns are neither invented nor developed, but have been observed. Giving account to this fact, Coplien [Co96] establishes the so-called *rule of three*, which specifies that a documented pattern must provide reference to at least three known uses in practice. Prior to that an observed solution represents a *pattern candidate*. Different ways to document a pattern, so-called pattern forms, have been developed each delineating constituents and parts considered relevant. Whereas the Alexandrian form is mainly narrative and comes without additional structuring, the canonical form of Buschmann et al. [BHS07] designates the following constituents as essential elements of patterns.

- A *pattern name*, which identifies the pattern and makes it memorable, usable, and distinct,
- an *example* illustrating the problem to be addressed by the pattern,
- a *context description*, representing the situations in which the pattern applies,
- a *problem description*, specifying the problem addressed by the pattern,
- a *solution description*, i.e., the elements of the solution design, their responsibilities, relationships, and collaborations, and
- *links to other patterns (see also)*, i.e., references to other patterns solving similar problems as well as to patterns, which help to refine the respective pattern.

Further detailing the above list of constituents Meszaros and Doble [MD97] propose a ‘meta-pattern’, i.e., a pattern about patterns, which constitutes of a set of patterns for pattern writing. Therein, they delineate two additional elements of a pattern—*pattern users* and *forces*. The *pattern users* make the intended target community of a pattern explicit. The concept of *force* is according to Meszaros and Doble defined as “contradictory considerations that must be taken into account when choosing a solution to a problem” [MD97, page 535].

Following this understanding of force, the solution as proposed by a pattern must explicitly describe how forces can be resolved. Furthermore, Meszaros and Doble list several optional elements of a pattern description, e.g. *indications* for the existence of the problem, a *rationale* explaining the appropriateness of the pattern's solution, *aliases*, i.e., alternative names under which the pattern might be known, *acknowledgments* to thank people involved in pattern identification or documentation, and the *resulting context*, i.e., "the context that we find ourselves in after the pattern has been applied" [MD97, pages 536–537]. The latter element is frequently alluded to as *consequence* in prevalent pattern descriptions [Ga94, Er08].

While the importance of patterns in closely related disciplines, as software engineering, is unquestioned, the role of patterns in IS research is subject to controversies. In the *discussion and opinion section* of the *Business & Information Systems Engineering* journal, researchers and practitioners outlined their attitudes towards patterns for IS research [Wi09b]. Thereby, special emphasis was put on advantages and disadvantages resulting from the practical nature of pattern-based research. Discussants of non-academic provenience, as e.g. Keller, outline that in particular the rule of three is in the academic context regarded as a lack of originality, such that patterns are not understood as results of 'true' research. Challenging former statements, Fettke and Loos argue that nowadays no distinction between reference modeling, which according to Becker et al. [BDK07] is a classical topic and outcome of IS research, and pattern identification should be made. Fettke and Loos go even further by stating that the terms *reference model* and *pattern* have become interchangeable representing 'two sides of the same coin'. Reminiscing about the rule of three, we advocate for making a differentiation between reference models, which are invented and developed and patterns as observed and documented occurrences in practice. While two active and distinct communities each promoting either term exist, current literature presenting IS research results frequently makes use of terms like 'good practices' or 'best practices' to avoid the explicit classification of their artifacts.

From a research perspective patterns can be regarded as coherent and self-contained design entities that describe a solution to a specific problem thus we understand patterns as elementary design principles in line with Markus et al. [MMG02]. In this sense, it seems reasonable for us to interpret patterns as pre-products of a design theory. Support for this interpretation can be found by Schermann et al. [SBK07a], who establish a conceptual structure of pattern-based design theories merging the pattern structure of Buschmann et al. [BHS07] with the structure of a design theory according to Walls et al. [WWES92]. We subsequently revisit a slightly adapted version of the contributions of patterns along the design theory framework introduced in Figure 2.5, which is based on the idea to use patterns to represent and describe knowledge as operational principles in terms of Vaishnavi and Kuchler [VK04].

Considering patterns as preliminary stage artifacts in abstracting design theories, we abstain from making a direct mapping of pattern description elements and design theory components, but exemplarily detail the relationships between patterns and design theories as proposed in [BMS10k]. The example given in a pattern describes a *situated problem in context* from the expository instantiation of a design theory. The context and problem descriptions from a pattern contribute to the *purpose and scope* of the framework of Gregor and Jones in Figure 2.5. The solution provided by the pattern refers to the *principles of form and function*. Furthermore, the *see also* section of patterns contains different kind of relationships (cf. Noble [No98]) which contribute to different components of a design theory. This links back to an

idea presented by Alexander et al. [A177], the idea of a *pattern language*, i.e., a system of interrelated patterns. To illustrate the components that can be composed from the interrelated set of patterns, three relationships are exemplarily introduced below and their contributions for theorizing are detailed:

**Refined by** A refining pattern targets a similar problem and context as its ‘parent’, but provides a more detailed solution model or outlines a broader variety of forces that have to be balanced. Relationships of that type contribute to the *artifact mutability* in the design theory, as they sketch possible trajectories for refining the design artifact.

**Used by** The usage relationship describes that and how a larger pattern, sometimes alluded to as ‘umbrella pattern’, employs another pattern for solving a sub-problem of the umbrella problem. Building on the usage relationship, it is possible to aggregate solution building blocks into more comprehensive *principles of form and function* for a coarse grained problem.

**Variant** A pattern variant targets a similar or closely related problem and context as the initial pattern, providing a solution that not far differs from the original one. Relationships of that type may help to refine the *purpose and scope* description by both broadening the scope of the corresponding classes and raising further dimensions of distinction.

Furthering the idea of a pattern-based design theory building, other general components of a design theory can be derived from patterns. The *principles for implementation*, for instance, can be fairly general and show up as rules and guidelines for selecting and integrating patterns. *Justificatory knowledge* can be derived from patterns related to the selected pattern. Similarly, *testable propositions* are a by-product of pattern documentation, i.e., each pattern itself is a solution model hypothesis<sup>3</sup>, stating that a specific class of problems situated in a distinct context can be addressed by a solution. The formulation of a consistent set of *constructs*, however is a challenge in developing a pattern-based design theory. While after careful revision, one can expect that a single pattern employs a consistent terminology, inter-pattern consistency of terms is usually not given.

In [BMS10k] we propose a four step approach for theory building based on patterns, which requires the involvement of a partnering organization, consisting of the activities

**Step 1** *observe and document design patterns*: Whenever at least three occurrences of a best practice solution are observed, the solution is documented and refined in cooperation with the industry partner, e.g. via so-called pattern workshops.

**Step 2** *elicit pattern terminology*: Newfound patterns need to be compared with prevalent patterns to identify synonyms and homonyms.

**Step 3** *devise pattern-relationships*: Newfound patterns have to be incorporated into the existing knowledge base of patterns by defining relationships to already existing ones.

**Step 4** *derive terminological compatibility relationships*: Two new types of relationships between patterns are introduced—**linguistically compatible** and **linguistically diverse**—to indicate that two patterns employ compatible or conflicting terminologies.

---

<sup>3</sup>While Gregor and Jones [GJ07] propose to define the *testable propositions* on a more abstract level, we distinguish different types relevant for theory evaluation and theory building (see Figure 2.7). The latter ones, the **testable solution model hypotheses** are of relevance in this context.

The last activity explicitly accounts for the terminological issues discussed before. With exception of the last one, the activities can be conducted in close cooperation with the industry partner of the research project. This is especially true, as each activity outputs worthwhile intermediary results that the partnering organization can readily re-use under the premise that the research projects targets a problem of relevance for the partner. With the emerging pattern language on the one hand and the indications of the patterns' utility gained from the application in the context of the partnering organization on the other hand, a researcher has a solid and sound basis to finally devise a comprehensive design theory or a set of design theories. At least during the final step of theory formulation, the formerly documented terminological issues can be resolved in favor of a well-defined and grounded terminology.

Reflecting the above described influence of pattern-based theory building, we subsequently propose a revised version of the framework delineating the components of a design theory [BMS10k]. The framework builds on the work of Gregor and Jones and the explanations of Schermann et al. in [SBK07a]. While the constituents described and presented by Schermann et al. are largely identical to those presented above, the used terminology is quite different and reflects the influence of pattern languages on theorizing in IS as discussed in [SBK07a, BMS10d, BMS10k]. Table 2.1 gives a comparison of the different approaches as provided by Gehlert et al. [Ge09, page 443]. In the table, we further highlight the terms used in our design theory framework as presented in [BMS10k], which also represents the terminology used in the remainder of this thesis.

Walls et al. [WWES92]	Gregor and Jones [GJ07]	Schermann et al. [SBK07b]
Meta-requirements	Purpose and scope	Context and problem
Meta-design	Principles of form and function	Solution model
Kernel theories	Justificatory knowledge	Theory references
Design process	Principles of implementation	Design method
- <sup>1</sup>	Artifact mutability <sup>2</sup>	Pattern references
Testable hypotheses	Testable propositions	Consequences
- <sup>3</sup>	Expository instantiation	(Instantiation) <sup>4</sup>

<sup>1</sup> Walls et al. do not account for the concept of artifact mutability in [WWES92].

<sup>2</sup> We use the term **principles of adaptation** in [BMS10k].

<sup>3</sup> Walls et al. do not demand an expository instantiation in [WWES92].

<sup>4</sup> Schermann et al. do not provide an explicit name for this concept in [SBK07b].

Table 2.1.: Existing approaches to design theorizing in IS research

Based on the work of Walls et al. in [WWES92] and the work of Gregor and Jones in [GJ07], we subsequently propose a framework for design theories in IS, which uses a terminology hinting to the pattern-based theory building as discussed above. Figure 2.7 provides an overview about the components of the framework and details on the relationships between the components. We thereby employ the color-coding outlined above, i.e., process-related constituents of the design theory are represented by red symbols, artifact-related ones are represented by blue symbols, and green symbols are used to represent elements resulting from the application of the design theory, i.e., relating to the design artifact.

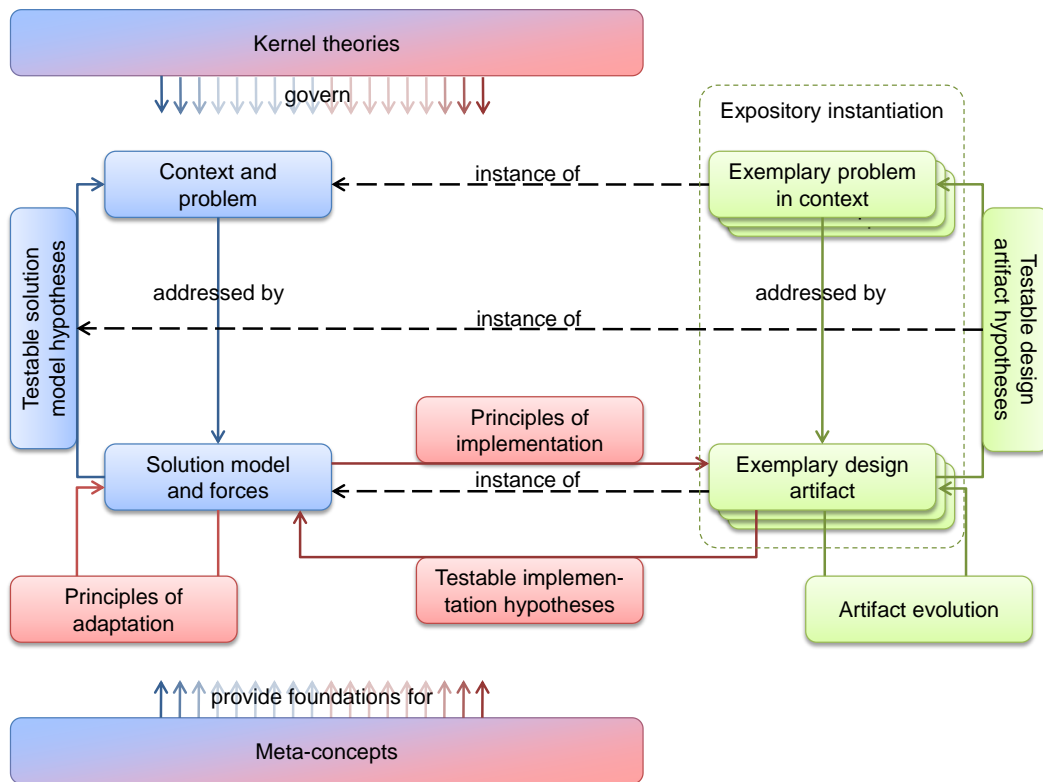


Figure 2.7.: Components of a design theory according to Buckl et al. [BMS10k]

Our framework expands the aforementioned ones by detailing the application domain of the design theory. Thereby, the *situated problem in context* represents the starting point for an actual **development process** and the **design artifact** that addresses the situated problem in a certain context and results from applying the principles of implementation. Following Venable's discussion in [Ve06b] on the different levels a testable hypotheses can be tested on, we introduce two additional kinds of testable hypotheses, which complement the testable solution model hypotheses of the meta-level. In contrast to Gregor and Jones who argue for the amalgamation of testable hypotheses formulated on a more general level to target both the design artifact and the principles of implementation in [GJ07], we regard the distinction to be advantageous to support hypotheses testing focused exclusively on artifact or method aspects. The concept of **artifact evolution** is added for reasons of completeness, as it complements the artifact mutability that on the meta-level introduces the capability to adapt the solution model to a changing environment. In addition, the component construct as introduced by Gregor and Jones in [GJ07] is renamed to **meta-concepts** to avoid an ambiguous term, which could be mixed up with the construct concept as part of the design artifact from March and Smith in [MS95, page 253]. The meta-concepts form a conceptualization of the context, problem, and solution domain interrelated by the design principle.

### 2.2.3. The concept of design theory nexus

In [PHB08] Pries-Heje and Baskerville present the idea of a **design theory nexus** as means to connect existing approaches, i.e., design theories, which provide competing<sup>4</sup> solutions for a problem domain. A design theory nexus not only connects competing design theories, but further helps “decision makers in choosing which of the theories are most suitable for their particular goals and their particular setting” [PHB08, page 733]. In line with this argumentation, Pries-Heje and Baskerville define the concept of design theory nexus as follows [PHB08, page 733].

**Definition: Design theory nexus**

A design theory nexus is a set of constructs and methods that enable the construction of models that connect numerous design theories with competing solutions.

Pries-Heje and Baskerville state that a design theory nexus is especially useful in cases of solving so-called [wicked problem]wicked problems or “ill-structured” design problems [PHB08, page 732]. Ill-structured thereby refers to the criteria used to decide, if a design theory is appropriate or suitable, namely the design goals and the design environment. Decision making requires one or more criteria. Whereas it is easy to rank competing solutions against one criteria, multiple criteria decision making heavily depends on the preferences of the decision maker. Decision making becomes even more complicated, if *asymmetric criteria* are considered. According to Pries-Heje and Baskerville in [PHB08, page 736], asymmetric criteria are those in which the criterion for selecting one design theory is entirely different from the criterion related to a competing design theory. In other words, the competing design theories are not alternatives with respect to a single set of criteria. While ‘classical’ contingency approaches establish a static relationship between the structure of the problem and context to the solution techniques and assume that one set of criteria can be used to assess all approaches [Ba90, page 62–64], they are not sufficient to provide decision support in a context with asymmetric criteria as the EA management domain.

As already discussed in the motivating Section 1 designing and establishing an organization-specific EA management function represents a wicked problem, for which a plethora of competing design theories exist. Each of the theories thereby employs an implicit or explicit description of the context in which it can be applied and the problems that are addressed. In terms of Pries-Heje and Baskerville, these criteria (context and problem) represent asymmetric criteria for which classic contingency-based approaches do not provide decision support. If the concept of the design theory nexus is applied to a specific design problem via connecting a respective set of design theories, this leads to what Pries-Heje and Baskerville call a **design theory nexus instantiation**. A design theory nexus instantiation for EA management supports organizations in choosing a suitable EA management approach according to the organizations’ goals, i.e., objectives of the EA management initiative, and the corresponding organizational context. In a design theory nexus, each competing design theory relates to one or more criteria that is a prerequisite or beneficial for the application of the theory. Accord-

<sup>4</sup>Pries-Heje and Baskerville use the term *alternative* instead of competing. We abstain from the term *alternative* in this thesis in favor of the term *competing* as *alternative* is sometimes understood to be limited to two choices representing mutually exclusive possibilities.

## 2. Research Design

---

ing to Pries-Heje and Baskerville in [PHB08, page 743], a design theory nexus instantiation generally consists of the following five constituents:

**Goals** describe what the system is intended for, i.e., the problems according to the terms introduced above (cf. Table 2.1).

**Environment** refers to contingencies, which are outside of the people involved, i.e., the context of a design theory (cf. Table 2.1)

**Competing design theories** each refers to alternate design theories providing a solution to a class of problems, i.e., the solution models provided by a set of design theories (cf. Table 2.1)

**Design theory nexus** defines the connection point at which the competing theories are bound with realities into a design solution.

**Design solution** (design artifact) represents the result constructed from competing design theories.

Figure 2.8 provides an overview about the components of a design theory nexus and illustrates their relationships.

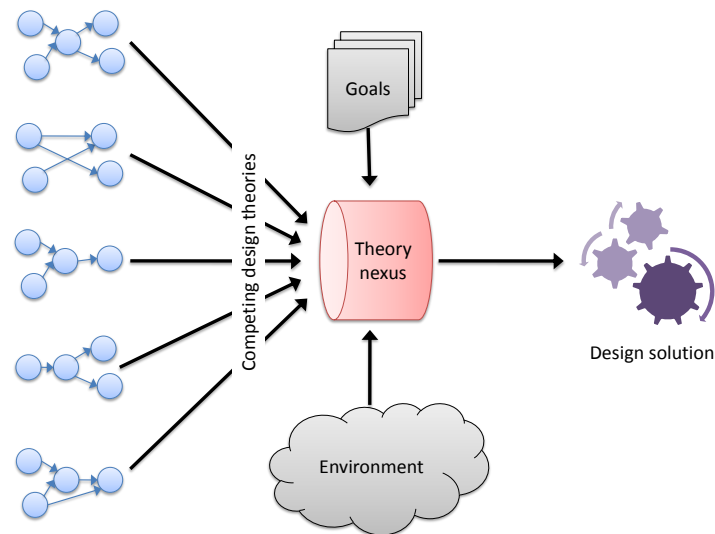


Figure 2.8.: Components of a design theory nexus according to Pries-Heje and Baskerville in [PHB08]

Revisiting our discussion on pattern-based theory building, the different backgrounds from which design theories originate typically hamper the development of a common terminology. The instantiation of a design theory nexus, which interlinks theories with different terminological backgrounds has to establish relationships between the different meta-concepts denoting terminological issues. Furthermore, relationships targeting the context, problem, and solution domains of distinct design theories exist. Reusing the relationship types as defined by Noble [No98] and exemplified in Section 2.2.2, two design theories can be independent, conflicting, or refinements. In a similar vein, theories can relate via the resulting and/or required context that must hold to enable utilization of a theory. That is a design theory can be considered a potential prerequisite to the application of another.



The competing design theories, which form the input for such a design theory nexus instantiation, can be described using the design theory framework illustrated in Figure 2.7. Subsequently, the construction of an instance of a design theory nexus according to Pries-Heje and Baskerville [PHB08] is detailed. Each of the five steps is sketched below.

- Step 1: *Identify approaches*** In the first step, the available approaches (design theories, patterns, solutions etc.) in the area under consideration are examined, e.g. via a literature analysis.
- Step 2: *Analyze approaches*** In a second step the identified approaches are investigated for explicit or implicit conditions, i.e., context and problem descriptions, which must hold for the approach to achieve the highest utility. Here, it has to be noted that these conditions might be unequal for any pairing of the theories.
- Step 3: *Formulate assertions*** The third step assesses the identified conditions for practical relevance and formulates them as assertions.
- Step 4: *Develop decision making process*** In the fourth step, a decision making process building on the assertions is designed.
- Step 5: *Develop tool*** Final step five combines the approaches, assertions, and the process into a tool (the artifact), which supports the evaluation regarding the fit for each design theory in a given situation.

While the first two steps of the described method for constructing a design theory nexus instantiation represent analytic steps, the third step shifts the process to constructive design. During the construction of a design theory nexus instantiation, the identified constituents of a design theory are used, e.g. kernel theories provide an analysis framework for identifying and analyzing existing approaches, or the meta-conceptualization provides a foundation for the description of the approaches and assertions. The latter two steps of the construction method can be performed on different levels of detail depending on the application area, i.e., the tool developed in step five, for instance, may be a spreadsheet-based calculation of the fit for purpose or a sophisticated software tool providing dedicated support.

#### 2.2.4. Information systems generators

In [WWES92, page 46] Walls et al. discuss different aspects of theory development as reflected in the development method. Prominent in these discussions is the role of IS development, which is regarded an essential cornerstone in devising a design theory. More precisely, Walls et al. introduce the generalized concept of the **information system generator**, a configurable IS that incorporates the essence of an underlying design theory, i.e., has the capability to construct IS. In line with the explanations given by Walls et al. in [WWES92, page 46], we define an IS generator as follows.

**Definition: Information system generator**

A tool incorporating the capabilities used to construct information systems derived from an underlying theory of the development method they are intended to support.

An information system generator reflects the prescriptions of a design theory and incorporates the underlying principles of implementation as well as the conceptualization of the problem, context, and solution domain. By configuring such a generator, the researchers apply the design theory in the situated context, i.e., instantiate the theory’s meta-design into an actual design. In other words the IS generator creates an IS implementing the theory-based solution, e.g. provides support for an organization-specific EA management function. An IS generator can be used to exemplify the application of the design theory through the creation of an expository instantiation and to support evaluation of the design hypotheses on instance and meta-level. On the instance level the testable hypotheses are evaluated via expository instantiations. On the meta-level, the testable implementation hypotheses can be evaluated through the operationalization of the theories’ principles of implementation, which produce testable artifacts, e.g. source code and operational models, that relate to the implementation hypotheses. As we can assume that incorrect implementation hypotheses lead to inconsistent artifacts, i.e., operating models, the testable implementation hypotheses can be evaluated, as these inconsistent artifacts prevent an successful operationalization.

In [PHB08, page 737] Pries-Heje and Baskerville reference the idea of tool support for the design theory nexus instantiation as “a package of elements that collectively embody the capability to build a specific DSS [decision support system]”. In line with step five of the process to construct a design theory nexus instantiation they call for a tool support to manage the amount of interlinked design theories that are incorporated into a nexus and to support nexus-based decisions. While an IS generator is per design limited to a single design theory, tool support for a nexus instantiation cannot directly rely on an IS generator. Thus, we further the conception of an *decision support system* (DSS) of Pries-Heje and Baskerville [PHB08, pages 736–737] to a tool supporting evaluation of competing design theories, i.e., a **nexus-based IS generator**. Our nexus-based IS generator a) computes the appropriateness of a particular design theory given the goals to be pursued and the environment description, i.e., the situated problem and context, b) keeps track of the interdependencies between already selected design theories and theories applying to yet not covered goals, as well as c) supports the integration of selected theories in a comprehensive solution. During the application of the nexus-based IS generator, the designer can in a first step search the solution domain for ideal solution models and integrate competing design theories. The results of this search are instantiated in a second step to a design solution to operate in the specific context and the situational problems. This adaptation is performed using the theories’ principles of implementation.

### 2.2.5. Synthesizing the research outcomes of this thesis

Developing a design theory nexus instantiation for EA management follows the explicit recommendation of pluralism of theories as stated by Zelewski and Winter in a discussion on “fundamental research in business and information systems engineering” (cf. Winter et al. [Wi09a]). This thesis aims at five distinct research outcomes contributing to a development method for organization-specific EA management, which are subsequently detailed:

**Meta-conceptualization for describing the context, problem, and solutions domain:** In the problem and context domain, the meta-concepts provide a common terminology to describe the goals and the organizational context of a situated EA management initiative. In the solution domain, the meta-concepts provide a common terminology and a framework to

describe the solution models. On the one hand, the meta-concepts provide the basis for a comprehensive language to document and relate solution models to design an EA management function that can be operationalized. On the other hand, the meta-concepts provide the basis for a comprehensive framework—the design theory nexus instantiation for EA management—to select, assembly, adapt, and integrate solution models to a comprehensive EA management function.

**Design theory nexus instantiation:** The design theory nexus instantiation describes the solution models that the design theories provide for constructing a situational EA management function. Thereby, each solution model corresponds to a distinct EA management goal in a distinct organizational context. Based on the common terminology established by the meta-conceptualization, each solution model defines steps to be taken, participants in these steps, forces, and consequences. Furthermore, the meta-conceptualization is employed to relate the different solution models based on the relationships identified during the construction of the design theory nexus instantiation for EA management.

**Method to develop and maintain the design theory nexus instantiation:** Detailing the five steps to instantiate a design theory nexus as proposed by Pries-Heje and Baskerville [PHB08, pages 737–738], the method depicts how the design theories derived from various sources are altered and linked in the design theory nexus instantiation. Well-grounded in the terminological basis established by the meta-conceptualization, the method provides techniques for adapting the design theories to the common terminology as established by the meta-concepts. Further, the method describes guidelines on how to incorporate newfound design theories in the design theory nexus instantiations using the concept of pattern-based theory building and how the meta-conceptualization can be evolved and adapted.

**Method and techniques for applying the design theory nexus instantiation:** The method and techniques describe guidelines how a situational EA management function can be developed or an existing EA management function can be adapted. Thereby the enterprise-specific goals to be pursued and the organizational context of the associated organization are taken into account. The method further provides techniques to integrate the solution models to a comprehensive EA management function and adapt it to the specific context of the associated organization. Creating an expository instantiation the method applies the principles of implementation in practice, i.e., in the environment of an organization willing to use the design theory nexus instantiation for EA management. In this vein, the steps taken to elicit the actual goals and organizational contexts of the EA management endeavor are described. Complementingly, the method describes guidelines on how relationships between solution models can be used to derive possible evolution paths and adaptation scenarios as well as techniques to support the transformation from the current state of the EA management function to the selected new configuration based on solution models.

**A nexus-based IS generator:** The nexus-based IS generator as developed in this thesis presents itself as a configurator for a highly configurable EA management tool. The resulting EA management tool can be classified according to Matthes et al. in [Ma08, pages 344–346] as a *process-driven* EA management tool that allows flexibly adaptation of the activities making up the EA management function. The nexus-based IS generator incorporates as basis the unified terminology provided by the meta-conceptualization of the solu-

tion domain. The goals and organizational context descriptions from the problem and context domain are represented in the generator and linked to methodical prescriptions as exponents of the corresponding solution models. Implementing the principles of implementation and adaptation the generator provides a wizard-based mechanism for (re-)configuration.

### 2.3. Research method

The discipline of IS research is characterized by an ongoing discussion on the research methods appropriate for the field of investigation [AK99, Le99]. Notwithstanding the importance of a multi-method approach for the IS field, the paradigm of design science takes up a prominent role especially in the German-speaking community of ‘Wirtschaftsinformatik’ [Ni06b, ÖWe10]. Thereby, the role of theory and theorizing for IS is still a topic of ongoing discussions, as stated by Vaishnavi and Kuchler who phrased the issue as follows: “Even within design research communities there is lack of consensus as to the precise objective—and therefore the desired outputs—of design research” [VK04]. As already discussed in Section 2.2, we in line with other researchers, e.g. Gregor [Gr06], Markus et al. [MMG02], Rossi and Sein [RS03], Venable [Ve06b], or Walls et al. [WWES92], argue for the need of theories in design science.

In the light of the controversial role of design theories, it is not surprising that only a few approaches exist, which target the development of such theories. Among them is the activity framework presented by Venable in [Ve06a] and in [Ve06b]. The framework embeds the activity of theory building into the activities of problem diagnosis, theory application, and theory evaluation. Figure 2.9 shows a slightly adapted version of the activity framework of Venable [Ve06b, page 17], which is used to structure the work presented in this thesis. Minor adaptations with respect to the naming are performed to reflect the combination of competing theories in a design theory nexus instantiation in contrast to the development of a single design theory. Figure 2.9 relates each activity to the according chapter(s) of this thesis. In addition, the color-coding introduced in Section 2.2 is used to demonstrate which chapters are concerned with method-related (red), nexus-related (blue) contributions, or the application thereof (green). According to Venable [Ve06a, page 185] a research program may contain any subset of the aforementioned activities. As indicated by the arrows between the activities, complete flexibility to move from one activity to another exists. In line with this understanding, Venable argues for performing the respective activities in an iterative and cooperative manner [Ve06a, page 185].

Developing an organization-specific EA management function forms a topic of high practical relevance, which is beneficially approached from a design science perspective. In line with the activity cycle of Venable [Ve06a, Ve06b] or the approach presented by Rossi and Sein [RS03] such a research topic can best be faced by

- exploring and understanding EA or EA management-related problems (*problem diagnosis* in terms of Venable or *identifying a need* according to Rossi and Sein),
- designing an appropriate EA management function (*application* according to Venable or the *build* step of Rossi and Sein),

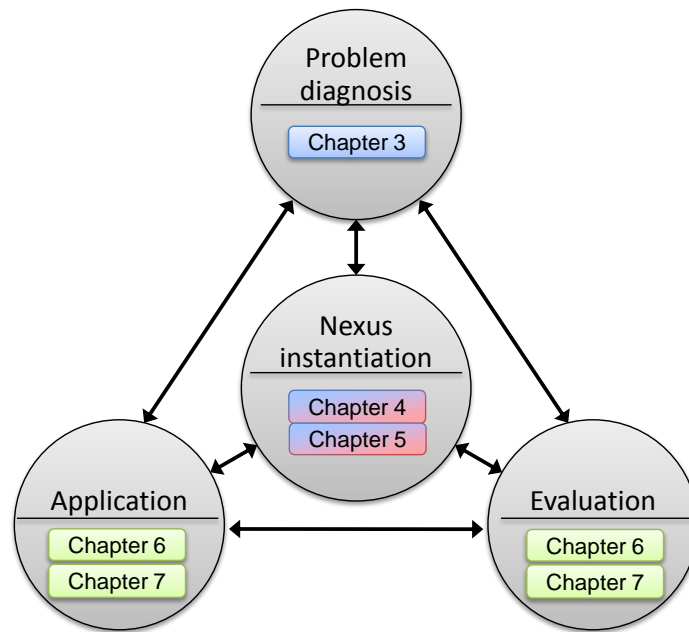


Figure 2.9.: The research activity framework of this thesis (based on Venable’s activity framework [Ve06a, Ve06b])

- evaluating the performance of the EA management function in the corresponding organizational setting (*evaluation* in the terminology of Venable or Rossi and Sein), and
- revisiting the design theory used (the embedded activity of *theorizing* as discussed by Venable or the steps *learn and theorize* according to Rossi and Sein).

Due to the complexity and scope of the research area, the last steps of evaluation and learning are hard to perform in the context of EA management, at least from the perspective of a dissertation. As establishing an EA management function is a strategic and long-term endeavor, conducting iterative research on the performance of the EA management function in the time-frame of a dissertation is infeasible, especially as an EA management function is intended to enhance the interaction between different enterprise-level management functions, e.g. project portfolio, synchronization, and release management. While immediate outcomes of EA-related projects can be analyzed, the improvement regarding coordination between management functions is expected to have late effects, which require a long period of observation. Ross and Beath who proposed four maturity stages for the evolution of an EA management function, estimate that large organizations need about five years to evolve from one stage to the next [RB06, page 185]. The research method targeting a development method for organization-specific EA management functions and underlying design theory nexus instantiation, accounts for this fact. Following the activity framework of Venable [Ve06a], we subsequently detail the constituting activities of the research method which incorporates the steps to construct a design theory nexus instantiation. Reflecting conflicts inherent to research conducted in close cooperation with an industry partner, i.e., between responding to practitioners demands and the methodological rigor required for academic contributions [Ga07], we propose a slightly adapted version of Venable’s activity framework. Our activity framework

combines the activities of applying and evaluating, incorporates the idea of the design theory nexus instantiation to link competing design theories, and uses the process of pattern-based theory building.

### 2.3.1. Problem diagnosis

In the problem diagnosis activity, we establish a comprehensive understanding of the class of problems to which the design theories apply as well as of the context domain that constrains the application area of the design theories. Problem diagnosis consists of three phases—**exploring the problem and context domain**, **structuring the domains**, and **meta-conceptualizing the domains**—for which no ordering exists, but which are carried out in an iterative manner following a hermeneutic cycle. The intended outcome of the problem diagnosis can be summarized as a set of problems (relating to the goals of the design theory nexus instantiation) as well as a set of context descriptions (relating to the environment of the design theory nexus instantiation), expressed in a well-defined terminology building on a well-founded framework as well as a conceptual model of both domains.

In the activity **exploring the problem and context domain**, we identify recurring problems to which the design theory nexus instantiation should be applicable. In addition, we collect typical contexts in which such problems occur. We investigate and interpret existing literature based on our initial knowledge base and extract relevant information shaping the problem and context domain. We apply the technique of “enumeration of problems” as outlined by Takeda et al. [Ta90, page 43] to revisit literature from researchers and practitioners illustrating cases of designing an EA management function. This technique can build on different sources to explore the domain of possible problems. Furthermore, we incorporate best practice knowledge, e.g. at a partnering organization, via observing and documenting patterns. During the problem diagnosis, especially the problem and context descriptions of the patterns are considered. Therein, we apply inductive techniques to generalize findings and form a description of the corresponding domain. Together, the descriptions from literature and practice outline the domain of possible problems and contexts in which the design theory nexus instantiation can be applied. To conceptualize the class of problems and organizational contexts, from the specific ones we use the method of “extracting case studies” as introduced by van Aken [Ak04, page 232]. In reviewing existing literature in the field of EA management, we infer, at least as far as possible, the epistemological and ontological assumptions underlying the described cases to ensure epistemological clarity. If necessary and possible, certain conclusions or statements from literature might have to be reformulated with respect to the thesis’ research design.

During the activity **structuring the domains**, we establish a conceptual framework that helps to place the concepts from the different domains and reason on the completeness of the domains’ descriptions. Thereby, we use kernel theories to complement the enumeration of problems and contexts. The conceptual framework provides a high-level structure for the two domains in the sense of Nunamaker et al. in [NCP91, page 635], who advocate for the establishment of structuring principles as part of problem understanding. We further employ the framework to additionally explore relationships between different problems and contexts. The kernel theories used to structure the problem and context for designing an organization-specific EA management function are discussed in Section 3.1.

The activity **meta-conceptualizing the domains** aims at establishing a consistent terminology of meta-concepts that represent classifications of relevant concepts from the problem and context domain. The meta-concepts are based on the aforementioned structuring kernel theories and on relevant generalizations as presented in the case studies from literature and the observed and documented patterns from practice. The meta-conceptualization provides a consistent terminological basis and enables the communication of statements by employing inductive techniques to aggregate and generalize our findings.

As mentioned before, no defined ordering of the activities of the problem diagnosis exists. Instead they are typically performed in an iterative manner. Following a hermeneutic cycle, newly identified problems and contexts might cause a restructuring of the domains' framework and can call for an adaptation of the meta-concepts. This richer framework for the problem and context domain might in turn support the exploration of additional problems and contexts. During the iterative problem diagnosis, we especially account for the aspect of *relevance*. Thus, relevance can be approached from two sides—relevance with respect to the overall problem and context domain and relevance with respect to scientific research. Exploring the former aspect, one must keep in mind that any identified problem or context can extend the problem and context domain and might change the subject of the design theory nexus instantiation. Especially in the context of designing organization-specific EA management functions, which forms a topic best to be approached in a multi-disciplinary manner, this may lead to an all-embracing problem domain, which is extremely difficult to handle. Complementingly, relevance of a newfound problem has to be assessed with respect to prevalent scientific research. In other words, such a problem should form a “wicked problem” in terms of Rittel and Weber [RW73], i.e., a problem in need for a (more effective) solution.

### 2.3.2. Nexus instantiation

In line with our understanding of designing EA management functions as wicked problems, to which a design theory nexus can be applied, this activity refers to the instantiation of the nexus and the pattern-based theory building process. Based on Venable's theory building activity [Ve06a, page 185–186] and the process to construct a design theory nexus instantiation of Pries-Heje and Baskerville [PHB08, page 737–738], this phase is composed of the activities of **eliciting method requirements**, **investigating solution models**, **meta-conceptualization the solution domain**, and **formulating assertions**. Subsequently, we describe each activity in detail and relate them to the procedure for pattern-based theory building.

In the activity **eliciting method requirements** the development method and the underlying design theory nexus instantiation are subject to further specification from a user perspective. In particular, the expectations of the target users, i.e., enterprise architects, with respect to the application of the development method and the creation and maintenance of the method base is analyzed. As far as possible this analysis is performed against the background of existing ad-hoc methods in this field. In addition, relevant theories and general requirements on methods, method fragments, and a method base are revisited to elicit requirements.

The activity **investigating solution models** is concerned with revisiting the solutions described by the identified design theories and observed and documented patterns. We again

use extracting case studies (cf. van Aken in [Ak04, page 232]) to re-read existing literature on designing EA management functions or constituents thereof. Complementing the theoretic investigation, we use the pattern-based method for theorizing to incorporate occurrences of best practice solutions. The identified solutions are gathered and used to develop solution models, i.e., competing design theories for the design theory nexus instantiation. Similar to the activity of problem diagnosis, we apply kernel theories to structure and investigate the solution domain. This helps to provide a more generic understanding of the exemplary solutions gathered in the exploration activity. In this respect especially theories from bordering fields, e.g. (situational) method engineering, provide valuable input. The kernel theories of the solution domain are discussed in Section 4.2. In the words of Hevner et al. [He04, pages 88–90], the aforementioned process to develop appropriate solution models can be regarded as a search process that iteratively broadens the search space to identify the most suitable solution models.

During the activity **meta-conceptualizing the solution domain** we develop a conceptual framework containing the meta-concepts to describe an EA management function complementing the meta-conceptualizing of the problem diagnosis. We establish a basis for understanding the solution models and the corresponding assertions with respect to the context and problem descriptions. The meta-concepts provide a common terminological basis, which is particularly important as the different solution models can originate from various sources thus also employing different or even conflicting terminologies. This activity relates to the final step of our pattern-based method, which is concerned with deriving terminological compatibility relationships between patterns. The developed meta-conceptualization lays the basis to compare, combine, and integrate the distinct solutions. Building the basis for the decision making process of the design theory nexus instantiation, the meta-conceptualization further acts as a conceptual framework in terms of Nunamaker et al. in [NCP91, page 635].

In the activity **formulating assertions** the solution models described using the meta-concepts are interlinked via assertions. Thus, we establish three distinct types of assertions. The **context and goal assertions** link the solutions to the problems and contexts identified. We operationalize the assertions delineating applicability of particular solutions for dedicated problem and contexts as specified by the design theories and patterns. The **solution assertions** make explicit the dependencies and relationships between solutions, e.g. successor- or predecessor relationships, based on the meta-concepts of problem, context, and solution domain. Complementing, the **general assertions** reflect the requirements that any possible solution for a particular problem needs to fulfill, i.e., describe beneficial and intended characteristics of a general solution. The general assertions represent the requirements elicited in the requirements elicitation

Again not only no defined ordering in which above activities should be carried out exists, but Venable explicitly votes for an iterative process [Ve06a, page 185]. We propose to use a hermeneutic method. Especially for the activities of investigating solution models as well as formulating assertions and meta-conceptualizing an iterative method to foster the alignment between the streams is beneficial. An iterative procedure is especially useful as the development of new solution models and the formulation of assertions can call for the design of more sophisticated meta-concepts. Complementing the domain knowledge, we employ kernel theories to facilitate the reassessment of already developed artifacts against the background of additionally grounding theories, thus leading to improved artifacts.



### 2.3.3. Application and evaluation

In line with the argumentation put forward by Sein et al. [Se11, page 3], we advocate for a stronger integration between the design research activities of building and evaluating. The development method, more precisely its related solution models and techniques from the nexus instantiation, are subsequently applied and evaluated in different ways. First, we **develop a nexus-based IS generator** supporting the development method. In this vein, we perform the final step of constructing a design theory nexus instantiation and follow the argumentation of Walls et al. [WWES92, page 46], who advocate for the creation of a tool to show applicability. Thus, we evaluate the nexus instantiations prescriptions and assess the fulfillment of the design theory nexus requirements elicited during the nexus instantiation. Second, the design theory nexus instantiation is applied and evaluated in practice. Partnering organizations apply the development method to design an organization-specific EA management function suitable for their specific problems and organizational contexts. Thus, the research outcomes are evaluated in practice. Finally, the different assertions established with respect to the solution models and their interplay with the problem and context domain are **theoretically evaluated**. The theoretic evaluation is performed by using formal and informal arguments which show that the assertions necessarily hold for all EA management functions created using the development method and the underlying design theory nexus instantiation.

#### 2.3.3.1. Developing an nexus-based IS generator

The activity **developing a nexus-based IS generator** is concerned with implementing the development method and the underlying design theory nexus instantiation for EA management into a prototypic tool. This tool supports enterprise architects in using the problem and context descriptions as well as the assertions to develop an organization-specific EA management function thereby enabling not only application but also evaluation thereof. The justification and evaluation of the resulting artifact should according to Pries-Heje and Baskerville [PHB08, page 742] focus on whether the artifact operates in the problem setting. A tool can be developed based on an adapted system development method as presented by Nunamaker et al. in [NCP91, page 635–637]. Taking into account the activities of meta-conceptualizing the problem and solution domains as illustrated before, the first step of the method of Nunamaker et al. can be regarded as completed, such that the activity of tool development can start with the “development of the system architecture” [NCP91, page 636]. Therefore, user requirements regarding tool support for the development method are elicited and derived from the description of the problem, context, and solution domain. Via building the system, the feasibility of the development method as well as the inner consistency of the underlying design theory nexus instantiation and its meta-conceptualization is shown.

#### 2.3.3.2. Application and evaluation in practice

Reflecting the relevance of the research topic in practice, we apply the development method and the underlying design theory nexus instantiation in a practical setting, i.e., real cases with industry partners participating in the research project. The activities of **nexus-driven**

**problem elicitation**, **nexus-driven artifact design**, and **nexus-driven evaluation** as defined by Venable [Ve06b]<sup>5</sup> are performed.

During the activity **nexus-driven problem elicitation**, we use the meta-conceptualization of the problem domain to elicit requirements for the development of an organization-specific EA management function in a structured manner. The elicitation is performed on the one hand at an organization willing to establish an EA management function in a developing case study [Ak04, page 232]. On the other hand, we complement the aforementioned case with an ex post analysis in an organization that has already established an EA management function. While the former case represents the ‘usual’ application of the development method, the latter approach reflects aspects of adapting, enhancing, or improving already existing EA management functions. It encompasses the subtle complexity that many EA management functions are established in an ad hoc manner without preceding problem elicitation leading to functions that do not explicitly delineate the problems they are intended to address (cf. our discussions in [Bu07d, Bu09b]). As discussed by Schermann et al. in [SBK07a], the problem-solution relationship can be used to drive the elicitation phase. In this respect, the nexus-driven problem elicitation should not be considered as a simple mapping from the situated EA management problem of the organization to the problem domain of the design theory nexus instantiation. In contrast, the problem elicitation explores the problem and context domain described by the nexus instantiation also via relationships to corresponding solution domain concepts in an inspired and iterative manner.

In the activity **nexus-driven artifact design** we develop an EA management function based on relevant solution models, which are selected according to the precedingly elicited EA management problems and organizational contexts. Thereby, the context and problem as well as the solution assertions as provided by the design theory nexus instantiation are used to identify appropriate solution models. The principles of implementation of these solution models are enacted into a coherent development method. In this vein, we develop an initial organization-specific EA management function based on the meta-conceptualizations underlying the solution domain, which is in a second step refined and adapted to the specific environment of the organization, e.g. the participant descriptions are mapped to organization-specific roles. Again, we perform the aforementioned steps both at an enterprise willing to establish an EA management function, i.e., in a developing case study, as well as ex post at an organization, which has an already established EA management function. In the former case, we execute the ‘interactive’ elements in the principles of implementation in cooperation with stakeholders from the enterprise. In the latter case, both the development method and the design artifact are analyzed after the fact. Therefore, the necessary input for the assertions is reverse engineered from design documents and existing method descriptions as well as from interviews with the involved stakeholders. As a byproduct to the implemented solution, we produce a **design rationale** as a second outcome, which details the decisions and assumptions that have led to the final design. This rationale is of interest for the organization under investigation as it can be helpful for the future evolution of the management function. The EA management functions developed in this activity serve as **expository instantiations** (cf. Section 2.2.1) of the design theory nexus instantiation. The nexus-driven artifact design can further be supported by the nexus-based IS generator as described before.

---

<sup>5</sup>Venable originally used the term *theory*.

During the activity **nexus-driven artifact evaluation**, we evaluate the designed EA management function with respect to its design artifact qualities, formulated as general assertions on the solution. Therefore, we use qualitative and observational techniques to evaluate the artifact based on different quality attributes. The design theory nexus instantiation and the selected design theories provide helpful input for the evaluation, e.g. via the selected problems. In addition to the findings from the developing case study, stakeholders of the EA management function in the ex post approach can revisit the design decisions documented in the design rationale and provide subjective statements on the quality attributes. In line with Gehlert et al. [Ge09, page 449] the aforementioned qualities can be regarded as *outcome-oriented* evaluation targeting the utility of the design artifact.

According to Hevner et al. [He04, page 85] an artifact can be evaluated with respect to the quality attributes “functionality, completeness, consistency, accuracy, usability,” and “fit with the organization”. The quality attributes functionality, completeness, and fit with the organization relate to the *outcome-oriented* evaluations of Gehlert et al. [Ge09, page 449] and can be assessed by observation. The outcome-oriented evaluation tests, if the solution models are suitable and comprehensive with respect to the intended problem and context, thus providing valuable statements helping to refute or support *testable solution model hypotheses*. Completeness is achieved, if the design artifact “satisfies the requirements and constraints of the problem it was meant to solve” [He04, page 85]. Evaluating consistency and accuracy is part of the *output-oriented* evaluation, which may help to assess the appropriate instantiation of the solution model elements, the quality of the **principles of implementation** and can contribute to refuting or supporting the **testable implementation hypotheses**. While the output- and outcome-oriented evaluations target the design artifact, the complementing theory-oriented evaluation advocated for by Gehlert et al. [Ge09, page 449] directly aims at evaluating the design artifact with respect to its underlying theory or the design theory nexus instantiation. The latter evaluation is discussed in the next section as it contributes to an evaluation from a theoretic point of view.

### 2.3.3.3. Theoretic evaluation

From the perspective of the design theory nexus instantiation, the designed EA management function can be referred to as **design artifact**. In the theoretic evaluation activity, we evaluate the development method and the underlying design theory nexus instantiation from a utilitarian perspective, i.e., we evaluate whether the development method is usable for creating appropriate products. From a theoretical point of view, our evaluation is twofold: First, we revisit the general research guidelines as proposed by Hevner [He04, pages 82–90] and second, we investigate the different types of assertions formulated during the nexus instantiation activity. Therein, we theoretically analyze whether an EA management function designed using the development method fulfills the general assertions ‘per design’. In particular the latter evaluation refers to a theory-oriented evaluation type according to Gehlert [Ge09, page 449], which analyzes whether in applying the theory an anomaly has arisen and the theory is suspect. In this sense an anomaly might be any kind of inconsistency occurring during the application of the theory’s prescriptions. Possible anomalies are an anomaly in the creation of a method for situated EA management problems based on the theory’s meta-conceptualizations, an assertion which does not hold, or an ambiguous or infeasible design step described in the de-

velopment method or in the principles of implementation. While the occurrence of an anomaly can help to refute implementation and solution model hypotheses, the absence of anomalies hints towards the consistency of the development method, the underlying design theory nexus instantiation, as well as the incorporated assertions and design theories.

Although the straightforward way, i.e., a successful application of the design theory nexus instantiation to create an EA management function solving situated problems, supports the assumption of a useful development method and underlying design theory nexus instantiation, different *levels of indirection* call for additional evaluation techniques. This necessity can concisely be explained along the different types of evaluation methodology presented by Verschuren and Hartog [VH05, pages 739–741]. They distinguish between evaluating the

**plan** assessing the “quality of the design on paper” [VH05, page 739] by performing a logical, ethical, and empirical check of the design requirements, the design assumptions, and the structural specifications, the

**process** which is to “improve the process and via this the product, of designing” [VH05, page 739] via qualitative methods, e.g. empirical ones based on sensory observation, and the

**product** which is concerned with figuring out “whether the artifact helps in achieving the goal” [VH05, page 740], thus a product evaluation is usually mostly *summative*, i.e., outcome-centric concluding in statements like satisfactory or unsatisfactory [VH05, page 741].

As already reflected in the above distinction of evaluation types, Verschuren and Hartog [VH05, pages 749–760] advocate for a staged evaluation during the development method. Performing a ‘backtracking evaluation’, i.e., an evaluation of plan, process, and product evaluation in reverse order, we aim at a more “formative” [VH05, page 748] perspective. We not only sense deviations from the expected outcome, but also aim at discovering inappropriate or incorrect steps in the development method or even in the plan that might have caused the deviation. The distinct types of assertion, i.e., context, goal, solution, and general assertions, of the design theory nexus instantiation relating solution models, problem and context descriptions are subjected to an argumentative evaluation process. Using the design theories and patterns from which the solution models, context, and problem descriptions have been derived, the assertions are formalized in a first step. Therein, the design theories and patterns are considered as already evaluated, i.e., are assumed to make correct prescriptions. In the second step, the development method is traversed backwards. With a potential design artifact in mind, the single steps of the development method are revisited to investigate the assertions from a pre-conditions and post-condition perspective. Thus, we analyze which post-conditions of the intermediary design artifact are expected to hold at some point within the method and investigate the respective pre-conditions for the technique applied in the step. Finally, the methods for developing and administering the design theory nexus instantiation are evaluated based on the findings from the aforementioned investigation. We analyze, if the integration of a correct design theory nexus instantiation and a correct design theory, results in a correct combined nexus instantiation.

As mentioned above the theoretic evaluation is concerned with the identification of design anomalies. To perform the theoretical evaluation, unexpected and unwanted behavioral and structural properties of the design artifact as detected during the nexus-based artifact evalua-

tion are described via instantiations of the meta-concepts. In a second step, the design theories constituting the design theory nexus instantiation are searched for prescriptions targeting the corresponding meta-concepts, as specifically also are the testable hypotheses that affect the meta-concepts under consideration. From there, the solution models and the assertions are traced to the respective context and problem prescriptions, which are subsequently analyzed with the complementing solution model hypotheses. The hypotheses are complementingly traversed in *inverse causal direction* from the effects to the causes, to seek for logical implications that do not hold. During this traversal at first the testable implementation hypotheses have to be analyzed and prescriptions from the principles of implementation have to be assessed. If prescriptions as well as their underlying assumptions turn out to hold, and the hypotheses cannot be refuted based on the application case evaluation, the traversal has to move on to the level of the plan. Again the assumptions backing the solution model and the context and problem descriptions underlying the plan have to be tested and support or refutation for the solution model hypotheses should be looked for.

## 2.4. Summary of the research design

In this chapter we discussed the research design underlying this thesis. According to the framework of Becker et al. [Be03, page 5] the research assumptions underlying this thesis, the intended research outcomes, and the research method used are explored. Figure 2.10 provides an overview on the ontological, epistemological, and linguistic assumptions underlying this thesis, which have been motivated and justified in Section 2.1. Summarizingly, the positions underlying this thesis are based on the ontological assumption of kantianism, accounting for the existence of non objective elements, i.e., strategies, processes, or guidelines. This position is reflected by the epistemological assumption of the thesis, which can be summarized as *consensus-oriented constructivism*, which is complemented by the methodological structure of the hermeneutic cycle, which addresses the complexity of the research subject—the EA management function—by the utilization of different methods (cf. arguments for multi-method research in [Ni06a, page 70]).

The objective of this thesis is to contribute to the field of developing organization-specific EA management functions. We seek to establish a development method that builds on practice proven solutions for developing EA management functions. This thesis focuses in particular on the methodical aspect of the EA management function. The development method should further be implemented in a corresponding tool-set supporting the users in developing an organization-specific EA management function. Understanding the design of organization-specific EA management functions as a wicked problem for which a multitude of practice proven solutions for specific solutions and dedicated organizational contexts exist, we outline how prevalent method prescriptions can be used, combined, and integrated to support the development of an organization-specific EA management function. Therefore, we revisit the notion of *design theory* as introduced by Walls et al. [WWES92] and furthered by Gregor and Jones [GJ07], discuss the role of *patterns* and *pattern languages* as described by Alexander et al. [Al77], and apply the concept of *design theory nexus* as proposed by Pries-Heje and Baskerville [PHB08] to link the competing solutions and enable their selection and integration. Referencing back to the above mentioned tool support we advance the idea of *IS generators* as discussed by Walls et al. [WWES92] and the idea of a decision support system supporting the

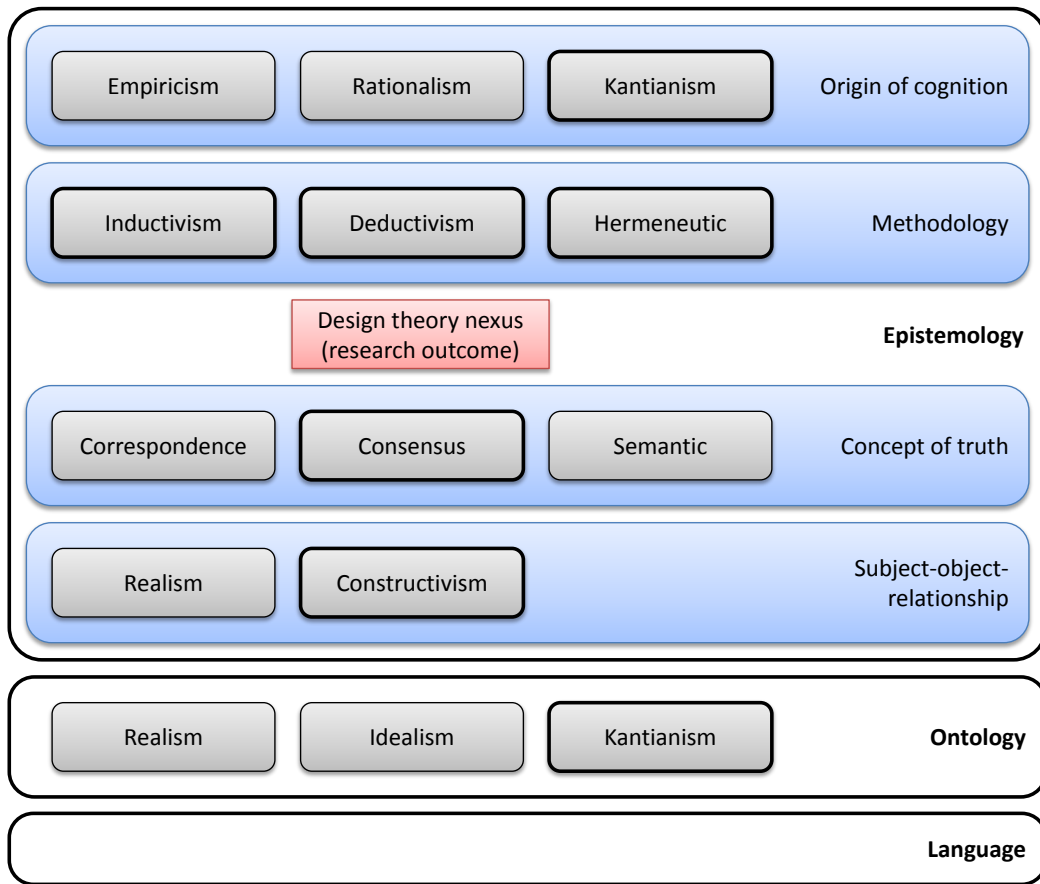


Figure 2.10.: The epistemological, ontological, and linguistic assumptions underlying this thesis

application of the design theory nexus as discussed by Pries-Heje and Baskerville [PHB08] and introduce a nexus-based IS generator supporting the users in the application of the development method for organization-specific EA management functions. Recapitulating, this thesis aims at the following five distinct research outcomes as discussed in Section 2.2 contributing to a development method for EA management

- the **meta-conceptualization** for describing the context, problem, and solution domain,
- a **design theory nexus instantiation**,
- a **method to develop and maintain the design theory nexus instantiation**,
- a **method and techniques for applying the design theory nexus instantiation**,
- a **nexus-based IS generator**

Table 2.2 displays how the research outcomes relate to the contributions of the thesis as introduced in Section 1.2. The symbol (●) denotes a direct relationship between the research outcome and the corresponding contribution of the thesis, while the symbol (○) in contrast is used to indicate that the research outcome is not the primary source of the contribution. The

structure of the thesis as presented in Section 1.3 follows the subsequently listed contributions (see Chapter 4 to 6). The method to develop and maintain as well as the method to apply the design theory nexus instantiation are subsumed in Chapter 5. Following Popper’s understanding of science as the process of trying to refute theories [Po80], we in line with Schermann et al. in [SBK07a] test our prescriptive statements of the theories through the development of corresponding instantiations in Chapter 7.

	Method modeling language	Method building blocks	Configurat techniques	Adminis- tration method	Configu- ration method	Prototypic tool
Meta- conceptualization	●		○	○	○	○
Design theory nexus instantiation		●				○
Method to develop and maintain the nexus instantiation				●		○
Method and techniques to apply the nexus instantiation	●		●		●	○
Nexus-based IS generator						●

Table 2.2.: The contributions of this thesis and their relations to the research outcomes

Complementing the research assumption and the research outcomes, Section 2.3 presents the the research method applied in this thesis to achieve the aforementioned research outcomes. The research method of this thesis represents a synthesis of the activity framework of Venable [Ve06b], the five-step approach to construct a design theory nexus instantiation as described by Pries-Heje and Baskerville [PHB08], and our pattern-based theory building process [BMS10k]. The synthesis results in a three step method containing the activities problem diagnosis, nexus instantiation, and application and evaluation. Table 2.3 shows the contributions of the aforementioned three sources to our synthesized method.

	Activity framework [Ve06a, Ve06b]	Nexus instantia- tion [PHB08]	Pattern- based theory building
Problem diagnosis	Problem diagnosis	Step 1 & 2	Step 1 & 2
Nexus instantiation	Theory building	Step 3 & 4	Step 3 & 4
Application and evaluation	Theory application Theory evaluation	Step 5	

Table 2.3.: The research method of this thesis and contributing works





---

## Analyzing the State-of-the-Art in Designing EA Management Functions

---

The research objective outlined in Section 1.1 needs to be critically reviewed against the current knowledge base of EA management. The development of a design theory nexus instantiation for EA management, requires up-to-date information on the state-of-the-art to ensure, that the artifact designed and produced is sufficiently innovative [VH05, page 749]. In this sense the subsequent research synthesis can according to Verschuren and Hartog in [VH05] be seen as a first evaluation stage. As a plethora of different approaches to design the EA management function exist, the aim of this chapter is to provide a firm foundation for advancing knowledge by summarizing the state-of-the-art. Thereby, theory development is facilitated by closing areas where a plethora of research exists and by uncovering areas where research is needed. Therefore, this chapter aims at performing an integrative research review targeting existing EA management approaches and their configurability to organization specificities—an area where according to our experience present approaches lack support.

To build on the existing knowledge base, we investigate the state-of-the-art in EA management literature with respect to the proposed methods and typical configuration aspects relevant to design an EA management function. Thus, we answer research question 2 (see Section 1.1). The literature review uses a systematic approach, which is based on the method of hermeneutic text comprehension as proposed by Gadamer in [Ga75] and follows the guidelines for literature reviews promoted in [WW02]. In line with Webster and Watson in [WW02, page xiv] and Bem in [Be95, page 174], we assume that “a coherent review emerges only from a coherent conceptual structuring of the topic itself”. Therefore, relevant kernel theories from related disciplines are presented subsequently and their impact on the context of designing an EA management function is discussed. Based on these kernel theories we develop an analysis framework, which builds the conceptual and cognitive background for the literature review. Further, we detail the scope of the literature analysis to fulfill the guidelines as proposed by Webster and Watson in [WW02].

Starting with an introduction to model theory in line with Stachowiak in [St73] and the ISO Std. 42010 [In07], which provides a terminological basis for architectural descriptions, terms

frequently used throughout this thesis are defined (cf. Section 3.1.1). we further call on different theories to establish distinct perspectives on EA management, namely

- a *design* perspective presented in line with Simon [Si96] (see Section 3.1.2),
- a *management* perspective discussed in the sense of Deming [De82] or Shewart [Sh86] (see Section 3.1.3),
- a *knowledge management* perspective derived from Probst [Pr98] (see Section 3.1.4), and
- a *systemic* perspective discussed in line with Beer [Be79, Be81, Be85] (see Section 3.1.5).

We briefly summarize these kernel theories in the following section and discuss their influence on the design of an EA management function. Based thereon analysis criteria for the design of an organization-specific EA management function are derived, which we use to establish a conceptual evaluation framework in Section 3.2. In Section 3.3 prominent approaches to design an EA management function are revisited against the background framed by the kernel theories. The results of the analysis are summarized and discussed in Section 3.4.

## 3.1. Fundamental perspectives on methods for EA management

Kernel theories govern the design requirements as well as the development method and therefore provide valuable input for an analysis framework to investigate the state-of-the-art in a discipline. Development, design, and maintenance of architectures have a long history in the engineering disciplines. Primary originating from construction engineering the objectives of architecture—to be strong or durable (*firmitas*), useful (*utilitas*), and beautiful (*venustas*) [Po96]—and their means have been applied to other disciplines to address challenges in related domains (cf. Freestone in [Fr00] as well as Reussner and Hasselbring in [RH06]). One of these disciplines is EA management. Architecture, thereby, is an *intrinsic* property of a system (the enterprise) that cannot be neglected, while it, in contrast, is not necessarily documented, i.e., made explicit. This idea yields a delicate but central distinction of the two terms **architecture** and **architectural description** that will reverberate through the remainder of this thesis.

### 3.1.1. Modeling perspective: Model theory and the ISO Std. 42010

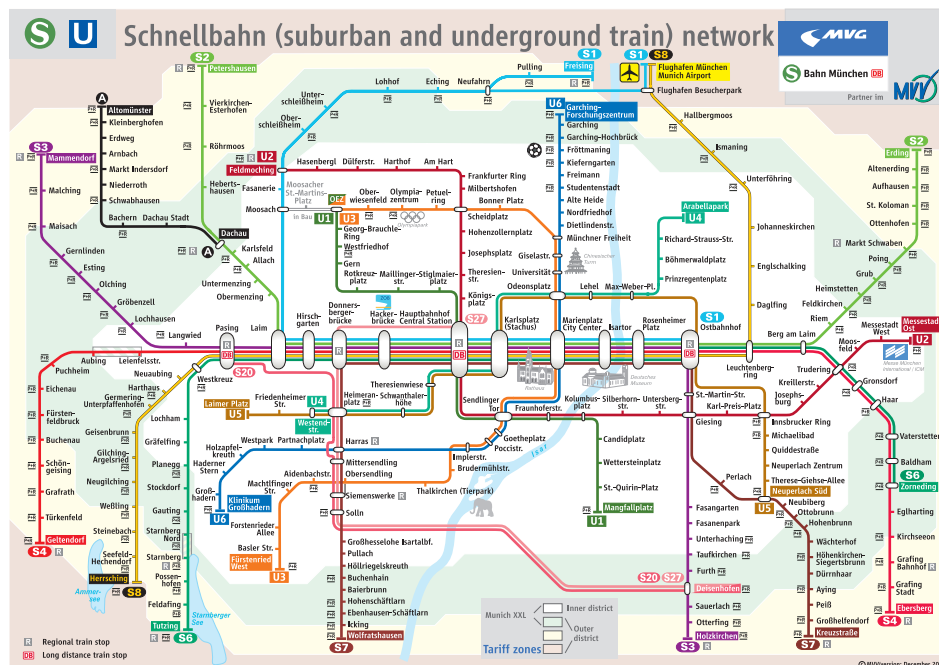
In 2007 the *International Organization for Standardization* (ISO) and the *International Electrotechnical Commission* (IEC)<sup>1</sup> provided a conceptual framework and terminology for discussions on architectural descriptions, the *ISO/IEC 42010: Systems and software engineering—Recommended practice for architectural description of software-intensive systems*. The concepts introduced by the ISO Std. 42010 are discussed and defined in the following alongside an example from city planning, an analogy frequently used in the context of EA management.

---

<sup>1</sup>The standard was initially released by the *Institute of Electrical and Electronics Engineers* (IEEE) in 2000 [IE00].



**Example 3.1: Interests in a system.** A city can be understood as an evolving and networked system of semi-autonomous systems with an unbounded lifetime. People are involved in the system as they create, manage, and finance it. Distinct people thereby have different interests in the system, e.g. a city major is interested in the crime rate, birth rate, the average age, while a merchant is interested in the traffic infrastructure or the purchasing power of the residents, and the public utility company is concerned with the power supply system or the sanitation. Specialized city maps, which address the above stated interests, typically exist (see subsequent figure, which shows the public transport system of Munich).



Specialized city map: The public transport system of Munich (Source <http://www.mvv-muenchen.de> (last accessed 2010-08-25))



An enterprise in analogy with a city can be understood as a system, i.e., a “collection of components organized to accomplish a specific function or set of functions” [In07, page 3]. In line with the above example, an enterprise does not only consist of ‘things’, but involves the people, e.g. employees, managers, or customers. To enable communication and discussion on the construction of the organization among these people, a common conceptualization, i.e., **model** of the enterprise, has to be developed. For our subsequent discussions it is important to distinguish between two different kinds of models—*mental* models, which relate to a person’s

intuitive understanding and perception of the things (noumena), and *explicit* models<sup>2</sup>, which represent conceptualizations made explicit via a defined modeling language (cf. our discussion in [BKS10]).

**Definition: Model**

A model according to Stachowiak in [St73] has three essential characteristics:

**Representation** A model is always a model of something, namely a surrogate or representation of natural or artificial originals, which can be models themselves.

**Reduction** A model commonly does not capture all attributes of its corresponding original, but only those, which are relevant for the model creators and/or model users.

**Pragmatism** Each model is made for a distinct time frame, a dedicated purpose, and certain users.

Exemplifying the characteristics along our city planning example, the model of the public transport system is considered in more detail. While this plan represents the reality as is in our city, it simplifies reality, e.g. by skipping details on buildings, roads, or the natural environment. By doing so, the model is dedicated to describing the currently existing public transport system, i.e., is dedicated to potential users. These people, can thereby be regarded as **stakeholders**, who have a certain **concern**, i.e., the how to get from point A to point B, regarding the system. According to the ISO Std. the concepts stakeholder and concern are defined as follows (cf. [In07, page 3]).

**Preliminary definition: Stakeholder**

An individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system.

**Definition: Concern**

Concerns are those interests which pertain to the system's development, its operation or any other aspects that are critical or otherwise important to one or more stakeholders

During EA management a variety of different documentations, so called 'views', of the system under consideration, i.e., the city map of our above analogy, are created and used to facilitate the communication between the involved stakeholders, e.g. application owners, CxOs, or enterprise architects. Each view is an instance of a **viewpoint**, which defines a perspective on the system, i.e., a selection of concepts relevant to a specific stakeholder, and their representation, i.e., symbols or texts. Both terms—view and viewpoint—are defined in line with the ISO Std. as follows (cf. [In07, pages 3–4]):

**Definition: Viewpoint**

A specification of the conventions for constructing and using a view. A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

---

<sup>2</sup>Whenever we use the term model in the following, we refer to an explicit model.

**Definition: View**

A representation of a whole system from the perspective of a related set of concerns.

Architectural descriptions representing the architecture of a system or parts thereof are typically used in the engineering disciplines to plan, develop, maintain, and manage complex systems. Similarly, existing approaches to EA management typically make explicit the artifacts, which are produced by the management function. In line with the ISO Std., we provide the subsequent definition of the term architectural description (cf. [In07, page 3]):

**Definition: Architectural description**

A collection of products to document an architecture.

Figure 3.1 provides an slightly adapted excerpt<sup>3</sup> of the conceptual model of the ISO Std. 42010 illustrating the relevant concepts as introduced above. In [BKS10], we provide formal definitions for the concepts concern, view, viewpoint, and architectural descriptions, which we will make use of in Chapter 5.

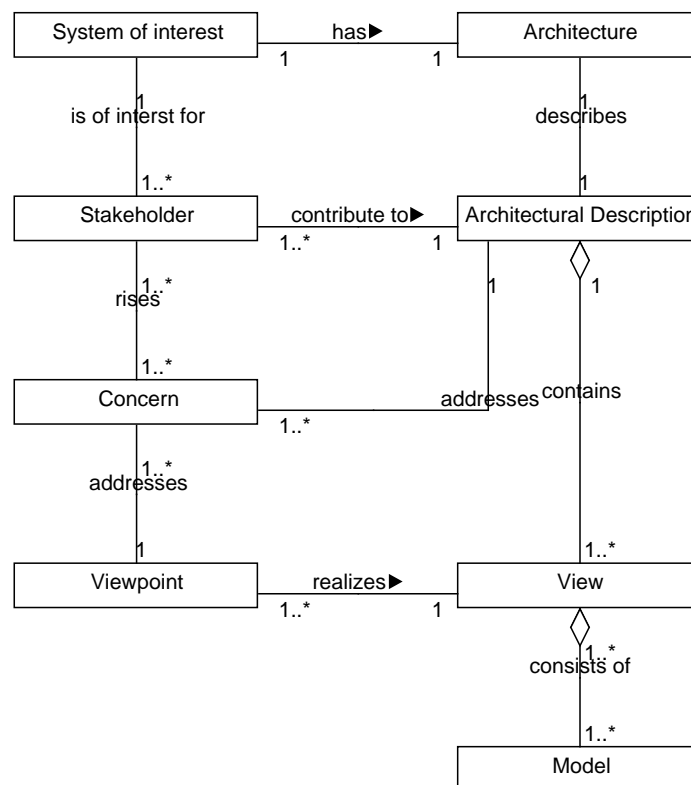


Figure 3.1.: Excerpt from the conceptual model of the ISO 42010 [In07]

<sup>3</sup>In the ISO Std., multiplicities are not provided for all associations. We adopted the illustrated version according to the textual descriptions.

#### 3.1.2. Design perspective: Simon's the sciences of the artificial

In [RV08] van der Raadt and van Vliet call for understanding EA management as a design activity targeting the enterprise in a comprehensive manner. One of the most prominent and early works discussing design activities was written by Simon in [Si96], who provided a more formal understanding of the activity based on the notion of *mean-end-relationships*. Rephrasing this understanding, any design activity is subject to domain-specific characteristics and relationships, so-called “natural laws” [Si96, page 3], that define how to connect dedicated means to corresponding ends. Transferring this understanding to the context of EA management, enterprise architects (*designers*), with a planned state (*end*) in mind, search for the *means* by which the EA will achieve those aims under the natural laws pertaining to the socio-technical system ‘enterprise’ as well as under artificial design constraints imposed by existing governance structures. As part of this search the architects develop different scenarios of intermediary planned states of the EA, which are evaluated with respect to the achievement of the desired end. The design activity may thereby be understood as a purely ‘mental’ one operating on a *mental model* of the enterprise also incorporating the according means-end-relationships. In [Si96, pages 115–118] Simon calls for a more formal understanding of design involving an imperative style of logic. In particular, he proposes to operationalize the means-end-relationships behind any design problem into logical statements relating

*command variables* describing objects (architecture elements) that may be changed by design activities,

*fixed parameters* describing architectural properties as well as environmental aspects that cannot be changed by design activities,

*constraints* limiting the space of changes that can be made by a design activity, and a

*utility function* evaluating a designed architecture with respect to the (experienced) utility for its stakeholders.

In terms of Simon the design of planned state scenarios to pursue can be reformulated as “given the constraints and fixed parameter; find values of the command variables that maximize utility” [Si96, page 117]. Every enterprise architect (as designer of the EA) uses a mental model of the enterprise to plan and evaluate the corresponding design alternatives. This mental model covers a specific concern in the overall architecture of the enterprise. Such concern can on the one hand be identified with a specific *conceptualization* of the enterprise, i.e., with a problem- and designer-specific classification of relevant elements of the enterprise, and on the other hand commits to a specific *filter* determining which parts of the enterprise are considered relevant [BKS10]. Two mental models as employed by two enterprise architects can hence differ with respect to both the conceptualization, i.e., the classification of elements, and the filtering, i.e., the selection of elements. To form the basis for a collaborative EA management conducted by a group of enterprise architects and other stakeholders, these people have to agree on a shared conceptualization. They have to be in one *linguistic community*, in terms of Kamlah and Lorenzen in [KL67], to be able to communicate their architecture understanding and collaboratively design as well as evolve the EA.

In the context of EA-related design activities more formal conceptualizations of the enterprise are widely used to facilitate communication between different stakeholders. These conceptual-

izations are reflected in corresponding EA modeling languages, more precisely their underlying **information models**<sup>4</sup>. These models are conceptual models committing to an agreed conceptualization of the EA or parts thereof as relevant with respect to the stakeholders and their design problems. While a stakeholder perspective on the EA would be useful for structuring the management subject, yet no comprehensive list of EA stakeholders has been developed (cf. the discussion of Bender in [Be09b]). Therefore, we draw structuring principles from different sources while conducting the state-of-the-art analysis. The operationalization of Simon in [Si96, page 117] provides a valuable foundation. In [Bu10b], we explore how the constituents of EA-specific design proposition look alike and provide a framework, which defines and relates

- the *current state* of the EA (command variables),
- the *strategies* and *projects*, which change the EA via affecting different constituents thereof (means),
- the *visions* and *goals* describing the target state of the EA (ends),
- the *principles* and *standards*, which guide and constrain the evolution of the EA (constraints), and
- the *questions* and *metrics* assessing and evaluating a state of the EA (utility function).

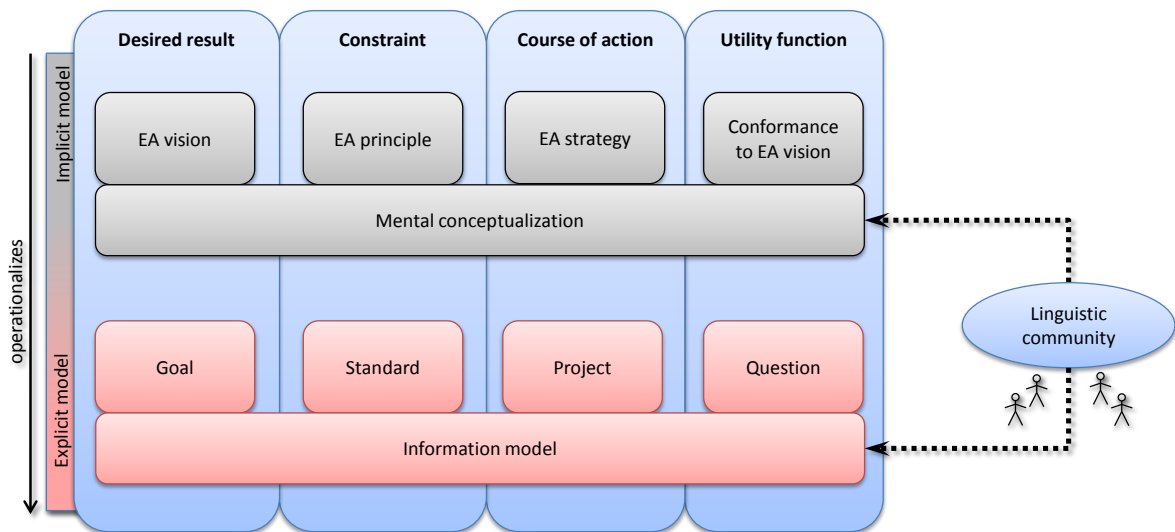


Figure 3.2.: Conceptual framework for EA design

Above framework builds on a distinction regarding the level of operationalization (**mental conceptualization** vs. **information model**). Figure 3.2 outlines this framework and shows how the different concepts can be related therein, which should accordingly be referred to by a proposed method to design an EA management function.

<sup>4</sup>The meta-models backing an EA modeling language are in line with our past publications named information models here (cf. [Bu07d]).

### 3.1.3. Management perspective: The Deming wheel or Shewart cycle

Approaching the topic of designing an organization-specific EA management function, from a management perspective as proposed by Harmsen et al. [HPK09, pages 162–163] or us [BMS10j] results in analyzing of which activities the “art of getting things done” consists (cf. quotation of Mary Parker Follett in van Aken [Ak05, page 26]). Although different definitions of and approaches to management have been proposed in academic literature (cf. Koontz and O’Donnell in [KO55], the St. Gallen Management Model in [UK72], or management by objectives [Dr06]), these approaches usually encompass the following activities: plan, decide, realize (via leading or directing), and control. Focusing on quality management, Deming [De82] and Shewhart [Sh86] proposed an additional phase—*act*—representing a meta-activity to improve the overall process. While abstaining from dedicating an own phase to decide, they propose a management cycle consisting of the steps *plan*, *do*, *check*, and *act* (PDCA cycle) also known as *Deming wheel* or *Shewhart cycle*. According to Deming in [De82, page 88] the PDCA cycle is incorporated in any transformation project. The single phases of the PDCA cycle are defined as follows. The *plan* step involves defining the problem as well as a hypothesis about possible causes and solutions as well as deciding which plan to accomplish. The *do* step covers the actual realization of the change and step *check* is concerned with observing the effects of the change. Final step *act* initiates the learning and feedback via studying the results and preparing the next iteration via improving the overall process.

Figure 3.3 illustrates the PDCA cycle emphasizing on the aspect of repetition, which according to Deming in [De82, pages 88–89] is a fundamental principle of continuous management. In other words, the final act phase is concerned with assessing the intended results of an iteration and improving the overall process via adapting the other phases. While the above definition may lead to an impression of sequentiality of the single phases, it has to be noted that in reality the phases are typically conducted in an overlapping manner. Mapping the different phases of the PDCA cycle to the context of EA management, the following main activities an EA management function consists of can be identified:

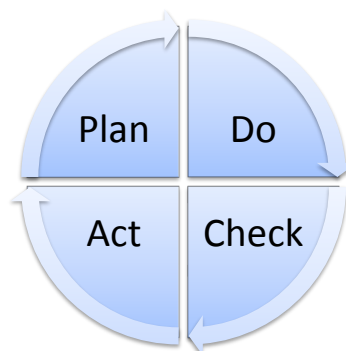


Figure 3.3.: The PDCA cycle according to Deming in [De82, page 88]

**Describe & develop** In this activity architectural descriptions of current, planned, and target states of the EA are described and developed. The development of the target state includes the establishment of architectural principles and standards to guide and constrain possible evolution paths of the EA. The current state of the EA describes the status quo and has to be updated regularly. The planned states describe the changes performed on



the current state as derived from possible project portfolio selections as well as roadmaps illustrating the transformation of the EA. The describe & develop activity corresponds to the *plan* activity of the PDCA cycle.

**Communicate & enact** In this activity EA artifacts are communicated and EA plans are enacted. Various ways how to enact exist, ranging from fairly non-interfering ways of informing and communicating via enacting to enforcing. While an EA management function realizes the developed plans via leading and directing, the communicate & enact activity reflects the *do* activity in terms of the PDCA cycle.

**Analyze & evaluate** In this activity parts of the EA (in terms of different states or concern-related excerpts) are analyzed and evaluated. Thereby, analyses can be performed to evaluate one state of the EA, e.g. the current state to identify potential for improvement as well as to measure the achievement of objectives. Similarly, planned states can be analyzed and evaluated regarding the strategic impact of the transformation planned. Furthermore, analysis and evaluation can be performed to compare two states of the EA, e.g. a delta analysis between the current state and a planned state or between a planned state and the target state can be performed. The analyze & evaluate activity mirrors the *check* activity introduced by the PDCA cycle.

**Configure & adapt** In this activity the EA management function itself is assessed and improved. To optimize the EA management function, the performance of the describe & develop, communicate & enact, and analyze & evaluate activities is assessed and the configuration of the different activities is adapted to better align to the enterprise's context and culture, as well as goals pursued by EA management. The configure & adapt activity corresponds to the *act* activity in terms of the PDCA cycle.

An EA management function has to contain all the aforementioned activities and has to provide suitable management methods.

#### 3.1.4. Knowledge management perspective: The cycle of Probst

Trends as globalization, downsizing, rapid change, and perhaps the most important one—the necessity of developing a company's sustainable competitive advantage have considerably increased the importance of *knowledge management* (KM) in organizations in comparison with the past (cf. [PS06]). The main goal of KM is making an organization aware of the knowledge it possesses so that it can make the most effective use of it (cf. Bennet and Bennet in [BB03, page 440]). According to Probst, “effective knowledge management creates sufficient internal and external transparency and supports employees in their knowledge-seeking activities” [Pr98, page 21]. Therefore, most KM initiatives in organizations pursue one of the following aims—making knowledge visible, developing a knowledge intensive culture, or building a knowledge infrastructure (cf. Davenport and Prusak in [DP00]). Hafner and Winter argue in [HW08, page 2] that an EA “serves as a transparent communication and design/evolution platform between the various IT stakeholders (e.g. application development sponsors in business and application developers in IT)”. This statement puts emphasis on a key similarity between EA management and KM as both disciplines involve information collection, communication, and exchange. In this sense, typical characteristics of an KM approach are likely to also apply in the context of EA management. Preparing more in depth analysis on this topic, we revisit

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

a KM approach in the following. In line with an advice of Probst in [Pr98, page 18], who stated that “While there is no single *right* model of knowledge management, there is a simple criterion for evaluating any model: how useful is it in relation to a chosen question?”, we select the most suitable one for our subsequent discussion<sup>5</sup>.

The KM cycle of Probst as presented in [Pr98, page 19] consists of eight typical activities that are carried out to avoid knowledge problems. As the cycle forms on the one hand a comprehensive model for KM and is on the other hand explained in very detail, it is subsequently sketched to provide the basis for the KM perspective on EA management. The KM cycle actually consists of the following two cycles, of which Figure 3.1.4 gives an overview:

- an *outer* cycle consisting of *goal setting*, *implementation*, and *measurement* as well as
- an *inner* cycle detailing the implementation activity into the sub-activities of *identification*, *acquisition*, *development*, *distribution*, *preservation*, and *use*

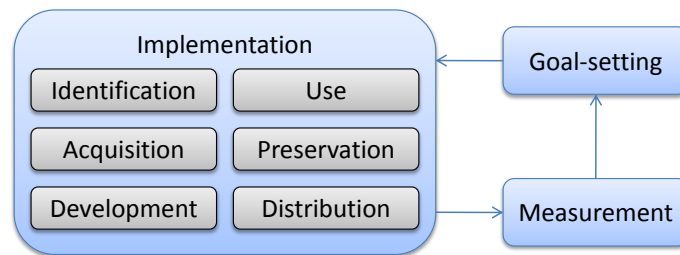


Figure 3.4.: The KM cycle of Probst (cf. [Pr98])

*Knowledge identification* is concerned with determining the knowledge that exists in an organization, and relating this to the knowledge existing in the organization’s environment. The activity increases transparency of knowledge, and may help to identify redundant as well as missing knowledge. *Knowledge identification* can, if the number of knowledge sources to process is abundant, resort itself to ‘critical’ knowledge as defined in the activity of *goal setting*.

*Knowledge acquisition* accounts for the fact that due to the growth of overall knowledge an organization is not capable to build up and maintain all needed know-how. Therefore, knowledge is imported over different ‘import channels’:

- acquisition of companies holding the corresponding knowledge,
- stakeholder participation, e.g. by involving the customers of the organization,
- counseling by experts that contribute to the organization’s knowledge, and
- acquisition of knowledge products that foster the development of new knowledge (does not directly improve the organization’s knowledge).

*Knowledge development* produces new knowledge on individual and collective level in a creative process, which can only to a very limited extent be discussed from a management perspective. Multiple sociological and psychological theories center around this activity and may be appropriate to study the process more in-depth. Linking back to the level of organizational

---

<sup>5</sup>See our findings in [BMS09b] for the discussion of the selection process.

KM and organizational development, for example an ‘atmosphere of trust’ in the organization is regarded as a prerequisite to effective knowledge development.

*Knowledge distribution* means making knowledge available across the organization. Put in the words of Probst, as stated in [Pr98, page 25], *knowledge distribution* is about the critical questions of ‘*Who*’ should know ‘*what*’, to ‘*what level of detail*’, and ‘*how*’ can the organization support these processes of knowledge distribution? These questions account for the fact that not everyone needs to know everything, as in contrast information overload might be as detrimental as a lack of information. Concerning the activity of knowledge distribution, the role of supporting tools and techniques should neither be underestimated nor overestimated. Useful and broadly accepted tools, and widely employed techniques can help to facilitate in the same ways as dysfunctional tools and not well adopted techniques can hamper effective *knowledge distribution*. As user acceptance is crucial for a tool or technique being an effective distribution facilitator, many organizational and non-technical issues have to be concerned regarding *knowledge distribution*.

*Knowledge use* forms the actual purpose of KM and refers to the application of knowledge in the production process of an organization. With respect to the later focus on EA management, which is no production process, the above statement can be reformulated as follows: *knowledge use* refers to the application of knowledge in the purpose-generating process of an organization. Here again, tools and techniques can be applied as facilitators; this is not surprising as especially in knowledge-intensive processes the borders between distribution and use are sometimes unclear. Notwithstanding, *knowledge use* should explicitly be accounted for, as the *goal setting* activity purposefully targets the use activity.

*Knowledge preservation* is concerned with avoiding the loss of valuable and purpose-relevant expertise in an organization. While tacit knowledge is more often subject to loss, e.g. due to an expert leaving, also explicit knowledge has to be preserved. Probst refers to outdated storage systems as ‘dead storage systems’, colloquially stating that a storage system, which is not longer maintained, may cause knowledge loss as well as a leaving expert. Techniques and tools used for knowledge distribution can also be helpful for knowledge preservation.

Complementing the inner cycle of *knowledge implementation*, two more activities constituting an embracing and sustainable KM exist. *Goal-setting*, i.e., , the development of knowledge goals, establishes a conceptual framework for organization-specific KM. The knowledge goals determine which capabilities should be built on which level. Different levels of abstraction with respect to the formulation of goals can be distinguished. Most important for the subsequent considerations are the levels of ‘strategic knowledge goals’ and ‘operational knowledge goals’. While the former goals describe a long-term vision of the knowledge portfolio of the organization, the latter goals operationalize the vision, i.e., , translate it into action. Making the knowledge goals explicit is regarded highly important to control the evolution of KM.

*Knowledge measurement* is concerned with measuring to which extent the knowledge goals have been fulfilled during the *implementation* activity. As knowledge is an intangible resource, indicators and measurement processes are hard to establish. To some degree the operational knowledge goals can be formalized such that they can help to objectively assess certain aspects of KM. Nevertheless, a commonly accepted way to measure knowledge has yet not been established, such that managers concerned with KM activities have to rely on their subjective perception of goal fulfillment. Additionally, surveys on user satisfaction with knowledge ac-

cess in distinct areas, which reflect certain knowledge goals, can be helpful during *knowledge measurement*.

Preparing the subsequent analysis, the KM model of Probst (1998), more precisely its activities, are mapped to the application domain of EA management. To ground the mapping in the application domain, the outer cycle's activities of KM are mapped, starting with the implementation activity. This activity can be identified with the core of EA management i.e., , with the “continuous management function seeking to improve the alignment of business and IT [...] of an organization”. This part of the preliminary definition of EA management (as discussed in Section 1.1) sketches the main goal of implementing EA management. Continuing with the activities from the outer cycle, both knowledge measurement and goal-setting can be identified with the aspect of ‘self maintenance’ of the EA management function, i.e., EA management governance. More precisely, an effective and continuous EA management, established as a management function within an enterprise, must define the part of the overall architecture of the enterprise that it covers. This can be understood as goal-setting, i.e., , defining which knowledge about the architecture is needed; multiple EA management approaches target this topic (for an evaluation of existing approaches see [BMS09b, BMS10e] or Struck et al. in [St10c]). The knowledge measurement closes a feedback loop in this respect by assessing to which extent the knowledge goals could be satisfied. Put in the EA management terminology, the measurement activity assesses, if the architecture concepts defined as relevant during goal-setting have adequately been considered. This provides input for revisiting the knowledge goals, if e.g. albeit a good coverage of relevant architecture concepts, an increased alignment between business and IT could not be achieved. In line with the KM perspective on the design of an EA management function, existing approaches should provide methods and means especially for the activities of the outer cycle of goal setting and measurement.

#### 3.1.5. A systemic perspective: The viable systems model

In line with Harmsen et al. in [HPK09, pages 168–170], Wegman in [We02], Pulkinnen in [Pu06], and the above definition of an organization as a socio-technical systems, i.e., a collective of human participants, processes, and technology jointly engaged in purposeful activity, we revisit the theories of the *viable systems model* (VSM) to derive analysis criteria for the state-of-the-art review.

The VSM, developed by Beer [Be79, Be81, Be85] provides a framework to describe complex systems that have to survive in a changing environment. According to Beer such systems consist of five interacting subsystems—*operation*, *coordination*, *control*, *planning*, and *identity*. The VSM has beneficially been applied in various contexts, e.g. project management [BP93] or organizational modeling (cf. [EH89, BC96]). The VSM can be used according to [BC96] as a tool to support an enterprise during the implementation of large scale organizational change. Whereas, a definition and description for each of the systems of the VSM is given in e.g. [Be79] no such common understanding about the constituents of the EA management function exists. Therefore, the five subsystems of the VSM are subsequently detailed in an EA management context and used to derive implications on the main constituents of an EA management function.

*System one*—operation—contains the primary activities of the system under consideration, which directly interact with the environment. In the context of EA management these pri-

mary activities should be identified with the enterprise-level management functions introduced in Section 1. The enterprise-level management functions form the system that changes the EA via projects, which have been initiated in the demand management, aligned in the strategies and goals management, selected in the project portfolio management, scheduled in the synchronization management, and realized with standards from the IT architecture management. A description of the function of EA management therefore must consider the role of related enterprise-level management functions.

*System two*—coordination—includes the information channels and bodies, which ensure that the primary activities of *System one* work harmoniously in coordination. EA management, as introduced above, provides a common basis and the means for communication between the various stakeholders with business and IT background involved in the enterprise-level management functions. Therein, especially visualizations to support communication are used and exchanged between the different enterprise-level management functions to coordinate their activities. All project proposals originating from the demand management for example, are used as input to create possible planned landscapes to prepare the project portfolio management [Wi07, Ma08, Bu09f, BMS09c]. Accordingly, a description for the EA management function should emphasize on the communication task.

*System three*—control—represents the structures and controls, which establish the responsibilities and rights to maintain the resource allocation of the operating system *System one*. Thereby, *System three* monitors the primary activities as well as the communication and coordination tasks of *System two* and adapts them according to the holistic view on the primary activities. If, for example, newly agreed standards from IT architecture management are not available for the project portfolio management, the projects considered therein cannot be checked for standard compliance. *System three* should therefore set up a structure e.g. an intranet, where the standards can be viewed and communicated to the corresponding stakeholders. *System three* can be referred to as reactive EA management and should be considered in the description of the management process.

*System four*—planning—contains the EA intelligence function. The system is concerned with a holistic and future-oriented perspective to support strategic decision making. Whereas *System three* is capable of dealing with immediate effects, *System four* focuses on future aspects, which emerge from the system's environment and also considers strategic opportunities, threats, and possible future directions. Typical processes in *System four* in the context of EA management include the analysis of the status quo of the architecture, the development of a target architecture representing the envisioned state in the future, and planning the transformation of the enterprise to pursue the target. Alongside the reactive aspect, an EA management approach must cover the aforementioned proactive aspect, containing a vision how a possible target enterprise should look like.

*System five*—identity—is responsible for managing the overall policy decisions. It should provide clarity about the overall direction, values, and purpose of the system under consideration. The main goal of *System five* is to balance present and future efforts, and to steer the system as a whole. In the context of EA management, *System five* addresses concerns like the scope and reach of EA management. Typically, a piloting project is performed in the initial phase of an EA management endeavor, e.g. starting with a limited number of concerns, e.g. compliance issues, availability aspects, or with restricted reach e.g. within one business department. Nevertheless, after the initial phase, when the EA management has matured and become

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

more adopted, an **EA management governance** is established to redefine EA management scope and reach. Accordingly, the EA management governance aspect should be part of a description of the EA management function.

Summarizingly, the *Systems one to three* can be regarded as managing the ‘inside and now’ of the EA whereas *System four* and *five* manage the ‘outside and future’ of the EA. In the context of EA management, the former systems relate to the operative EA management tasks—‘running the enterprise’—while the latter ones consider the strategic EA management tasks—‘changing the enterprise’. The application of the VSM to the EA management process as described above is illustrated in Figure 3.5.

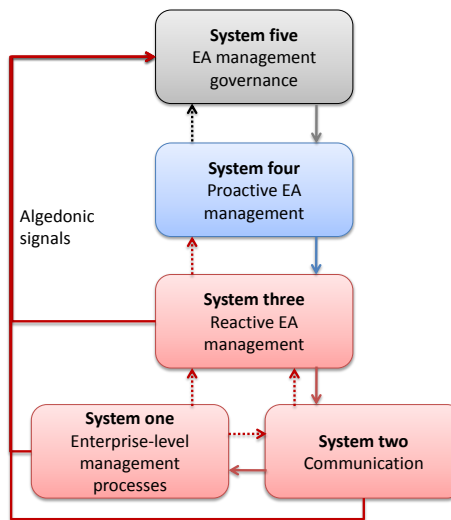


Figure 3.5.: Applying a viable system perspective to EA management

The systemic view on EA management is further complemented with the concept of the *algedonic signals* from the VSM. These signals, originating from *Systems one to three*, provide an alerting mechanism, which is employed, if one of these systems is not able to perform as intended in the current situation. Such a signal is propagated to *System five*, which in response can adapt the overall management function and can provide guidance to maintain the identity, i.e., the purpose of the EA management system. To exemplify these considerations, one may think of an EA getting increasingly heterogeneous albeit a standardization board has been established. At the point, this board notices that it has no means to counteract the tendency, an alert is escalated to the EA management governance. The governance function then has to e.g. either empower the board to stop non standard conform projects, to enact the envisioned homogenization, or to rise the question, if standardization is still a goal that should be pursued with the EA management endeavor in the future.

From an systemic perspective an approach to design an EA management function should include method descriptions mirroring the tasks of the five systems as well as provide method descriptions to handle algedonic signals in an appropriate manner.

## 3.2. Framework for analyzing EA management methods

As extensively discussed above, the design of the EA management function can be approached from multiple perspectives. Besides the area of management in general further related research topics exist, from which analysis criteria for our review can be derived. In line with the understanding of EA management as methodical and language-based means to develop and evolve the overall architecture of an enterprise, we subsequently analyze different EA management-related approaches as outlined in scientific literature with respect to the provided methods for designing an EA management function. The complementing analysis results relating to language aspects is published by us in [BS11]. Thereby, the cognition and comprehension of the subject is driven by the different perspectives on the design of an EA management function as described in Section 3.1, i.e., we derive analysis dimensions from the above discussions focusing on the method part of EA management in the following.

Beside the analysis of the method-specific prescriptions made by the different approaches, each approach is shortly summarized in a fact sheet. In this sheet general information on the approach is provided, such as NAME, ISSUING ORGANIZATION, dedicated TOOL SUPPORT, PERIOD OF ACTIVITY and the corresponding list of PUBLICATIONS. Thereby, the period of activity starts with the first publication of the group that can be ascribed to the area of EA management. Two additional characteristics are described in any fact sheet. The FOCUS AREA relates to the method-language dichotomy of the development method of EA management. Existing approaches typically can be characterized to either emphasizing on LANGUAGES or METHODS for EA management. The embracing nature of the management subject with floating boundaries to related management areas as strategic IT management, as well as the corresponding function further influences the way in which an approach can be presented. Regarding each approach in itself as one (composite) artifact, the approaches may strongly differ with respect to their inner organization. In detail, an approach may be presented as one comprehensive MONOLITH without an apparent inner structure, but may further be composed of different, clear distinguishable components. In the latter case two forms of organization may be distinguished, namely an EXPLICIT ORGANIZATION, in which the components establish explicit links to each other or an IMPLICIT ORGANIZATION, where the components are grounded in a unified and linking terminology.

Besides the general fact sheet, the perspectives on EA management as introduced in Section 3.1 motivate different analysis dimensions to characterize existing EA management approaches. These dimensions are introduced subsequently and complemented by possible characteristic types of EA management approaches.

### Analyzing integration

Taking a systemic perspective, the EA management function does not exist as an isolated management function in an organization but is embedded into the context of other enterprise-level management functions as project portfolio or strategy management (cf. system one of the VSM). The successful management of the EA is in this sense inevitably connected with linking these management functions, i.e., defining the tasks, means, and triggers for the exchange of management-relevant information. From this perspective, we classify the existing approaches according to their level of ‘integration’ in approaches that do not provide

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

mechanisms for integration, i.e., which do not account for information exchange with other management functions, and

- approaches that provide **UNIDIRECTIONAL** integration mechanisms. An EA management function according to such an approach describes the information source, i.e., management function. Information on the exchanged content, triggers, and tasks can optionally be supplied. Thereby, the EA management function is limited to one direction of exchange either being a receiver or a sender of information.
- approaches that provide **BIDIRECTIONAL** integration mechanisms. An EA management function according to such an approach describes the originating or targeted management function, i.e., acts as receiver as well as sender each in at least one case. Information on triggers, tasks, and the exchanged information can also be supplied.

#### Analyzing develop & describe

Central prerequisite for EA management are means to understand the management body—the EA—in a problem-adequate manner. Such means are EA descriptions and plans of the EA or parts thereof, i.e., models of different types. In line with the *plan* phase of the PDCA cycle, the dimension **DEVELOP & DESCRIBE** comprises different tasks aiming at the creation of EA models of different architectural states, i.e., current, planned, and target states. Rephrased in the words of knowledge management, the objective of the activity is to make tacit knowledge explicit. Besides the architectural states, principles, and standards guiding future evolution of the EA by constraining the solution space (cf. Simon’s design of the artificial) are relevant. Complementing, the activity is concerned with the concretization and documentation of questions as utility functions that apply on different architectural states. The method descriptions of the approaches can be complemented by participants involved or responsible for the different tasks. Detailed against the different objects, we characterize the methods provided by the different approaches as

- approaches describing the **current state**. An EA management function building on such an approach describes tasks and steps for documenting the current state of the EA. Describing the current state, thereby not only relates to an initial information gathering but instead refers to tasks and steps for maintaining the initially gathered information.
- approaches developing a **planned state**. An EA Management function committing to such an approach describes tasks and steps to be taken for developing planned states for an EA from projects, more precisely the architectural changes performed by these projects.
- approaches developing a **target state**. An EA management function based on such an approach describes tasks and steps for developing target states of an EA, i.e., to formulate architecture visions. Optionally these steps can describe how the target state can be derived from the strategies of the enterprise, namely the business and the IT strategy.
- approaches developing **EA principles**. Approaches of that kind describe tasks and steps that can be taken to devise organization-specific development guidelines, i.e., principles and standards, and to document these guidelines. Optionally these steps delineate how the principles can be derived from strategic input, as e.g. the business or IT strategy.



- approaches developing EA-relevant **questions**. An EA management function building on such an approach describes tasks and steps to commit a set of EA-relevant questions, i.e., methods for agreeing on an understanding of ‘better’ and ‘worse’ with respect to EAs. These questions may be formulated on a fairly abstract level.

#### Analyzing communicate & enact

Using the terminology of KM, knowledge distribution is frequently discussed in literature as a the communication challenge of EA management (cf. Lankhorst in [La05, pages 67–82] or Schekkerman in [Sc06d, page 88]). In line with that understanding an approach for EA management must cover the topic of COMMUNICATE & ENACT, which maps to the do-phase of the PDCA cycle and the system two of the VSM. We analyze, whether the approach describes steps to be taken and tasks to be performed to communicate EA-related information to the corresponding stakeholders. Complementing the communication nature of EA management, we further analyze, if the approach delineates tasks that may be applied to govern projects as the implementors of organizational change and enterprise-level management functions according to EA plans, visions, and principles. Detailed onto the level of the different classification this means:

- approaches communicating the **current state**. Approaches of that kind describe steps and tasks for communicating the current state of the EA, or resort to the provision of visualizations together with a statement on the corresponding stakeholders.
- approaches communicating and enacting **planned states**. An EA management function building on such approach describes tasks and steps for communicating planned states, or delineates visualizations and their corresponding stakeholders. Further, tasks and steps for enforcing architecture plans in related management processes may be given.
- approaches communicating a **target state**. An EA management function committing to such an approach describes tasks and steps for communicating target states of the EA, or describes visualizations for doing so as well as the corresponding stakeholders.
- approaches communicating and enacting EA **principles**. For the communication of principles an approach should describe steps and tasks or provide structured templates for communicating principles together with information on the intended audience thereof. Enactment mechanisms for principles, i.e., via dedicated steps in planning functions like quality gates, are further described.
- approaches communicating EA-relevant **questions**. Approaches of that kind describe uniform templates for communicating questions and link these to the relevant stakeholders. Instead of doing so the approaches may delineate steps and tasks for communicating putting special emphasis on the informed stakeholders.

#### Analyzing analyze & evaluate

In the course of developing future, i.e., planned and target, states of the EA different alternatives for implementation may be developed and have to be analyzed to make an informed decision. Mapping the check phase of the PDCA cycle to the EA management context, a comprehensive approach must in this respect cover methods and responsibilities concerned

with analyzing architecture states and plans as well as for comparing different states of the EA. Regarding the corresponding state, we classify to

- approaches analyzing the **current state**. An EA management function building on such approach describes steps and tasks to be taken to (collaboratively) analyze the current state of the EA with respect to given goals and principles. In this context the stakeholders of the corresponding analyses may be denoted and responsibilities for performing the analyses may be specified.
- approaches analyzing **planned states**. An EA management function building on such an approach describes which steps and tasks are necessary for analyzing planned states and may optionally specify the addressees of the analyses as well as the responsible participants. Goal- as well as principle-based analyses are expected mechanisms here.
- approaches analyzing a **target state**. For analyzing target state especially expert-based analysis techniques are to be described. Such techniques, more precisely the steps and tasks performed therein, are necessary to evaluate a target state with respect to principles and goals.
- approaches performing comparative analyses (**delta analysis**) targeting two states. Approaches of that kind provide steps and tasks for comparing different EA states highlighting the corresponding differences and similarities. Comparisons between current and target states, planned and target states, as well as between different planned states are of interest here.

#### Analyzing configure

An EA management function is an organization-specific artifact, i.e., has to be ‘configured’ to fit the organizational context as well as the intended scope and reach, i.e., the goals pursued (cf. goals setting of KM). As not any kind of implementing a management activity is suited in every context and for every intension, an EA management approach supporting its configuration must supply mechanisms to specifically design an EA management function with respect to the goals pursued and the organizational context, which surrounds the management function. With the distinction between context on the one and scope and reach, i.e., goals pursued, on the other hand, we classify each approach as follows:

- approaches providing no mechanisms for configuration. Approaches of that kind do not regard EA management as organization-specific or make prescriptions on an abstract level abstaining from organizational implementation.
- approaches providing mechanisms to configure the EA management function to the ORGANIZATIONAL CONTEXT. Configurable approaches of this type delineate organizational contexts, e.g. management structures that are beneficial or detrimental for some of the provided management methods. In other words, these approaches describe organizational contexts and link them to tasks, steps, and responsibilities.
- approaches providing mechanisms to configure to SCOPE AND REACH. Approaches that are configurable to an organization-specific scope and reach, i.e., the goals pursued by the EA management endeavor link tasks, steps, and responsibilities to the specific management goals that are considered helpful for pursuing.

## Analyzing adapt

Complementing the PDCA-cycle, the act phase has to be mapped to the EA management context. With the ongoing change of the organization itself as well as its environment, the EA management function may need to be ‘adapted’ as well (cf. discussions on the system five and algedonic signals of the VSM). Further, the need to adapt the EA management function may arise from the successful implementation of such function in the enterprise, which calls for an increased reach of the function. In the latter sense the adaptation reflects an increased level of maturity in EA management<sup>6</sup>. According to the provided mechanisms, we classify EA management approaches as follows:

- approaches providing no mechanisms for adaptation. Approaches of that kind make no or only abstract prescriptions on how to react to changes in the organizational context or on how to adapt to a changed scope and reach. Especially these approaches do not describe how to transform an already implemented EA management function to an adapted one.
- approaches providing mechanisms to adapt to the ORGANIZATIONAL CONTEXT. Adaptable approaches of this type delineate organizational context changes and describe transformations for implemented management tasks, steps, or responsibilities.
- approaches providing mechanisms to adapt to SCOPE AND REACH. Approaches of that kind describe transformations for increasing and reducing the reach of the EA management function, e.g. by delineating how certain tasks and steps can be extended to relate to other enterprise-level management functions. Concerning the scope, such approaches describe mechanisms to perform the one-time change of the scope, i.e., detail transformation methods encompassing documentation, communication, and analysis.

Table 3.1 provides a morphological box, which summarizes the analysis dimensions and characteristics as outlined above. With this morphological box, the results of the review are summarized in the subsequent sections.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.1.: Method classification for an EA management approach

## 3.3. Revisiting prominent to EA management

Several publications targeting the state-of-the-art in EA management literature have been written in the last years (cf. Aier et al. in [ARW08b], Schönherr in [Sc08b], Schelp and

<sup>6</sup>See Szyszka in [Sz09] for a in-depth discussion on maturity models in the context of EA management.

Winter in [SW09], and Aier and Schelp in [AS09]). While each of them targets a dedicated area-of-interest, e.g. Schönherr in [Sc08b] focuses on definitions for EA or EA management respectively, and Schelp and Winter in [SW09] emphasize on the research methods of the academic groups, the concluding call for developing a common understanding, i.e., forming the basis for a language community, remains the same. For our subsequent literature analysis, we make in line with Webster and Watson in [WW02, page xv] scope and limits of the literature included in the review explicit by discussing the way the literature was identified. In the area of EA management the identification of literature is hampered by the increasing importance of the topic of EA management in recent years, which has lead to a vast amount of literature published in this area (cf. study of Langenberg and Wegman in [LW04a] or Mykhashchuk in [My11]). At the same time no common understanding on the topic has evolved, leading to distinct research groups each forming a so-called language community. As EA management is a new discipline, for which different terms, e.g. *strategic alignment* (cf. Henderson and Venkatraman in [HV93]), *information systems architecture* (cf. Zachman in [Za87]), or *business IT alignment* (cf. Luftman in [Lu03]) have been used in the past, before the term **enterprise architecture** was coined. The identification of relevant literature accordingly can be regarded as complex task, as existing databases, e.g. the web of science<sup>7</sup>, the ACM digital library<sup>8</sup>, or IEEE Explore<sup>9</sup> cannot be searched using a single search string. In addition, research results concerning the topic of EA management are until now typically published as books in case of practitioners' experiences or presented on workshops (cf. Trends in Enterprise Architecture Management Research (TEAR) or the Enterprise Architecture Challenges and Responses (WEACR) international workshop) and therefore not included in scientific databases, which typically focus on journal publications. Due to this fact, we identified literature relevant for our synthesis by

- identifying research group via existing state-of-the-art analysis on EA management of Aier et al. in [ARW08b] as well as Schelp and Winter in [SW09],
- searching the DBLP<sup>10</sup> and the websites of the author for further publications,
- going backwards by reviewing the citations of the publications identified in the first two steps, and
- removing research groups, whose publications have been cited sporadic or are not available in English.

Using the above method, we identified 22 research groups with a publication record in the area of EA management, of which the 15 most cited ones are included in the subsequent state-of-the-art review. We present an in-depth discussion and classification of the remaining seven approaches in [BS11]. Whereas a practitioner framework like *The Open Group Architecture Framework* (TOGAF) is included in our analysis, frameworks developed by governmental agencies, e.g. the *Department of Defense Architecture Framework* (DoDAF) in [De09a, De09b, De09c] and the *NATO Architecture Framework* (NAF) in [NA07], are not discussed. As these frameworks typically focus on the public sector and the intended audience of our subsequent review are practitioners and researchers in the area of EA management, we

---

<sup>7</sup><http://www.webofscience.com>

<sup>8</sup><http://portal.acm.org>

<sup>9</sup><http://ieeexplore.ieee.org>

<sup>10</sup><http://www.informatik.uni-trier.de/~ley/db/>

abstain from discussing these frameworks. The following sections each detail on one prominent EA management approach in chronological ordering and present analysis results focusing on methodical aspects of the presented approach, which are summarized according to the framework introduced above. The complementing evaluation from a language perspective is detailed in [BS11].

To ensure methodological soundness of the subsequent review synthesis, we detail the characteristics of our review according to the framework developed by Fettke [Fe06]. In 2006 Fettke conducted a state-of-the-art analysis of the state-of-the-art in the German-speaking IS community (cf. Fettke in [Fe06]). Thereby, he identified a characterization framework for literature reviews [Fe06, page 259]. This framework is used in the following to summarize the scientific method that the review results are based on. According to Fettke, two different types of reviews can be distinguished—natural language and statistical reviews. Our review emphasizes on the natural language characteristic. While each review has a distinct focal points, e.g. results, research method, theory, and experience, we focus on the theories presented by the different approaches, in particular the methodical prescriptions made. In line with Fettke in [Fe06, page 265], we decide to make the objective of the review explicit. Therefore, the aim of the state-of-the-art analysis is detailed at the beginning of this chapter, represented in Research question 2 (cf. Section 1.1), and can be summarized as investigating existing theories for prescriptions how to design an EA management function and as identifying configuration aspects supporting the organization-specific design. To do so, an analysis framework is developed based on existing kernel theories in Section 3.1, which is further influenced by the epistemological assumptions presented in Section 2.1. We made the boundaries of our work explicit and stated the criteria how the literature was selected, thus providing a comprehensive overview of existing EA management approaches, although proof for complete coverage is unfeasible to give. According to Fettke, a review can be structured historically, thematically, or methodically. In our review we used the historic structuring, resulting as a side effect in an overview how long the different research groups have been active in this area. Although a dissertation is typically aimed at the scientific community, we believe that the topic has a strong relation to industry, therefore, the thesis results as well as the results of this research synthesis address researchers in general as well as practitioners. Table 3.2 summarizes the classification of this review synthesis according to the framework developed by Fettke in [Fe06].

TYPE		natural language		mathematic-statistical	
FOCUS		research results	research method	theory	experience
TARGET	FORMULATION CONTENT	not explicit		explicit	
		integration	criticism	central topics	
PERSPECTIVE		neutral		position	
LITERATURE	SELECTION EXTENSIVENESS	not explicit		explicit	
		foundations	representative	selective	complete
STRUCTURE		historical	thematically	methodical	
TARGET GROUP		common public	practitioners	common researcher	specialized researcher
FUTURE RESEARCH		not explicit		explicit	

Table 3.2.: Characterization of the review synthesis presented in this chapter according to Fettke in [Fe06, page 259]

### 3.3.1. The Zachman Framework

EA management approach	
Name of approach:	Zachman Framework
Issuing organization:	Zachman Institute
Focus area:	Modeling
Tool support:	-
Period of activity:	since 1987
Publications:	[Za87], [SZ92]
Inner organization:	monolith

In 1987, Zachman developed what was initially (cf. [Za87]) called a “framework for information systems architecture” and has ever since been extended to a more holistic perspective, resulting in the perhaps most well-known framework for EA—the Zachman Framework. In its most recent version<sup>11</sup> the framework consists of five modeling layers and six dimensions. The modeling layers are *scope*, *business*, *logical systems*, *technical systems*, and *detailed representations*. The latter, however, according to Zachman is not in the scope of EA management. On these different layers, the questions of *what*, *how*, *where*, *who*, *when*, and *why* apply. Figure 3.6 outlines the structure of the Zachman Framework. Putting the interrogatives and the modeling layers together, the core question that the Zachman framework associates with the EA can be summarized as “Who does what in which way (how), when, where and why does he/she do it?” This question is answered on each layer with increasing level of detail reflecting the addressees of the different layers, i.e., the planner on the contextual level (scope), the owner on the conceptual level (business), the designer on the logical level (logical systems), the builder on the physical level (technical systems), and the subcontractor on the out-of-context level (detailed representations). Illustrating the levels of details along the question-dimension of *who*, major business divisions (on the scope level) are decomposed and operationalized to organizational units (on the business level), to roles (on the logical system level), to users (on the technical system level), and finally to specific identities (on the detailed representation level).

In line with the understanding of the Zachman framework as an EA framework and not an EA management framework, no detailed descriptions on methods, management activities, or tasks can be found. Consequently, only some minor method-related information is provided, outlining that the framework may be applied both in describing the current state of the EA as well as in describing requirements for a future state, i.e., in developing a target state [Za87]. Additionally, the framework gives several remarks on the importance of transformation activities to get from the current to a future state, thus highlighting the importance of planning processes. More detailed information on how to plan states of the EA are however not directly given. With respect to the communication of information corresponding to the framework’s prescriptions, Zachman delineates in [Za87, pages 282–284] the variety of purposes that such architecture descriptions may serve as well as the plurality of addressed stakeholders, e.g. business owners or information system designers. All this aligns with the basic notion of the

---

<sup>11</sup>An overview on the framework is available online at <http://www.zifa.com/framework.pdf>. The recent version was accessed on October, 17<sup>th</sup>, 2010.

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

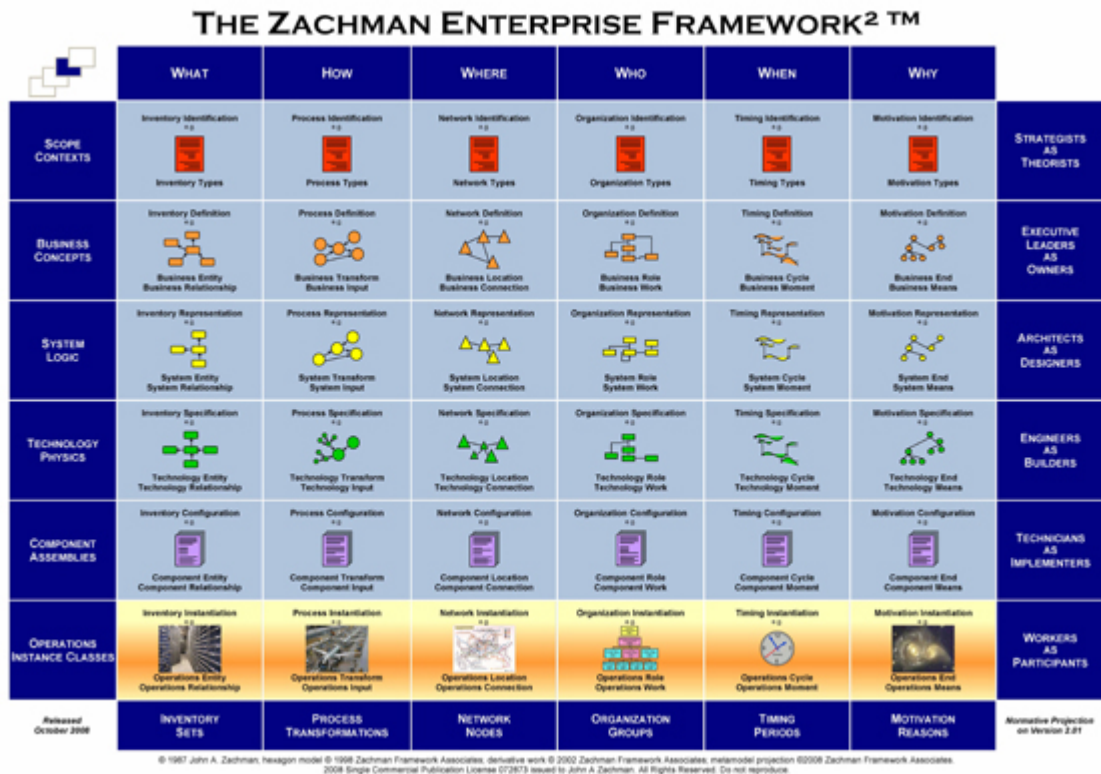


Figure 3.6.: Two dimensional schema behind the Zachman Framework (Source: <http://zachmanframeworkassociates.com/index.php/>)

framework understanding itself as structuring principle to be used in information system architecture development activities to get an embracing perspective. In this vein, no methodical integration points are discussed but a flexible utilization of the framework, e.g. in combination with a framework focusing on methodic guidance as TOGAF (see Section 3.3.5) is advocated for. In line with the requirements put forward in Section 3.2, we classify the Zachman Framework as shown in Table 3.3.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.3.: Method classification for the Zachman Framework

### 3.3.2. Architecture of Integrated Information Systems (ARIS)

EA management approach	
Name of approach:	Architecture of Integrated Information Systems (ARIS)
Issuing organization:	University of Saarbrücken, IDS Scheer AG
Focus area:	Modeling
Tool support:	ARIS Toolset
Period of activity:	since 1992
Publications:	[KNS92], [Ki99], [Sc01], [Sc02]
Inner organization:	monolith

The *Architecture of Integrated Information Systems* (ARIS) is a framework for holistic modeling of business information systems, targeting the design and development of such systems from a process-based perspective (cf. Scheer in [Sc01, Sc02]). Originating from information system development, the ARIS method mirrors its roots by incorporating the ‘classic’ software development phases *requirements elicitation*, *design specification*, and *implementation description*. The waterfall-like method is nevertheless not executed once, but is applied on the different views<sup>12</sup> that pertain to a business information system. The focus on business processes and the control view is additionally reflected in a corresponding modeling method, namely the one of *event-driven process chain* (EPC) as introduced by Keller et al. in [KNS92]. An event-driven process chain details the structure of events and functions with additional *operators* that may be used to denote splits, joins, and decisions in the process execution. The ARIS approach defines dedicated symbols to represent concepts like events, functions, inputs, outputs, organizational units, etc.

The so-called *ARIS house* (see Figure 3.7) introduces these views as follows:

- organization view** describing the structure of the organization together with the lines of authority and the communication channels in the organization,
- data view** describing the business data objects created, manipulated, and exchanged between the business functions
- function view** describing the business functions that are to be executed by the organization as part of its value proposition
- output view** describing the values, goods, and services delivered by the organization in executing its business functions
- control view** interlinking the other views from a process-oriented perspective, i.e., describing the business processes that are executed by organizations, the business data objects used, the involved participants, the executed business functions, and the delivered goods and services in time flow.

Beyond the waterfall-like software development method focusing on the development of models, ARIS does not directly provide methodical guidance. Extending the basic ARIS approach, Kirsch in [Ki99] presents a method for “process-oriented management of client-server-systems”.

---

<sup>12</sup>In line with the terminology used throughout the report, the *views* would more correctly be alluded to as *viewpoints*. According to the ARIS approach they may nevertheless be identified with specific views on an actual information system, thus being characterized as views.



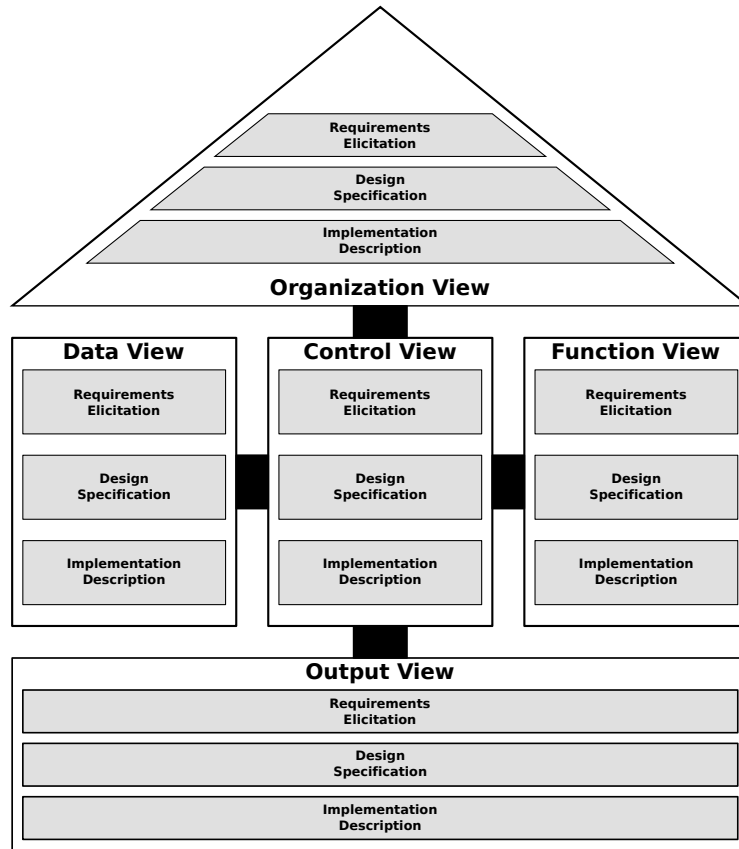


Figure 3.7.: ARIS house

Central to this approach is an iterative development method consisting of the phases *plan*, *realize*, as well as *apply and control*. During the different phases of the method, process-oriented analysis models are created and refined towards implementation models, which are fed to implementation. Kirsch further emphasizes related processes, e.g. release management, that are to be supplied with information incorporated in the ARIS models, leading to an overall classification of the ARIS approach as shown in Table 3.4.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.4.: Method classification for ARIS approach

### 3.3.3. The Generalised Enterprise Reference Architecture and Methodology (GERAM)

#### EA management approach

Name of approach: The Generalised Enterprise Reference Architecture and Methodology (GERAM)

Issuing organization: IFIP-IFAC Task Force on Architectures for Enterprise Integration

Focus area:

Tool support:

Period of activity: since 1994

Publications: [BN94] [BN96] [IF99] [In99] [BNS03] [IF03] [No03] [In06]

Inner organization: monolith

In the 1970s and the 1980s several EA-related frameworks have been developed. In response to the emerging number of frameworks in this area, the *International Federation of Information Processing* (IFIP) and the *International Federation of Automatic Control* (IFAC) established the *International Task Force on Enterprise Integration* aiming at the development of a reference framework that supports comparison, evaluation, and combination of existing approaches (cf. Bernus and Nemes in [BNS03, page 13]). As a result of the investigation, the Task Force developed the *Generalised Enterprise Reference Architecture and Methodology* (GERAM) which in 2000 became part of the ISO 15740:2000 [In99]. GERAM can be used to identify missing elements in existing approaches but can also be used as an EA (management) framework itself. Mappings of existing approaches to GERAM exist (cf. Noran in [No03]). GERAM consists of the subsequently described nine components (cf. Figure 3.8), which in line with the understanding as reference model do not impose particular methods or models but define criteria to be satisfied by an EA management approach (cf. [IF03, page 25]).

*Generalised Enterprise Reference Architecture* (GERA): GERA describes the basic concepts to be used in enterprise engineering and integration projects. According to GERAM these concepts can be categorized into *human oriented concepts*, e.g. capabilities, skills, know-how, as well as the roles of humans in the enterprise organization and operation, *process-oriented concepts*, e.g. functionality, behavior, entity life-cycles, and activities, and *technology oriented concepts* describing the supporting technology involved in enterprise transformation and operation.

*Enterprise Engineering Methodology* (EEM): EEMs provide process models or structured procedures with detailed instructions for enterprise engineering and integration.

*Enterprise Modeling Languages* (EMLs): EMLs define the generic modeling constructs for enterprise modeling. In particular, the EMLs provide constructs to describe and model human roles, operational processes, supporting information, and technologies.

*Generic Enterprise Modeling Concepts* (GEMCs): GEMCs define and formalize the generic concepts of enterprise modeling. In increasing order of formality, the following ways do define the generic concepts exist: natural language explanations (*glossaries*), meta models describing the elements and their relationships (*information models*), and theories defining the meaning, i.e., semantics of enterprise modeling languages (*ontologies*).

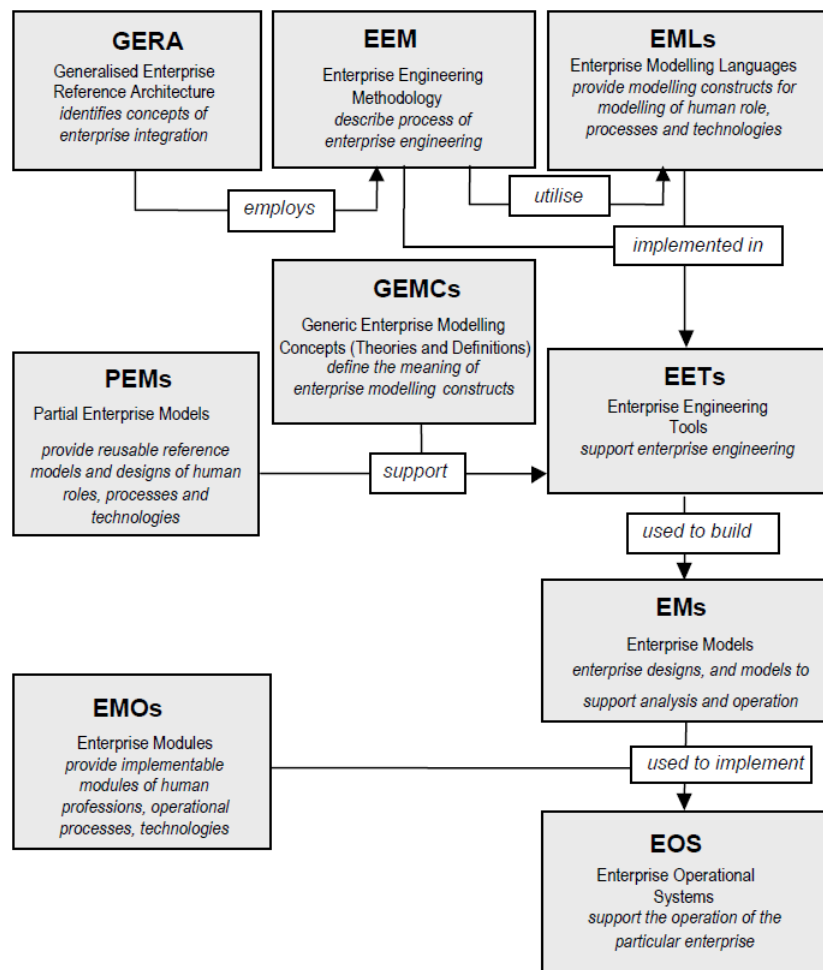


Figure 3.8.: The components of the GERAM framework [IF99, page 5]

*Partial Enterprise Models (PEMs):* PEMs represent reusable, paradigmatic models capturing characteristics of enterprises. They capitalize on previous knowledge by supporting the development of model libraries in a ‘plug-and-play’ manner rather than developing models from scratch. PEMs may cover the entire enterprise or a part thereof and typically concern a variety of enterprise entities such as products, projects, or companies.

*Enterprise Engineering Tools (EETs):* EETs provide implementation support for the method and modeling language used for enterprise transformation, e.g. a shared repository that enables creation and maintenance of PEMs.

*(Particular) Enterprise Models (EMs):* EMs capture concepts common to many enterprises and are expressed using a certain enterprise modeling language. They are used for analysis or represent executable models to support enterprise operation. EMs may consist of several models (views), which describe certain aspects on the enterprise.

*Enterprise Modules (EMOs):* EMOs are components that can be used in implementing the organization. Exemplary EMOs are human resources with given skill profiles, common

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

business procedures (e.g. banking and tax rules), or IT infrastructure. The EMOs can be used to model portability and interoperability, provide decision support as well as operation monitoring and control via real-time access to the enterprise environment.

*Enterprise Operational Systems (EOS)*: EOS represents the operation of an enterprise, which is typically guided by a particular enterprise model. This model provides the specification of the system and defines the EMOs used, i.e., the EOS consists of all hardware, software, and socio-technical elements needed to fulfill the enterprise objectives and goals.

Reflecting the method-language-dichotomy as proposed by our analysis framework, GERAM distinguishes between the methods for enterprise engineering (EEMs) and the modeling languages (EMLSs) used by the methods. Furthermore, GERAM defines three dimensions for defining the scope and content of the management body (cf. [IF03, pages 42–44]<sup>13</sup>), namely

*life-cycle dimension* providing means for modeling entities according to the life-cycle activities, *genericity dimension* supporting controlled particularization (from generic via partial to particular), and

*view dimension* enabling visualization of specific views of the enterprise entities.

Besides these dimensions, GERAM advocates for defining the pragmatic purpose for each view and thus concepts to be considered by the EA endeavor. Possible pragmatic purposes, e.g. support of design choices, simulation of processes to identify characteristics as cost or duration, are given in [IF03, page 45].

In GERA, a life-cycle for each constituting concept of the enterprise is introduced, which consists of the phases *identification*, *concept*, *requirements*, (*preliminary and detailed*) *design*, *implementation*, *operation*, and *decommission*. In the concept phase, the entity's mission, vision, strategies, objectives, etc. are defined, thereby linking cross-cutting aspects to the concepts considered during enterprise transformation [IF03, pages 32–34]. The concept of *life history* is referred to as main aspect of EA management approaches by GERA and the link to different kind of projects, e.g. engineering, redesign, or improvement projects, is discussed and related to the phases of the EA concepts. From a systemic perspective, GERA proposes a *recursive enterprise entity type* concept, which partitions into five different subtypes (cf. [IF03, page 39]). Entity type 1—strategic management entity—defines the necessity and the starting of any EA-related effort. Entity type 2—engineering implementation entity—provides the means to carry out the EA-related effort, i.e., uses a methodology (entity type 5) to define, design, implement, and build the operation of the enterprise entity (entity type 3). Entity type 3—enterprise entity—uses the methodology (entity type 5) and the operational system provided by entity type 2 to define, design, implement, and build the products of the enterprise (entity type 4). Entity type 4—product entity—represents all products (or services) of the enterprise. Complementing, entity type 5—methodology entity—represents the used methodology in the course of operation, which in general leads to the creation of another entity type.

In line with the objective of GERAM to represent an evaluation framework for EA (management) approaches, the life-cycle concept introduced above does not only apply for the

---

<sup>13</sup>The *GERA Modelling Framework* represents the basis for the International Standard ISO 19439 : 2006. Framework for Enterprise Modelling [In06].

constituting concepts of the enterprise but also to the enterprise itself. Besides this basic description, the EEMs define further requirements for an EA management function. According to [IF03, pages 49–50], the methodologies should be described in terms of process models or descriptions with detailed instructions for each activity, the information used and produced, resources needed, and relevant responsibilities assigned to the tasks. Again, special emphasis is put on the ‘human aspect’ as the EEMs call for an explicit modeling of humans and their relations to tasks, responsibilities, and influences. The role of humans as supporter or opponent of an EA management initiative and therefore the human role as success factor is alluded to (cf. [IF03, pages 50–52]). Dedicated methods and means how to overcome this challenge are however not provided. The aspect of human knowledge and tacit knowledge in particular is accentuated and specialized models are proposed to address this challenge. Emphasizing on the requirements defined by GERAM and the EEMs in particular, the overall evaluation of the method-related prescriptions of GERAM is shown in Table 3.5.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.5.: Method classification for GERAM

### 3.3.4. Semantic Object Model approach (SOM)

EA management approach	
Name of approach:	Semantic Object Model (SOM) Approach
Issuing organization:	University of Bamberg
Focus area:	Modeling
Tool support:	SOM Modeling Environment (in development, see <a href="http://www.openmodels.at/web/som">http://www.openmodels.at/web/som</a> , last-cited 2010-10-18)
Period of activity:	since 1994
Publications:	[Fe94], [FS95], [FS97]
Inner organization:	monolith

The *Semantic Object Model* (SOM) is rooted in the system theory, organizational theory, and cybernetics [FS08]. It was developed by Ferstl and Sinz who diagnosed a fundamental change in the way business information systems are understood via models in [FS95]. Whereas up to this point IS modeling centered around structural aspects of the systems, more recent approaches in those days started to identify IS with the set of interlinked business processes that the systems support. In this vein, the focus of IS modeling is broadened to not only incorporate the single system but also its enterprise environment (cf. Ferstl and Sinz in [FS95]). Reflecting this understanding of an enterprise, SOM introduces two key abstractions in [Fe94]: the *business transactions* reflecting the exchange of services between *business objects* that conversely provide or consume such services. Key principle of the approach is the *decomposition* of

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

both objects and transactions into smaller parts thereof, getting from an abstract perspective on an enterprise to more specific descriptions. This is further mirrored in the approach's EA framework, which consists of three layers [FS95, page 8], namely

**Enterprise plan** representing an external perspective on the organization concerned with the world of discourse, the environment of, and the goals pursued by the organization,

**Business objects & processes** are concerned with the tasks to be performed with respect to the enterprise plan, and

**Resources** considers human resources, information systems, infrastructure, etc. to support the business processes.

According to Ferstl and Sinz in [FS95], each layer of the EA has both a structural and a behavioral aspect, which need to be covered by an appropriate description method. This method is described via the so called *V-model* of the SOM approach as shown in Figure 3.9. On the top level the method calls for informal (textual) descriptions of the enterprise plan both from a structural and a behavioral perspective. The latter perspective covers the enterprise's value proposition and strategic goals. The former perspective is used to distinguish between the enterprise systems and its environment. In a subsequent step the structural plan of the enterprise is refined into an *interaction model* (cf. [FS97]) describing the interacting internal and external business objects, as ORGANIZATIONAL UNITS or CLIENTS. Building on the interaction model's description of the participants, the *value model* of the enterprise is described by connecting the participants via transactions. The preceding models are further detailed in subsequent steps, finally concluding in the *conceptual object design* which in turn forms the basis for implementing business information systems (cf. [FS97]).

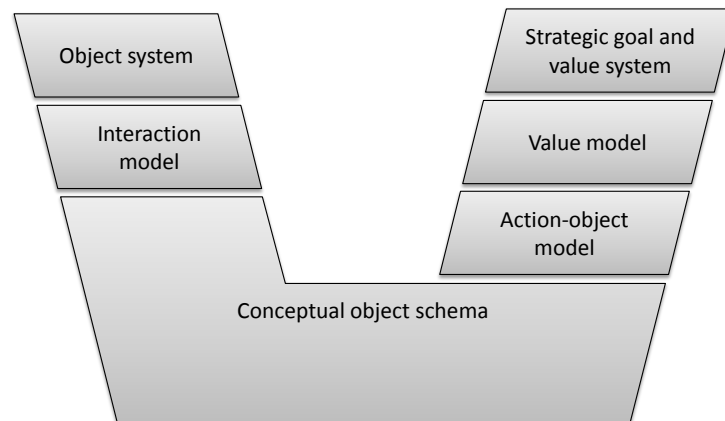


Figure 3.9.: V-Model (method model) of the SOM approach [FS95, page 9]

For the different modeling levels below the enterprise plan, Ferstl and Sinz [FS95, page 16] call for formal modeling techniques and languages based on the principle of decomposition. Using a BNF-like syntax, they describe decomposition rules which are to be applied in the V-model method. An exemplary rule reads as follows

$$O ::= \{O_1, O_2, [T(O_1, O_2)]\}$$

and describes that a modeler may decompose an object into a set of two sub-objects that are optionally linked with a transaction. Similar rules for decomposing transactions also exist. The

SOM approach similar like the Zachman Framework (see Section 3.3.1) can be characterized as emphasizing on EA modeling. Thus, no explicit methods for developing a target state for an organization or to re-engineer the current state are described. The aforementioned method nevertheless can be applied therefore. Reflecting these characteristics, we classify the method-related prescriptions of the SOM approach as shown in Table 3.6.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.6.: Method classification for the SOM approach

### 3.3.5. The Open Group Architecture Framework (TOGAF)

EA management approach	
Name of approach:	TOGAF
Issuing organization:	The Open Group
Focus area:	Method
Tool support:	TOGAF 9 Method Plugin for the Eclipse Process Framework Composer tool <sup>14</sup>
Period of activity:	since 1995 (TOGAF version 1.0)
Publications:	[Jo09], [Th09a], [Th09b]
Inner organization:	explicit organization

*The Open Group* is a vendor and technology-neutral consortium with the objective to foster information flow via open standards for enterprises<sup>15</sup>. In 1995, The Open Group published the first version of *The Open Group Architecture Framework* (TOGAF) which was based on the *Technical Architecture Framework for Information Management* (TAFIM) published by the Department of Defense. The current version 9.0 of TOGAF has been released in October 2009 [Th09a]. TOGAF is based on the terminology introduced in the ISO Standard 42010 (see Section 3.1.1) and provides a method, supporting models, and techniques for developing an EA management function. As a widely-used and known framework, the major players in the market of EA management tools have incorporated TOGAF in their tools (cf. the analysis of sebis in [se05] and Matthes et al. in [Ma08]). In addition, a method plugin for the open source eclipse process framework composer exists<sup>16</sup>. TOGAF 9 consists of six main parts, namely

- the *architecture development method* (ADM) describing an iterative process consisting of eight interconnected phases of EA development and a complementary preliminary phase (see Figure 3.10),

<sup>15</sup>See <http://www.opengroup.org/overview>, last accessed 2010-10-19

<sup>16</sup>See [http://www.opengroup.org/architecture/togaf/epf\\_intro.html](http://www.opengroup.org/architecture/togaf/epf_intro.html) (last accessed 2010-11-08)

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

- the *ADM guidelines and techniques* covering aspects of adaptability and configuration of the ADM to different process styles or specific architectures, i.e., security
- the *content framework* providing a conceptual meta-model for describing architectural artifacts,
- the *enterprise continuum and tools* representing a view on the architecture repository providing methods to structure and classify architecture and solution artifacts to enable communication and reuse of EA-related descriptions, and
- the *TOGAF reference models* being divided into the *TOGAF foundation architecture* and the *integrated information infrastructure reference model* (III-RM). The foundation architecture is embodied in the *technical reference model* (TRM), which is universally applicable and can be used to build any system architecture. The III-RM helps to address the need to design an integrated information infrastructure with reference designs.

In line with other EA management approaches, TOGAF proposes to structure the EA in different architecture domains representing subsets of the overall EA [Th09a, page 10]. Thus, TOGAF distinguishes between *business architecture*, which is concerned with strategic, governmental, organizational, and process-related aspects, the *data architecture* describing the structure of an organization's data assets and data management resources, the *application architecture* considering the application systems, their interactions, and their relationships to the business processes, and the *technology architecture* describing the logical software and hardware capabilities required to support the deployment of business, data, and application services.

Focusing on methodical aspects, the best-known part of TOGAF is the ADM, which describes an iterative process consisting of eight phases, which are complemented by a preliminary preparation phase and the central activity of requirements management (see Figure 3.10). The TOGAF ADM cycle starts with the *preliminary* phase, which prepares and initializes the EA management project. Typical tasks executed in this phase include the establishment of the EA team, the selection and implementation of supporting tools, as well as the definition of architecture guidelines and principles. After the preparation and initialization activities are performed, the scope of the EA management endeavor is defined within the *architecture vision* phase (*A*). A core objective of this phase is to identify the relevant stakeholders and their concerns. Based on the identified stakeholders and concerns a high-level architecture vision of the enterprise is derived in this phase. Succeeding phase *A*, the business, information systems, and technology architecture are developed in the phases *business architecture* (*B*), *information systems architectures* (*C*), and *technology architecture* (*D*). The fundamental make up of these three phases is very similar: initially, the baseline architecture (current state of the EA) is described. Based on this architecture, a target architecture is developed taking the architecture vision into account. This vision was formulated as part of the preceding phase *A*. A delta analysis is performed to evaluate the differences between the current and the target architecture and roadmap components enabling the transition from baseline architecture to target architectures are identified. The phase *opportunities and solutions* (*E*) is concerned with linking the separate business, information system, and technology architecture plans and deriving projects and programs, which describe the transformation from the current to the target architecture via intermediate transition architectures (planned states). The steps to be performed in this phase are the consolidation of the delta analyses from phases *B* to *D*, the



identification, refinement, and validation of dependencies between the different architectural layers, and the establishment on an integrated project and program portfolio. The transition architectures form the input of the *migration planning* phase (*F*), which is concerned with the formulation of an implementation and migration plan that schedules and realizes some or all of the planned architectures. The steps within this phase are the assignment of a business value to each project, the prioritization of projects, and the generation of a roadmap and migration plan. In the phase *implementation governance* (*G*) the projects selected for realization in the preceding phase are executed. Tasks to be conducted in this phase are the identification of deployment resources and skills, monitoring of the execution, and the conduction of reviews, e.g. regarding architecture compliance. The final phase (*H*) *architecture change management* concludes an ADM cycle and prepares the initiation of the next iteration. As part of the phase, the changes of the architecture are assessed. Key tasks of this phase are the deployment of monitoring techniques for the architecture process, the development of change requirements to meet performance targets, and the management of the governance process.

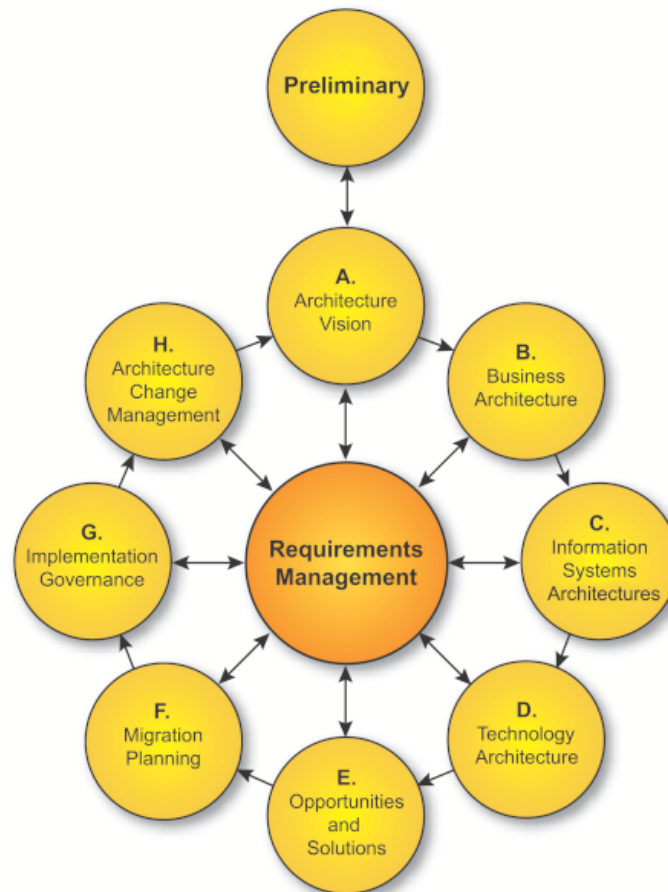


Figure 3.10.: The architecture development method of TOGAF [Th09a, page 54]

The ADM of TOGAF thereby focuses on EA management-projects instead of a continuous EA management function. While this approach ensures that a sponsor for the EA management endeavor is available (see preliminary phase), it entails the disadvantage that each project has to start with information gathering as no up-to-date information and description of the

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

EA is available. Complementing the high-level description of the phases, TOGAF provides exemplary guidelines and techniques for adapting or complementing the ADM (cf. The Open Group in [Th09a, pages 213–358]):

- *Capability-based planning*: A method for capability-based planning is presented by TOGAF, which enables a black-box view on the business level, i.e., the business capabilities of TOGAF include people, process, and material dimensions (cf. [Th09a, pages 353–358]).
- *Organizational contexts*: TOGAF for instance proposes to use hierarchies of ADM processes, if the EA management-related project is too complex. As organizations typically differ strongly regarding the situation in which EA management is intended to be established, TOGAF discusses different situations in which an adaptation of the ADM might be required (cf. The Open Group in [Th09a, pages 56–57]). The ordering of the single phases, for instance, may be subject to adaptation (cf. discussion in [Th09a, page 217]).
- *Architectural principles*: Different best practices how to develop, document, and apply principles are alluded to in TOGAF (cf. [Th09a, pages 167–280]). While exemplary principles are presented, methods to communicate and enact them are not discussed.
- *Delta analysis*: A matrix-based approach to perform delta analysis is presented in [Th09a, pages 321–323].

To configure the management body of the ADM, TOGAF proposes three different dimensions for segmentation (cf. The Open Group in [Th09a, pages 58–63]). First, the EA can be segmented with respect to the *scope*, i.e., which specific business sectors, functions, organizations, geographical areas are to be included. Second, a segmentation with respect to *architecture depth*, i.e., are all types of architecture covered or only a subset thereof, e.g. the data and application architecture. Third, the management body of the ADM can be tailored with respect to *time*, i.e., only the baseline architecture (current state) is included. While the importance of tailoring the management body is alluded to by TOGAF, no mechanisms how to guide and perform this configuration are given. Table 3.7 summarizes the key characteristics of TOGAF and especially the ADM classified against the background of the analysis framework.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.7.: Method classification for TOGAF

### 3.3.6. Extended Enterprise Architecture (E2A)

EA management approach	
Name of approach:	Extended Enterprise Architecture (E2A)
Issuing organization:	<i>Institute For Enterprise Architecture Developments</i> (IFEAD)
Focus area:	Method
Tool support:	-
Period of activity:	since 2001 (foundation of the IFEAD)
Publications:	[SK04], [Sc06a], [Sc06b], [Sc06c], [Sc06d], [Sc08a], [Sc10]
Inner organization:	explicit organization

In 2001 Jaap Schekkerman, who according to own statements has more than 25 years of experience in managing complex and large EA programs in the governmental area, healthcare, and high tech industry, founded the *Institute For Enterprise Architecture Developments* (IFEAD) a non-profit research and information organization aiming at fostering the EA-related knowledge exchange. In 2002 the IFEAD published the first version of the *Extended Enterprise Architecture* (E2A) framework. The E2A framework is influenced by the Zachman framework (see Section 3.3.1) and by the IAF (see van't Wout et al. in [Wo10]). Approaching the topic of EA management from a holistic perspective, the aspect of linking the EA management function with other closely related functions and processes, e.g. human capital management, information security management, and budgeting, is alluded to in [Sc08a, pages 36–37].

A core contribution of the E2A is the so-called *Enterprise Architecture Program* (EAP) which details on implementation steps for establishing an EA management function (see Figure 3.11). The EAP consists of eight steps, which are described subsequently.

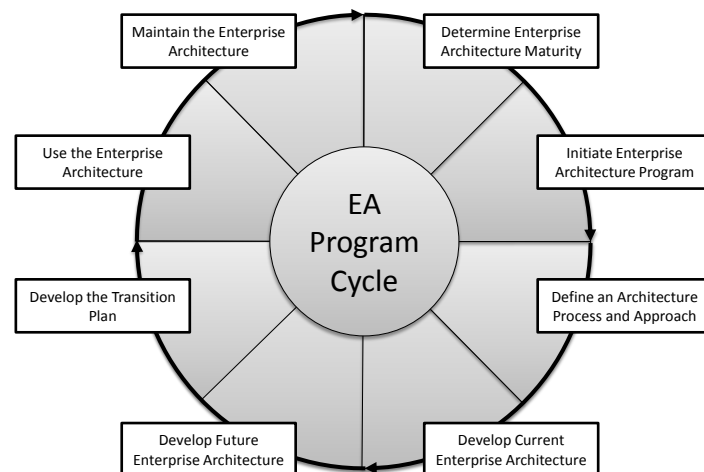


Figure 3.11.: The enterprise architecture program cycle according to Schekkerman in [Sc08a, page 38]

- *Determine enterprise architecture maturity*: Preparing the establishment of an EA management function the current maturity level needs to be determined. Therefore the E2A proposes the use of either existing maturity models or provides an own maturity model called *Extended Enterprise Architecture Maturity Model* (E2AMM) [Sc06c].

- *Initiate enterprise architecture program:* The setup of the EA management function is performed in this step. This includes linking the relevant management functions, e.g. IT portfolio management, with EA management and establishing the management structure and control, e.g. the EA steering committee, the chief enterprise architect, and the EA core team. For each role, E2A presents the associated responsibilities assigned to the team members, e.g. the information architect being responsible for documenting and analyzing business information and associated relationships (cf. Schekkerman in [Sc08a, pages 213–215]). Furthermore, the activities and results of the EA program are defined.
- *Define an architecture process and approach:* In this step, the intended use (goals), scope, and depth of the EA management endeavor are defined to ensure an EA management function sufficient for its purpose. The definition of scope thereby includes decisions on, e.g. geographical areas to be considered or the relevance of timeframes ([Sc08a, page 59]). The utilization of existing EA frameworks is proposed to answer the questions on goals, scope, and depth (a description of different EA frameworks is given in [SK04]). The final activity of this step is the definition of the EA process. Supporting this step the utilization of existing frameworks like TOGAF (see Section 3.3.5) or the Enterprise architecture process model introduced below is proposed as well as the selection of an appropriate tool is discussed.
- *Develop current enterprise architecture:* Phases of this step are a) discovery and data collection, b) design and preliminary results generation, c) review and revision, and d) publication and delivery of the EA results to an appropriate repository [Sc08a, pages 93–97]. For each of the aforementioned phases, methods and techniques how they could be performed, e.g. via interviews, ‘quick looks’, or documentation review, are discussed and basic questions to be answered are presented.
- *Develop future enterprise architecture:* In the same vein as the current state of the EA is documented in the preceding phase, the target state of the EA is documented. Essentials in creating the future state of the EA are thereby discussed, as e.g. the alignment with the strategic plan or the focus on business areas with the greatest potential payoff.
- *Develop the transition plan:* Based on the descriptions of the current and future state of the EA, a transformation plan is derived via a delta analysis. Additional dependency analyses between projects and the transition plans respectively are performed to provide input to project portfolio management [Sc08a, pages 99–104].
- *Use the enterprise architecture:* Enacting the transition plan as developed in the preceding step, this step provides good practices how the EA management function interacts with other enterprise-level management functions as e.g. project portfolio management. To ensure architectural compliance of projects on the one hand, trainings, reviews, consequences, etc. for deviations are proposed (cf. Schekkerman in [Sc08a, pages 107–118]). On the other hand, the importance of reflecting the performed changes in the ‘new’ current EA description is referred to in the next step.
- *Maintain the enterprise architecture:* As organizations represent highly dynamic systems, which evolve over time, this step is concerned with maintaining the EA artifacts, e.g. current and future state, and with the continuous control and oversee of the overall EA management function. The latter should be performed as a continuous process, which takes quick and decisive actions to correct problems. Examples of that actions

are redefinition of purpose and scope of the EA management function, introduction or strengthening of existing control mechanisms to ensure continuous improvement of the overall function (cf. Schekkerman in [Sc08a, page 126]). In [Sc04] Schekkerman proposes an *Enterprise Architecture Score Card<sup>TM</sup>* for assessing the performance of the EA management function.

Besides the above introduced program and its steps, the E2A proposes an *enterprise architecture process model* consisting of the steps a) enterprise architecture visioning, b) EA scope & context, c) EA goals / objectives & requirements, d) opportunities & solutions, e) organizational impact, f) benefits / business case, g) transformation planning, and h) implementation governance structure. A spiral model, which iterates through the aforementioned steps, is proposed to adapt the idea of “think big but start small” [Sc08a, pages 81–84]. Addressing the communication challenge, the E2A proposes techniques to identify and classify stakeholders using a *power-interest matrix* and proposes different sets of viewpoint types [Sc06a]. Furthermore, aspects of how to establish the EA governance, i.e., centralized, decentralized, or federated are alluded to in [Sc08a, pages 132–136] and aspects as roles and responsibilities are discussed.

E2A introduces three different types of principles (cf. Schekkerman in [Sc08a, page 236]: *enterprise principles* providing support for decision making on an enterprise level by informing how an organization seeks to fulfill its mission; *EA principles* reflect the spirit and thinking of the EA states and govern the EA management function as well as the implementation of its plans; *information technology principles* guide the use and deployment of all IT resources and assets. A general method how these principles can be developed as well as the responsible roles are provided by the E2A. Complementing best practice principles are presented [Sc08a, pages 238–255].

Recently, the IFEAD published a new and agile approach to EA management called *Speedy, Traceable, Result-driven Enterprise Architecture Management (STREAM)* [Sc10]. The characteristics of STREAM are traceability of choices and decisions from the business side (traceable), focus on elements that directly contribute to the objectives (pragmatic), deliver results within a short time frame (rapid), deliver predefined type of results (productive), and always start at the business side and deliver significant value (relevant). The STREAM approach consists of five steps of which two address the current situation of Business and IT (step 1 & 2), two the future situation (step 3 & 4) and the final step the transformation plan. The steps are carried out in an agile way with iterations within and over the different phases. In summary, this leads to an overall evaluation of the method-related prescriptions of the E2A as shown in Table 3.8.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.8.: Method classification for E2A

### 3.3.7. The EA management approach of MIT

EA management approach	
Name of approach:	(Approach of MIT)
Issuing organization:	MIT Sloan <i>Center for Information Systems Research</i> (CISR)
Focus area:	Method
Tool support:	-
Period of activity:	since 2003
Publications:	[Ro03], [WR04], [RB06], [RWR06]
Inner organization:	monolith

In 1995 researchers at the *Massachusetts Institute of Technology* (MIT) started their work in the area of EA management, with the first book concerning the topic published in the year 2003. Their work focuses on governance aspects of EA management (cf. Weill and Ross in [WR04]) and is mostly based on empirical surveys and case studies from industry. According to Ross et al. in [RWR06, pages 8–10], a foundation consisting of an *operating model*, an *enterprise architecture*, and an *IT engagement model* has to be built to execute the enterprise’s strategy. The operating model reflects the (envisioned) situation of the enterprise with respect to the two dimensions *business process integration* as well as *standardization* and therefore involves a commitment to the way the organization will operate. The EA provides a holistic view on the organization’s business processes, systems, and technologies. The IT engagement model describes the governance mechanisms used to ensure the achievement of objectives, by coordinating decisions from business and IT, and linking the enterprise-level management functions. Figure 3.12 illustrates how the three disciplines work together to create a foundation for execution.

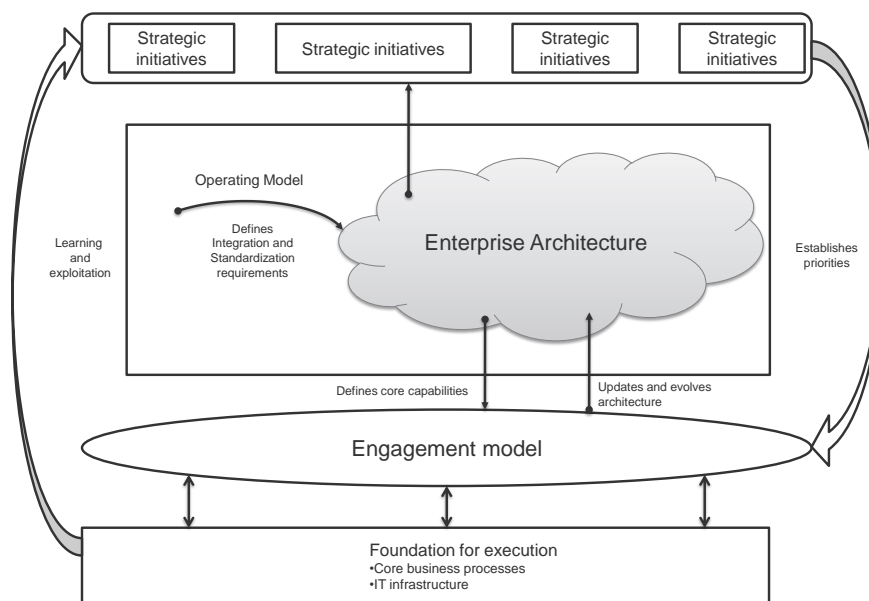


Figure 3.12.: The foundation for execution according to Ross et al. [RWR06, page 10]

In [RWR06, pages 28–33] Ross et al. introduce four different types of operating models which can be interpreted as different types of organizational context and goal descriptions defining requirements how the EA management function has to be configured. While abstaining from providing a precise procedure how to configure the EA management function to the specific needs of the operating model pursued, the authors exemplify the requirements and influences on the chosen model by a description of case studies. Ross et al. propose in [RWR06, page 195] a six step iterative approach to design and revise an EA management function, which read as follows: 1) analyze the existing foundation for execution<sup>17</sup>, 2) define the operating model, 3) design the EA, 4) set priorities, 5) design and implement an IT engagement model, and 6) exploit the foundation for execution for growth.

To define an operating model, one of the four different types has to be chosen, whereas, different operation models may apply for distinct parts of the organization. In the second step, Ross et al. propose to develop an *envisioned state* of the EA based on a so-called *core model*. A core model describes the relevant elements of the organization, which typically contain four common elements: business processes, data, technologies, and customers. For each operating model, Ross et al. propose a ‘best practice’ core diagram, which can be used as starting point for the development. Figure 3.13 shows the core diagram for the operating model *unification*. Each core diagram requires organization-specific adaptation and specification, i.e., the relevant elements have to be defined in an iterative collaboration process by the responsible managers, e.g. senior managers or IT leaders [RWR06, pages 65–67].

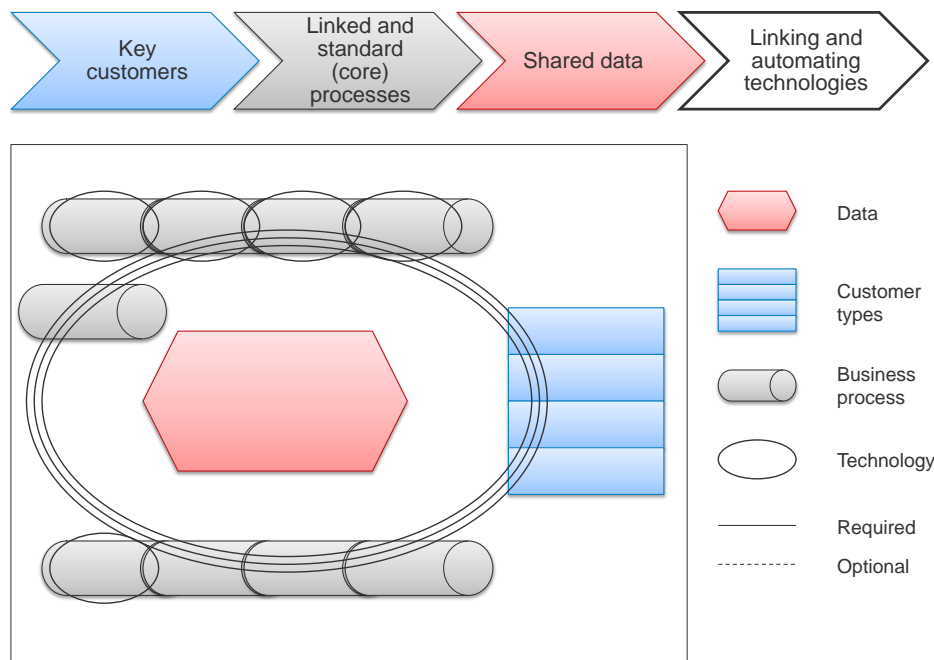


Figure 3.13.: The unification core diagram according to Ross et al. [RWR06, page 54]

While the approach presented by Ross et al. in [RWR06] discusses general aspects of EA management, no detailed method description on how to perform the documentation, commu-

<sup>17</sup>This step is the starting point in the first iteration, during initialization the method starts with the second step.

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

nication, or analysis tasks is presented. Nevertheless, the case studies from industry sketch such methods e.g. during the Toyota case study, the concept of architectural principles is introduced and different methods to enact the principles, e.g. through incentives, funding, or enforcement, are discussed [RWR06, pages 130–135]. For the communicate & enact activity, Ross et al. have derived key principles of successful engagement based on case studies from eighteen organizations, which can be seen as hints for the design of an organization-specific communication method [RWR06, pages 135–136].

In line with the idea of an organization as a vivid system, Ross et al. propose four stages of EA management maturity [RWR06, pages 71–79], which provide a path for the development of the EA management function. Considering the scope and reach of the EA management function, the maturity stages are defined as:

- *business silos architecture*, i.e., focusing the IT investments on individual business units needs,
- *standardized technology architecture*, i.e., shift from local optimization to global optimization via centralization of technology management and establishment of standards,
- *optimized core architecture*, i.e., shift from local applications and shared data to enterprise systems through organization-wide data and process standardization, and
- *business modularity architecture*, which enables strategic agility through reusability of loosely coupled IT-enabled business processes based on global standards [RWR06, pages 71–79].

Ross et al. discuss implications which emerge, while moving from one step to the next, and further present different architectural elements, tasks, and responsibilities that have to change during the maturity process [RWR06, pages 79–86]. Furthermore, Ross details on case studies using the maturity stages to evolve their EA management function in [Ro03] and sketches lessons learned. Ross et al. further discuss the utilization of the operating models and maturity stages in different application contexts, e.g. merger and acquisitions [RWR06, pages 176–182] or outsourcing [RB06]. In line with the above argumentation, the approach of the MIT can be classified as illustrated in table 3.9.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.9.: Method classification for the approach of MIT



### 3.3.8. The EA management approach of TU Lisbon

EA management approach	
Name of approach:	(Approach of TU Lisbon)
Issuing organization:	TU Lisbon
Focus area:	Modeling
Tool support:	-
Period of activity:	since 2003
Publications:	[Va01], [VST03], [Va04], [VST05], [Ca07], [MaZT07], [VST07], [AMT08], [VST08], [CST09], [AST10a], [AST10b], [Av10], [CST10], [MZT10], [Za10]
Inner organization:	implicit organization

Originating from “information system architecture” (ISA) (cf. [VST03, pages 78–79]), an intermediary level between EA and software architecture, the research group of José Tribolet at TU Lisbon lays a focus on “the representation of the IS components structure, its relationships, principles, and directives, with the main purpose of supporting business”. With this broad definition, it is sensible to reconcile the group’s research as contribution to the field of EA management, especially as more recent publications of Vasconcelos et al. [VST07] and of Aveiro et al. [AST10b] give indications towards an embracing EA perspective. The initial work emphasizes on the ISA aspect of EA modeling, e.g. by introducing the so-called *CEO framework* that introduces a high-level meta-model for describing ISAs as presented by Vasconcelos et al. in [VST03, page 79], with the work of Caetano et al. [CST09] broadening the scope to organizational and business aspects. In latter publications also more emphasis is put on method aspects related to ISA.

Addressing the “matching problem” (cf. Vasconcelos et al. in [Va01]), i.e., the missing link between established disciplines as business process modeling and management, goal modeling, as well as IS modeling and development, ISA outlines a trifecta of *ISA modeling*, *ISA evaluation*, and *IS/Business alignment assessment* (cf. [Va04]) as key activities necessary for solving this problem. Thus, the early work with its strong emphasis on modeling-related aspects abstains from detailing actual steps and tasks for performing these activities, but gives some abstract indications, e.g. on how to adapt analysis methods from related disciplines like the *architecture trade-off analysis method* (ATAM) (cf. [VST05]). Picking up this idea, Vasconcelos et al. provide in [VST07, pages 95–111] more precise analysis prescriptions as specific metrics, which are described via a uniform template called *ISA metric template*. In later work, namely [MaZT07, pages 62–63] the authors detail on the importance of ISAs and EAs as means of communication and elaborate on the stakeholder-specificity of communication methods reflecting “well-articulated preferences [of users]” and emphasized on the “different perspectives and viewpoints from which the company is considered” in [VST07, page 92]. Although the aspect of configuration and adaptation of the EA management function is not directly alluded to in the ISA approach, indications for the importance of the topic can be found. In [MaZT07, pages 66–67], for instance, the necessity to ‘co-evolve’ the described organization and the descriptive methods is briefly discussed along the *boundary* nature of the EA inviting *reflexivity about the organization*. In [AMT08] Aveiro et al. discuss on the topic of communicating architecture states, delineating the need to account for current, planned, and target states equivocally. The more recent publication of Aveiro et al. [AST10b] furthers these

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

discussions and develops the abstract indications on management methods towards more detailed prescriptions on activities in *operational engineering*, namely *(re)generation, operation and deletion of the enterprise*. Central to these considerations is a *viability* perspective on the enterprise, regarding the associated management processes as means to ensure viability of the organization by resolving dysfunctional interplay, i.e., to keep the organization working as intended in a corresponding to-be model. Where the work of Aveiro et al. in [AST10b, page 157] stays to abstract descriptions of the contained feedback loops “in tune with the well known PDCA cycle”, more specific prescriptions can be found in [AST10a, pages 231–233]. A generic *exception handling cycle* presented there and complemented with an organization-specific *monitoring, diagnosis, exception and recovery table* adds some detail on how viability of the enterprise system may be ensured reflecting the idea of algedonic signals as introduced by the VSM. The focus of the approach is nevertheless on modeling the feedback loops in such systems by providing a meta-model capable for describing actions taken as well as participants and resources involved (cf. Aveiro et al. in [AST10a, page 238]). With the intention to generically cover organizational feedback and control processes, no actual prescriptions are made with respect to the tasks and responsibilities involved in activities for ensuring viability. Aveiro nevertheless builds on these foundations in [Av10] and establishes methods for deriving planned states and for establishing EA principles. Regarding the question of the interplay between EA management and related management functions, Caetano et al. introduce in [Ca07] the concept of the *competency*. Competencies are therein understood as classifications of human participants according to their ability of performing certain tasks. Based on these classifications, the actor’s involvement in management activities comprised of several tasks can be discussed. Furthering this understanding, Marques et al. explain in [MZT10] the complex net of interlinked competencies necessary to manage on an enterprise level, thus giving strong indications on how to understand and establish integration between management processes. In the light of the requirements discussed in Section 3.2, we classify the approach of TU Lisbon as shown in Table 3.10.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.10.: Method classification for the approach of TU Lisbon

### 3.3.9. The Systemic Enterprise Architecture Methodology (SEAM)

EA management approach	
Name of approach:	Systemic Enterprise Architecture Methodology (SEAM)
Issuing organization:	École Polytechnique Fédérale de Lausanne (EPFL)
Focus area:	Modeling
Tool support:	SeamCAD [LW06]
Period of activity:	since 2003
Publications:	[RBW03], [We03], [LW04b], [LW05], [LW06], [BW06], [RW06], [RW07], [We07a], [We07b], [We08], [Re09]
Inner organization:	explicit organization

Driven by the the multi-disciplinary nature of EA projects (cf. Rychkova et al. [RBW03], Lê and Wegmann [LW04b]) and the resulting need to support these projects with methods and models the *systemic enterprise architecture methodology* (SEAM) was initially presented by Wegmann in [We03]. Outlining the central method-model-dichotomy, also alluded to as “method-notation”-dichotomy by Rychkova et al. in [RBW03], SEAM diagnoses a lack of support with respect to the method part of EA project support, thus seeking to complement existing approaches with additional methodical guidance. In [We08] Wegman provides an example for such a synergy by augmenting the Zachman Framework (cf. Section 3.3.1 and [Za87, SZ92]) with a *systemic conceptualization* based on SEAM. Constituting a predominant characteristic of the SEAM approach, the systemic nature of SEAM-based conceptualizations, reverberates through the different publications. Wegmann formulates in [We03, pages 486–488] the underlying *systemic paradigm* based on the SEAM ontology, which itself builds on the work of RM-ODP [In96], and constructivism as epistemological perspective. In line with the constructivism principle, SEAM assumes that knowledge about a system is relative to the observer, meaning that no observer-independent descriptions of reality exist. Caused by this understanding, Wegmann motivates in [We03, page 487] a hierarchical understanding of any system considering different *levels of reality* owned by dedicated stakeholders. This understanding reverberates through the work on SEAM, especially through the foundational language descriptions by Lê and Wegmann [LW04b, LW05], and is further mirrored in the methodology’s tool support (SeamCAD) presented by the same authors in [LW06].

In line with the understanding of TOGAF, the notion of the *EA project* as discussed by Wegmann in [We03, page 485] is central to SEAM. Such a project is initiated by an organization to *react to* or to *anticipate change* (cf. system three and four of the VSM) and starts with creating an *as-is model* reflecting the project-relevant entities. Complementing this model, a *to-be model* outlining the expected reaction to the change is created. A stepwise method for creating both models (as-is and to-be) is described by Rychkova et al. in [RBW03, pages 11–12]. This method is recursive in its nature spanning different levels of abstraction in the system hierarchy, such that lowest level models provide *all necessary details for [subsystem] implementation*. As part of the method delta analyses are to be conducted on each abstraction level, identifying one delta on each level such that a multi-level set of deltas has to be accounted for in finding the optimal design (cf. [We03, page 485]). Figure 3.14 taken from [We08] summarizes the cyclic development method as incorporated in SEAM.

The project nature of SEAM is reflected in the notion of the to-be models created in the development method, which actually represent planned states for the EA or parts thereof,

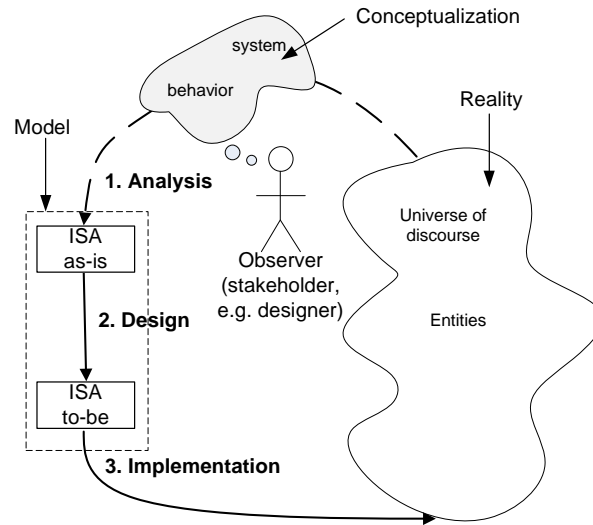


Figure 3.14.: Basic development method of SEAM according to Wegmann et al. [We08]

and is further reflected in the context descriptions of SEAM, where a linkage to change and requirements management is briefly alluded by Wegmann et al. in [We07a, pages 397–398]. The linkage is mediated via graphical models of as-is and to-be, which are discussed in different works by Rychkova and Wegmann [RW07] as well as by Wegmann et al. [We07a]. The importance of stakeholder-specific models is further emphasized by Wegmann et al. in [We07b, pages 118–119], where they—in line with the systemic paradigm from [We03]—delineate that different designers use different *views* representing relevant parts of the overall system. These discussions are detailed with remarks on the ways for developing and designing the SEAM models namely via workshops or using collaborative tools. In an earlier publication [BW06] Balabko and Wegmann reflect on the topic of methods for designing architecture models. As part of this reflection they analyze various methods from different disciplines with respect to their suitability for designing as-is and to-be models on various hierarchy levels. Based on the analysis’ results an EA project can select a method well-suited for the specific design purpose, although Balabko and Wegmann abstain from giving details on how to integrate different methods.

Fostering the analysis of SEAM’s basic development method (cf. Figure 3.14), Rychkova et al. propose in [RBW03] a conceptual groundwork for analyses. Thereby, they complement the SEAM notation with an operational semantics described in terms of *abstract state machines* (ASMs). In particular, they outline transformations for translating a SEAM model into a model of an ASM. The underlying idea is concretized and furthered by Rychkova and Wegmann in [RW06], where the transformation is rewritten based on the ASM description language of AsmL<sup>18</sup> via an *AsmL interpretation* of SEAM graphical models. Based on the executable ASM description as well as the notion of *behavioral substitutability*, Rychkova and Wegmann devise a method for verifying the alignment of a system design and the according behavioral requirements. Complementing these behavioral verification, Wegmann et al. ex-

<sup>18</sup>For a description of AsmL see <http://research.microsoft.com/en-us/projects/asml/> (last accessed 2010-11-08).

emplify in [We07b, pages 113–118] three methods for analyzing the planned design from a customer and an organizational perspective, critically relying on beliefs of the system’s stakeholders. This analysis perspective is revisited by Regev et al. in [Re09], who further introduce a classification of stakeholders in *favored*, *disfavored*, and *ignored* ones. Based on the classification, a conceptual basis for defining key qualities of any system such as *utility* or *risk* via the existence or absence of a perception for different stakeholder groups is established. Table 3.11 summarizes the key characteristics of SEAM against the background of the analysis framework.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.11.: Method classification for the Systemic Enterprise Architecture Methodology

### 3.3.10. Archimate

EA management approach	
Name of approach:	ArchiMate
Issuing organization:	Telematica Institute / Novay
Focus area:	Modeling
Tool support:	ArchiMate Workbench [La05, La09a]
Period of activity:	since 2003
Publications:	[Jo03], [Jo04b], [La04], [La05], [Jo06], [Ar07], [La09a], [Jo10], [QEJ10], and <a href="http://www.opengroup.org/archimate/doc/ts_archimate/">http://www.opengroup.org/archimate/doc/ts_archimate/</a>
Inner organization:	monolith

The ArchiMate modeling language for EAs is intended to support the description, analysis and visualization of EAs based on the ISO Std. 42010 (cf. Section 3.1.1) and has been adopted as an open standard hosted by The Open Group<sup>19</sup>. The development history of ArchiMate dates back to the work of Jonkers et al. in [Jo03], who outline the key requirements and principles of what would later become a “language for coherent enterprise architecture descriptions”. In particular, they introduce a notion of flexibility with respect to the model, plurality with respect to visualizations as well as viewpoints, and integrability with respect to existing modeling documentations. Building on this basic understanding, Jonkers et al. describe the three core aspects of an enterprise that any suitable modeling language should account for, namely *structure*, *information*, and *behavior*. For each of these aspects as well as for the three relevant layer, namely *business*, *application*, and *technology*, the ArchiMate modeling language provides appropriate concepts and conceptualizations. Figure 3.15 summarizes the *architecture framework* behind the ArchiMate language as made up of the aforementioned aspects and layers.

<sup>19</sup>See [http://www.opengroup.org/archimate/doc/ts\\_archimate/](http://www.opengroup.org/archimate/doc/ts_archimate/) (last accessed 2010-10-24) for further details.

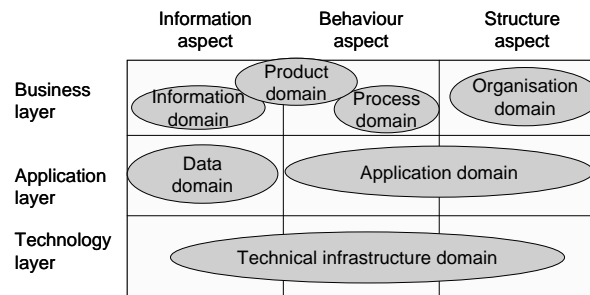


Figure 3.15.: The ArchiMate architecture framework according to Jonkers et al. [Jo03]

Since its initial presentation in the first edition of Lankhorst’s book *Enterprise Architecture at Work* [La05], ArchiMate has become more than the modeling language, meaning that the language is embedded into the context of a comprehensive set of related methods and guidelines, resulting in the ArchiMate 1.0 specification (cf. Lankhorst in [La09a] as adopted by The Open Group). Furthermore, a tool support for ArchiMate EA modeling has been developed, the so-called *ArchiMate Workbench*. In [Ar07] Arbab et al. describe the *architecture life cycle* that should be supported by appropriate architecture models. Similar versions of this life cycle can also be found in [La05, pages 46–47] and in [La09a, pages 49–50]. The life cycle consists of the phases a) design, b) communication, c) realization, and d) feedback.

Emphasizing the communication challenge of EA management, the importance to represent architecture aspects relevant to particular stakeholders is repeatedly discussed e.g. in [Jo03, Jo04b] and especially emphasized upon by Jonkers et al. in [Jo06]. This central communication challenge is further mirrored by Lankhorst et al. in [La04], where he deems EA modeling to be an “issue of integration” that has to bring together information from many different sources, and as Lankhorst later puts in [La09a, pages 12–22] related governance instruments. Adding more detail to answer the question how EAs can be communicated in an appropriate and stakeholder-specific way, Lankhorst introduces in [La09a, pages 80–84] the notion of the ‘architectural conversation’. Such conversations are employed to proliferate architectural knowledge respecting the scope and perspective of the intended audience. Further, conversations relate to specific knowledge goals as *introduce*, *agree*, or *commit*, of which each demands for a specific conversation technique. Making the relationship between knowledge goal and conversation technique more explicit, Lankhorst presents a *suitability matrix* in [La09a, page 83]. In a similar sense, Lankhorst adds in-depth discussions on how to select and adapt viewpoints for creating stakeholder-specific visualizations. Special attention is thereby paid to the stakeholder *commitment*, i.e., stakeholder awareness and agreement on the possible social implications of a certain viewpoint (cf. [La09a, page 171]). Central to this discussion is the understanding that every viewpoint creates transparency with respect to a certain part of the organization, meaning that stakeholders responsible for this part might be ‘overseen’ by ones having access to according visualizations. In line with this argumentation, Lankhorst discusses on the topic of *scoping* viewpoints to convey the information needed to perform certain activities but not necessarily more. Exemplary viewpoints related to dedicated EA stakeholders and possible activities are delineate in [La09a, pages 176–194].

First outlined by Jonkers et al. in [Jo03], techniques for analyzing EAs represented in corresponding models are detailed both in [La05] and [La09a]. In the first edition of his book

Lankhorst discusses two different types of possibly interesting EA characteristics, namely *quantitative* and *functional* ones. Detailing on these concepts, he delineates quantitative methods for assessing *performance*, *reliability*, and *costs*, where for the latter case process algebras for assessing the dynamic behavior of an architecture are introduced. In the more recent edition of the book [La09a, pages 231–255], the analyses’ subject are extended to incorporate *architecture alignment*, which may be regarded an intrinsic property of the EA. More precisely, alignment may be regarded a key goal of EA management and the presented *guidelines regarding architecture alignment* as well as the complementing analysis techniques represent part of a governance structure for EA management, itself. In two recent Open Group whitepapers [Jo10, QEJ10] more detailed prescriptions on how to perform EA management are provided. The first whitepaper targets the linkage between EA management and the enterprise-wide requirements management processes, describing how change demands and information on the current state of the EA can be used to derive “architecture requirements” [QEJ10, page 8–9]. These are subsequently incorporated in a target state of the EA, from which in turn “realization requirements” are derived and finally converted to realization plans. This yields the linkage to the activities described in the second whitepaper [Jo10], which delineates a how projects and as well as their results are reflected in the EA management function. Relating the notion of the project from an EA management-perspective with the understanding of projects promoted by PRINCE2 [Of09], the presented techniques are useful as means of integrating EA management and project management. Summarizing the above, ArchiMate presents itself as approach with a strong model focus, whose method-related prescriptions are limited as indicated in Table 3.12.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.12.: Method classification for Archimate

### 3.3.11. The EA management approach of KTH Stockholm

<b>EA management approach</b>	
Name of approach:	(Approach of KTH Stockholm)
Issuing organization:	KTH Stockholm
Focus area:	Modeling
Tool support:	EA Tool [Ek09]
Period of activity:	since 2004
Publications:	[Ek04], [Jo04a], [GLS06], [JNL06], [JE07], [Jo07], [La07], [La08], [LJ08], [Nä08], [Bu09g], [Ek09], [KUJ09], [RNE09]
Inner organization:	monolith

The EA management approach developed at KTH Stockholm centers around methods and techniques for analyzing EAs with respect to specific qualities. Key addressee of the approach,

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

which aims at providing decision support for IT management in enterprises, is the CIO, the key responsible for strategic IT-related decisions (cf. Ekstedt et al. in [Ek04]). Relating the topic of EA management to the disciplines of software engineering and IS engineering, the approach of KTH Stockholm seeks to complement existing approaches for modeling EAs, such as the pattern-based approach to TU München (cf. [Bu09g]) or the ArchiMate modeling language [Nä08]. The theoretic discussions on evaluating architecture qualities are complemented by Ekstedt et al. in [Ek09] by the outline of an analysis tool.

Reflecting the core topic of the approach, Ekstedt et al. discuss in [Ek04], the basic method for performing analyses on an EA-relevant level. A user of the method, e.g. the CIO, is required to prioritize the EA-related questions that should be answered, thereby assigning an expected utility to each question. Based on the prioritization, the set of required information is derived and complemented with estimates on costs for gathering the information. Evaluating the cost/utility ratio for each question, an appropriate organization-specific information model is derived, the corresponding information is gathered, and the analyses are finally performed. Aforementioned steps may further be embedded into the environment of the method outlined by Johnson et al. in [Jo04a]. In the first step of the method, architects create *scenarios*, i.e., modified versions of the current state of the architecture. Quality criteria for these scenarios are established in the second step and the scenarios are analyzed in the third step (both steps using aforementioned method). The scenario to be implemented is selected in the final step. The proposed method well aligns with the understanding of *IT management* as outlined by Gammelgård et al. in [GLS06, page 30], where three steps *understand*, *decide*, and *monitor* are introduced. In [JE07, pages 272–273] Johnson and Ekstedt further discuss methods for EA management, proposing a process as shown in Figure 3.16, in which each step is assigned individual responsibilities and participants. Reflecting integration, the process is further linked with typical processes of IS management, as introduced in the CobIT guidelines [IT09]. To make the interaction explicit, CobIT-specific artifacts are denoted as input and output artifacts of the different process steps.

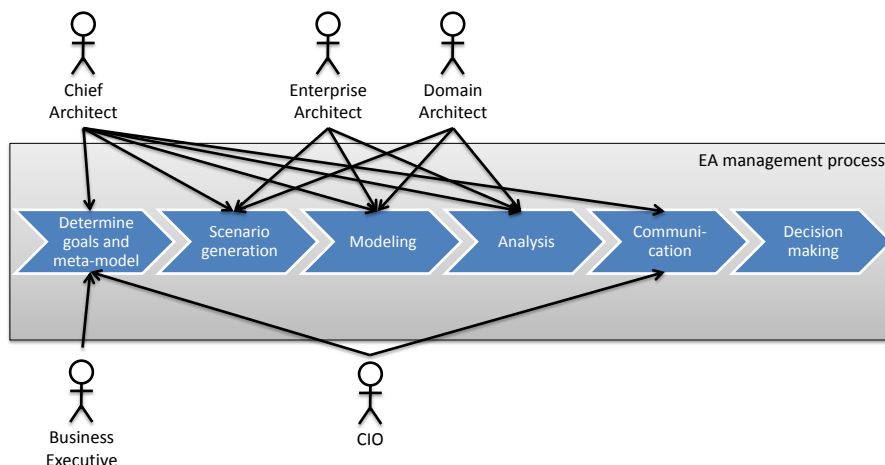


Figure 3.16.: The normative EA management process of Johnson and Ekstedt [JE07]

Furthering the process proposed by the KTH Stockholm, a consolidated view on the method-related prescriptions is taken by Källgren et al. in [KUJ09], where guidelines for constructing



a company-specific *EA model framework* are delineated. The guidelines mirror the basic idea of Ekstedt [Ek04] and are constituted of three major steps *make EA categorization*, *identify desired information*, and *finalize EA model framework*. In the first step the relevant business and IS goals for EA management are selected and linked to the relevant EA stakeholders in the second step. In the third step the appropriate viewpoints for the EA management function are identified, i.e., using the list of viewpoints provided by Johnson and Ekstedt in [JE07]. These viewpoints are further linked to the underlying information models, which are in turn related to existing information sources in the organization to develop a collection procedure for the integrated information model. The decision and control centric focus of the approach of KTH Stockholm is reflected in the classification as shown in Table 3.13.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.13.: Method classification for approach of KTH Stockholm

### 3.3.12. The EA<sup>3</sup> Cube<sup>TM</sup>

<b>EA management approach</b>	
Name of approach:	EA <sup>3</sup> Cube <sup>TM</sup>
Issuing organization:	Scott A. Bernard, Syracuse University
Focus area:	Method
Tool support:	-
Period of activity:	since 2005
Publications:	[Be05], [Do08], [BG09], [Do09a], [Do09b], [Do09c]
Inner organization:	monolith

In [Be05] Bernard presents his experience gained through his work in practice and academia in the area of EA management. Identifying the need for a textbook for students, Bernard wrote *An Introduction to Enterprise Architecture Management*, in which he presents the *EA<sup>3</sup> Cube<sup>TM</sup>* approach. According to Bernard, a framework for EA management follows the dichotomy of language (*what*) and method (*how*) [Be05, page 75] and consists of six basic aspects:

1. an EA governance process, which links the the EA management function to other enterprise-level management functions,
2. a repeatable methodology describing the management function,
3. a framework representing the core elements and layers, i.e., the scope, of the initiative,
4. an integrated set of artifacts, i.e., architectural descriptions,
5. documentation tools with a repository to support architectural descriptions, and
6. associated best practices, which guide the implementation of the management function [BG09, page 220].

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

These constituents are further detailed subsequently along the EA<sup>3</sup> Cube™.

In [Be05, pages 38–40] Bernard introduces a framework for EA descriptions—the EA cube, which consists of three dimensions (see Figure 3.17). The first dimension is concerned with the different architectural *levels* ranging from high-level strategic goals and initiatives to technical network and infrastructure aspects. The *segments* dimension divides the overall EA in different parts covering one or more lines of business, i.e., distinct areas of activity in the organization, from a holistic perspective, i.e., all architectural levels. Complementing, the third dimension *artifacts* refers to the components that make up the organization. Thereby, vertical components that serve one line of business but may affect more than one architectural levels and horizontal, i.e., crosscutting components, which serve several lines of business, are distinguished.

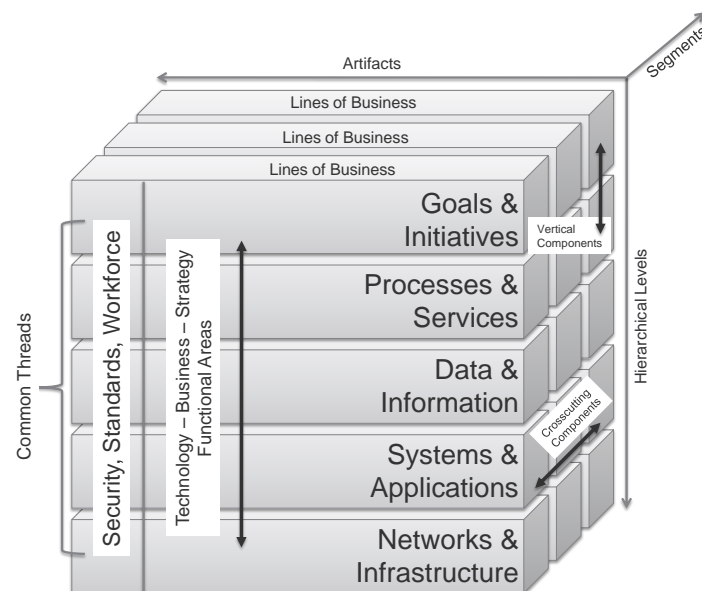


Figure 3.17.: The EA<sup>3</sup> cube description framework [Be05, page 40]

The above introduced concept of cross-cutting components cannot be put on a level with the idea of cross-cutting aspects as introduced in Chapter 1. As the cross-cutting components introduced by Bernard in [Be05, page 40] relate to instances, while the cross-cutting aspects introduced in the motivating chapter relate to the class level. Goals of an EA management endeavor, which represent a cross-cutting aspect in terms of the analysis framework, are referred to by Bernard in [Be05, pages 64–69] and their relation to supporting components of the EA is detailed [Be05, page 181]. Similarly, strategies, goals, and measures are discussed to quantify the EA management endeavor [Be05, pages 72–74] but only from a methodical perspective.

The need to understand the organizational contexts, i.e., the “amalgamation of values, beliefs, habits, and preferences of all of the people throughout the enterprise” [Be05, page 56] to design a suitable EA management function is emphasized by Bernard. Besides a list of prospective key success factors for avoiding cultural misinterpretations, Bernard presents detailed examples for the impact of the organizational context on the design of an EA management function, e.g. he proposes a *segmented approach*, which is limited in scope for large or decentralized

organizations [Be05, page 55] or he indicates that the schedule for updating architectural descriptions has to be defined [Be05, page 83]. While providing hints and suggestions, which method is suitable in which organizational context, he abstains from directly linking them or providing mechanisms for configuration.

In [Be05, pages 83–94] a 20-step process to implement an EA management function is introduced. The steps can be grouped in four phases: Establishment of the EA management function, framework and tool selection, documentation of the EA, and use and maintain the EA. In the first phase, aspects as integration with other enterprise-level management functions, e.g. the project management and investment planning (cf. [Be05, pages 198–211]), as well as the configuration aspects with respect to the organizational context and the intended scope and reach (cf [Be05, page 87]) are discussed. Thereby, no explicit mechanisms how to perform this configuration are presented but their importance is accentuated and questions, which guide the configuration, are provided (cf. [Be05, page 83]). In the second phase especially the aspect of tool support and best practices for EA management is alluded to. Phase three is concerned with the development and description of current and future states of the architecture. Thus, Bernard distinguishes between two different types of future views, the long-term strategic future views (with a time horizon of 4-10 years) and the medium-term, planned, tactic views (1-3 years) [Be05, page 41]. The latter ones are derived from changes as described in the planned initiatives [Be05, page 160]. Thereby, also the importance of variant development and historization is alluded to (cf. [Be05, pages 160–164]). The *Concept of Operations* (CONOPS) (cf. Neilson in [Ne00]) is described as exemplary method to develop the future states. Complementing the temporal snapshots on the EA, an EA management plan representing a roadmap illustrating the transformation from the current to the future state is developed. This plan further includes the definition of roles, responsibilities [Be05, page 170] and boards and their competencies providing the link to other enterprise-level management functions [Be05, page 204]. While Bernard abstains from detailing methods in terms of tasks and participants to develop descriptions of the current and future state as well as roadmaps, he provides exemplary viewpoints how an EA description could look alike, thus leading to a classification of the approach as shown in Table 3.14.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.14.: Method classification for EA<sup>3</sup> Cube Framework

### 3.3.13. The EA management approach of Niemann

EA management approach	
Name of approach:	Niemann
Issuing organization:	act! consulting (consultant)
Focus area:	Method
Tool support:	-
Period of activity:	since 2006
Publications:	[Ni06c]
Inner organization:	monolith

Klaus D. Niemann is the managing director of act! consulting, which is specialized in the development of EAs. Klaus D. Niemann has more than 20 years experience in the area of EA management (cf. [Ni06c]), which he has written down in a book "From Enterprise Architecture to IT Governance" [Ni06c].

The book presents the so-called *EA Cycle* consisting of the phases document, analyze, plan, act, and a central check activity (cf. Niemann in [Ni06c, page 37]) as illustrated in Figure 3.18. According to Niemann in [Ni06c, pages 170–177], EA management as to be integrated in existing management structures, i.e., the interaction with other processes and functions has to be defined. Relevant functions are program and service management as well as requirements and portfolio management, which can have a bidirectional connection in the sense that e.g. the portfolio management on the one hand receives decisions from the requirements management as input, whose decisions in turn provide input for the planning activity of EA management on the other hand.

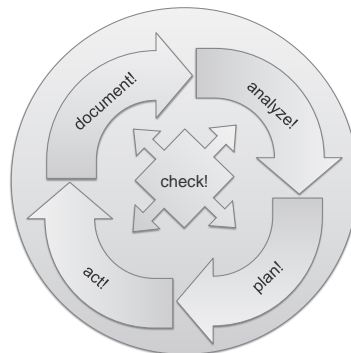


Figure 3.18.: The EA Cycle of Niemann [Ni06c]

The documentation method is concerned with defining the scope and reach of the EA management function, implementing, and populating the model [Ni06c, page 41]. Thereby, typical pitfalls of EA management are sketched and solutions to avoid these shortcomings are proposed. Niemann proposes to structure an EA model according to three main levels called *business architecture*, *application architecture*, and *systems architecture* [Ni06c, page 77]. For each of the layers a detailed descriptions of the elements and relationships contained is given and cross-layer relationships are introduced. Furthermore, functional and non-functional requirements are introduced as cross-cutting aspects, which influence elements on all layers (cf. Figure 3.19).

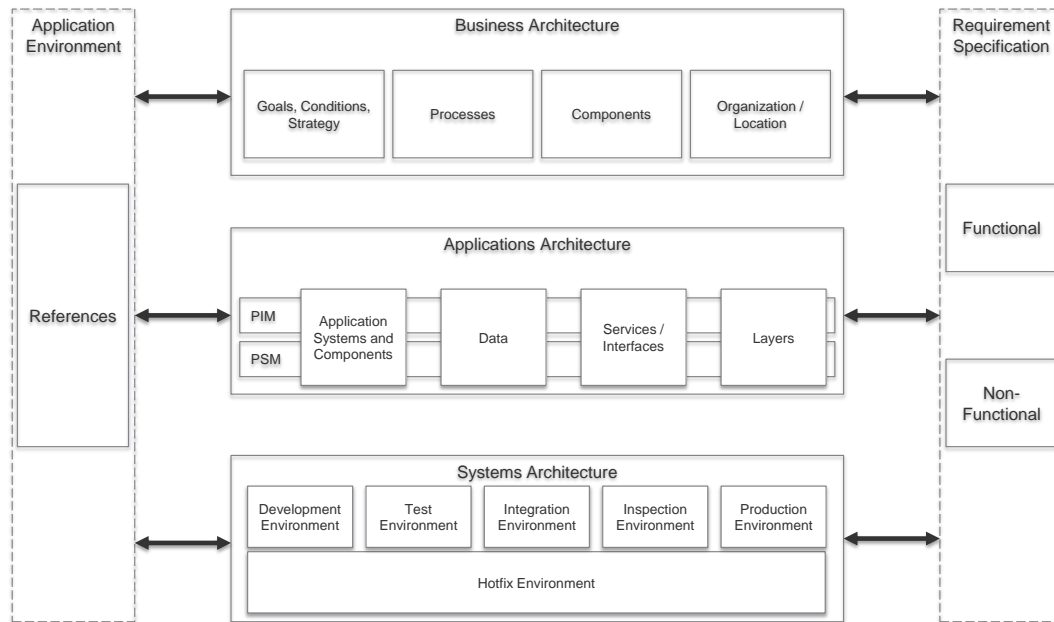


Figure 3.19.: The information model of Niemann [Ni06c]

For the document phase, Niemann emphasizes on the description of the current state of the EA, the development of planned future states is sketched as part of the plan activity. Thus, projects from the project portfolio management are considered in development planning [Ni06c, page 161]. The coverage of projects is neither exemplified nor is the effect of projects on the EA referred to. To derive and establish principles, i.e., reference models in the terminology of Niemann [Ni06c, page 97], guiding the future evolution of the EA, Niemann proposes to evaluate existing development lines and set up standards. Niemann lists different application scenarios for using these reference architecture models but limits their reach to the IT-related elements (cf. [Ni06c, pages 102–105]). While Niemann specifies different viewpoints applicable to populate architectural descriptions, no hints how this population can be performed, e.g. via the intranet or e-mail, are given. Similarly, the importance of providing *stakeholder-specific views* [Ni06c, page 82] is explicitly referred to, while in contrast no dedicated audience for the proposed visualizations is given.

To analyze different states of the EA, Niemann proposes several questions for e.g. evaluating dependency, coverage, heterogeneity, or complexity (cf. [Ni06c, pages 126–152]), which are linked to all architectural layers. Thereby, qualitative as well as quantitative analysis techniques are provided. According to Niemann, these questions can be used to analyze the current state of the EA and identify potentials for improvement. For each of the proposed questions an analysis method is provided. In contrast, methods for evaluating planned states or for performing delta analyses are not provided by Niemann, although he describes criteria for such an analyses, like cost efficiency, ability to reduce risks, or impact on functional requirements [Ni06c, page 163].

To support the establishment of an EA management function, Niemann details on different line organizations of an enterprise, which should be considered and require a different organization of the EA management function (cf. [Ni06c, pages 178–181]). In the same sense, the author discusses the need of “steering” [Ni06c, page 121] EA management, emphasizing on aspects like scope and reach and the need for adapting an EA management function. Nevertheless, precise

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

mechanisms to adapt the methods of the EA cycle or how to include an organization-specific terminology are not given.

Table 3.15 summarizes the analysis results of the EA management approach of Niemann with respect to methodical aspects.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.15.: Method classification for the approach of Niemann

#### 3.3.14. The EA management approach of the University of St. Gallen

EA management approach	
Name of approach:	(Approach of the university of St. Gallen)
Issuing organization:	University of St. Gallen
Focus area:	Method
Tool support:	ADOben [Ai09b, Ai09c]
Period of activity:	since 2007 (2003)
Publications:	[ÖW03], [WF06], [AS07], [Br07], [KW07], [Ös07], [SS07], [Ai08b], [ARW08a], [Fi08], [HW08], [WS08], [Ai09b], [Ai09c], [AW09], [Ku09], [KW09], [RA09], [AG10]
Inner organization:	explicit organization

The University of St. Gallen has a long history in the field of *business engineering* (cf. Österle and Winter in [ÖW03]). The comprehension of business engineering as a holistic perspective for designing organizations in the information-age, nevertheless closely relates to the objectives and goals of EA management. Although abstaining from using the term *enterprise architecture* until the year 2007 in which more and more publications specifically devoted to EA management topics, the engagement of the University of St. Gallen can be dated back to 2003. In 2007 Winter and Fischer introduced a layered framework for the EA [WF07], on which more recent publications of St. Gallen are based. The different architectural layers of the framework reflect a holistic perspective of EA management ranging from business to technology architecture, which the so-called *integration architecture* inbetween as illustrated in Figure 3.20.

According to Winter and Fischer, an EA seeks to provide a *cross-layer view of aggregated artifacts* to address challenges that are not confined to a single layer. Three main aims of EA management are denoted: a) support business IT alignment, b) support business development, especially business re-engineering as well as IS re-engineering, and c) support maintenance. In line with this investigation, Aier et al. apply statistical techniques on the outcomes of a survey in [ARW08a] intended to confirm these aims. In this survey practitioners were asked

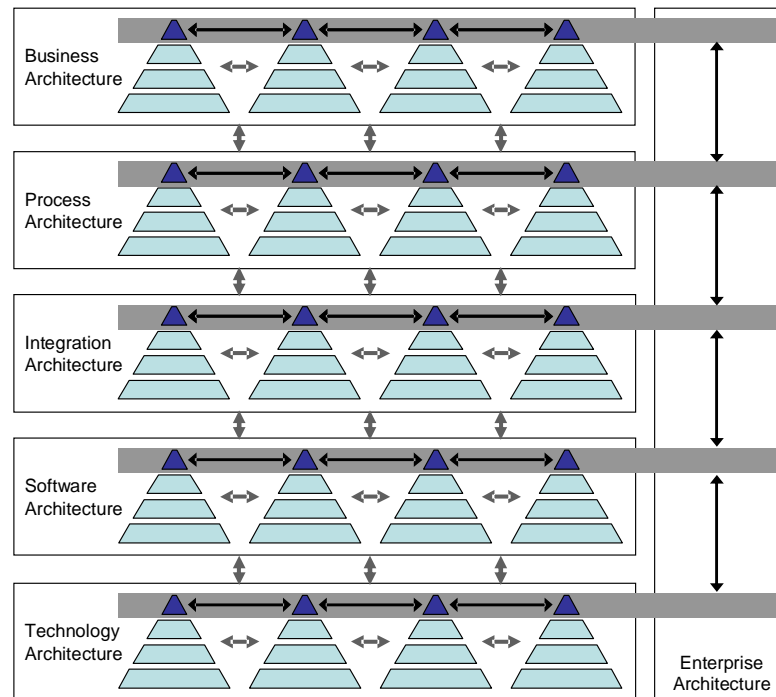


Figure 3.20.: Essential layers of an EA [WF06]

to rate 15 properties of an EA according to the level of implementation experienced in the specific organization. Clustering the survey's results, Aier et al. discover what they call three different *EA scenarios*, reflecting typical stages that an organization managing its EA can be in. In [RA09] Riege and Aier further the above idea towards a *contingency framework* for EA management, i.e., an organized set of factors that may influence the actual make-up of an EA management function in an organization. Relating the factors back to the aims that the corresponding organization seeks to pursue with EA management, Riege and Aier are able to predict, which of the subsequent aims is—based on the contingency factors—most important for an organization:

- support of business strategy development,
- support of business operations, or
- support of IT management.

Further emphasizing on configuration to organizational specificities, Schelp and Stutz approach the topic of EA management in [SS07] from a strategic perspective, thereby showing how to apply the balanced scorecard mechanism onto the topic. They devise the *EA scorecard framework*, which relates the different perspectives of the scorecard to the above introduced EA layers. Further, they delineate the method for applying scorecards consisting of four stereotypic steps (cf. [SS07, page 9]) as 1) *develop strategy and metrics on business level*, 2) *define business goals*, 3) *monitor metrics with the framework*, and 4) *adjust strategy, goals, and metrics*. While abstaining from providing details on the steps, a four step PDCA-like (cf. [De82, Sh86]) method is a recurring principle throughout the EA management approach

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

of the University of St. Gallen. For the control activity of the PDCA cycle, a technique for analyzing EAs is outlined by Aier and Schelp in [AS07], which interprets the EA as untyped graph. The method uses clustering algorithms for determining (candidates for) structuring principles, so called *domains*. This idea is later furthered by Aier and Winter in [AW09] to create proposals on how to organize the decoupling of business and IT. Enhancing the topic of EA analysis, Winter and Schelp reflect in [WS08, pages 549–550] on the different types of analyses that may be performed on an EA organized in layers as shown in Figure 3.20. Basically distinguishing between intra-layer, inter-layer, and extra-layer analyses, they expatiate on seven different kinds of analyses ranging from simple *dependency analyses* over *complexity analyses* to more economically motivated *cost* or *benefit analyses*. These analyses may be understood as techniques embedded into the larger whole of a consistent EA management method.

Taking a holistic perspective on the design of an EA management function, Hafner and Winter discuss requirements and the general make-up of an EA management method in [HW08], demanding at foremost that the model is both *scalable* with respect to the covered part of the organization and *evolutionary* accommodating a changing level of process maturity. Further, they require a method to be *organizationally compatible*, meaning that each organization has its specific culture, stakeholder setting and involved participants, which the method has to adapt to, without detailing on how to perform this adaptation. Deriving from three case studies at Credit Suisse, Die Mobiliar, and HypoVereinsBank as well as from theoretic underpinnings in literature, Hafner and Winter delineate the four core activities of the process model with related sub-activities as show in Figure 3.21.

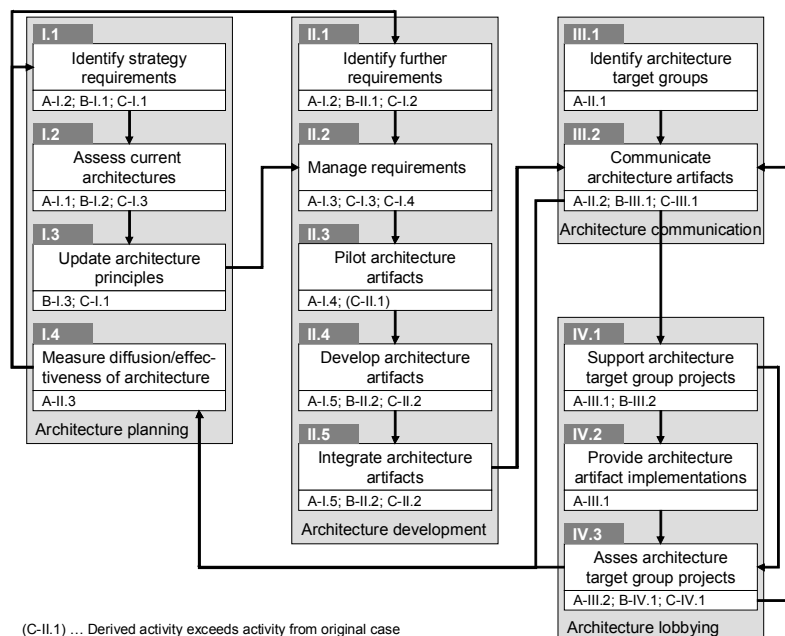


Figure 3.21.: EA process model of Hafner and Winter [HW08]

In line with the KM perspective presented before, the four core activities may be identified as relating to typical knowledge intense disciplines with the possible exception of the *architecture lobbying*, which is quite specific for EA management. This may be ascribed to the EA management function often not being empowered to actually make prescriptions for the organization.



The core activities of Hafner and Winter are further mirrored in the work [Fi08, pages 114–118] of Fischer, although the latter identifies four slightly different core activities<sup>20</sup> namely *strategic dialog*, *architecture development*, *architecture implementation*, and *architecture maintenance*. Following a well-defined method, Fischer details the activities via an intermediary M1-level to specific processes described in an activity-diagram like syntax [Ob10d] on M0-level. Iterating over the the different processes, Fischer describes [Fi08, pages 145–184] the distinct tasks, their execution order as well as the assigned participants in detail. Furthering the multi-level understanding of the EA management method, Fischer further discusses the organizational structures and roles (cf. [Fi08, pages 185–205]) that are required to support the aforementioned M0-level processes. Beside this organizational embedding of the processes, no references to other contingency factors of EA management are provided.

Furthering the findings of Fischer, Aier et al. return in [Ai08b] to an understanding of EA management as a design discipline mirroring characteristics of classical engineering disciplines, summarized in line with Shaw [Sh90] as “creating cost-effective solutions for relevant problems using scientific knowledge in service to society”. From this, Aier et al. derive two relevant consequences, namely the question of *depth vs. width* and a set of general mechanisms used in EA management. While the former question relates to scoping an EA management endeavor in terms of the concerns covered, the general mechanisms described may be used as part of the M0-level processes of Fischer. In particular, Aier et al. describe model navigation mechanisms as well as viewpoints that may be used to comprise specific information to a stakeholder. These mechanisms are revisited by Aier et al. in [Ai09b] and [Ai09c], where they show how the mechanisms can be implemented in a meta-modeling platform (ADOxx<sup>21</sup> of BOC), creating the *business engineering navigator* (ADOben). The question of organization-specificity of EA management that reverberates especially throughout more recent publications on the topic of the University of St. Gallen is central subject to the discussions undertaken by Kurpjuweit and Winter in [KW09]. This article, furthering the considerations from [KW07], especially describes on how to configure an EA perspective to the stakeholder’s architecture perception. In the recent work [AG10, page 60] of Aier and Gleichauf the topic of “EA planning” is discussed and the understanding of different states of the EA (current, planned, and roadmap) is put on a sound methodical basis. In particular, dedicated activities for transformation design, namely a) delta analysis, b) identify projects, c) identify temporal interdependencies, and d) schedule projects, are described and linked to the underlying conceptualizations of the EA. In the light of the results and contributions described above the method related prescriptions of the EA management approach of the University of St. Gallen can be classified as shown in Table 3.16.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.16.: Method classification for the University of St. Gallen

<sup>20</sup>The EA management method consisting of these activities is called *M2* model.

<sup>21</sup>For more information on the ADOxx meta-modeling platform, see e.g. [http://www.openmodels.at/c/document\\_library/get\\_file?p\\_l\\_id=65121&folderId=65129&name=DLE-2505.pdf](http://www.openmodels.at/c/document_library/get_file?p_l_id=65121&folderId=65129&name=DLE-2505.pdf) (last accessed 2010-10-24).

### 3.3.15. Strategic IT management of Hanschke

EA management approach	
Name of approach:	Strategic IT management
Issuing organization:	iteratec GmbH (consultancy)
Focus area:	-
Tool support:	iteraplan
Period of activity:	since 2010
Publications:	[Ha10]
Inner organization:	explicit organization

Using the term *strategic IT management*, Hanschke presents in [Ha10] a pragmatic approach to EA management. By means of providing a *workable toolkit* consisting of a collection of practice-proven prescriptions and guidelines for the EA management context, Hanschke complements existing literature on strategic IT management, which according to her perspective is patchy and lacking real-life application [Ha10]. Reflecting the pragmatic nature of the approach, the majority of statements contained in [Ha10] do not detail on the intricacies of doing research on a sound and coherent terminological basis, leading to issues of ambiguity. The proposed prescriptions and guidelines are implemented in an open source tool *iteraplan*<sup>22</sup>. Thereby, the tool in particular incorporates the language perspective as promoted by Hanschke in [Ha10, page 125]. Central to the language perspective on the management subject, is a set of interrelated sub-architectures as shown in Figure 3.22 (cf. Hanschke in [Ha10, pages 65–67]). Although Hanschke emphasizes on the application architecture and the corresponding management activity of *IT landscape management*, best practices and prescriptions for each of the sub-architectures are provided by the strategic IT management toolkit in [Ha10].

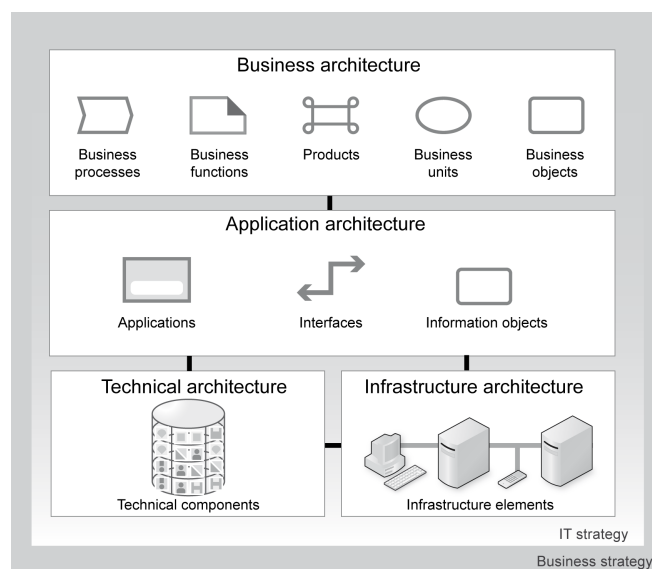


Figure 3.22.: EA framework “Best practice enterprise architecture” of Hanschke [Ha10, page 66]

<sup>22</sup>For more details on *iteraplan* see <http://www.iteraplan.de/en> (last accessed 2010-10-26).

In line with the best practice EA framework, Hanschke presents methodic guidance for the business architecture—*business landscape management* (cf. [Ha10, pages 89–96])—the application architecture—*IT landscape management* (cf. [Ha10, pages 105–218])—and the technical architecture—*technical standardization* (cf. [Ha10, pages 219–250]). Strong emphasis is thereby laid on IT landscape management, whereas no dedicated methodical prescriptions are made concerning the infrastructure architecture. Regarding IT landscape management, Hanschke describes four distinct main-processes: *documenting*, *analyzing*, *planning*, and *governing*. For each of these processes, the IT management toolkit details on activities to be performed, constraints to be accounted for, and participants to be involved in. The level of prescriptions provided can be exemplified with the process of *IT landscape documentation*, for which Hanschke recommends to devise a “maintenance concept” [Ha10, page 232–236], which not only describes, who is responsible for keeping which information up-to-date, but further supplies information on related enterprise-level management functions, e.g. project portfolio management, that may serve as sources of according information. Considering the process of *analyzing IT landscapes* Hanschke describes coarse-grained categories for typical questions and provides a template that can be used to describe organization-specific “analysis patterns” in [Ha10, pages 140–142]. Exemplary applications of this template illustrate, how detailed steps for performing analyses may look like. Thereby, also *delta analyses* are detailed, which can be used to compare current and to-be (plan) states of an IT landscape [Ha10, page 158].

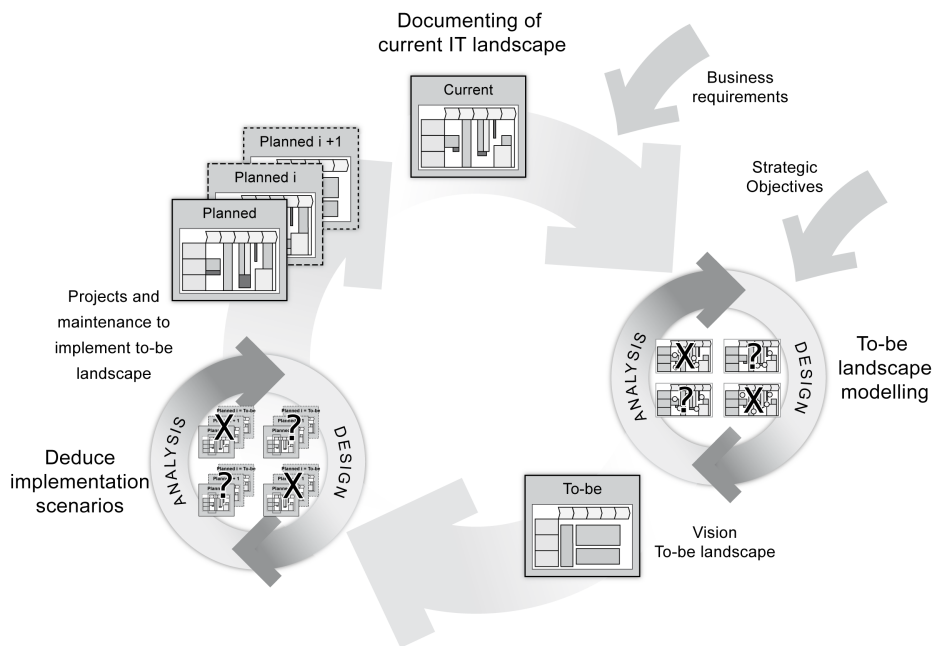


Figure 3.23.: IT landscape planning process according to Hanschke [Ha10, page 158]

Sharing the view on EA management as a management function, Hanschke details on the process of *IT landscape planning* as a circular and ongoing process, which iteratively performs steps of design/documentation and analysis of dedicated landscape states (cf. Figure 3.23). Further detailing this process, Hanschke describes how intermediary planned landscapes can be derived from the to-be landscape, which in turn is based on the current landscape and on additional information on business requirements and strategic objectives. Practice-proven method fragments, so-called *planning patterns*, are thereby applied. Complementing the tri-

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

---

fected of documenting, analyzing, and planning, the topic of communicating EA plans in a stakeholder- and organization-specific manner reverberates through the work of Hanschke in [Ha10]. The communication aspect is further concretized in multiple exemplary visualizations and dedicated communication processes especially for the current landscape along with mechanisms to integrate IT landscape management with other enterprise-level management functions (cf. [Ha10, pages 190–193]). The communication aspect is picked up with respect to standards as part of *technology landscape management*. In detail Hanschke describes steps for developing, maintaining, communicating, and enacting technological standards reflecting EA principles.

Incorporating the idea of *EA management governance*, i.e., system five in terms of the VSM, the strategic IT management approach of Hanschke discusses structures, roles, and responsibilities necessary to successfully implement the toolkit in a using organization (cf. [Ha10, pages 261–312]). Furthermore, different *maturity levels* of IT landscape management (cf. [Ha10, pages 194–206]) describe a possible roadmap for adapting scope and reach of the management function, but also discuss how additional stakeholders may be involved into the management processes by making them *beneficiaries* of the artifacts, documentations, and visualizations. Table 3.17 shows how the approach of Hanschke [Ha10] can be classified with respect to its methodical coverage against the framework devised in Section 3.2. It nevertheless has to be added that the methodical prescriptions are limited to methods for IT landscape management and technology landscape management, hence not targeting an embracing EA management approach.

INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 3.17.: Method classification for Strategic IT management

### 3.4. Summary and conclusion

Recapitulating the above analysis results, indications towards an ongoing consolidation process can be identified in the EA management discipline. Revisiting the discussions of the single approaches a general agreement on the main activities of EA management as derived from the kernel theories can be stated, which can additionally be backed from the perspective of EA management practitioners as the survey of Winter et al. showed in [Wi10]. Notwithstanding these agreement, different approaches emphasize on different aspects not only with respect to the method-language dichotomy, but also with respect to the used level of detail concerning their prescriptions. Revisiting the above approaches against the background of the analysis frameworks from Section 3.2, we could further show that as of today a comprehensive approach addressing all aspects of EA management is still missing. Nevertheless, methodi-

cal prescriptions on varying levels of detail can be found for the different dimensions of our analysis framework as summarized in Table 3.18 and presented subsequently.

**Develop & describe** are well covered by a majority of approaches, especially when it comes to prescriptions on how to document the current state and how to create a vision for a target state of the EA as well as a transformation roadmap. Prescriptions on how to develop architecture principles and how to develop as well as describe architecture questions are on the contrary scarce.

**Communicate & enact** are also well addressed by many approaches, nevertheless with a significant drop in the frequency of communication-related prescriptions, when it comes to architecture roadmaps, architecture principles, and architecture-related questions. Latter fact may nevertheless not be considered a surprise, as different literature on EA management [Ch10, Ku09, La09a] reports on difficulties in communicating stakeholder concerns. This would in turn be a prerequisite of communicating the relevant questions.

**Analyze & evaluate** have a fairly good coverage, but the lowest of the three core processes. In particular, only a few approaches concern themselves with methods for analyzing target states and development of roadmaps. The need for different techniques to perform analyses however is put in different approaches, leading to a first classification in EXPERT-BASED, RULE-BASED, and INDICATOR-BASED analysis techniques as identified by us in [BMS09a].

**Integration** is, despite EA management being a topic heavily relying on coordinating and informing existing management functions, not addressed by about one third of the approaches. Regarding the other approaches both integration scenarios are nearly equally alluded to. This points to the fact that many EA management approaches seek to install EA management as a super- or sub-ordinate management function instead of establishing a dense web of bidirectional linkages.

**Configure & adapt to** are not well covered by the approaches. In particular, questions on how to tailor and re-tailor the methods in a changing organization are not addressed in depth, although some approaches discuss *contingency factors* for an EA management function. In contrary, many approaches do nevertheless mention the need for organization-specific tailoring and give indications on organization-specific aspects as the goals pursued by the EA management endeavor and the organizational context in which the management function has to be embedded.

Building on the overview on the existing knowledge base of EA management method prescriptions, the developed method framework consisting of the main activities DEVELOP & DESCRIBE, COMMUNICATE & ENACT, ANALYZE & EVALUATE, and CONFIGURE & ADAPT is substantiated. Existing approaches and their method prescriptions can be characterized according to this framework. This makes the method framework a good starting point for a mechanism to find, select, configure, integrate, and adapt existing method prescriptions. A prerequisite for such a configurable approach grounded in existing prescriptions is a conceptual foundation, i.e., a model for EA management method prescriptions, which supports organization-specific configuration. The requirements for such a conceptual model are elicited in the subsequent chapter.

### 3. Analyzing the State-of-the-Art in Designing EA Management Functions

INTEGRATION	unidirectional (2, 3, 6, 7, 15)		bidirectional (9, 10, 11, 13, 17, 18, 19, 20, 21)		
DEVELOP & DESCRIBE	current (1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21)	planned (8, 9, 12, 13, 14, 17, 19, 20, 21)	target (2, 3, 4, 6, 8, 9, 10, 11, 13, 15, 16, 17, 18, 21)	principle (8, 13, 18, 19, 21)	question (7, 11, 13, 14, 19, 21)
COMMUNICATE & ENACT	current (1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13, 14, 15, 16, 19, 20, 21)	planned (9, 10, 11, 12, 20, 21)	target (2, 3, 4, 6, 9, 10, 11, 13, 15, 16)	principle (18, 21)	question
ANALYZE & EVALUATE	current (5, 7, 11, 13, 14, 16, 19, 20, 21)	planned (12, 14, 20)	target (11, 13, 15, 21)	delta analysis (4, 8, 12, 20, 21)	
CONFIGURE TO	organizational context (8, 10, 17, 19, 21)		scope and reach (4, 6, 13, 14, 17, 18, 20)		
ADAPT TO	organizational context (9, 10, 21)		scope and reach (9, 13, 14, 18, 20, 21)		

<sup>1</sup> The Zachman Framework

<sup>2</sup> Architecture of Integrated Information Systems (ARIS)

<sup>3</sup> The Integrated Architecture Framework (IAF) [Wo10]

<sup>4</sup> Enterprise Architecture Planning (EAP) [SH93]

<sup>5</sup> The Generalised Enterprise Reference Architecture and Methodology (GERAM)

<sup>6</sup> Semantic Object Model Approach

<sup>7</sup> Multi-perspective Enterprise Modeling (MEMO) [Fr94, Fr98a, Fr98b, Fr99, Ju07, Fr08, He08, Ki08, Fr09]

<sup>8</sup> The Open Group Architecture Framework (TOGAF)

<sup>9</sup> Extended Enterprise Architecture (E2A)

<sup>10</sup> EA management approach of MIT

<sup>11</sup> EA management approach of TU Lisbon

<sup>12</sup> Systemic Enterprise Architecture Methodology (SEAM)

<sup>13</sup> Archimate

<sup>14</sup> EA management approach of KTH Stockholm

<sup>15</sup> Finnish Enterprise Architecture Research (FEAR) [HP04, PH05, Pu06, IL08, LHS08, VS08, PNL07, SHL09, VSL09]

<sup>16</sup> Methodology for (re)design and (re)engineering organizations (DEMO) [Di05, Di06, LD08, ED09]

<sup>17</sup> The EA<sup>3</sup> Cube<sup>TM</sup>

<sup>18</sup> Dynamic Architecture for modelling and development (DYA) [Wa05, St05, BS06, SBB07a, SBB07b, Lu09, SB09, Br10, St10a]

<sup>19</sup> EA management approach of Niemann

<sup>20</sup> EA management approach of the University of St. Gallen

<sup>21</sup> Strategic IT management of Hanschke

Table 3.18.: Summary of method classifications

---

## Towards a Building Block-Based Method to Design EA Management Functions

---

Taking into account the complex management subject an EA management function has to deal with and considering the changing problems EA management endeavors have addressed in the last years (cf. [BES10a, BMS10]), the presumption that “there is no method that fits all situations” [Ha97, page 6] turns out to be true for the context of EA management. As the state-of-the-art analysis in the preceding chapter showed, the topic of designing an EA management function has been approached from various directions in the past with each approach having a different focus. The large number of papers, books, and theses published in the last years on methods, models, and case studies in the context of EA management (cf. Mykhashchuk in [My11]) mirrors the ill-structured nature of the research field lacking theory development either with respect to an embracing approach to EA management or in the sense of integrating competing approaches with differing foci. Incorporating the existing knowledge base on EA management, the integration approach seems to be most valuable to us. Thereby, an organization willing to establish an EA management function, has to perform configuration, i.e., to customize, integrate, and adapt prevalent approaches. Nevertheless, existing approaches typically do not provide techniques and methods for integration and adaptation as the literature review revealed.

In this chapter we establish the foundations for our approach—*building blocks for EA management solutions* (BEAMS)—that closes the aforementioned gap. The outline of the basic solution idea of BEAMS starts with an elicitation of characteristics of the EA management function in Section 4.1. This elicitation uses a twofold approach: First, we seek to understand quality criteria of a ‘good’ EA management function and second we elicit requirements for the design of an EA management function. Starting with an academic perspective, we take into account our lessons learned from the analysis of current EA management approaches in Chapter 3 and revisit existing lists of requirements for an EA management function as proposed by Hafner and Winter [HW08] as well as GERAM (Bernus and Nemes [BN94]). The aca-

demographic perspective is complemented by a practitioners' perspective using a comprehensive set of scenario descriptions we gathered during workshops with industry partners. Section 4.1.1 closes with a comprehensive set of nine quality criteria that describe the intended result of our development method. In contrast, Section 4.1.2 takes a utilitarian perspective and elicits requirements based on the expectations of the intended users of the development method. The comprehensive list of requirements is complemented by the general guidelines of Hevner et al. [He04]. To address the characteristics of EA management functions, Section 4.2 discusses kernel theories from the area of EA management and related domains that lay the groundwork for our development method. Starting with the field of *method engineering* [Br96], we further revisit the contributions of *situational method engineering* [Ha97], explore the idea of *intentional modeling* [Yu96], and conclude with the idea of a *pattern language for EA management* as initially proposed by us in [Bu07c] and developed further by Ernst in [Er10]. Based on the prevalent work, Section 4.3 presents the conceptual basis of BEAMS and the associated development method for designing organization-specific EA management functions. The development method builds on a method base, which we identify with the *design theory nexus* as introduced by Pries-Heje and Baskerville [PHB08] as revisited in Section 2.2.3. While we postpone the detailed description of the constituents of the method base and the development method to Chapter 5, the main constituents of the solution are discussed and their interplay is sketched to outline the general characteristics and design of our solution.

### 4.1. Characteristics of the development method

Answering research question 3 (cf. Section 1.1), this section elicits characteristics, i.e., quality criteria and requirements that the EA management domain exerts on the design of corresponding management functions. In line with the argumentation from Chapter 2, we address a twofold design problem. While an organization-specific EA management function represents a design artifact, this thesis does not aim at the design of a single EA management function for an associated organization, but targets the aspect of 'designing' in general. Therefore the research outcome of this thesis is not a classical design artifact in the sense of Hevner in [He04], but is a development method for design artifacts, i.e., a prescriptive theory in terms of Gregor [Gr06, page 12]. The elicitation of the characteristics is accordingly performed on two levels; the level of the design artifact—the organization-specific EA management function (see Section 4.1.1)—and the method level of developing an organization-specific EA management function (see Section 4.1.2)—the development method. In the former case, quality criteria are derived directly from the EA management domain, while in the latter case, the requirements are elicited against the general knowledge base derived from theoretical and practical experiences in EA management endeavors and further complemented by guidelines from related domains as the *guidelines of modeling* as proposed by Becker et al. [BRS95] and the general *research guidelines for design science* of Hevner et al. [He04, page 83].

#### 4.1.1. Quality criteria of an organization-specific EA management function

Existing EA management approaches typically have a distinct perspective on the problems that can be addressed. This perspective leads to a specific set of quality criteria, which the approach fulfills. As no commonly agreed set of quality criteria or requirements an EA man-



agement function has to fulfill exists, each (research) group has defined its own quality criteria (cf. Op't Land et al. [La09c, pages 22–23]). The different problems discussed in prevalent approaches are nevertheless too detailed and lack the necessary generality for reasoning what characteristics make up a 'good' EA management function. Therefore, we subsequently use four main sources to elicit quality criteria for an organization-specific EA management function on a more general level, namely

- the lessons learned from the literature review presented in Chapter 3,
- the set of requirements for the design of an EA management function as presented by Hafner and Winter [HW08, page 3], which build on the work of Birkhölzer and Vaupel [BV03],
- the requirements specified by GERAM (cf. Bernus and Nemes [BN94, pages 4–5], Bernus and Nemes in [BNS03, pages 14–15], and the ISO Std. 15704 [In99]), and
- a comprehensive set of scenario descriptions gathered in three workshops with 30 industry partners and compiled as part of the *Enterprise Architecture Management Tool Survey 2008* (EAMTS 2008) [Ma08].

Hafner and Winter [HW08] introduced in 2008 a process model for architecture management that according to their own statement can be considered a “reference for establishing organization-specific architecture management” [HW08, page 8]. The process model is developed based on a set of proposed requirements for the EA management function, which is taken from previous work on IT architecture by Birkhölzer and Vaupel [BV03]. With the focus on IT architecture, Birkhölzer and Vaupel elicit in [BV03, page 37] the following requirements an architecture should fulfill: 1) expressiveness, 2) economic and operational efficiency, 3) encapsulation, 4) scalability, 5) capability to evolution, 6) stability, and 7) comprehensibility. These requirements are later applied onto the architecture management process, where especially the *capability to evolution* is emphasized as major paradigm [BV03, pages 50–51]. Building on the idea of Birkhölzer and Vaupel, Hafner and Winter provide in [HW08, page 3] the subsequent list of requirements an EA management approach needs to fulfill:

- **HW1** Since application architecture takes on a crucial role for IT business alignment, an approach must **explicitly address application architecture management**<sup>1</sup>
- **HW2** To operationalize architecture management and to embed it in the organization, an approach must propose **a detailed process model**.
- **HW3** Architecture management should be **scalable** with increasing requirements.
- **HW4** Architecture management must have an **evolutionary character** as its influencing factors mean that it must continually balance the long-term alignment of the architecture against short-term entrepreneurial success.
- **HW5** Architecture management should be **organizationally compatible** with its associated tasks, particularly in information systems management.

---

<sup>1</sup>The term *application architecture management* is used by Hafner and Winter [HW08] as a synonym for EA management. Although application architecture is typically understood to have a narrower focus, we use the term here to stay to the original terminology as employed by Hafner and Winter.

- **HW6** Architecture management should be able to corroborate its effectiveness and efficiency in the form of **performance indicators**.
- **HW7** Architecture management should produce **methodological results** in the form of architecture artifacts such as models, standards, etc.
- **HW8** Architecture management should allow for the constant analysis of its **influencing factors** and its long-term objectives.
- **HW9** Architecture management should allow for the development of **visions for to-be architecture alternatives**.
- **HW10** Obviously, it is not possible to assume a consistent enterprise architecture at a specific point in time, which means there should be a prime focus on **dealing with inconsistencies**.
- **HW11** While maintaining an exchange with its stakeholders and associated task areas, application architecture management should adopt a **service-oriented approach** in the virtual absence of other options for pushing through its points of view or of other benefit arguments.

Joining the list of quality criteria from an academic point of view, the requirements of GERAM (see Section 3.3.3 are presented subsequently (cf. Bernus and Nemes [BN94, pages 4–5], Bernus and Nemes in [BNS03, pages 14–15], and ISO Std. 15704 [In99, pages 4–8]). GERAM is a generalized framework against which EA management-related approaches can be mapped. In other words, GERAM can be understood as “common baseline” or “to-do list” [No03, page 200] for EA management approaches. According to these requirements, an EA management function should

- **G1** cover all activities necessary for describing, designing, implementing, operating, maintaining, and improving enterprises and their constituents.
- **G2** provide a consistent modeling environment leading to executable code, which is modularly constructed, incorporates alternative methodologies, and therefore can be extended (**G2.1**). The modeling environment should not restrict the methodologies used (**G2.2**).
- **G3** encompass an easy to use method, which is understandable and usable by the communities targeted (**G3.1**). The method should identify the application circumstances, which must hold (**G3.2**) and be expandable to incorporate new engineering methods that come into existence (**G3.3**). It should be technically correct and organizational implementable (**G3.4**). The method should specify roles and responsibilities (**G3.5**) as well as allow cost and performance evaluation, i.e., address the need for an economic aspect (**G3.6**). The method should be specified both on a general and detailed level (**G3.7**).
- **G4** allow the adoption of good practices for building reusable, tested building blocks (**G4.1**). The best practice building blocks should thereby be the results of rigorously conducted and theoretically well-founded research to ensure durability (**G4.2**).
- **G5** provide a unified perspective on the constituents of the EA, e.g. processes or products, to tie and relate other enterprise processes (**G5.1**) and support evolutionary integration as well as adaptation (**G5.2**).

Complementing the academic view above, we elicit practitioners' interests and quality criteria for the development of an EA management function. Therefore, we use the set of scenarios that we developed for the EAMTS 2008. Each of the 18 scenarios is described by a core concern on an abstract level, which is further concretized with questions that are answered during performing typical EA management activities. While the set of scenarios was initially developed as a set of requirements for analyzing EA management tools, the scenarios also provide a valuable basis for deriving quality criteria for an EA management function. This is especially true, as the methodical formalism which was applied by us [Ma08, page 4–5] to develop the scenarios comprises the involvement of 30 industry partners from global acting enterprises (e.g. BMW Group, Deutsche Bahn, Deutsche Bank, EWE, Kühne+Nagel, Nokia Siemens Networks, Procter & Gamble, Unicredit Group, and Wacker Chemie). The applied methodology can be summarized in five sequential steps as follows:

1. Derive initial scenario description from project experiences and scientific literature,
2. conduct on-site workshops with the survey's industry partners,
3. revise scenarios descriptions to on-site feedback provided by reviewers and add or reshape scenarios appropriately,
4. conduct remote reviews with industry partners of the survey, and
5. revise and finalize scenario descriptions based on the feedback.

With this methodology, we compiled a comprehensive set of scenarios based on existing literature but further leveraging the knowledge of 30 industry partners, of which a majority (25) sent at least one leading EA representative to the on-site workshops. The developed scenarios are classified into two distinct sets: scenarios concerned with specific tool requirements and scenarios of dedicated EA management tasks. The scenarios of the former set are:

- **E1.1** *Importing, editing, and validating model data* reflects requirements for importing and exporting data from different formats as EA-relevant data sources usually already exist in an organization prior to the introduction of an EA management tool.
- **E1.2** *Creating visualizations of the application landscape* focuses on presentation capabilities, i.e., the creation of EA-related artifacts like matrices, diagrams, reports, or other visualizations.
- **E1.3** *Interacting with, and editing of visualizations of the application landscape* deals with functionalities for handling and interacting with visualizations.
- **E1.4** *Annotating visualizations with certain aspects* refers to the desired capability to annotate visualizations, e.g. with calculated indicators (metrics).
- **E1.5** *Supporting lightweight access* reflects requirements concerned with the multitude of stakeholders participating in the EA management initiative and concerned with easy access to EA-related information, e.g. via a web interface.
- **E1.6** *Editing model data using an external editor* is concerned with the capability to enable offline access and manipulation of data.

#### 4. Towards a Building Block-Based Method to Design EA Management Functions

---

- **E1.7** *Adapting the information model* mirrors the requirement to adapt the tool's underlying conceptual information model of the EA in the initial configuration but also at a later date in the EA management initiative.
- **E1.8** *Handling large scale application landscapes* refers to techniques for dealing with the complexity of the management subject, i.e., large-scale EAs.
- **E1.9** *Supporting multiple users and collaborative work* is concerned with requirements for concurrent working, notification, or the possibility to specify workflows for EA management activities.

While above set of scenarios puts emphasis on desired capabilities a tool for the domain should provide, the second set of scenarios represent specific application scenarios, namely:

- **E2.1** *Landscape management* is concerned with support for describing the current state, future planning, development, and historization of the EA.
- **E2.2** *Demand management* reflects the capability to gather, document, and process demands originating from business or IT and linking them to affected elements.
- **E2.3** *Project portfolio management* is concerned with providing a holistic perspective for the project portfolio management and support the decision making process.
- **E2.4** *Synchronization management* mirrors requirements concerned with synchronizing projects according to their interdependencies, e.g. avoiding conflicts and managing delayed projects.
- **E2.5** *Strategies and goals management* elicits requirements for aligning the EA management initiatives with the organization's strategies and goals, enabling traceability of decisions, and controlling goal fulfillment.
- **E2.6** *Business object management* refers to the capabilities to manage business objects, the operations performed on them, and their exchange between application systems.
- **E2.7** *SOA transformation management* discusses requirements for supporting enterprise transformation with respect to evolving from an application-centric architecture to a service-oriented one.
- **E2.8** *IT architecture management* elicits requirements for standardization and homogenization of applications.
- **E2.9** *Infrastructure management* is concerned with managing the IT infrastructure to reduce operating costs, identify potentials for improvement, and calls for action.

While the former sources represent academic findings, the EAMTS 2008 reflects the practitioners' expectations and points of view. Based on the two perspectives, nine requirements representing quality criteria of a 'good' EA management function are elicited subsequently. The quality criteria are further supported by related work on

- the characteristics of an EA management function as described by Op't Land et al. [La09c],
- critical issues in EA management as identified by Lucke et al. in [LKL10],

- the contingency framework as well as factors as discussed by Leppänen et al. [LVP07] and Riege and Aier et al. [RA09], as well as
- the stakeholder perception of EA as investigated by van der Raadt et al. [RV08].

Finally, prevalent maturity models for EA management are revisited to support the quality criteria (cf. *Architecture Capability Maturity Model* (ACMM) [Un03], NASCIO *Enterprise Architecture Maturity Model* (EAMM) [Na03], and *Enterprise Architecture Capability Maturity Model* (EACMM) [Un07], *Enterprise Architecture Assessment Framework* (EAAF) [Ex09], or *Dynamic Architecture Maturity Matrix* (DYAMM) [St10b]). Using a hermeneutic method (see Chapter 2), we iteratively revisit and re-interpret our findings against the literature. For reasons of readability, only the findings from the last iteration are given subsequently. In this vein, we elicit the following quality criteria for an EA management function.

A result from the state-of-the-art review is the finding that there is no ‘one-size-fits-all’ EA management function (see Table 3.18). This finding backs our assumption that such a method does not exist, instead, prevalent EA management approaches have to be evaluated, if they are appropriate for a specific situation or how they can be adapted accordingly. Requirement **HW5** accordingly demands for organizational compatibility of the EA management function. The EAMTS 2008 further supports this assumption by the approach to evaluate tools. Due to the missing ‘standard’ EA management function no simple ranking of current tools is used to summarize the evaluation results, instead we use a scorecard-based evaluation enabling organizations to match their individual requirements against the different scenarios to “find out which tool best satisfies their specific needs” [Ma08, page 2]. Similarly, requirement **G3.2** demands a method to identify the circumstances under which the method can be applied. Additionally, requirement **HW8** details the role of *influencing factors*, which reflect the impact of the organizational culture, structure, size, or further conditions on the EA management function. Possible influencing factors are identified in the analysis of Riege and Aier in [RA09, pages 391–392] or the *contingency framework of EA method engineering* of Leppänen et al. [LVP07]. Van der Raadt et al. and Lucke et al. further discuss the social skills and organizational status of the enterprise architects as influencing factor impacting the suitability of a method [RV08, page 27] and [LKL10, page 5 and 7]. Similarly, maturity models as the EACMM typically encompass areas dedicated to *governance* aspects, reflecting the influence of the organizational culture or other ancillary conditions, e.g. in terms of management commitment to the EA management initiative [Un07, page 10].

#### Quality criterion Q1: Being organization-specific

An EA management function must be compatible with the specificities of the using organization like the organizational culture, structure, and embedding conditions.

While the management subject of the EA management function, the EA, represents a complex socio-technical system, the management function should be tailored to the specific problems the associated organization wants to pursue. Accordingly, one third of the analyzed approaches from Section 3.3 provides mechanisms to configure the EA management function with respect to the problems of the associated organization. Taking into account the distinct and changing information demands of stakeholders, scalability as expressed in above requirement **HW3** can be ensured by tailoring the EA management function with respect to the problems the associated stakeholders want to address. As yet no standardized information model for EA

management has evolved and indications raise doubts that such a model exists, the scenario *adapting the information model* [Ma08, page 49–50] reflects the demand for problem-specificity (**E1.7**). Complementing the problem-orientation of an appropriate EA management function, GERAM calls for a human-orientation in the sense that the method must be understandable by the communities targeted (**G3.1**). Similar calls for a) making explicit the problems that the EA management function is intended to solve and b) restrict the management function to these problems can be found in Lyytinen et al. [LKL10, pages 6–7], where stakeholder involvement and dealing with complexity are listed as critical issues in enterprise architecting, Riege and Aier [RA09, page 392], where coverage analyses regarding EA data and artifacts are discussed, as well as in Leppänen et al. [LVP07, page 13], who place emphasis on defining the scope of the method. Furthermore, higher stages of maturity models as e.g. the EACMM which deals with problem orientation [Un07, page 8] or the EAMMF, which proposes to scope the management body to address issues of problem-specificity and stakeholder-specificity [Un03, page 8]. Anticipating the presentation of our approach in Section 4.3, we need to clarify the terms problem, goal, and concern in the area of EA management as these are typically used interchangeable by current approaches. In the terminology used in this thesis a **problem** to be addressed by the EA management function encompasses the **concern**, i.e., the area of interest in the system as well as a **goal**, i.e., a description of the envisioned state.

**Quality criterion Q2: Being problem-oriented & stakeholder-oriented**

An EA management function must be tailored to the specific problems, i.e., goals and concerns, of the associated organization to appropriately address the pursued objectives of the stakeholders.

Revisiting the state-of-the-art in EA management, it becomes apparent that the approaches propose methods, which can be characterized by describing, communicating, or analyzing activities. All approaches propose methods for describing, while the amount of approaches also presenting methods for communicating, enacting, and analyzing is limited (see Table 3.18). These main activities are also reflected in the scenario descriptions for dedicated EA management tasks (**E2.1 to E2.9**) of the EAMTS 2008 [Ma08, pages 54–77]. The scenarios are presented in a uniform manner describing which information has to be documented, how to communicate that information via visualizations, and according to which criteria this information should be analyzed to find answers for the problems expatiated by the scenario. GERAM emphasizes the activities of “designing, operating, maintaining, and improving” [BNS03, page 14], which must be covered by an EA management function (**G1**). Similar considerations regarding the main activities an EA management function consists of according to Op’t Land et al. [La09c] are developing architecture descriptions, analyzing, communicating, and enacting these descriptions (see also van der Raadt et al. [RV08, page 27] or Lucke et al. [LKL10, page 7]). Providing methods for shaping the evolutionary character of the EA management function (see requirement **HW2** and **HW4**), an EA management function must accordingly propose a detailed process model encompassing methods for

**develop & describe** Providing an overall picture of the management subject is understood as a prerequisite for managed evolution. Thus, the development and description of different architecture states is considered an important part of EA management for which dedicated methods must be provided (cf. *architecture development* in [Un03, page 8], [Un07,

page 8], or in [St10b, page 3] as well as the capability to develop, maintain, and oversee EAs in [Ex09, page 26]).

**communicate & enact** Enabling and fostering communication as well as enacting plans guiding the evolution of the EA is an important aspect of an EA management function, which is, especially with respect to the current state of the EA, frequently discussed in existing approaches as the literature review showed. Means for communication thereby need to take the specificities of different stakeholders into account, i.e., cover the areas of interest of these stakeholders, and provide a terminological basis for shared understanding (cf. [LKL10, page 7]). The aspect of communication is mirrored in scenario **E1.2** as contained in the EAMTS 2008, which is concerned with *creating visualizations*. Riege and Aier discuss different ways of enacting EA, which they refer to as “organizational penetration” [RA09, page 392]. Backing the importance of communication and enactment, maturity models typically propose own evaluation areas dedicated to the aspect of communication (e.g. *architecture communication* in [Un07, page 9], *consultation* in [St10b, page 3], or deployment of “EA content out to [the] user community” in [Ex09, page 33]).

**analyze & evaluate** Corresponding to the famous quotation ‘you can’t manage what you can’t measure’ means and techniques to analyze and evaluate different states of the EA are frequently referred to in existing literature and also reflected in the EAMTS 2008, in which scenario **E1.4** is concerned with illustrating analysis and evaluation results (cf. *annotating visualizations* [Ma08, pages 45–47]). Riege and Aier explicitly discuss “monitoring of EA data and services” [RA09, page 392] as a contingency factor of EA. A prominent analysis type discussed e.g. in existing maturity models is thereby the so-called *gap* or *delta analysis* (cf. [Un07, page 8] or [Ex09, page 19]).

In line with the above introduced understanding of the main activities, i.e., method descriptions on a general level, and the call for more detailed method descriptions for each of these activities, we support the specification of methods on a general and detailed level (cf. requirement **G3.7** of GERAM). In addition, GERAM calls for a modeling environment for both the language used to describe the EA as well as the language used to describe the EA management function, which leads to executable code. While the former aspect relates to an own area of research dedicated to EA modeling languages [BMS10d], we address the latter aspect by proposing a method modeling language that facilitates transformation into dedicated workflows. The language incorporates the capability to cope with alternatives and extensions while ensuring consistency (**G2.1**).

#### **Quality criterion Q3: Provide detailed method prescriptions**

An EA management function must provide detailed methods containing prescriptions to develop & describe, communicate & enact, and analyze & evaluate the EA.

To ensure applicability, a management function must define when (trigger) and who (responsible role) has to perform which activity. Although existing approaches more often than not neglect this aspect, practitioners call for EA management tools encompassing the capability to specify workflows and support collaborative work (cf. the scenarios *supporting multiple users and collaborative work* (**E1.9**) [Ma08, pages 51–53] and *lightweight access* (**E1.5**) [Ma08, pages 48–49] of the EAMTS 2008). Taking into account the amount of stakeholders that serve as information suppliers in the context of EA management, defined triggers, roles, and

responsibilities for each task must be specified to ensure execution of steps (**G3.5**). Similarly, governance structures as boards, committees, and councils have to be established. Van der Raadt et al. [RSV08, page 27] in addition name awareness for these roles within the organization and accountability of the enterprise architects for their advices and the outcome of their work as key attributes of an EA management function. In agreement, Lucke et al. have identified “insufficiently defined roles, responsibilities, processes, and procedures” and undefined “timelines” as critical issues of EA management [LKL10, page 6]. Requests for defined triggers, roles, and responsibilities for the activities and tasks constituting the EA management function can further be found in [LVP07, page 11] and in higher stages of maturity models (cf. [Un03, page 7], [Na03, page 9], [Un07, page 7], and [St10b, page 50],).

**Quality criterion Q4: Define triggers, roles & responsibilities**

To operationalize an architecture management, the EA management function must define triggers, roles, and responsibilities for each task.

To foster communication among the different stakeholders of EA management, artifacts describing certain parts or aspects of the EA are created. The literature review revealed a prominence of descriptions concerning the current state and envisioned long-term future states (cf. Table 3.18). Additional short-term future states, i.e., planned states, are only discussed by about half of the approaches. The documentation of principles and questions as prerequisite for a managed evolution plays a minor role. Comparing the findings from the literature review with the practitioners’ point of view from the application scenarios of the EAMTS 2008, the *landscape management* scenario (**E2.1**) [Ma08, pages 55–57] requires further investigation. The scenario describes a distinction between current, planned, and target states of the EA as a prerequisite for managing the evolution of the EA. Dealing with inconsistencies as demanded by requirement **HW9** and **HW10**, the scenario **E2.1** of the EAMTS 2008 also discusses that different variants of a planned state can exist. The role of artifacts in general is frequently alluded to in the second set of scenarios of the EAMTS 2008 as for each scenario (**E2.1** to **E2.9**) the expected resulting artifacts are described. In addition, scenarios dedicated to visualization aspects are contained in the first set. Examples for such scenarios are *creating visualizations*, *interacting with and editing of visualizations*, and *annotating visualizations* [Ma08, pages 41–48] (**E1.2-E1.4**). Further support for the importance of EA artifacts as results of the EA management function can be found in requirement **HW7**. A distinction of descriptions as relating to current, planned, and target architecture states is forced by Op ’t Land et al. in [La09c, pages 22-23], van der Raadt and colleagues in [RSV08, pages 27–28], Riege and Aier in [RA09, page 391], as well as by existing maturity models (cf. [Un03, page 8], [Un07, page 8], [Ex09, page 18], and [St10b, page 50]). Notwithstanding the importance of EA artifacts, GERAM emphasizes the autonomy of the method from the language, i.e., demands that the modeling language used for creating the EA artifact should not restrict the methodologies used (**G2.2**).

**Quality criterion Q5: Being artifact-centric**

An EA management function must produce results in the form of EA artifacts such as models, visualizations, or guidelines.

To be implementable in an organization, the EA management function must describe a coherent and sound process, which builds on available resources of the associated organization.



Driven by the diversity of approaches to EA management, which typically have differing focus and are developed for varying purposes (see Table 3.18), today's organizations face the challenge to integrate or adapt existing approaches. Thus, terminological differences between the approaches have to be eliminated and consistency as well as coherency between different approaches have to be ensured. Consistency issues can, for instance, arise, if the approach to document the current state for instance does not provide the information necessary for the successive analysis from another approach or, if the resources necessary for conducting a certain activity or task are not available in the organization. While the criteria to be organizationally implementable is obvious, it is not directly alluded to in our literature review. This quality criteria is nonetheless regarded a prerequisite of EA management (**G3.4**) and a critical success factor for the EA management function cf. Lucke et al. [LKL10, page 8]. The authors further discuss the importance of sufficient tool support, by which aspects of coherency, consistency, integration, and implementation can be ensured. Some maturity models further address the aspect of organizational implementability either direct or indirect via tool support (cf. [Un07, pages 7–8] and [Ex09, page 33]).

**Quality criterion Q6: Being implementable**

To support organizational implementation, an EA management function must be detailed, executable, consistent, and self-contained.

Two-thirds of the approaches analyzed in our literature review deal with aspects of establishing links between the EA management function and related enterprise-level management functions (see Table 3.18). Emphasizing the role of integration with other management functions, the majority of these approaches discusses bidirectional links. The integration aspect is also mirrored in the scenarios of the EAMTS 2008, of which one is concerned with *importing model data* (**E1.1**) from other sources as related management functions (cf. [Ma08, pages 40–41 and page 350]). Hafner and Winter call for a service-oriented approach to foster exchange with stakeholders from associated tasks (see requirement **HW11**) and van der Raadt et al. discuss “collaboration between architects” [RSV08, page 27] as important means to establish an EA management function. Supporting the importance of an integrated EA management function GERAM advocates for a unified perspective on the EA to tie and relate existing management functions (**G5.1**). Reflecting the criticality of establishing information exchange and linking existing management functions, like demand or project portfolio management with the EA management endeavor, maturity models for EA management describe evaluation criteria for classification, e.g. “The agency uses its EA to inform strategic planning, information resources management, IT management, and capital planning and investment control processes” [Ex09, pages 22 and 26] and “organization works with other states to share ideas for improved integration, including procurement and project management practices” [Na03, page 13]. Further support for the need for an integrated EA management function can be found in [Un03, page 9], [Un07, page 8–9], and [St10b, page 50].

**Quality criterion Q7: Being integrable**

An EA management function must be embedded in the context of surrounding management processes, therefore links in both directions should be defined.

EA management typically represents a long-term endeavor, which requires a high investment to be started, e.g. to set up the governance structure, develop an information model, perform

initial information gathering, and convince the stakeholders. Therefore the overall performance of the EA management function should be made visible, i.e., measurable. Although no analysis dimension in the analysis framework as presented in Section 3.2 is directly concerned with performance measurement, the dimension of adaptation is related to performance measurement. An adaptation is typically triggered, if the objectives have not been reached, or if the organization matures. Although the literature review revealed that only one third of the analyzed approaches discusses performance measurement of the EA management function (cf. Table 3.18), we in line with requirement **HW6** of Hafner and Winter advocate for an EA management function, which is able “corroborate its effectiveness and efficiency” [HW08, page 3]. Similar, GERAM in requirement **G3.6** calls for addressing the economic aspect of EA management by assessing the performance of the EA management function. An assessment of the EA management function can thereby be performed subjectively, e.g. evaluating stakeholder or management satisfaction (cf. [La09c, page 22] or [RA09, page 392]), or objectively, e.g. measuring time-to-market of products, degree of heterogeneity, or return on investment [LKL10, page 6]. Complementing, EA management maturity models discuss performance measurements of the EA management function typically in the higher maturity stages (cf. [Un03, page 7], [Un07, page 7], [Ex09, pages 39–40], and [St10b, page 50]).

**Quality criterion Q8: Facilitate performance measurement**

An EA management function must provide means and techniques to measure its effectiveness and efficiency.

Revisiting existing approaches to EA management, the diversity of problems to be addressed by the proposed methods becomes obvious. In line with this diversity some of the approaches (about one third) discusses sustainability of the EA management function in terms of the need for encompassing methods and techniques to adapt the management function to changing organizational contexts or changing scope and reach, i.e., the concerned management subject (cf. Table 3.18). With regards to requirement **HW10**, we demand an EA management function to incorporate means for adaptation, which ensure sustainability. Scenario **E2.2** also reflects this demand. This scenario named *SOA transformation* is concerned with the transformation of an application-oriented architecture to a service-oriented one [Ma08, pages 69–72]. Similarly, Lucke et al. list “rapidly changing conditions” [LKL10, page 8] as a challenging situation, the EA management function has to deal with or has to adapt to. GERAM specifies the ability to support evolutionary integration and adaptation (**G5.2**) as a core quality that needs to be provided by EA management approaches. Prevalent maturity models for EA management emphasize the need for change as e.g. the EAMMF discusses the capability of “leveraging the EA to manage change” in which the aspect of “adjustments [which] are continuously made to both the EA management process and the EA products” is stated (see [Un03, page 7], [Un07, page 7], [Ex09, page 40], and [St10b, page 50] for further examples).

**Quality criterion Q9: Being sustainable & adaptable**

An EA management function must be sustainable in the sense that it must be adaptable to changing organizational contexts and goals to be pursued.

To discuss the completeness of the above list of quality criteria, we subsequently shortly revisit the sources from which the quality criteria have been derived. Table 4.1 provides an overview on the elicited quality criteria and details on originating sources.

Quality criteria		Hafner & Winter	GERAM	EAMTS 2008
<b>Q1</b>	Being organization-specific	HW5, HW8	G3.2	
<b>Q2</b>	Being problem-oriented & stakeholder-oriented	HW3	G3.1	E1.7
<b>Q3</b>	Provide detailed method prescriptions	HW2, HW4	G1, G2.1, G3.7	E2.1-E2.9, E1.2, E1.4
<b>Q4</b>	Define triggers, roles & responsibilities		G3.5	E1.5, E1.9
<b>Q5</b>	Being artifact-centric	HW7, HW9, HW10	G2.2	E1.2-E1.4, E2.1-E2.9
<b>Q6</b>	Being implementable		G3.4	
<b>Q7</b>	Being integrable	HW11	G5.1	E1.1
<b>Q8</b>	Facilitate performance measurement	HW6	G3.6	
<b>Q9</b>	Being sustainable & adaptable	HW10	G5.2	E2.2

Table 4.1.: Quality criteria for EA management function and the originating sources

Starting with the requirements identified by Hafner and Winter [HW08, page 3], all requirements except requirement **HW1** are taken up. Requirement **HW1** is not considered a quality criterion in itself as it is obviously satisfied by a method for developing organization-specific EA management functions. The scenarios **E1.3**, **E1.6** and **E1.8** from the EAMTS 2008 are also not considered in our list of quality criteria as they centrally target EA management tool support. While the other scenarios could be easily used to derive general quality criteria of an EA management function, the aforementioned three scenarios are specific tooling scenarios and therefore not considered. Concluding, we critically reflect the requirements brought up by GERAM. In Table 4.1 the requirements **G4.1**, **G4.2** and **G3.3** are missing. As already mentioned at the beginning of this section, we distinguish between quality criteria for the design artifact, i.e., the EA management function, and requirements for designing, i.e., the development method. Therefore, the requirements **G4.1**, **G4.2**—*adoption of good practices by a rigorous method*—and **G3.3**—*expandable to incorporate new methods*—are discussed in the next section. A ‘good practice’ according to GERAM can be understood in two different ways. Firstly, good practices can refer to templates of architectures, i.e., “solution building blocks” in terms of TOGAF [Th09a, page 501–502], and secondly good practices can relate to templates for architecture management, i.e., “EA management patterns” [Bu08b]. In line with our pattern-based approach [Bu08b], practices of the second type can be used to design an EA management function. As the left requirements are concerned with the process of designing, we postpone their discussion to the next section.

#### 4.1.2. Requirements for a development method to design EA management functions

Complementing above requirements for an EA management function, we subsequently elicit requirements targeting a development method that builds on practice-proven solutions to develop organization-specific EA management functions. We subsequently elicit requirements for

this method against the background of existing approaches, their deficiencies and benefits, as well as the practical experience we gained in past EA management-related projects. As already motivated in Section 1.1, three different ways to develop an EA management function can be distinguished, namely the a) *greenfield approach*, b) *customization approach*, or c) *integration approach*. Against the background of the requirements **G4.1** and **G4.2** of GERAM [BNS03, page 14], we consider the customization and integration approaches to be most useful for a method to develop organization-specific EA management functions. Whereas both approaches benefit from the fact that they are based on practice-proven method prescriptions, they also entail the challenge of incorporating techniques for selecting the most appropriate method (parts) for the situation at hand and for keeping the resulting method coherent and consistent. Thereby, inconsistency in the case of customization can arise, if parts of the overall method are restricted or omitted, while in the context of integration inconsistency can occur as model fragments from different sources are integrated. Caused by the similarity of challenges of both approaches, requirements for the development method can be elicited subsequently independent from the approach taken. The requirements are captured in Section 4.1.2.1 from a utilitarian perspective and are further supported by the *guidelines for modeling* (GoM) of Becker et al. [BRS95] as well as Schütte and Rotthowe [SR98], which have already been applied in the context of reference modeling [Sc98] and business process modeling [BRU00]. Coming back to the fundamentals, the guidelines formulated by Hevner et al. [He04, pages 86–90] for design science research in general are revisited in Section 4.1.2.2 and the development of the method itself is discussed.

### 4.1.2.1. Domain-specific requirements

Eliciting requirements for a method demands as a prerequisite an understanding of the intended audience, i.e., the targeted user community, of the development method. In the context of EA management approaches, the user group of the method is typically only implicitly defined, e.g. in the ‘who should read this document’ sections of the frameworks. However, the role of the **enterprise architect** is frequently used in method descriptions of the EA management function and can therefore be assumed to be the key user group of a method to develop an EA management function. Supported by statements of practitioners in [Bu09b, page 20], the profession of enterprise architects is assumed to have a particular skill set enabling them to deal with the intrinsic complexity of the EA management subject, which makes them the most appropriate user group for our development method.

According to James “a good enterprise architect needs not only excellent technical skills, but business and behavioral competencies as well” [Ja02]. These skills are further detailed by Strano and Rehmani [SR07, pages 386–389], who identified core competencies needed by enterprise architects, namely

- analytic and problem solving skills to identify problems and break them down into manageable pieces by structuring them (e.g. into goals and concerns),
- communication skills to communicate concepts to stakeholders from various disciplines and bridge communication gaps by translating between different terminologies,

- leadership skills to scope and run the architecture program, motivate people involved, and emanate an image of helping rather than hindering (interpersonal skills), as well as
- modeling skills to structure reality and conceptualize the enterprise to illustrate complex relationships, e.g. the impact of projects.

In line with the above discussion, we assume enterprise architects to be the most appropriate user group of our development method, although the enterprise architects typically are not the only stakeholders raising problems, i.e., goals and concerns, to be addressed by the EA management function. The enterprise architects nevertheless typically act as mediators for the different stakeholders, who develop an appropriate method to address the stated goals and concerns with the help of the aforementioned skills. In this vein, we assume that enterprise architects with the above competencies are available to use the development method for which the following requirements must be met.

**Requirement R0: Produce a ‘good’ EA management function**

The result of an application of the development method must be a ‘good’ EA management function in the sense of the stated quality attributes (requirements Q1–Q9).

A method according to the IEEE is “a standard that describes the characteristics of the orderly process or procedure used in the engineering of a product or performing a service” [IE91, page 10]. This definition already emphasizes the outcome of a method, which in the case of our development method is the organization-specific EA management function. In line with above requirements for a ‘good’ EA management function, the development method should incorporate techniques, which ensure that the requirements **R1–R9** are met.

**Requirement R1: Being easy to use**

The development method and incorporated techniques must be easy to use by the users, who are familiar with the underlying terminology and conceptualization.

With the complexity of the management subject in mind and the need for customization due to organizational specificities, it is obvious that the development method cannot be used without prior familiarization with the concepts entailed. In particular the terminology used and the criteria as well as concepts relevant for selecting and filtering the method prescriptions, e.g. organizational contexts and problems, must be understood by the prospective users. Therefore, a uniform and consistent way of presenting the method prescriptions as well as structural guidelines for arranging and filtering them must be provided. This enables the users after an initial phase of familiarization to intuitively understand and use the method prescriptions. In this vein, the development method adheres to the *principle of clarity* of GoM [SR98, pages 248–249] and [BRU00, page 33], which states that a good model needs to be addressee-oriented and comprehensible.

**Requirement R2: Providing practice-proven prescriptions**

The development method must provide the users with practice-proven design prescriptions as well as context-specific method fragments for EA management.

Notwithstanding the importance of a theoretical foundation, this requirement reflects the demand for practical relevance. In other words, the prescriptions made by the method must have proven themselves as appropriate solutions to problems in dedicated contexts of practical relevance. Thus, a prescription must fulfill three characteristics to be adopted. First, the method fragments and contextual descriptions must be abstracted from observed practice cases instead of being theoretically invented. Second, the method prescriptions must have proven their applicability through repeated occurrences and third, the method prescriptions must have proven to be relevant for practitioners. This requirements aligns with the *principle of relevance* of GoM [BRU00, page 33], which demands that any model selects a relevant universe of discourse and thereby abstains from modeling elements that can be eliminated without loss of meaning to the users.

**Requirement R3: Ensuring consistency**

The development method must be formulated in a consistent terminology with respect to the method fragments and the conceptualization used to enable configuration, i.e., customization, integration, and adaptation.

Following the idea of a development method, which supplies practice-proven method prescriptions, aspects of coherency and consistency arise. As comprehensibility is a prerequisite of using any method, semantic and syntactical correctness needs to be ensured. This leads to the demand for a clear conceptualization and the use of a consistent terminology. In particular consistency issues arise, if a method building on prescriptions from various origins as practitioners' experience, academic research, or consultancy knowledge is considered. The terms that make up the corresponding terminology need to be embedded in a consistent overall semantics to avoid ambiguities and possibly conflicting understandings, e.g. with respect to defined roles and responsibilities or descriptions of the organizational context in which a method prescription can be applied. A consistent conceptualization and meta-model further facilitate a well-defined syntactical structure, which is employed by the development method for providing the method prescriptions. This requirement aligns with the *principle of correctness* of GoM [BRU00, page 32], which requires a model to be syntactic and semantically correct and additionally emphasizes the aspect of consistency between different models.

**Requirement R4: Being systematic extensible**

The development method and the corresponding method base must be adaptable and extensible with respect to new practice-proven prescriptions.

With EA management being continuously forced to adapt to the changing environment, the discipline itself has to develop techniques and means to adapt the corresponding management function. Influencing factors as economic diversification, regulatory forces, social changes, or technical innovations shape and change the way EA management is conducted. These changes can directly affect the applicability of single method prescriptions or give rise to new ones addressing these changes. The development method accordingly needs to incorporate parts, which support adaptation of existing method prescriptions and extension with new ones that come into existence while ensuring usability and applicability of the overall method. Therefore, the adaptation and extensions must adhere to the underlying structure of the development

method and the provided method fragments. Dealing with inter-model consistency, i.e., consistency of the method prior to an adaptation or extension and afterwards, this requirements reflects the *principle of systematic design* of GoM [SR98, page 249] and [BRU00, page 33].

**Requirement R5: Enabling tool supported design**

The development method must be supported with a tool that guides the users in conducting the steps of the development method as well as supports the users in accessing the method base and glossary of used concepts.

Revisiting the variability of the design of EA management functions and recalling the plurality of different method prescriptions as proposed by existing EA management approaches (cf. Section 3.3), tool support to access, filter, select, and configure the method fragments is necessary. The amount of method fragments, which the development method supplies to design an EA management function, requires tool support for the users to identify appropriate method prescriptions. Furthermore, a tool can guide the users through the single steps of the method and document decisions made, e.g. selections, customization, and adaptation and support remaining ones. Thus, traceability of decision and consistency of the configuration made is ensured by the tool. The requirement for tool supported design, aligns with the *principle of economic efficiency* of GoM [SR98, pages 247–248] and [BRU00, page 33] in the sense that tool support lowers the overall investment of the users in using the method.

**Requirement R6: Being integrable with other EA management approaches**

The development method and corresponding method fragments must be linkable with method prescriptions of prevalent approaches to enable integration.

As typically EA management initiatives do not start with a *greenfield approach*, the development method should support integration with prevalent approaches. Especially the method fragments that are supplied by the development method should not be developed to be used in isolation, but instead should facilitate integration with other approaches, in particular those from which they originally have been derived. To enable the integration, the method fragments which adhere to a consistent terminology must retain links to their origin. The integration requirement reflects the *principle of comparability* of GoM [SR98, page 249] and [BRU00, page 33], which demands each ‘good’ model must be comparable with other models.

#### 4.1.2.2. General research guidelines of Hevner et al.

Complementing the requirement elicitation for our development method, we resort ourself to the fundamentals of designing and revisit the general design guidelines as formulated by Hevner et al. in [He04]. According to Hevner et al. design science is inherently a problem-solving process in which “knowledge and understanding of a design problem and its solution is acquired in the building and application of an artifact” [He04, page 82]. Thereby, a dichotomy regarding the term *design* has to be taken into consideration, as design describes both a product and the process leading to the product. The design problem addressed in this thesis, the innovative and purposeful artifact [He04, page 82] should not be mistaken to be an organization-specific EA management function, but an appropriate method to develop

such management functions. Hevner et al. have proposed seven guidelines for conducting and evaluating ‘good’ design science research, which we subsequently summarize and link to the contributions of our thesis.

The first guideline—*design as an artifact*—requires that the result of design science research is a purposeful and viable artifact that addresses an important organizational problem. The artifact must be described effectively to enable its implementation and application in an appropriate domain. An artifact in terms of Hevner et al. as well as March and Smith is a *construct, model, method, or instantiation* or a combination of those types [MS95, page 256–258] and [He04, page 82–83]. The different types of artifacts can in the context of this thesis be mapped to our contributions as follows.

**Constructs** provide a conceptualization of the problem, context, and solution domain on which a common language used for discussions on the field can be built. In the context of this thesis, appropriate constructs<sup>2</sup> for organizational context descriptions and problems as well as an adequate conceptualizations for the method prescriptions are developed as part of the contribution.

**Models** capture the structure of reality by a set of properties or statements expressing relationships among constructs with respect to utility, thereby representing problems and solutions. With respect to our contribution, models link problem descriptions, i.e., EA-related problems and organizational context descriptions, to the method prescriptions in the method base.

**Methods** define processes, i.e., represent a set of steps, providing guidance on how to solve problems. The method is thereby based on a set of underlying constructs and a model of the solution space. In terms of our contribution, the method defines which concepts from the problem and context domain have to be documented by the users and how an appropriate solution from the solution domain can be derived, integrated, configured, and adapted by using which techniques.

**Instantiations** are the realization of an artifact in an environment, i.e., represent problem-specific aggregates of constructs, models, and methods, to demonstrate feasibility and effectiveness of the artifact. As discussed in Section 2.2.4 a nexus-based IS generator represents the instantiation in our contribution, which guides the users in performing the steps of the development method, accessing the model, i.e., the method base, and in applying the relevant techniques leading to an organization-specific EA management function.

Guideline two of Hevner et al.—*problem relevance*—demands that problems addressed through design science research represent “heretofore unsolved and important business problems” [He04, page 84]. The objective of the constructed innovative artifact is to change the phenomena that occur. Therefore, a combination of technology-based, organization-based, and people-based artifacts is necessary. In the light of our contribution, this means that enterprise architects commissioned with the task to design an EA management function should be supported in a way that they create ‘good’ EA management functions, i.e., management functions fulfilling the quality criteria **Q1–Q9**.

---

<sup>2</sup>Constructs in this sense correspond to the meta-concepts of problem, context, and solution domain as discussed in Section 2.2.



The third guideline—*design evaluation*—targets the rigorous demonstration of utility, quality, and efficacy of the designed artifact. Evaluation thereby includes integrating or applying the design artifact in the target environment and evaluating which requirements are satisfied. Due to the iterative and incremental nature of design, the evaluation phase provides essential feedback to the construction, which can be used to enhance the constructs, models, methods, and instantiations in an iterative manner. For performing the evaluation, a variety of different methods is available of which in particular observational techniques in the form of case studies are used in our contribution (see Section 2.3).

Guideline four of Hevner et al.—*research contributions*—deals with the research contributions of effective design science research. According to Hevner et al. research can always be assessed, whether it provides a clear answer to the question “What are the new and interesting contributions?” [He04, page 87]. Three different types of research contributions can be distinguished of which at least one should represent the result of each research project, namely

**design artifact** providing a solution to a heretofore unsolved problem, thereby it may extend the knowledge base,

**foundations** representing novel, appropriately evaluated constructs, models, methods, or instantiations, thereby refining and improving the existing knowledge base

**methodologies** introducing new ways to evaluate design science research contributions.

According to Hevner et al. each of the above research contribution types can be assessed with a focus on “representational fidelity” and “implementability” [He04, page 87]. Our work contributes new foundations in terms of constructs, models, methods, and instantiations for developing organization-specific EA management functions.

Relating to the challenge of relevance versus rigor, the fifth guideline of Hevner et al.—*research rigor*—deals with the ways in which design science research is conducted [He04, page 87–88]. Thereby, the use of rigorous methods in the construction and evaluation of the designed artifact is required. Rigor can be derived from the effective use of the knowledge base. In other words, rigor depends on the researchers’ skill to select the appropriate techniques to develop, construct, and evaluate the artifact. For our contribution, rigor has to be considered in a twofold way. First, the findings presented should be grounded in the theoretical foundations of the discipline and its literature (cf. Section 3.3) and second, the method base and the development method integrating the method prescriptions has to be evaluated.

Guideline six of Hevner et al.—*design as a search process*—accounts for the iterative nature of design science research [He04, page 88–90]. As design science research is typically carried out in the context of wicked problems, the size and complexity of the solution space makes it infeasible to identify an optimal solution in one step. Simon therefore describes the design process as a *generate/test cycle* [Si96, page 128–130]. Iterative search procedures are typically used in this cycle, i.e., alternative solutions are created and tested against the elicited requirements. The requirements elicitation requires knowledge of the application, i.e., problem and context domain and the solution domain. Furthermore, decomposing the design problem into sub-problems is another beneficial strategy, which is used in the context of this thesis to develop building blocks for designing the EA management function. However, the solutions have to be analyzed for unforeseen interdependencies that arise after integration.

Final guideline seven of Hevner et al.—*communication of research*—is concerned with populating the designed artifact to receive feedback [He04, page 90]. Thus, two different target communities have been proposed by Hevner et al. Communication to a technology-oriented audience is required for which in particular the process by which the artifact was constructed and evaluated should be explained. This on the one hand enables practitioners to beneficially use, replicate, as well as instantiate the artifact and on the other hand enables researchers to build a cumulative knowledge base facilitating further extension and evaluation. In addition, communication to the management-oriented audience is required. To enable decision makers to determine, if the designed artifact should be used in their specific organizational context, the focus of the presentation should be put on knowledge required to effectively apply the design artifact. Our research findings are made available on the one hand via the nexus-based IS generator and on the other hand discussed and publicized in different scientific conference proceedings.

Keeping the above quality criteria, requirements, and the guidelines for design science research of Hevner et al. in mind, we subsequently discuss kernel theories relevant for the solution domain. The requirements and guidelines are additionally picked up in Chapter 7 to evaluate the design artifact.

### 4.2. Contributing theories

Preparing our method to develop organization-specific EA management functions, we subsequently reason on kernel theories for the solution domain. Starting with investigating the fundamentals of *methods* in Section 4.2.1, we discuss the constituents of a method as proposed by Gutzwiller [Gu94]. Based on the developed understanding we revisit the disciplines of *method engineering* introduced by Brinkkemper [Br96] in general and of *situational method engineering* as presented by Harmsen [Ha97] in special in Section 4.2.2. Taking into account the problem-specificity of the EA management function and the influence of stakeholders as discussed in the preceding section, we sketch the discipline of goal-oriented and actor-oriented modeling by an overview on the i\* framework of Yu [Yu96] in Section 4.2.3. Complementing the list of contributing theories, we introduce our *pattern-based* approach to EA management as initially described in [Bu07d] and furthered by Ernst in [Er10] in Section 4.2.4.

#### 4.2.1. The fundamentals of methods

While a common understanding of the term ‘method’ as a procedure which is planned and systematic in terms of its means and purpose exists, IS practitioners and theoreticians frequently use the term without clearly defining its means and purpose. This circumstance can best be exemplified if the terms *method* and *methodology* are considered that are frequently used as synonyms. We in line with Brinkkemper sense that the “misuse of the term methodology standing for method is a sign of the immaturity of our field, [... which] should consequently be abandoned” [Br96, page 276].

A method according to Gutzwiller is described on the basis of the elements “activity<sup>3</sup>, role, result, meta-model, and technique” [Gu94, pages 12–15]. Figure 4.1 relates the fundamental elements of a method as adopted from Gutzwiller by Gericke and Winter [GW06, page 231]. *Activities* are construction tasks which create certain *results*, i.e., specification documents or artifacts, and are executed by *roles*. The results of an activity are documented via a predefined specification document, which in turn is compiled using a certain *technique* that provides detailed instructions for the development of a certain type of result. *Tools* support the application of the techniques and the *meta-model* specifies the conceptual data model<sup>4</sup>.

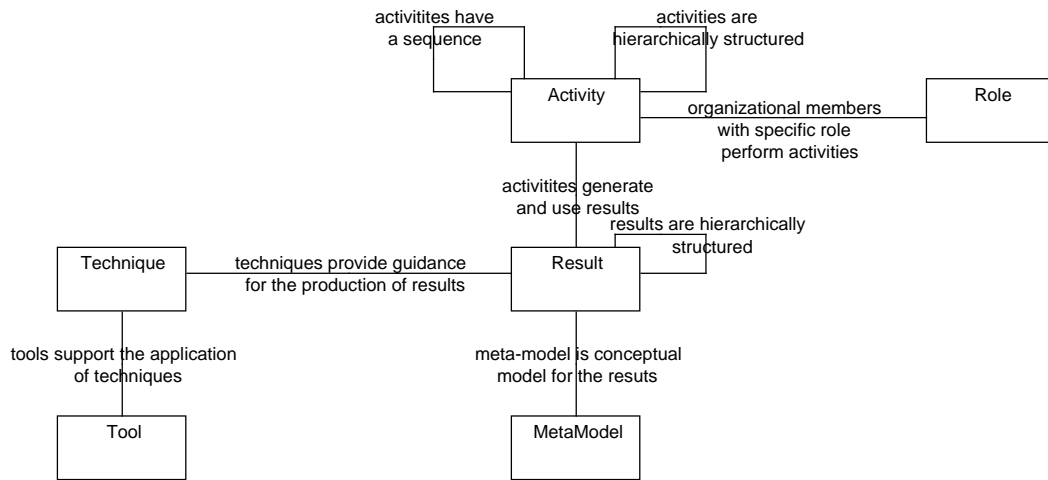


Figure 4.1.: Fundamental elements of a method according to Gutzwiller [Gu94, page 13]

In 2005 Braun et al. conducted a literature review on the state-of-the-art of *methods in IS research* in which they investigated twelve approaches to gain a deeper understanding of the concept of method [Br05, pages 1296–1297]. According to the review results, the fundamental attributes of methods in IS are a) *goal-orientation*, i.e., methods delineate how to proceed to achieve a defined goal or solve predefined problems, and b) a *systematic approach*, i.e., methods provide a systematic structure that enables the deduction of precise work steps or tasks for achieving goals or solving problems. Additionally, *procedure models* are identified as the only fundamental element of a method which is referred to in all twelve approaches analyzed. In line with these findings, we subsequently define the term method based on the explanations given by Henderson-Sellers [HS95] and Brinkkemper [Br96, pages 275–276].

#### Definition: Method

A method is a procedure of doing something, i.e., to achieve a certain goal. Methods are structured in a systematic way in development tasks with corresponding development artifacts.

In practice, methods are typically developed using a spiral model (cf. the IS engineering method life cycle presented by Harmsen [Ha97, pages 12–14]). In the first stage of method

<sup>3</sup>We in line with the *Business Process Model and Notation* [Ob10b] use the term **task** in this thesis to denote a piece of work that has to be performed to execute a method and which is part of a role’s duties.

<sup>4</sup>In our terminology the *data model* would be labeled information model (see definition in Section 3.1.1).

development, an approach appropriate for the situation at hand is developed ‘ad-hoc’ [Ha97, page 12] by an individual or a group of individuals. In the context of EA management these approaches often turn out of the necessity just because no suitable method to address the specific problem or context is available. Being in use for some time, the ad-hoc method turns out to be a “best practice” [Ha97, page 12] and is documented, e.g. as a pattern (cf. Section 4.2.4) to enable dissemination throughout an organization. The description of these best practices evolves over time, and becomes more and more complete until it has evolved to a “de facto” method [Ha97, page 12]. The diffusion rate of de facto methods thereby is still limited to one organization and is subject to change. If a de facto method is published in a reasonable amount of copies, thereby being made available outside of the originating organization, the method develops to a “de jure” method [Ha97, page 12]. Coming full circle, de jure methods are used in different organizations in a deviated manner, i.e., adding or modifying steps or outcomes, resulting again in an ad hoc method. This iterative life cycle of methods aligns with our pattern-based approach to theorizing as discussed in Section 2.2.2. Chapter 5 details how the practice-proven solutions contribute to our development method for organization-specific EA management functions.

#### 4.2.2. (Situational) method engineering

Over the years a plethora of different methods and tools for the area of software or IS engineering and the management of IS has been developed. Lacking a proper framework for research, the discipline of *method engineering* has developed in this area, which focuses on formalizing the use of methods for system development. Method engineering according to Brinkkemper [Br96, page 276] can be defined as follows.

**Definition: Method engineering**

Method engineering is the discipline to design, construct, and adapt methods, techniques and tools for the development of information systems.

Motivated by the plurality of proposed methods for standardizing IS engineering and the increasing application area diversification and complexity, Harmsen presented [Ha97] an approach to *situational method engineering*. The driving idea behind situational method engineering can be summarized by the following quote. “There is no method that fits all situations” [Ha97, page 6] which represents the prominent idea in the area of method engineering (cf. [BJ87, AWH91, KW92, SB93, HV97, FRO03]). Introducing the term *controlled flexibility*, Harmsen [Ha97, page 34] elicits requirements for a method engineering approach, which accomplishes standardization and at the same time is flexible enough to match the situation at hand. A *situation* thereby refers to a combination of circumstances at a given point in time in a given organization [Ha97, pages 32–33]. To address these requirements, for each situation a suitable method—so-called *situational* method—is *constructed* that takes into account the circumstances applicable in the corresponding situation. In the construction process uniform pieces of a method are selected. Two different ways, how these method pieces are framed, have evolved in the area of situational method engineering, namely

**method fragment** Harmsen [HBO94] and Brinkkemper [Br96, page 278] coined the term method fragment as a “coherent piece of a method”. Two types of method fragments are usually distinguished, namely *process fragments* representing the activities that are

to be carried out and *product fragments* representing deliverables, diagrams etc. to be produced or that are required (cf. [Br96, RP96, Ha97, Ra02, Åg07]). Method fragments can be defined on different levels of granularity, i.e., both a whole method can represent a method fragment as well as every single task used within a method represents a method fragment. Gradual growth, i.e., maturity stages, are represented by method fragments ranging from simple method fragments to more complex ones (cf. van de Weerd et al. [We06, WBV07]).

**method chunk** Rolland and colleagues [RP96, RPR98, RDR03] introduce the concept of the method chunk, i.e., an “autonomous, cohesive and coherent part of a method providing guidelines and related concepts to support the realization of some specific system engineering activity” [Åg07, page 361]. Thus, a method chunk represents the combination of a process fragment and a product fragment. The knowledge of a method chunk is captured in the chunk’s *body*, defining the a) *products*, i.e., input and output of the work, b) *process*, i.e., how to target products are created from input products, and c) *interface* defining the pre-conditions and post-conditions of a method chunk.

In line with the method-language dichotomy, reflecting tasks (processes) and results (products in terms of Gutzwiller [Gu94]), and the focus of this thesis on method aspects of the EA management function, we use the term **method fragments** in the following to denote the method prescriptions or coherent parts thereof. Thus, we in line with Harmsen [Ha97, page 28] define situational method engineering as follows.

**Definition: Situational method engineering**

Situational method engineering is the sub-area of method engineering directed towards the controlled, formal, and computer-assisted construction of situational methods out of method fragments.

Engineering a situational method requires standardized method fragments, which are typically stored in a data base called *method base* [Sa93, Br96, Ha97, RPR98, RR01], and guidelines or techniques how to assemble these fragments to a situational method. Following a pragmatic nature, the source of the method fragments contained in the method base is according to Henderson-Sellers and Ralyé [HSR10, page 424] not critical to the use in situational method engineering. Abstaining from further discussions on the origin of the method fragments, situational method engineering typically provides two generic processes for *applying* the method base to construct a situational method and *administering* the method base. The generic process to construct situational methods consists of the three steps which are subsequently sketched.

- Input to the configuration process is the specific situation, in which the method should be applied, e.g. the environment of the initiative, including users, organizational culture, management commitment, etc. This situation is analyzed in the *characterization of the situation* to describe the application characteristics and contingencies that must be taken into account. Methods and techniques to determine the contingencies of a certain situation can be found e.g. in van Slooten et al. [SB93] and Bucher [Bu07a].
- The characterization of the situation is used in the *selection of method fragments* to identify and select suitable method fragments from the method base. Heuristics can be applied to support the filtering and selection process.

- In the *method assembly* step the method fragments corresponding to the situation characterization are combined to a situational method. During the assembly of method fragments, aspects like completeness, consistency, efficiency, soundness, and applicability are accounted for (cf. Harmsen [Ha97, pages 204–216]).

After being organizationally implemented, i.e., being practically applied in an organization, the performance of the situational method is measured and requests for adaptations are raised, if potentials for improvements or unsuitable method tasks in the method fragments are detected. Figure 4.2 provides an overview on the configuration process and illustrates the relationships between the different steps.

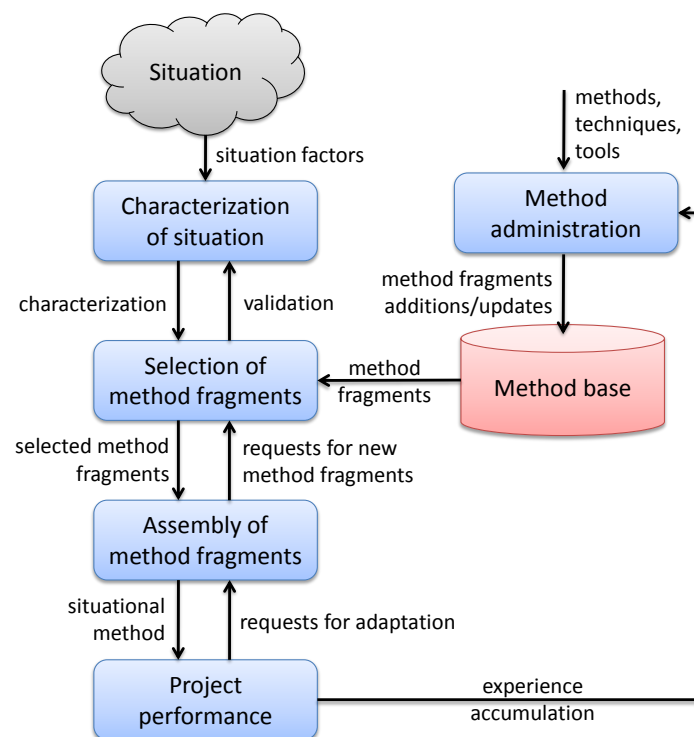


Figure 4.2.: The process of situational method engineering according to [Ha97]

The complementing process of *method administration* is concerned with the initial creation and maintenance of the method base. Representing a continuous process, administering the method base identifies new method fragments and provides means as well as techniques to incorporate the newfound method fragments into the method base. Specifying the latter process, Rolland [RP96] emphasizes the need to include knowledge about the context of use, i.e., define the pair *situation* and *decision* to store knowledge with each method fragment in which situation it is relevant and the associated decisions that need to be made (cf. context descriptions and forces of patterns as presented in Section 4.2.4). While the source of the method fragments contained in the method base is only of minor importance for method engineering according to [HSR10, page 424], a method to incorporate emerging techniques or newfound best practices into the method base needs to be established as part of the method administration. The contribution of patterns to evolve the method base is for instance discussed by Ambler [Am98], D’Souza and Wills [DW99], or Tasharofi and Ramsin [TR07].

Developing a harmonization in the area of method engineering and at the same time emphasizing the influence of the particular situation, which a method should be applied in, represents the core idea in situational method engineering as presented by Harmsen [Ha97]. We identified a similar objective in the area of EA management in the motivating chapter of this thesis. The situation described by Harmsen in IS engineering is quite similar to the one in developing and designing organization-specific EA management functions. A multitude of approaches exists but none of these has gained prominence due to the situation-specificity and organization-specificity of the subject. The typical questions of situational method engineering like ‘how to create method fragments, how to store and retrieve them, how to assemble a full method from the fragments, and how to formalize and structure method fragments’ taking especially into account the application field of EA management are answered in Section 4.3, which additionally refers to the role of pre-conditions and post-conditions in answering the aforementioned questions.

#### 4.2.3. Goal-oriented and actor-oriented modeling: the i\* framework

According to Harmon [Ha05, page 78] enterprises have a *system nature*, i.e., they represent systems, which are composed of systems and act in an environment of interrelated systems. Furthermore, organizations form socio-technical systems, meaning that they comprise interdependent resources of people, information, and technology interacting with each other in support of a common mission [Gi10, page 30]. The motivation of these different people has already been identified as an important aspect of EA management by Zachmann, who introduced the “why” dimension in his framework (see Section 3.3.1 for a detailed discussion of the framework). Most EA management approaches nevertheless focus on structural aspects of EA management neglecting the *intentional* perspective which is concerned with documenting, investigating, and making explicit the “reasons behind choices [...] and the exploration of alternatives” [YSD06, page 32].

The i\* model, developed by Yu [YM94, Yu96] and ever since applied and furthered in manifold publications not at least from an own workshop series<sup>5</sup>, is a model that aims at making explicit the intensional relationships between different actors of a complex system. In 2008 i\* has been adopted as part of the *ITU-T Recommendation Z.151—User requirements notation (URN)—Language definition* [In08]. In its current form the i\* model is comprised of two submodels, namely the *strategic dependency* (SD) and the *strategic rationale* (SR) model. The former model describes the dependencies between two or more actors in terms of *dependor*, *dependee*, and *dependum*. Thus, it expresses the strategic dependencies among actors. An actor according to the i\* model [Ab07, Section 4.1] can be defined as follows.

**Preliminary definition: Actor**

An actor represents an active entity that carries out actions to achieve goals by exercising its know-how.

A dependum in the SR models can be

- a *goal* i.e., the dependor needs the dependee to get the (measurable) goal fulfilled,
- a *tasks* i.e., the dependor relies on the dependee for getting a task executed,

---

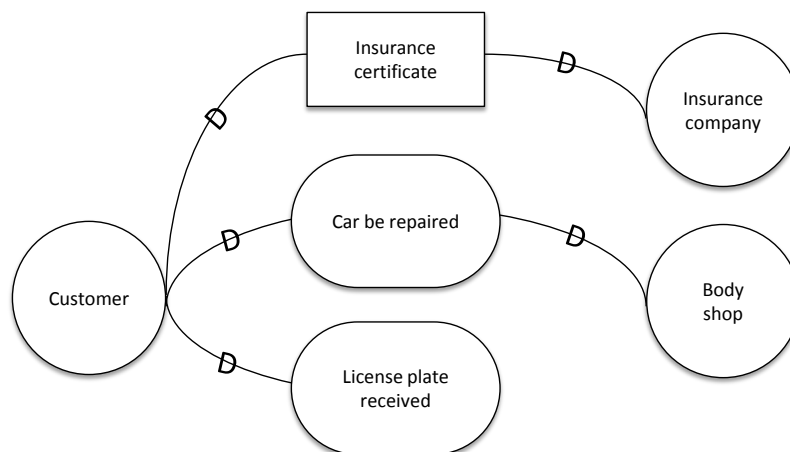
<sup>5</sup>See <http://istar.rwth-aachen.de> (cited 2010-12-20).

- a *resource* i.e., the depender needs the dependee to provide a certain artifact, or
- a *softgoal* i.e., the dependee can satisfy a not-measurable goal of the depender.

Below an exemplary SD model is given illustrating a goal and resource-dependency. Additionally, the example illustrates the notion of the one-side dependency. In such dependencies, only depender and dependum or dependee and dependum are known, leaving the opposite actor unknown.



**Example 4.1: The i\* framework.** If my car is broken by an accident and I want my car to be repaired, I take the role of a customer having a goal-dependency to the body shop, in which my car should be fixed. Similarly, I also need my insurance company to issue a certificate for insurance, i.e., I have a resource dependency to the insurance company. I have the goal to receive a license plate although, the opposite actor who fulfills this goal is not known. The following figure exemplifies the aforementioned dependencies using an SD model.



**Legend:**



Modelling dependers, dependa, and dependees with i\*



These dependencies are explained in the SR model, especially detailing on the intentionally desired elements for the corresponding actors. In other words, the SD model provides an abstract black-box perspective on actors and their relationships, whereas the SR model explains dependencies via a white-box perspective on actors and their intentions. Therefore, the SR model specializes the contribution links to positive and negative contributions. Exemplary contribution links as defined by i\* are a positive contribution strong enough to satisfy a soft-



goal (*make*), a positive contribution whose strength is unknown (*some+*), a partial positive contribution (*help*), a contribution whose polarity is unknown (*unknown*), a negative contribution sufficient enough to break a softgoal (*break*), a negative contribution with unknown strength (*some-*), or a partial negative contribution (*hurt*).

In the context of EA management methods, especially the task-dependencies are of interest. Using the mechanisms of the SD and SR models intentional dependencies between different actors participating in one or more EA management tasks can be made explicit. The transparency on the actor dependencies can be subject to systematic analysis regarding the organizational implementability of an EA management function [YSD06, page 2]. Considering further the aspect of information demand and supply in the context of EA management, dependency models as proposed by the i\* framework can be used to relate an EA management-problem and the corresponding stakeholder that triggered the information demand. Taking into account the amount of information suppliers participating in typical EA management tasks of documentation, the aforementioned transparency can be used to motivate information suppliers by making explicit the drivers and rationale of their ‘additional’ work (cf. the problem statement of Yu et al. [YSD06, page 2]).

#### 4.2.4. A pattern language for EA management

Following the integration approach introduced in Section 1.1, the *pattern-based approach to EA management* addresses typical problems of prevalent EA management approaches like too abstract guidelines, which lack appropriate guidance to be used in practice, or monolithic, ‘all or nothing’ approaches neglecting the specific EA-related problems of the associated organization. The idea of patterns as reusable solutions to recurring problems was initially introduced by Alexander et al. [Al77] in the field of architecture and has since that time been adopted to a variety of different fields<sup>6</sup>. In 2007 we applied the idea of patterns to the area of developing EA management functions [Bu07d]. In line with the definition of a pattern given in Section 2.2.2 an EA management pattern is defined by Ernst [Er10] as follows.

**Definition: EA management pattern**

An EA management pattern is a proven practice-based, general, reusable solution to a common problem in EA management, for a given context, identifying driving forces, denoting known usages, and consequences.

Furthering the idea of patterns for EA management, we compiled an initial set of patterns collected from literature and practice resulting in a first version of the so-called *EA management pattern catalog* (EAMPC) [Bu08b]. Alike its relatives from software engineering or architecture, an EA management pattern is not invented, but represents a “small, re-usable unit preferably based on established practices” which has been observed in practice [Bu07d, pages 154–155]. We distinguish three different types of EA management patterns [Bu08b, page 21].

---

<sup>6</sup>A general introduction to the concept of *patterns* is given in Section 2.2.2.

<sup>7</sup>While the utilization of *methodology* meaning ‘method’ is widespread [Ja94, page 35], we refer to its etymology meaning and definition as the study of methods. In line with this understanding ‘methodology patterns’ should correctly be labeled ‘method patterns’.

*Methodology*<sup>7</sup> *pattern* (M-Pattern) defines steps to be taken to address given concerns. It also documents roles, inputs, and outputs of the process as well as known variants and consequences related to its usage. The documented steps can further use one or more viewpoint pattern and information model patterns during its execution.

*Viewpoint pattern* (V-Pattern) proposes a language, i.e., a way how to present certain information on the EA to the participants of the method, e.g. diagrams, reports, or bar charts. It also documents techniques for view creation and usage, as well as known variants of the viewpoint. Each viewpoint pattern references an information model pattern that provides the underlying conceptualization of reality. To ensure understandability of a viewpoint, a legend is regarded to be mandatory.

*Information model pattern* (I-Pattern) supplies the underlying conceptual model of the EA, i.e., an information model fragment. It also includes definitions and descriptions of the used concepts as well as documents techniques for information model fragment implementation and usage.

In addition to the above pattern types, we also described so-called *anti-patterns* for EA management documenting recurring ‘solutions’, which have proven not to work. Examples for such anti-pattern can be found in [Bu09e] and in [Er10, pages 54–55]. The initial collection of patterns as documented in [Bu08b] has been evolved by Ernst to a *pattern language* [Ch10, Er10]. A pattern language according to Alexander et al. is defined as “a structure on the patterns, which describes how each pattern is itself a pattern of smaller patterns. And there are also rules, embedded in the patterns, which describe the way that they can be created, and the ways that they must be arranged with respect to other patterns” [Al79a, pages 185–186].

Besides the anti-patterns, the patterns of the different types follow a standardized documentation schema, a so-called *pattern form* [Bu96]. Ernst extends the initial documentation form for EA management patterns in [Er08, Er10]. An EA management pattern according to that format, which is also used in [Ch10] always provides a fact sheet with an expressive name, a unique identifier, and versioning information. In addition to the fact sheet, an EA management pattern typically consists of eight components, namely

- an *example* illustrating the problem to be addressed by the pattern either containing real world or anonymized information,
- a *context* describing the situation in which the pattern works, i.e., environmental factors that have to hold, if the pattern should be applied,
- a *problem* referencing the issue a pattern addresses in an appellative description, which directly addresses the users. The problem statement is further detailed with a discussion about its associated forces, i.e., goals and constraints that occur in the context,
- a *solution* presenting the fundamental solution principle underlying the pattern, i.e., steps to be taken (M-Patterns), exemplary visualizations (V-Patterns), or information model fragments (I-Patterns). Possible resolutions to the forces of the problem descriptions are delineated as part of the solution description,
- an *implementation* prescribing guidelines for realizing the pattern and operationalize the solution, e.g. the establishment of specific organizational roles or required tool support,

- *variants*, i.e., links other patterns, which present variant or specialized solutions of the pattern under consideration,
- *known uses* describe successfully usages of the pattern, e.g. in organizations, tools, or literature,
- *consequences* list known side-effects that arise from the implementation of the pattern. These side-effects can either be benefits that the pattern provides or potential liabilities (also including one or more new problems),
- a *see also* section that references other patterns, which solve similar problems, refine the pattern under consideration, or are used by the pattern, and
- *credits* that thank other authors, reviewers, or shepherds, which have been involved in the pattern elicitation process.

Although patterns according to their typical understanding represent self-contained solutions to recurring problems, the different types of EA management patterns are strongly interconnected in a way that a combination consisting of at least one M-, V-, and I-Pattern is used to address a given EA management problem. This circumstance can be exemplified with the M-Patterns, which reference one or more V-Pattern to be used during the conduction of a step, see the M-Patterns proposed by us in [Bu08c, Bu09a, Bu09c, Bu09d, Bu09i, BES10b] or Lau et al. [La09b]. In a similar vein, the V-Patterns are strongly linked to the I-Patterns in the sense, that a V-Pattern exemplifies the specified viewpoint, using the concepts supplied by a specific I-Pattern. Besides these relationships, which relate ‘prerequisite’ patterns, another type of relationship between EA management pattern exists, referencing ‘alternative’ patterns. Using the latter relationships evolution paths, i.e., maturity stages, for the EA management function can be defined (cf. [BMS10f, Bu10c] for a maturity discussion on the information model level).

Reflecting the pragmatic nature of patterns, which mainly target practitioners, pattern typically do not account for terminological issues. Pattern languages as systems of interrelated patterns accordingly do not provide specialized means and techniques to ensure terminological consistency between the interrelated elements. The pattern language for EA management [Ch10] also follows this pragmatic nature. Whereas a common glossary for the EA concepts, i.e., the constituents of the I-Patterns, is given, no entries for other terms, as contained e.g. in the method prescriptions of the M-Patterns are provided. Thus, the wording as used in different M-Patterns has to be adapted prior to integrating them into a comprehensive EA management function to avoid terminological disruption. This in particular applies as the method prescriptions in the M-Pattern are described on different levels of granularity mostly staying on an abstract and vague level that can be summarized as ‘something must be documented in an architectural artifact, the artifact must be analyzed and communicated to the according participants to support their decision making process’. This general process of documenting, communicating, analyzing, and deciding is repeated in different ordering in the various M-Patterns. The redundancy can thereby be ascribed to the direct linkage between an M-Pattern and the EA management problem to be addressed. Two similar problems, e.g. increasing homogenization of applications and increasing homogenization of infrastructure elements, result in two different M-Patterns containing the same method prescriptions but referencing different V-Patterns and I-Patterns.

The pattern language for EA management provides a way to reuse proven-practice solutions for EA management problems and presents a good starting point for our method to design organization-specific EA management functions. Whereas the overall structure of the EA management pattern catalog gives rise to redundancy, e.g. each M-Pattern specifies distinct ways to collect information, to prepare or to enact decisions and plans, the collection of best practice method prescriptions is reconsidered in our solution. Besides the aforementioned drawbacks of the pattern-based approach, an overall picture of the EA management function is missing. Such an understanding of the overall make up of the EA management function is needed to develop a comprehensive, consistent, and coherent management function out of method fragments. Furthermore, the level of detail in the method prescriptions should be lowered to make the sometimes abstract and vague prescriptions as contained in the pattern language for EA management organizationally implementable. Finally, the explicit relationships between patterns form an issue to be addressed in our method, as these hamper the evolution of the overall knowledge base. The aspect of consistency deserves special attention and techniques to ensure that a combination of method fragments results in a consistent and coherent EA management function need to be established.

### 4.3. BEAMS: A conceptual overview

The central contribution of this thesis are the *building blocks for EA management solutions* (BEAMS) and the associated development method which guides the design of organization-specific EA management functions. The development method builds on prescriptions that have been proven useful to address specific EA management problems in a given context in practice. While a multitude of such prescriptions exists (see the analysis in Section 3.3), they are typically described using different terminologies and levels of abstractions. To facilitate re-usability in our development method the concept of **building block** is used which provides a unifying basis by specifying the necessary design and components of the prescriptions. We identify two different types of building blocks—*method building block* (MBB) and *language building block* (LBB). This distinction is supported by the idea of method-language dichotomy as promoted by Schelp and Winter [SW08, page 81] as well as Aier and Schelp [AS09, page 55] and can also be found in our pattern-based approach to EA management [Bu08b, Ch10]. In our solution, MBBs describe the steps to be taken, the decisions to be made, and the participants involved in addressing a specific EA management problem in a given organizational context. The language counterpart is described in the LBBs which provide the language primitives used during executing the MBB. As this thesis focuses on the method part of EA management functions it emphasizes the MBBs which are subsequently discussed in detail. In this section, we briefly sketch the main constituents of our solution and outline the general method for developing organization-specific EA management functions. The relationship and linking to LBBs, which are researched in a complementary workstream [BMS10d], is drawn where necessary.

As our literature review in Section 3.3 on prevalent EA management approaches revealed, a plurality of method prescriptions, i.e., solutions, addressing specific EA management problems in given organizational contexts exists. The applicability of these solutions depends on the one hand on the organizational context which delineates facilitating factors or impediments and on the other hand on the specific EA management problem which describes the goal pursued and

the concern addressed. Therein, not only the granularity in which the solution is described but also the level of abstraction used to document the organizational context and the problem varies widely in the different approaches. Taking into account these inhomogeneous description levels, the organizational context and problem descriptions can be categorized according to the classification of Pries-Heje and Baskerville [PHB08, pages 732–733] as *asymmetric criteria*, i.e., criteria that apply only to a subset of the solutions<sup>8</sup>. As a result, ‘classic’ contingency-based approaches are not sufficient to determine the solutions suitable for a specific problem in a given context, as they use *symmetric criteria*, i.e., criteria that apply to all solutions. For the context of decision support in situations where asymmetric criteria exist, Pries-Heje and Baskerville present the idea of a *design theory nexus* (cf. Section 2.2.3). A design theory nexus links competing design theories that apply to problems in given contexts. Thus, the concept of the nexus can be mapped to the idea of a method base in terms of situational method engineering (cf. our discussion in [BMS10c] and [BMS10a]). The linked design theories are not unified but retain their levels of abstraction. In the context of EA management, the existing approaches to develop an EA management function are design theories in the sense of Walls et al. [WWES92] as discussed in Section 2.2. A selection of solutions, i.e., design theories, from a nexus instantiation or method base for EA management, respectively, can therefore, if it is a consistent one, be used to develop an organization-specific EA management function. Figure 4.3 provides an overview on the design and the components of a design theory nexus instantiation that supports the development of organization-specific EA management functions.

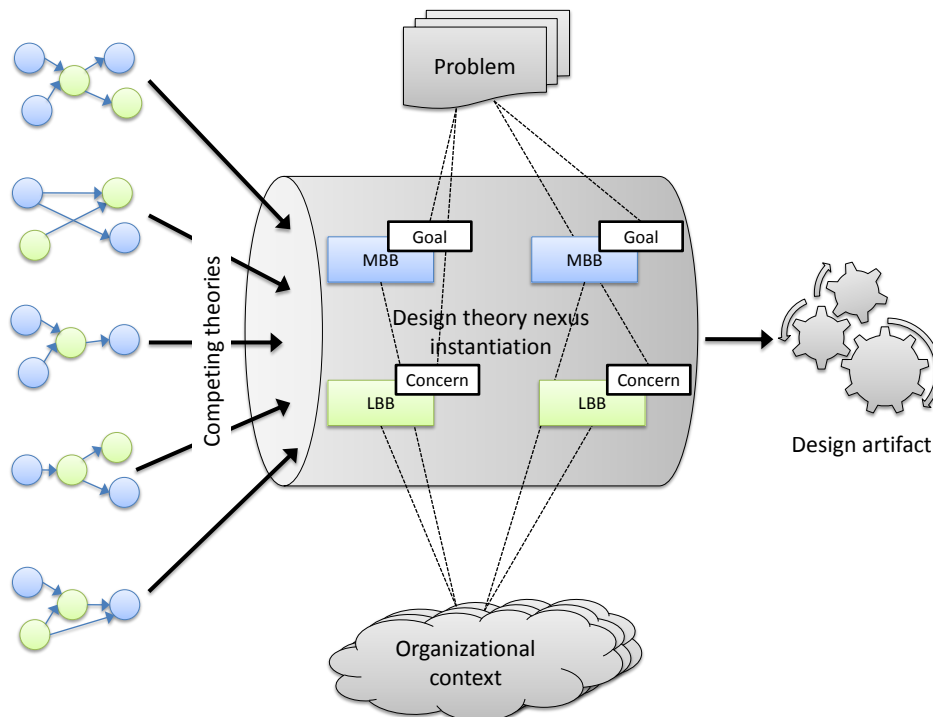


Figure 4.3.: Design theory nexus instantiation for EA management functions

<sup>8</sup>For an explanation of symmetric and asymmetric criteria see Section 2.2.3.

With the thesis focus on the development of an organization-specific method for an EA management function, we define a method base as follows.

**Definition: Method base**

A method base is an organized collection that contains a set of interlinked method building blocks (MBBs). The method base enables the enterprise architect to select MBBs that fit to the situation at hand by interlinking MBBs with organizational contexts.

With the method base concerned with the MBBs and organizational contexts, we subsequently sketch the core constituents of a design theory nexus instantiation for EA management from a black box perspective as illustrated in Figure 4.3.

**Competing theories:** The competing theories represent prevalent method prescriptions (see the literature review performed in Section 3.3) that are linked in the design theory nexus instantiation in the form of building blocks. Reflecting the nature of EA management as a practice-oriented field of research, we build on theories which have been proven beneficial in practice (see the pattern-based process to theory building in Section 2.2.2).

**Problem:** A problem represents the issue to be solved by applying the theory. A problem in the area of EA management consists of a

goal representing an abstract objective of the EA management initiative, e.g. increase homogeneity, provide transparency, and a concern, i.e., an area of interest in the enterprise<sup>9</sup>.

**Organizational context:** The organizational context represents the situation in which the EA management function operates. Typical facilitating factors and impediments, which are considered in the organizational context, are the enterprise culture, management commitment, or the organizational structure.

**Building block:** The building blocks represent the theories to be combined to an organization-specific EA management function in a consistent and coherent form. Reflecting the dichotomy of method and language, two kinds of building blocks exist,

MBBs describing who has to perform which tasks to address a problem in the situated context and

LBBs referring to which EA-related information is necessary to perform the tasks (specified in an *information model building block* (IBB)) and how it can be visualized (defined in the *viewpoint building block* (VBB)).

In terms of Gutzwiller [Gu94, page 3], MBBs describe tasks, participants<sup>10</sup>, and techniques whereas the LBBs describe the meta-model. The results are referred to by both building block types and represent *boundary objects*, i.e., objects that are relevant to two perspectives in terms of Star and Griesemer [SG89]. To get into detail, the MBBs specify certain requirements that the results must fulfill and the LBBs specify how the results should be presented. In the application area of EA management results of EA management tasks are typically referred to

---

<sup>9</sup>See definition of concern in Section 3.1.1.

<sup>10</sup>Gutzwiller originally used the term *activity* instead of **task** and *role* instead of **participant**.

as *architecture artifacts*, *architecture descriptions*, or *viewpoints* (cf. Lankhorst [La09a]) and the meta-model is contemplated *information model*<sup>11</sup>. To avoid a situation as we encountered in the pattern-based approach to EA management (see Section 4.2.4), where especially in the M-Patterns redundancies in the prescriptions exists, the MBBs contained in the method base represent ‘underspecified’ building blocks extracted from the best practice solutions. We introduce the **variable concept**, that represents a ‘underspecified placeholders’. This placeholder is configured during the development method. In the context of BEAMS, we supply four different types of variables which are in the following introduced by example.



**Example 4.2: The variable concept.** Two fictitious organizations from the finance industry have each established an EA management function to document their as-is situation via face-to-face interviews with the corresponding information suppliers.

In organization A the interviews are conducted on demand with application owners to gain information on the application landscape. Therefore a structured questionnaire is used.

In organization B the interviews are conducted on a yearly basis with business process owners to gain information on the current process portfolio of the organization. The enterprise architects do not use any auxiliary materials except a note to put down the information gathered, as they have a profound understanding of the overall architecture and can decide ad hoc which questions should be asked.



The general method how the as-is situation is documented in the above example can be regarded similar in both organizations. Hence the precise realization varies. The tasks that need to be performed, e.g. contacting the interviewee, arranging an interview date, preparing the interview, conducting the interview, post-processing of the interview, etc. can be described in a uniform way. In contrast, the participants vary as do the viewpoints used to gather the information, the triggers of the activities, and the underlying information model describing the problems to be addressed. Reflecting these different concepts, we introduce four types of variables for MBB descriptions, namely

- **participant variables** that denote a certain participant involved in executing a task for whom certain requirements hold. In the above example both the application owners as well as the business process owners can be generalized to a participant called ‘information steward’, i.e., persons which acts as information suppliers.
- **viewpoint variables** that replace specific perspectives in architecture descriptions. The viewpoints used in the above example, the questionnaire and the note, are replaced in a method description by the general concept of viewpoint variable, which are either of

---

<sup>11</sup>For a definition of viewpoint in the context of architectural descriptions and information model see Section 3.1.1.

the type *representation function*, i.e., used to communicate information on the EA as introduced by Ernst et al. [Er06], or are of the type *notation function*, i.e., used to model the EA as alluded to by Kühn [Kü04, pages 30–37].

- **information model variables** that replace the specific EA concepts, e.g. the concepts that make up the application landscape and the processes of the process portfolio from the above example, to decouple the method prescriptions from the specific conceptualization of the EA, which is regarded to be organization-specific (cf. discussions by Kurpjuweit [Ku09] or Kurpjuweit et al. [KW09]). A method description can nevertheless specify some concepts which must be contained in an information model variable without confining the information model to these concepts. Typically, these concepts represent cross-cutting aspects as principles, projects, or standards (see Chapter 1 for an explanation of cross-cutting aspects).
- **trigger variables** that specify when an MBB is performed. Different types of trigger variables exist, namely event triggers and temporal triggers. An event that triggers the execution of an MBB can be invoked by a task from another MBB, e.g. the publication of an architectural description or the documentation of a project. A temporal trigger represents an external trigger, i.e., specifies time intervals when an MBB is executed.

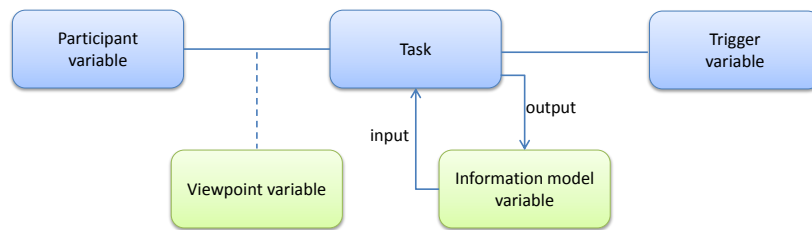


Figure 4.4.: Interlinking MBBs and LBBs via different types of variables

Figure 4.4 shows the relationships between the introduced types of variables and links them to the MBBs via the central concept of **task**. A color-coding is employed which classifies the concepts as MBB-related (blue) and LBB-related (green). The trigger variable specifies when a certain task is executed. Each task of an MBB relates to an information model variable which specifies the concepts of the EA used during the conduction of the task. Furthermore, each task is associated to at least one participant variable, which is responsible for executing the task. During execution of the task, information on the EA, specified in an information model, is modeled, read, or updated using a viewpoint, which has to be specified in the configuration step of the development method and assigned to the viewpoint variable. Different viewpoints thus are based on different information models. The different information models hence can be integrated into a consistent information model representing the conceptual model of the associated organization. The variable concept thus not only interlinks the methodical perspective of an EA management function with the corresponding EA modeling language but further supports organization-specific customization of e.g. participant descriptions or viewpoint specifications.

With the focus on a development method for organization-specific EA management functions, we in the following we place emphasis on the method-related constituents and challenges. Recapitulating the definition of method as provided in Section 4.2.1 and taking into account



requirement **Q2** which demands an EA management function to be problem-oriented and stakeholder-oriented, two different participants can be identified that relate to the EA management function, namely

- **stakeholders** who own the problems to be addressed by the EA management function, i.e., represent information consumers and
- **actors** who participate in an EA management-related task, i.e., are information providers.

In the sense of the i\* SD-model the stakeholders can be interpreted as the *dependers* who rely on the actors, i.e., the *dependees* to get a task executed, a goal fulfilled, or a resource provided (see Section 4.2.3). Using the i\* SD-model, organizational implementability of an EA management function and the constituting activities and tasks can be analyzed. Thereto, we propose a technique that makes explicit the dependency between the stakeholders of an EA management problem and the corresponding actors that are involved in addressing the problem. Comparing the dependencies with the organizational structure the organizational implementability can be evaluated. Section 5.4.2 presents a technique to analyze the organizational implementability and discusses different relations between organizational control and information dependencies.

Based on the organizational contexts and specific problems of the stakeholders, the MBBs from the method base are selected. The linking of competing theories from different origins into a method base is aggravated by the fact that the method prescriptions from the theories can use different terminologies, be defined on different abstraction levels, as well as refer to distinct organizational contexts and specific problems. To facilitate the selection and integration of MBBs, a general framework of the main activities of an EA management function that supports classification and structuring of method prescriptions from prevalent theories has to be developed. Therefore, the high-level objective of EA management to guide the managed evolution of an organization is revisited. Figure 4.5 shows the main activities of an EA management function and illustrates the interplay between methods and models for EA management.

In response to the complexity of the EA, models describing the current state are developed, which are analyzed to identify potentials for improvement. Based on the analyses and planned projects, roadmaps for the evolution of the EA are developed and documented in planned states, which are again evaluated to identify the most appropriate one. This state is finally enacted and communicated to guide the transformation process.

In line with our findings from Chapter 3, we identified in [BMS10j] the main activities of an EA management function: **develop & describe**, **communicate & enact**, **analyze & evaluate**, and **configure & adapt**. Further investigating the develop & describe activity of the EA management function, we identified three different time-related dimensions that can be used to classify the resulting architecture descriptions [Bu08d, page 65], [Bu09f, page 14], and [Bu11b]. Firstly, an EA description has been **modeled at** a certain time. Secondly, an EA description is **planned for** a specific time, i.e., specifies at which point in time it should become a description of the status quo and thirdly, different **variants** of an future EA description can exist, e.g. reflecting different evolution paths. In line with these time-related dimensions the develop & describe activity can produce EA descriptions of three different states of the EA.

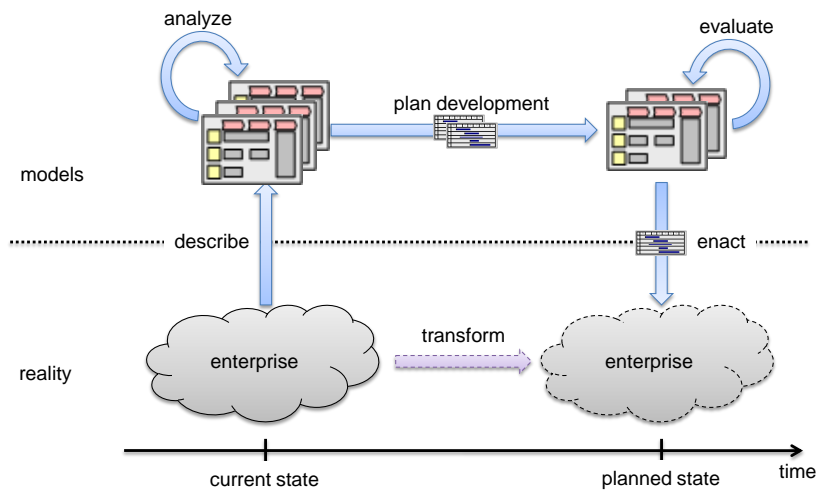


Figure 4.5.: Transforming enterprises: the interplay between models and methods for EA management

- **Current state** represents the status quo of the landscape as-is modeled at a certain time.
- **Planned states** represent medium-term future states of the EA as to be at a specific point in time that are modeled at a certain time emphasizing the changes performed by planned projects up to the specific future dates. Reflecting different project portfolio choices one or more variants for a specific future date (planned for) can exist.
- **Target state** represents a long-term envisioned state of the EA which is typically defined on a more abstract level and must not relate to the current state as well as consider specific projects. The target state is typically derived from the organization’s vision and strategy.

Figure 4.6 illustrates the different states of the EA and relates them to the two dimensions of time and the variants dimension.

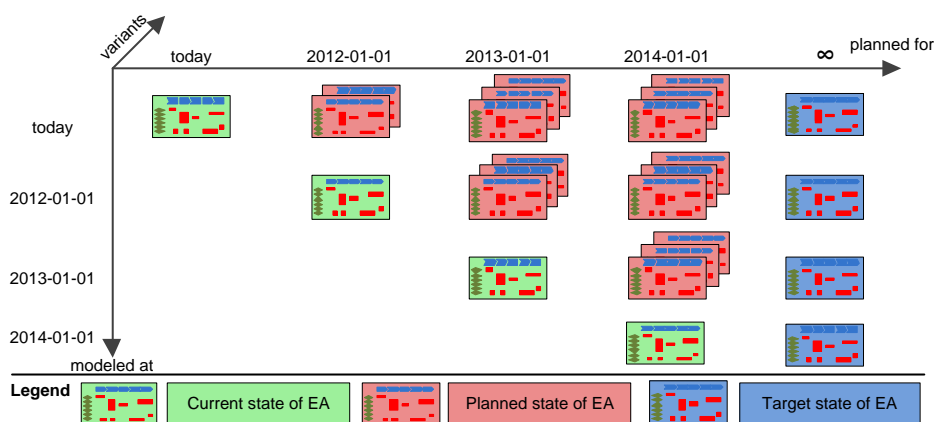


Figure 4.6.: Current, planned, and target state of the EA

Based on the above understanding of EA state and the findings from the literature review, we devise a general method framework for the EA management function, which can be used to classify method prescriptions for the integration in our nexus instantiation. The framework further enables subsequent selection and integration into a consistent and coherent EA management function. It is problem-independent, i.e., does not refer to specific concepts of the EA but might reference the different states as introduced above. We subsequently introduce the activities of the method framework and provide additional details on the EA descriptions that are created and consumed by each activity in terms of above EA states.

**Develop & describe**—This activity is concerned with creating EA descriptions of the current, planned, and target state of the EA. The target state is developed based on the business and IT visions and strategies that the enterprise seeks to implement. In creating a description of the current state, a defined area-of-interest, i.e., a concern, is documented via an EA description and planned states making the evolution roadmap explicit are developed. Further, the activity is concerned with establishing maintenance procedures that ensure the actuality of the EA descriptions. For implementing the activity, different best-practices can be used, ranging from documentation endeavors on regular basis, to continuous endeavors accompanying the EA relevant projects (cf. Moser et al. [Mo09]). As the develop & describe activity is concerned with developing architecture descriptions, the viewpoint variable is of the type notation function, as introduced above.

**Communicate & enact**—The developed architecture descriptions need to be made available to the different participants via propagating them to the enterprise-level management functions. This propagation aims at influencing the decision making in the related functions. Therefore, communicating and enacting contributes to the decision making in the enterprise-level management functions, e.g. project portfolio management. Different ways to implement the activity exist. These range from the non-interfering way of informing the decision makers to the most powerful method of having the right to stop projects, which are non-conformant with respect to planned and target states of the EA. This activity hence always takes the EA description as input, but can create multiple output artifacts that are handed over to the enterprise-level management functions. The types of artifacts to be exchanged thereby depend on the method of communication and enactment chosen. Typically a viewpoint variable of the type representation function is used to communicate and enact the different states of the EA.

**Analyze & evaluate**—Architecture descriptions of different states of the EA are created and maintained by the EA management function. The analyze & evaluate activity makes these descriptions comparable to prepare a subsequent decision on the future evolution. Different properties of the architecture can thereby be of interest, ranging from compliance characteristics to economic properties. Functional properties of the architecture, as e.g. the provided business support, can also be important (cf. Niemann [Ni06c]). Most commonly non-functional properties, e.g. the availability of certain business services (cf. Johnson et al. [JNL06]) or the flexibility of the overall architecture are used for analyzing different states. In literature a broad variety of approaches to EA analysis exist, differing with respect to the employed level of formalization, ranging from expert-based assessments (cf. Niemann [Ni06c]) to indicator-based computations (cf. Frank et al. [Fr08] as well as Iacob and Jonkers [IJ06]). The approaches also vary concerning their time reference: some approaches are designed to analyze current architectures (cf. Niemann [Ni06c]), while other approaches (cf. De Boer et al. [Bo05]) provide

prediction capabilities that can be used to analyze architectures not yet realized. On the one hand viewpoint variables of the type representation function are typically used during analysis or evaluations as input, on the other hand viewpoint variables of the type notation function are used to incorporate the results into the architecture description.

**Configure & adapt**—Before starting an EA management endeavor the EA management problems to be addressed by the initiative should be clearly defined. Based on these problems the goals to be pursued and the concerns in the EA can be defined. As part of the configure & adapt activity also decisions on the scope and reach of the EA management function must be made. The choices range from bottom-up approaches, in which only a certain division of the enterprise is considered regarding a certain aspect like standardization, to top down approaches, where the whole enterprise is examined with respect to multiple aspects. After the initial establishment of an EA management function, the configure & adapt activity is concerned with measuring the overall performance of the EA management function. Adaptations can be necessary, if e.g. goals are achieved and the enterprises matures or the organizational context changes. The configure & adapt activity results in a (re-) configuration of the EA management function. Thus a documentation of the management function can be regarded to be the output of this activity.

Figure 4.7 provides an overview of the method framework. Based on the above developed understanding, we revise our definition of EA management as follows.

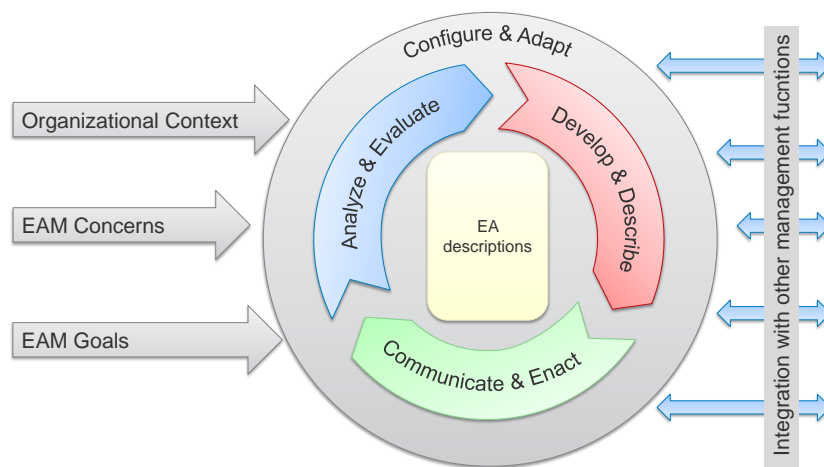


Figure 4.7.: Method framework detailing the main activities of an EA management function

**Definition: EA management**

EA management is a continuous and self maintaining management function seeking to improve the alignment of business and IT and to guide the managed evolution of an (virtual) enterprise. Based on a holistic perspective on the enterprise furnished with information from other enterprise-level management functions it provides input to, exerts control over, and defines guidelines for these enterprise-level management functions. The EA management function consists of the activities develop & describe, communicate & enact, and analyze & evaluate.

Subsequently, we exemplarily apply the generalized method framework to classify single activities of an EA management function as established by an industry partner.



**Example 4.3: The EA management activity framework.** The topic of EA management has a long history in the internationally operating bank from Germany since a merger in the year 1996. The management function established at the company consists of the following activities:

- (1) *Create and adjust IT strategy:* Based on the enterprise business strategy, the IT strategy is developed, which includes information on core competencies, products, business areas, etc, and is used to design a target state of the EA.
- (2) *Develop and update architectural guidelines and standards:* Architecture principles are identified and guidelines as well as standards are developed and updated on this basis. To decide on new guidelines or standards, an architecture board was introduced.
- (3) *Identify needs for action originating from business and IT:* Business and IT demands are collected and analyzed with respect to their strategic or operative importance. The identified needs are further assessed and prioritized according to the architectural principles identified.
- (4) *Develop and update architecture artifacts:* EA descriptions, like viewpoints, artifacts, guidelines, and standards are developed from three perspectives: the functional, technical, and security perspective. They are updated on a yearly basis either prior to or after the creation of the annual project plan.
- (5) *Check architecture conformity:* The EA conformity is ensured via quality gates for projects, i.e., each project is assessed for its conformity with the EA plans. If deviations are detected the decision on the continuation of the project is vertically escalated.

The EA management function as presented above was subject to various changes in the past, where the performance of the function itself was assessed. Such an assessment took place in the year 2005, where impediments, which hampered the successful management of the EA, were identified. As a consequence of this assessment, decisions on architectural guidelines in activity (2) were not longer taken in a central board, if activities with local impact are concerned. Thus, overloading the architecture board is prevented.



The EA management function established at the banking company can be mapped to the activities of our method framework as follows: the activities (1),(2), and (4) relate to the develop & describe activity, activity (3) relates to analyze & evaluate, and activity (5) relates

to communicate & enact. Although the industry partner has a long history in EA management, no dedicated activity for the adapting the EA management function itself is established.

Using our general method framework for EA management functions, prevalent method prescriptions can be structured and organized in the method base as problem-independent MBBs that are linked to the contexts in which they can be applied. Integrating these MBBs into an organization-specific EA management function with a coherent and consistent method prescription raises some challenges. Obviously not every combination of (in terms of the organizational context) suitable MBBs is consistent. Furthermore, MBBs have to be placed in a certain order, e.g. an architectural description of the current state can only be analyzed, if it has been documented, i.e., described, before. Some MBBs might require other MBBs to be included and finally the variables of participants, viewpoint, information model, and trigger type employed by an MBB have to be configured during the development method to result in an implementable method prescription of the EA management function. Further considering the former challenges, three types of relationships between MBBs can be introduced, namely **integrable**, **alternatives**, and **related**. To facilitate maintenance of the method base, no ‘explicit’ links between the MBBs are established (cf. our discussion of the drawbacks from the pattern language in Section 4.2.4), instead we use the concepts of **pre-conditions** and **post-conditions** to establish **implicit relationships** between MBBs. A post-condition thereby is a specialized form of a *consequence* as presented in our pattern-based approach to EA management (see Section 4.2.4) that specifies conditions for the information model variable that hold after the tasks described by the MBB have been applied. The pre-condition represents the counterpart specifying what conditions must hold prior to applying the method.



**Example 4.4: Relationships and conditions.** Different MBBs to develop an architectural description of the current state of the EA exist.

- MBB<sub>1</sub> uses an interview based technique to gather information,
- in MBB<sub>2</sub> e-mails containing a questionnaire are sent to the information stewards, and
- in MBB<sub>3</sub> a dedicated tool for EA management is used.

The pre-conditions of all these MBBs are empty, i.e., they do not impose any constraints for the information model variable. The post-conditions of MBB<sub>1</sub> and MBB<sub>2</sub> are identical as they both specify that the concern assigned to their information model variable a) has been documented and b) that during documentation inconsistencies might have occurred. In contrast MBB<sub>3</sub>, which uses a tool-based technique, can ensure consistency during documentation such that its post-condition is limited to the concern being documented.



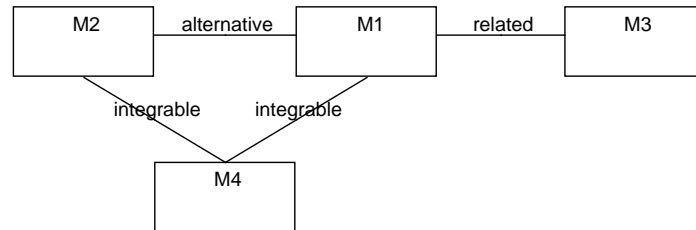
With the above understanding of pre-conditions and post-conditions, the different relationship types of MBBs are defined as follows: two MBBs, MBB<sub>1</sub> and MBB<sub>2</sub> are

- **integrable**, if the post-condition of  $M_1$  fulfills the pre-condition of  $M_2$ ,
- **alternatives**, if both MBBs,  $MBB_1$  and  $MBB_2$ , specify the same pre-conditions as well as the same post-conditions, and
- **related**, if both MBBs,  $MBB_1$  and  $MBB_2$  contain the same pre-conditions but specify differing post-conditions.

These relationships can be detailed along the above example of MBBs belonging to the develop & describe activity as follows.



**Example 4.5: Integrable, alternatives, and related MBBs.** Between above MBBs the following relationships can be identified.



Exemplifying the relationship of integrable one could think of an MBB ( $MBB_4$ ) that specifies a method to resolve inconsistencies based on textual information descriptions.



In Section 5.3.2 we introduce a technique to ensure consistency based on above concepts of pre-condition and post-condition. The technique is applied as part of our development method for organization-specific EA management functions based on the MBBs.

In line with the constituents of a method as discussed by Gutzwiller [Gu94], above concepts constituting a method description in the context of EA management and in line with the framework of the main activities of an EA management function, we define an MBB as follows.

**Definition: Method building block (MBB)**

A method building block is a context-specific, problem-independent description of a coherent part of an EA management method describing the tasks to be performed, triggers, participants, known forces, consequences, as well as pre-conditions and post-conditions. An MBB can be classified as contributing to the develop & describe, communicate & enact, or analyze & evaluate activity of an EA management function.

The configure & adapt activity as mentioned before represents a meta-activity to the three other activities, in the sense that it is concerned with determining the design of the aforementioned activities. Such activity is sometimes referred to as *EA management governance* (cf. Harmsen et al. [HPK09, pages 120–121]) and maps to the **development method** of

our solution. Therein, especially the aspect of constructing a coherent and consistent EA management function based on the MBBs from the different activities is challenging. As mentioned before, we use the method framework to facilitate the structuring and organization of method prescriptions to MBBs and further incorporating them into the method base. Further the framework is used in the development method to support the selection and integration of MBBs. Our understanding of the EA management function as a continuous management function is thereby in particular reflected by the last activity of *configure & adapt*. It builds on the concept of reusable MBBs to develop an organization-specific EA management function, thus concretizing the *configure & adapt* activity. The development method consists of three parts, namely **characterize situation**, **configure EA management function**, and **analyze EA management function**. The characterization of the situation is a simple activity which consists only of two steps.

1. Characterize the environment of the EA management function by selecting the organizational contexts that apply in the organization.
2. Select the problem to be addressed by the endeavor, i.e., the goal to be pursued and the concern on which the goal applies.

For each combination of organizational context and problem, an appropriate method to solve the problem is configured. Configuring the method starts with an empty set of method specifications that is expanded by iteratively applying the following steps

1. Select an appropriate MBB from the method base
2. Customize the MBB via assigning the variables, i.e.,
  - a) assign organization-specific roles to the participant variables,
  - b) define triggers that specify when a task is executed, and
  - c) detail appropriate viewpoint descriptions to be used for executing a task.
3. Update the organization-specific configuration, i.e., integrate configured method fragments with into the already configured EA management function (this step is omitted in the first iteration).

While the organizational contexts provide ‘static’ filtering criteria, appropriate MBBs to be selected in the first iteration are only those which do not specify any pre-conditions. The latter criteria represents a ‘dynamic’ one, i.e., is adapted with each MBB selection. In other words, while the set of selected MBBs and thus the predefined method prescription matures, the post-conditions of already selected MBBs extend the set of applicable MBBs.

Complementing, the above two parts of the method, the organization-specific configuration is finally analyzed for organizational implementability by

- the communication to the stakeholders as sponsors of the endeavor and
- by analyzing the stakeholder-actor-dependencies with the organizational control structures.

Reflecting the changing topics of the EA management function the development method is additionally complemented with guidelines how to adapt an organization-specific configuration to changing situations, i.e., changing organizational contexts or varying problems to be addressed.



## 4.4. Summary

In this chapter we outlined the basis idea of our development method that builds on practice-proven building blocks for developing organization-specific EA management functions. To further shape the solution domain, we started with characterizing the development method in Section 4.1. Thereto we use a twofold approach reflecting on the one hand quality criteria which designate general characteristics of the resulting artifact, i.e., an EA management function, and on the other hand users' expectations for using the development method and underlying method base to develop such management functions. Regarding the former case, we elicited nine quality criteria using a hermeneutic method. We iteratively revisited our findings from the literature review in Chapter 3 against the set of requirements as presented by Hafner and Winter [HW08], the requirements specified by GERAM [In99], and a practitioners view as reflected in a comprehensive set of scenarios gathered during the EAMTS 2008 [Ma08]. Complementing the first set of quality criteria, we elicited requirements on the 'meta-level', i.e., regarding the development method and the method base using the *guidelines of modeling* as proposed by Becker et al. [BRS95] as well as Schütte and Rotthowe [SR98]. Complementing these two sets of characteristics that answer our research question 3, we approached our contributions from a design perspective. Based on the guidelines for design science research of Hevner et al. [He04] we established a basis to evaluate the quality of our contribution.

Preparing the presentation of our development method and the underlying method base, we revisited contributing theories from related disciplines in Section 4.2. We started with further investigating the notion of method which relates to our contribution in a dual way. First, our main contribution is a method to develop organization-specific EA management functions and second, we focus on the methodical part of EA management functions, i.e., the result from applying the development method itself is a method. In particular, we detailed on the constituents of a method according to Gutzwiller [Gu94] and indicated their relevance for the application in the EA management domain. Referencing to the characteristics of organization-specificity, we outlined the basic idea of *situational method engineering* for developing situation-specific methods as proposed by Harmsen [Ha97]. Discussing two different types of constructs representing 'method pieces', we identified the concept of *method fragments* as appropriate to describe our practice-proven method prescriptions. Furthermore, Harmsen [Ha97] proposes two generic processes in the context of situational method engineering one relating to applying the method base, which we linked to our development method, and one for administering the *method base*. Providing a preliminary framework for our development method, the generic process for applying the method base is discussed. It consists of the steps of *characterization of the situation*, *selection of method fragments*, and *method assembly* [Ha97]. Complementing the influence of situations on the development of EA management functions, we revisited Yu's work on *intentional modeling* [Yu96] and discussed how the different *actors*, i.e., participants, of an EA management function depend on each other. We also described how the explicit dependencies can be used to ensure organizational implementability. Finally, we prepared the exposition of our solution by revisiting our pattern-based approach to EA management [Bu07d], which was further developed to a pattern language by Ernst [Er10]. Exploring the benefits and shortcomings of EA management patterns, we delineated how patterns can contribute to the development of our solution.

Section 4.3 finally presented the general design of our solution. Central concept of our solution is the **method base** which can be identified with the method-related part of a *design theory*

*nexus instantiation* for EA management in the sense of Pries-Heje and Baskerville [PHB08]. Further detailing the solution, the design and components of the method base, are detailed. Thereto, we introduced the concepts of **problem**, **organizational context**, and **method building block** (MBB) and described the interplay between the concepts by taking into account the idea of situational method engineering. By introducing the **variable** concept, we enabled a unified description of the MBBs independent from the specific problem to which they are applied. Further the variable concept enables customization of MBBs to organization-specificities, e.g. already existing roles and responsibilities as well as suitable time schedules for triggering the MBBs. Following the idea of intentional modeling, we introduced a distinction of participants who act as **stakeholders**, i.e., in the role of information consumers and **actors** who represent information providers. Preparing further discussions on the integration of different MBBs, we proposed a framework describing the main activities of an EA management function, which consists of three activities: develop & describe, communicate & enact, and analyze & evaluate activity. We detailed concepts of **pre-condition** and **post-condition** which are used to determine whether two MBBs can be consistently integrated. The chapter concludes with an exposition of the **development method** to design organization-specific EA management functions consisting of the activities **characterize situation**, **configure EA management function**, and **analyze EA management function**. While we provided an overview on the steps of each of these activities taking into account the work on situational method engineering, we postpone their in-depth discussion to Chapter 5.

---

## BEAMS: Building Blocks for EA Management Solutions

---

BEAMS supplies the **method base** to store best practice knowledge on method descriptions for the EA management function, provides techniques to support the organization-specific identification and configuration of MBBs, and guides the customization and integration of MBBs into a comprehensive EA management function by a defined method. Central concept of BEAMS is the concept of MBBs that represent reusable, problem-independent design prescriptions applicable in a defined context. The MBBs in the method base are organized according to our general EA management activity framework along the activities, develop & describe, communicate & enact, and analyze & evaluate. The framework's fourth activity—configure & adapt—is a meta-activity for which BEAMS provides the **development method** to design organization-specific EA management functions using the method base as well as the **administration method** to maintain and evolve the method base. The administration method describes how best practice-proven solutions are documented in a structured manner, decomposed into MBBs to minimizing redundancies, and integrated into the method base.

The building block-based integration approach of BEAMS leads to challenges with respect to the consistency of the integrated solution. During the integration of method fragments we must ensure consistency and coherency of the resulting method description. Thereto, we provide specialized integration and configuration techniques that do not use explicit relationships between MBBs, but are based on consistency rules linking MBBs via implicit relationships. We use the concept of implicit relationships to ease contribution of new MBBs to the method base. Implicit relationships do not have to be maintained explicitly, but can be derived from MBB characteristics. The techniques also refer to language-aspects of EA management, especially to the information model used to describe the relevant part of the EA. Thereto, we discuss how meta-attributes describing certain characteristics of the information model can be used to ensure consistency. Figure 5.1 illustrates the three main contributions of BEAMS: the method base, the development method, and the administration method.

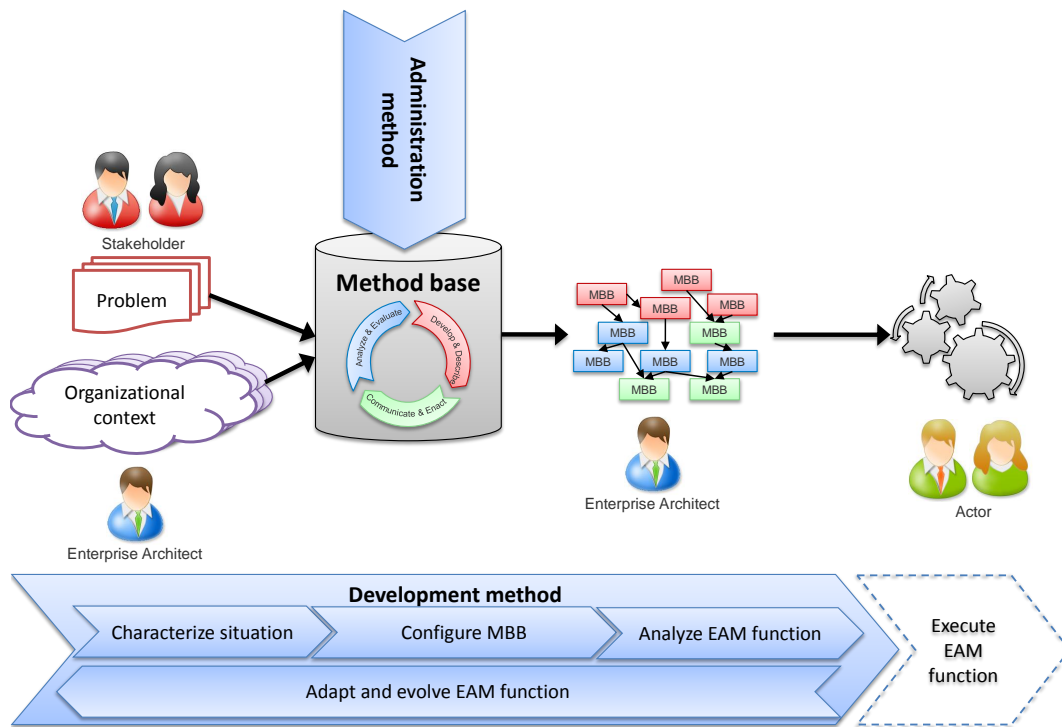


Figure 5.1.: The main contributions of BEAMS: method base, development, and administration method

This chapter is structured according to the main contributions as illustrated in Figure 5.1. Starting with the MBBs representing the central concept of BEAMS, we present a dedicated modeling language for describing the MBBs as contained in the method base in Section 5.1. The MBBs represent problem-independent method descriptions that build on the variable concept to enable configuration during the development method. The modeling language makes the MBBs accessible in a uniform way for the intended users of the development method. In line with Mylopoulos [My92] and Kühn [Kü04], we understand a modeling language as constituted of

- a *syntax* that describes the language primitives as well as principles and rules for constructing correct language expressions,
- a *semantics* that describes the meaning of the primitives and the language expressions, and
- a *notation* that describes how the primitives and expressions are represented.

In Section 5.1.1 we incrementally revisit the characteristics of a ‘good’ EA management function and the associated relevant language primitives and supply definitions of their meaning. Thus, we delineate the syntax and semantics of our language. The triple of syntax, semantics, and notation is completed with the presentation of a graphical notation in Section 5.1.2. The graphical notation builds on the familiar notation of the *business process model and notation* [Ob10b].

The development method representing the main contribution of this thesis is presented in Section 5.2 to Section 5.5. The development method uses the practice-proven MBBs contained in the method base to design an organization-specific EA management function. The development method consists of four phases, namely **characterize the situation** (see Section 5.2), **configure EA management function** (see Section 5.3), **analyze EA management function** (see Section 5.4), and **adapt and evolve EA management function** (see Section 5.5). For each of the above phases, the single steps to be performed are detailed in subsequent sections. Thereto, we first lay the basis for the different parts of the development method by augmenting the conceptual model of MBBs developed in Section 5.1 with the relevant concepts. The concepts are thereby derived along the characteristics of a ‘good’ EA management function defined in Section 4.1.1. Second, we present associated consistency rules and supply additional techniques. Third, we present the methodical part that builds on the introduced concepts and techniques. The single parts of the method are thereby presented using a three-fold approach: first, we provide an overview on the corresponding phase using a UML activity diagram; second, the single activities of the diagram are described textually. Third, the theoretic presentation of the phases is complemented by an expository example illustrating the execution of the method and the application of the techniques. The output of the single phases of the development method is an **organization-specific configuration** that is iteratively enhanced and contains the selected problems, organizational contexts, and the resulting EA management function. Case studies describing the application of the development method in real-life cases are presented in Chapter 7.

Final Section 5.6 introduces the **administration method** concerned with the development and maintenance of the method base. The section presents the findings from the industry-funded research project *EA management method library* (EAMML) in which we developed the first version of the method base. We provide an overview on the project and the participating industry partners and discuss how the method base was initially developed. Thereto, we refer to the single steps of our research method presented in Section 2.3. Furthermore, we supply the administration method detailing how newfound practice-proven solutions can be incorporated into the method base and how already described solutions can be adapted. The administration method is exemplified using a practice-proven solution gathered during the EAMML project.

## 5.1. Method building blocks

Central to BEAMS is the notion of the **MBB** as a re-usable, practice-proven, problem-independent method description that has proven to be applicable in a certain organizational context. Method descriptions of the EA management function are typically documented textually and thus are subject to further interpretation. Hence, a modeling language for EA methods can improve the state-of-the-art in documenting the EA management function. Furthermore, a respective modeling language facilitates comparing of methods, or in case of BEAMS MBBs by a defined syntax, semantics, and notation. Aiming at establishing such a language, we discuss requirements for describing EA management methods based on the quality criteria of an EA management function from Section 4.1.1. The application purpose-specific requirements are complemented with more general requirements for a modeling language elicited by Frank in [Fr09, pages 4–6]. Along these requirements, we derive relevant constituents of an MBB.

Thereby, we restrict our discussions on problem-independent and organization-independent constituents, by using the perspective on MBBs as contained in the method base. We develop a conceptual model that supplies the syntax and specify our language in Section 5.1.1. The model is described using the *Unified Modeling Language* (UML) [Ob10e] and the *Object Constraint Language* (OCL) [Ob10c]. Complementing the specified syntax and semantics, Section 5.1.2 analyzes prevalent process modeling languages and presents a notation for modeling MBBs based on the BPMN [Ob10b].

### 5.1.1. Constituents of an MBB

We subsequently supply a conceptual model defining the constituents of an MBB. Thereto, we stepwise discuss the single quality criteria, derive requirements for our language, and supply the corresponding concepts. In this vein, we iteratively develop a conceptual model representing on the one hand the syntax of our modeling language and on the other hand the starting point for the conceptual model of BEAMS. Reflecting the organization- and problem-independent nature of the MBBs as contained in the method base, we postpone the discussion of the quality criteria **Q1** and **Q2** to Section 5.2, where the MBBs are interlinked with the organizational contexts and problems. The configurability of the MBBs to a specific problem and the organizational context is nevertheless reflected by the variable concept to which we frequently refer in our subsequent discussion. The organization-specific configuration, i.e. determining the value of the variable, is discussed in Section 5.3. We subsequently start with the quality criterion **Q3** that centers around the method description itself.

According to quality criterion **Q3** a method description for EA management function should not only describe ‘what to do’ but additionally make specific descriptions on ‘how to do it’. Accordingly, we define an MBB to be constituted of single tasks, i.e. steps to be performed. A detailed method description links the different tasks and describes possible alternatives in execution (cf. [KNS92, OA09, KNS92]). The tasks constituting an MBB are connected via a combined control and information flow thereby providing detailed process execution information. An MBB specifies the ordering of the tasks in terms of sequential FLOWS as well as decisions (SPLITS) and mergers (UNIONS). For every SPLIT the MBB describes the GUARD that has to hold. These discussions lead to our first requirement for a modeling language for MBBs:

- L1 The modeling language must support to model named **tasks** and the information and control **flows** connecting the tasks. Thereby, the language must provide means to express the direction of information exchange (**source** and **target**), decisions in the information flow (**split** and **union**), and optional or alternative tasks, i.e. it must be possible to model that different tasks are executed, if different circumstances apply. These decisions must be complemented with rules (**guards**) for deciding which tasks should be executed.

Furthermore, each MBB denotes the SOURCE triggering the execution of the MBB and defines a final target, i.e. SINK that represents the end of the MBB execution. The TRIGGER of an MBB can be of two different types, namely

**event triggers** that fire on changes of EA-related concepts, e.g. creation, update, or deletion and

**temporal triggers** that fire once every specified time-interval. Typical time-intervals in this context are e.g. on a yearly or half-yearly basis.

In general both types of trigger can be applied to an MBB during configuration. Thereby, the MBB's TRIGGER VARIABLE is bound to an actual configured trigger. The method base can supply restrictions of the admissible trigger types for certain MBBs. A trigger variable of type event can further specify that certain elements are part of the trigger specification, i.e. reference concepts as contained in the information model associated with the task.

L2 The modeling language must supply mechanisms to describe the **triggers** initiating the execution of a task as well as the **sinks** that represent execution ends. The language should further allow to specify different types of triggers, i.e. **event triggers** and **temporal triggers**.

Figure 5.2 displays the concepts constituting an MBB and their relationships.

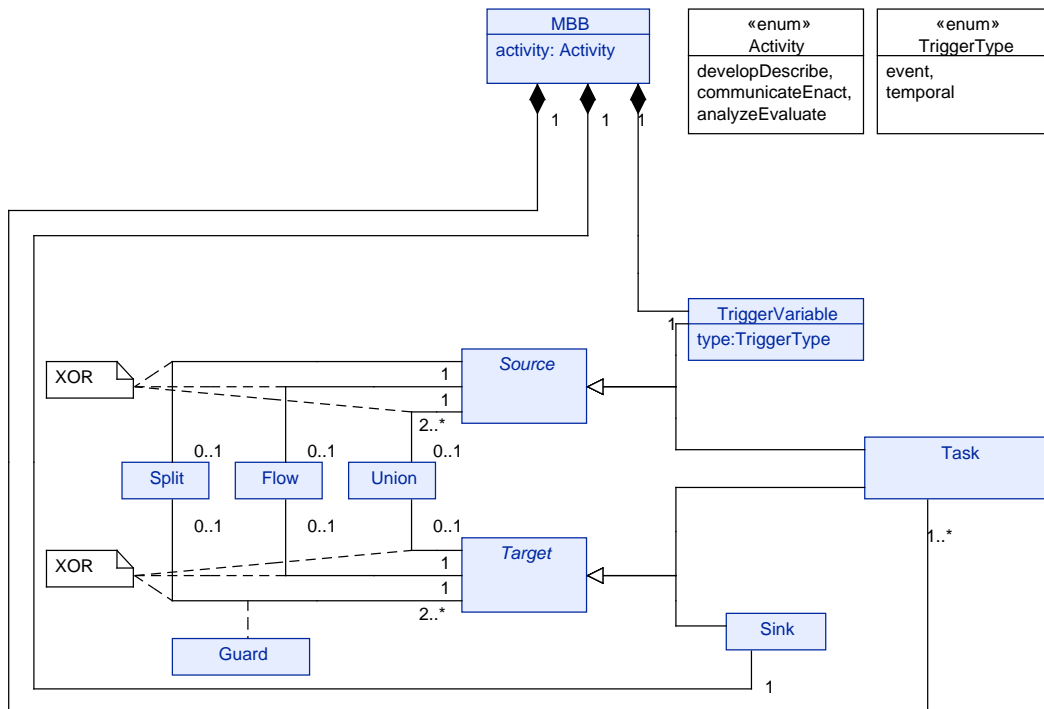


Figure 5.2.: Constituents of an MBB—detailed method description

The pattern-based approach to EA management supplies further elements used to detail the tasks to be performed in addressing a problem. For each task of the MBB, different FORCES and CONSEQUENCES can be defined. Forces describe different alternatives for the detailed implementation of a task and relate to the *techniques* as introduced by Gutzwiller (cf. Section 4.2.1). These alternatives are not selected during configuration (in the development method) but are chosen each time the associated task is conducted, i.e. ‘on runtime’. The consequences list known side-effects of the execution of a task, i.e. delineate benefits and potential liabilities. In Section 5.3.3 we revisit the concept of consequences and show how they to support the selection of MBBs from the method base.

Single MBBs are connected via information flows, i.e. exchanged information on the EA. Information flow modeling enables flexible combination of different MBBs. The execution of an MBB or a particular tasks is dependent from the availability of certain information. In

this vein, we do not need to provide an explicit concept for parallelism of EA-related activities but rely on the concept of TRIGGER, explained before.

- L3 The modeling language should support to model **consequences** that may occur, if applying a task.
- L4 The modeling language should supply mechanisms to optionally describe **forces** which influence the execution of a task, i.e. to make explicit different techniques that can be applied in the implementation of a task.

Figure 5.3 illustrates the concept of TASK, FORCE, and CONSEQUENCE.

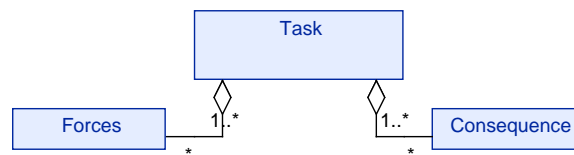


Figure 5.3.: Constituents of an MBB—task, forces, and consequences

In the EA management pattern catalog [Ch10], the tasks in a method are not explicitly linked via control or information flows. We nevertheless provide a short example from the catalog to illustrate how a textual method description with forces looks like.



**Example 5.1: Forces influencing task execution.** The process of standard conformity management according to Ernst [Er08, pages 7–14] consists of the activities *setting standards*, *analyzing standards*, *enforcing standards*, and *evaluating standards*. For reasons of brevity, we restrict our example to the activity of setting standards in the following.

To define a set of standards a multi expert evaluation method can be used. Therein, a group of experts agrees on a stop criterion (task 1), e.g. five standard programming languages, performs a group discussion (task 2) and summarizes the results of the discussion (task 3). If the stop criterion is reached, e.g. the experts have agreed on five programming languages, the method ends, otherwise the tasks 2 and 3 are repeated in an iterative manner.

Exemplary force of task 2, i.e. group discussion, is the election of a moderator who ensures that all experts get the chance to present their opinion and that all participants have a similar share of the conversation. While a moderator for a group discussion has the advantage that all opinions are considered, the participation of a moderator also requires an additional person to be involved in the discussion. Further, moderators need dedicated skills and must be accepted within the group.





Quality criterion **Q4** demands two different characteristics to be fulfilled by a method descriptions, namely a definition of ‘when’ and ‘who’. As the triggers supply information on when to execute a particular method, further concepts are required to specify the ‘who’. We distinguish different levels of involvement in tasks: being **INFORMED** that a task is executed or a result has been achieved, being **CONSULTED** during the conduction of a task, and being **RESPONSIBLE** for the execution.

**L5** The modeling language must support to model named **participants**, which are associated to corresponding tasks. By this association, the languages express that a participant is involved in the execution of the task. The language must further allow to specify the type of involvement, i.e. if a participant is **responsible**, **consulted**, or **informed**.

The actual participants depend on the using organization. Therefore, we introduce the organization independent concept **PARTICIPANT VARIABLE**. This variable is bound during the development method to an organization-specific role that can be a stakeholder of an associated EA management-problem or an actor involved in the execution of a task. Figure 5.4 illustrates the relationships between **PARTICIPANTS** and **TASKS**.

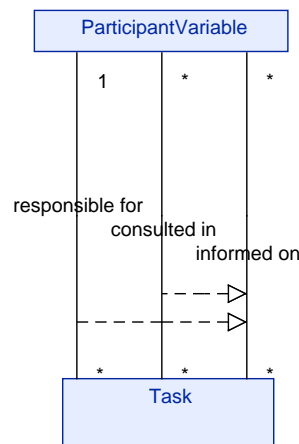


Figure 5.4.: Constituents of an MBB—participants and tasks

Each task is executed by a responsible participant represented by a **PARTICIPANT VARIABLE** in the description of the method. Beside the mandatory relationship to the executing, i.e. **responsible** participant variable, each task can relate to other participant variables as well, namely variables representing participants that are **consulted** or **informed**. The distinction between the different levels of involvement pertaining to a single task is based on the *responsible, accountable, consulted, and informed* (RACI) model of the *Control Objectives for Information and Related Technology* (CobiT) [IT09]. Two adoptions to the RACI model apply: 1) we assume that every involved participant is informed about the relevant information. 2) we assume that exactly one participant is responsible for executing a task rendering a distinction between *accountable* and *responsible* superfluous. The three different levels of involvement in an EA management task are subsequently exemplified using an EA management method of an industry partner from the telecommunication industry.

**Example 5.2: Participant involvement.** The EA management-related problem of the industry partner is to enact principles and standards for business applications. The enactment of standards is performed by requesting a ‘statement of defense’, if a standard is violated.

If project managers (responsible participant) want to realize a non-standard compliant business application, they are requested to provide a statement of defense. The statement has to achieve accreditation from the EA review board (consulted participant) which consists of the enterprise architects and the heads of the business departments. Further, the CIO receives a notification of the deviation (informed participant).

Each participant involved in an EA management task must be provided with information on relevant parts of the EA. In line with quality criteria **Q5** EA artifacts are used as means to communicate relevant information to the participant. A `VIEWPOINT VARIABLE` further details the participant’s perspective on the EA taken in the corresponding task. Such a variable acts as placeholder for an appropriate architectural viewpoint. In line with Wittenburg [Wi07] and Matthes et al. [Ma08], we distinguish different types of viewpoints that can be supplied in an architectural description: textual and graphical viewpoints. We showed in [Bu08b, Bu09b] that these different participants of EA management preference different types. The method base incorporates the best-practice knowledge by `RECOMMENDING` and `DISCOURAGING` certain viewpoints for a viewpoint variable associated with a dedicated task. Further, the viewpoint variables are distinguished with respect to the intended usage of the viewpoint in **representation** and **notation**. A viewpoint of type `representation` is used to inform a stakeholder about a relevant part of the EA, i.e. to grant reading access. Conversely, a viewpoint of type `NOTATION` is used when an actor has to supply architectural information, i.e. to grant read and write access.

**Example 5.3: Types of viewpoints.** Questionnaires used during EA-related information gathering are obvious examples of viewpoints of the type `notation`. In contrast organigrams represent viewpoints of the type `representation` as they are typically used in a read-only fashion.

The actual configuration of the viewpoint is a language-related aspect of the EA management function and thus not discussed in the scope of this thesis. We use a building block-based transformation language to perform these configurations as discussed in [Bu10a, pages 29–31].

- L6 The modeling language must support to model named **viewpoints**, which are used by a participant that is involved in the execution of a task. The modeling language must support to specify the type of viewpoint, i.e. **representation** function or **notation** function. Furthermore, for each viewpoint the modeling language must support the specification of recommended or discouraged viewpoint types.

Figure 5.5 displays how VIEWPOINT VARIABLES objectify the INFORMS-relationship between PARTICIPANT VARIABLE and TASK.

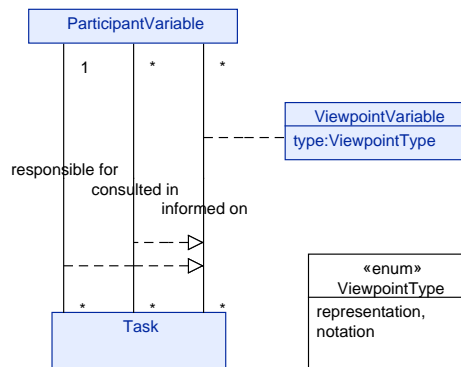


Figure 5.5.: Constituents of an MBB—participants and viewpoints

A participant that is only informed during the execution of a task cannot perform updates to the provided information and is hence limited to a viewpoint of the type REPRESENTATION. Expressed in OCL this constraint reads as:

```

context ViewpointVariable
inv: self.oclIsTypeOf(informed on) implies self.type = representation

```

The above list of application purpose-specific requirements can be extended with three general requirements for a modeling language as elicited by Frank in [Fr09, pages 4–6].

- L7 The concepts as introduced in the modeling language should correspond to concepts, which modeling experts from the modeling domain are familiar with. Similarly, the graphical notation of the language should correspond to prevalent graphical notations in this modeling domain (cf. requirements *U1* and *U3* of Frank [Fr09]).
- L8 The modeling language should facilitate the development of tools for the creation of and for providing execution guidance for the described methods (cf. requirement *A4* of [Fr09]).
- L9 The modeling language must offer all concepts needed to describe methods in the context of EA management, but must support restriction to exactly these concepts to prevent the introduction of “accidental complexity” (cf. requirements *A1* and *A2* of Frank [Fr09]).

Within the list of requirements, the key words “must”, “should”, and “optional” are to be interpreted as described in RFC 2119 (cf. Bracher [Br97]). This interpretation yields a distinction between ‘mandatory’ and ‘optional’ requirements, which can be exemplified with requirements **L3**, **L4**, **L7**, and **L8**. These have strictly optional character, while the other requirements have both optional and mandatory constituents.

Figure 5.6 provides the synthesized conceptual model describing the problem-independent MBBs as contained in the method base.

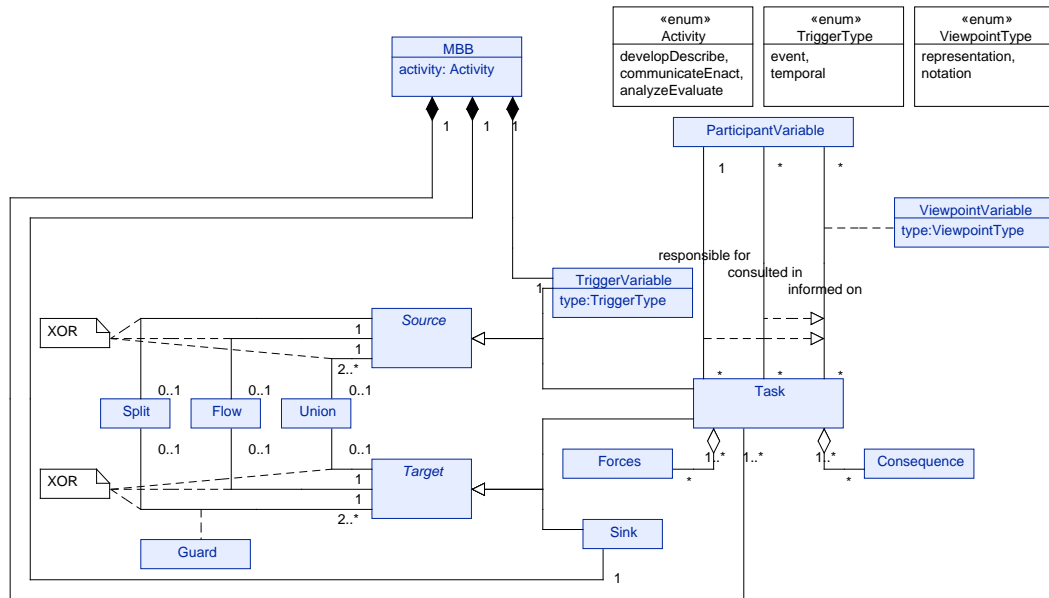


Figure 5.6.: Conceptual model of an MBB

### 5.1.2. A notation for method building blocks

The requirements elicited above give rise to the assumption that process modeling languages is appropriate for modeling EA management methods. In doing so, we can identify the tasks of a method with the processes or process steps as contained in a process modeling language. This nevertheless points to a subtle complexity as process modeling languages are designed to describe the sequence of process steps on a rather specific level, i.e. typically describe control flows, while method descriptions in the context of EA management are restricted to information flows at least if a comprehensive EA management function is considered. Revisiting existing process modeling languages as the *extended event driven process chains* (eEPC) [KNS92], *business process model and notation* (BPMN) [Ob10b], *business process execution language* (BPEL) [OA09], *integrated definition for process description capture method* (IDEF3) [Ma95], *petri nets* [Ba96], *yet another workflow language* (YAWL) [Ho10], and the general purpose modeling language *unified modeling language* (UML) [Ob10d, Ob10e] based on the above requirements, it becomes apparent that none of the existing languages fulfills our requirements out-of-the-box. Nevertheless, some of the languages well address some of the requirements while in contrast providing more sophisticated concepts that are not necessary for the context of modeling EA management methods (cf. BPEL, BPMN, and eEPC). Table 5.1 provides an overview on the analysis results of the aforementioned languages. The detailed description of the analysis can be found in [BMS10h, pages 4–6]. The symbols used in the table range from (●) indicating complete fulfillment over (◐) indicating medium fulfillment to (○) indicating total lack of support.

	UML <sup>1</sup>	BPEL	BPMN	eEPC	IDEF3	Petri nets	YAWL
L1	○	○	○	○	●	○	●
L2	○	● <sup>2</sup>	●	● <sup>2</sup>	○	○	●
L3 <sup>3</sup>	○	○	○	○	○	○	○
L4	○	○	○	○	○	○	○
L5	○	●	●	●	○	○	○
L6	○	○	○	○	○	○	○
L7	○	○	○	○	○	○	○
L8	○	○	○	○	○	○	○
L9	○	○	○	○	○	○	○

<sup>1</sup> The analyzed diagram type is the *activity diagram*.

<sup>2</sup> BPEL and eEPCs do not provide a dedicated trigger or source concept but use the concept of *events* that represent initiations, results, and connections between activities.

<sup>3</sup> Concern and execution context can mostly be supplied only by textual annotations.

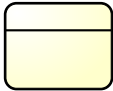








Table 5.1.: Fulfillment of requirements of the selected process modeling languages

In line with requirement **L7** which votes for re-using a notation a domain expert is familiar with, we propose a notation that is grounded in the notation of the BPMN in the following. While all three languages BPEL, BPMN, and eEPC achieved similar results in our above analysis, we decided to use a BPMN similar notation for two reasons. First BPMN is commonly used and further developed by the *Object Management Group* (OMG). Second, a configurable open source and web-based tool support for BPMN exists that can be re-used in the context of this thesis to enable tool-supported creation and manipulation of MBBs (see the BEAMS configurator presented in Chapter 6).

With the syntax and semantics of our modeling language for MBBs defined in Section 5.1.1, we subsequently propose a graphical notation for the concepts. We do not provide a graphical representation for all concepts specified by the syntax of our modeling language. The consequence of a task execution, for example, is typically documented using a textual description and therefore not represented in the notation. Due to clarity, we omit concepts that do not interlink different elements but can be interpreted as ‘attributes’ of one concept from receiving an own symbol. These concepts should nevertheless be made available for a user during the development method as they provide decision support for the selection and configuration of MBBs (an respective prototypic tool is discussed in Chapter 6). For each concept for which a symbol in the notation exists a short explanation is given in the following. If a corresponding concept is available in the BPMN 2.0 specification a mapping is given in brackets. Table 5.2 provides a summary of the graphical elements provided by the notation.

- **Task** (activity): An EA management method consists of different steps, so called tasks.
- **Participant**: Tasks are executed by acting systems or persons, of which the later, in a well defined method, act in a distinct role, e.g. CIO, project manager, etc. The *pool* element of the BPMN can be used to represent participants. Different types of involvement for participants exist. A participant can be
  - **responsible** for executing a task,
  - be **consulted** in executing the task, or
  - be **informed** on the execution of the task.

- **Viewpoint**: A viewpoint represents an EA-related artifact, i.e. an architectural description, that represents certain information, e.g. a visualization, questionnaire, or report. The viewpoints are presented to the task's associated participants in an appropriate fashion and are used by them during the execution of the task. Viewpoints are thereby either of type
  - **representation** function, i.e. to visualize information that is accessed read-only or
  - **notation** function, i.e. to visualize, gather, or update information.
- **Control flow** (sequence flow): The different tasks constituting a method are connected via control and information flows. An information flow specifies that information objects are handed over from a source to a target. Two specializations of the information flow concept exist.
  - **Split** (gateway): The information flow is not necessarily linear, but may contain points, where decisions take place. The subsequent tasks, which receive information, are decided based on **guards** associated with the different options.
  - **Union** (gateway): Information flows originating from different tasks are integrating via a merge.
- **Trigger** (start event): A trigger initiates the execution of a sequence of tasks. A trigger can be of two different types.
  - **Event trigger**: Changes in the EA description, e.g. new documented concepts or updated architectural descriptions, can trigger the execution of a task sequence.
  - **Temporal trigger**: Some tasks are not triggered by changes in the architectural description but are initiated at certain points in time, corresponding to specific time intervals.
- **Sink** (end event): A sequence of tasks terminates with at a sink. After the sink has been reached, the post-conditions of the task are fulfilled.

Concept	Notation	Concept	Notation
Task		Control flow	
Participant		split	
responsible for		union	
consulted in		Trigger	
informed on		event	





Viewpoint		temporal 
representation function		Sink 
notation function		

Table 5.2.: Graphic elements for describing EA management methods

Together, syntax, semantics, and notation form a modeling language for EA management methods that relates to the language counterpart by referencing the concept of information model, delineating the constituents of the EA, and viewpoints, defining the visualizations used. Complementing the theoretic discussion on our modeling language for MBBs, we present an exemplary method description below that is modeled using the above defined notation in Figure 5.7.



**Example 5.4: An exemplary EA management method.** The method starts by gathering information about the current state of the application landscape. In our example this information is gathered via revisiting existing documentation and filling a questionnaire. Thereby, the enterprise architect has to decide, which information should be gathered, e.g. information about applications and the used technology. To validate the gathered information, the filled questionnaire needs to be reviewed by the application owners. Based on the description of the current state, the technology homogeneity is analyzed. Thereby the enterprise architect is the responsible participant and uses a defined viewpoint *Standard Conformity (V-5)*<sup>1</sup>. The analysis results are used in the following phase by the standard managers to create, update, or delete standards. Thereby bar charts like e.g. *Effects of Project Proposals on Technology (V-38)* are used and the EA board is informed on changed standards. The subsequent process step, applying standards, is concerned with defining which business application should conform to one of the standards defined before, therefore the enterprise architect uses the viewpoint *Clustering by Standards (V-6)*, which details on the environment of the used technology. The enforcement of standards, is either performed via vertical or horizontal escalation (see

<sup>1</sup>The viewpoints are taken from the EA management pattern catalog [Ch10].

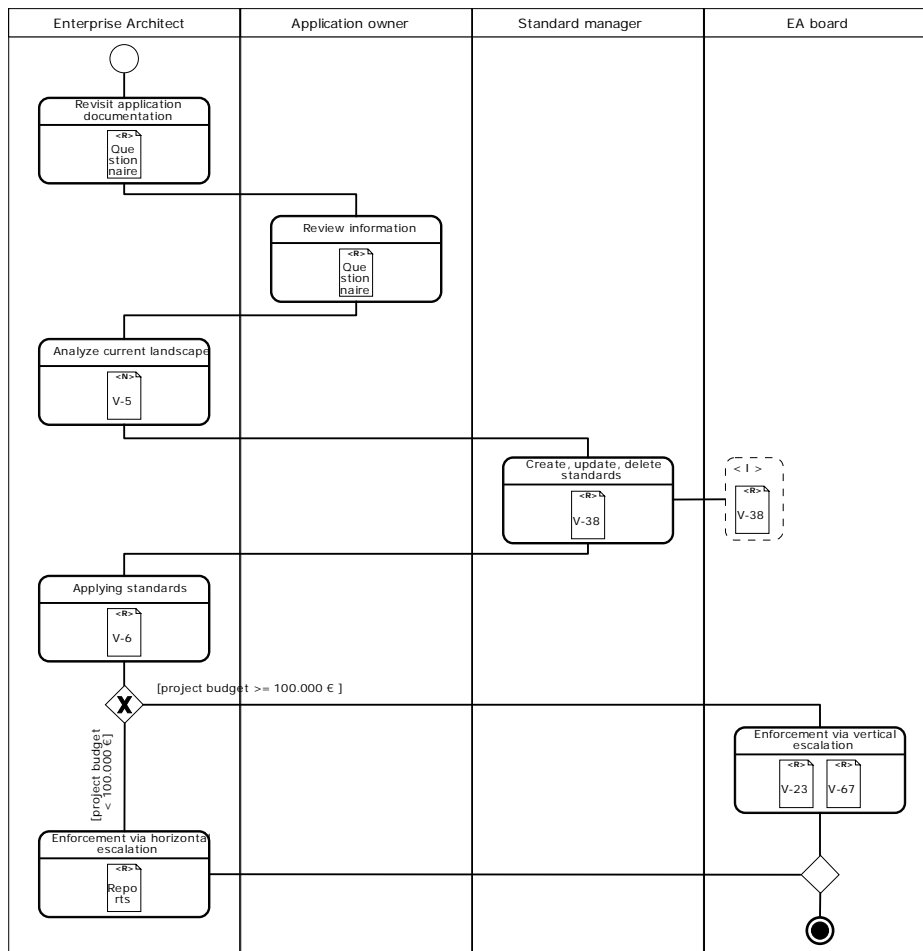


Figure 5.7.: A method description of the EA management function

our discussion in [Bu09b]). A typical decision criterion for horizontal vs. vertical escalation is the estimated budget of the project. In our example the vertical integration is chosen, if the project budget is more than €100000, then the decision about the future of the application is handed over to a EA board, which uses a report-based viewpoints, *Technology by Architectural Standard (V-23)* and *Standard Conformity Exception (V-67)*. In the case where horizontal escalation is used, the enterprise architect is allowed to impose obligations e.g. taxing the project (see [Bu09b]). These obligations are documented using reports like word documents.





## 5.2. Characterize situation

In the preceding section, we discussed the constituents of an MBB from the method base perspective, i.e. as problem- and organizational-independent method descriptions. Complementing the examination of the quality criteria, we present a perspective on MBBs in this section, which details the ‘environmental’ concepts of an MBB. While some of these concepts are structurally interlinked with the MBBs in the method base, others are used to establish **implicit relationships** to organize the MBBs in the method base. In Section 5.2.1 we augment the conceptual model of an MBB with the environmental concepts. The resulting conceptual model supports the BEAMS **development method**. In addition, a technique supporting the characterization of the situation is supplied in Section 5.3.2. The first part of applying the method base during the phase **characterize situation** of the development method is outlined in Section 5.2.3.

### 5.2.1. Constituents of the method base

We define a first set of concepts of BEAMS that represent the embedding environment of the MBBs. According to quality criteria **Q1** and **Q2** this environment consists of the organizational context and a specific EA management-related problem. This environment also reflects the idea of a design theory nexus instantiation as introduced in Section 2.2.3. The MBB and its environmental concepts are shown in Figure 5.8. The figure employs a color-coding to classify the illustrated concepts as belonging to the method-related part of an EA management function (highlighted in blue) or belonging to the organization-specific parts.

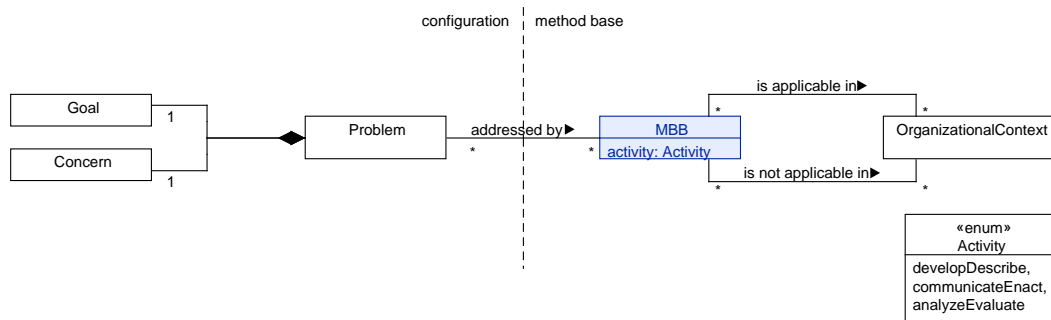


Figure 5.8.: BEAMS: interrelating context, problem, and MBB

An MBB can be categorized along the activity that it contributes to. Therefore, an MBB belongs either to the develop & describe, communicate & enact, or analyze & evaluate activity. The problem-independent MBBs from the method base have to be configured during the development method. As part of the configuration each selected MBB is linked to the specific problem. An EA management-related PROBLEM consists of an abstract GOAL to pursue and the CONCERN to which the goal applies. The distinction between the ‘where’, i.e. the concern, and the ‘what’, i.e. the goal, enables flexible configuration of the MBBs. Following example demonstrates the resulting flexibility along three *problems*<sup>2</sup> as contained in the EA management pattern catalog [Bu08b, pages 32–38].

<sup>2</sup>The problems listed in the EA management pattern catalog have been called *concerns* in the first version.



**Example 5.5: Problems from the EA management pattern catalog.**

C4: Which technologies, e.g. programming languages, middleware, operating systems, database management systems, used in the application landscape should be replaced, which ones should be kept?

C5: Which activities or projects have to be started, to increase conformance to standards? What has to be done to modify the current business applications to increase their conformance to standards and reduce heterogeneity?

C101: Which activities or projects have to be started to improve conformance to architecture standards? Which modifications to the currently used business applications are necessary to achieve conformity?

The EA management pattern catalog groups its problems in different categories. Above problems belong to the category “technology homogeneity” [Ma08, page 31]. A more general category ‘increase homogeneity’ would be possible, but operates on different concerns, e.g. programming languages, projects, or architectural standards.



Each of the three problems (C4, C5, and C101) from the EA management pattern catalog relates to a particular method (M-Pattern). These M-Patterns differ with respect to the performed analyses of the EA as well as with respect to the underlying information models. They nevertheless are identical in the steps that are taken to gather the relevant architectural information. This leads to redundant descriptions in the pattern catalog (cf. our discussion in Section 4.2.4). We seek to avoid such redundancies and increase flexibility in method configuration by detaching methods and problems as far as possible. Furthering this idea, we decompose any EA management-related problem into a goal and a concern. Thereto, we understand a goal in line with Basili et al. in [BCR94] and the *Business Motivation Model* (BMM) of the OMG [Ob10a, page 23] as follows.

**Definition: Goal**

A goal represents an abstract objective of the EA management function that describes a state or condition of the enterprise to be brought about or sustained through appropriate means. A goal is thereby defined for one or more objects of the EA.

An EA management-related goal has to be substantiated during the development of an appropriate management function to identify the corresponding concepts of the EA. Thereto, the information model of the corresponding concern is extended with modeling elements that reflect the goal and enable measurement of goal attainment.

---

To not be mixed up with the term “concern” from the ISO Std. 42010 (see Section 3.1.1), as used in this thesis, the *concerns* should aptly be named **problems**.

**Example 5.6: Operationalizing goals.** The goal *increase homogeneity* applied on the concern *business application uses technology* can be operationalized by augmenting the EA concept TECHNOLOGY by a boolean attribute ISSTANDARD.

Any problem in an EA management function is reflected by an information model composed by the models of the corresponding goal and concern. We subsequently provide a definition of the term problem, based on the notions of goal, as introduced above, and concern, as introduced in Section 3.1.1.

**Definition: Problem**

A problem specifies the objective of the EA management function by defining what to achieve, i.e. the goal, and where the goal should be applied, i.e. the concern.

Each MBB describes the organizational contexts in which it has proven to be applicable or the organizational contexts in which it has proven not to be applicable. The different organizational contexts are explicitly linked to the associated MBBs for which they impact applicability. These relationships represents practice knowledge that cannot be generalized. We use an M-Pattern from the EA management pattern catalog [Bu08b, pages 44–47] to illustrate this kind of relationship.

**Example 5.7: Organizational context.** In the method descriptions of *M4: management of blueprint conformity of the application landscape* the enactment of standards is performed by informing the developers of new application systems on the organization-specific standards early in the specification process. An evaluation of conformance to these standards is performed after project completion. The evaluation does not result in consequences. Instead the enterprise architects use the findings to assess, if the right standards have been chosen with respect to investigating, if the developers adhere to these standards.

While the approach proposed in M4 represents the non-interfering way to enact standards, it can only be beneficially applied, if the culture of the organization is an open one, i.e. one where the developers voluntarily adhere to proposed standards, are open for changes, and willing to follow optional guidelines. The method is hence also applicable in ‘bottom-up’ endeavors, i.e. ones where no official commitment or support for EA management exist. In those contexts other ways of enacting, which would require upper management support, are not feasible.

Based on the understanding of the term *situation* from situational method engineering and the context description as defined in the pattern language for EA management, we define an organizational context as follows.

**Definition: Organizational context**

The organizational context is the combination of circumstances in which the EA management function is intended to operate in a given organization. It describes the circumstances that constraint the applicability of different MBBs.

During configuration MBBs are bound to problems of the organization. Thereby, the users specify how the problem-related information is gathered, documented, updated, or deleted during the execution of the tasks. An MBB is parameterized in the development method with an information model that represents the selected problem. The MBBs as contained in the method base link to an INFORMATION MODEL VARIABLE that is assigned with a specific information model during configuration in the development method.

An MBB is organized from TASKS of which each acts on an INFORMATION MODEL VARIABLE. In the method base, this variable is used as placeholder for an actual information model, which is supplied during configuration (cf. Section 5.3). For any information model variable a lower bound can be specified (REQUIRES). This bound designates an information model constituted of the concepts that must be supplied with the information model assigned during configuration.



**Example 5.8: Specifying a lower bound for an information model.**

A typical circumstance in which a lower bound for the information model must be specified, is a task concerned with the assessment of projects. While no assumptions have to be made with respect to the exact information model during the assessment, it is nevertheless necessary that the information model at least covers PROJECTS.



The concepts specified as ‘lower bound’ of tasks are limited to the concepts representing **cross-cutting aspects** of an EA as introduced in Section 3.1.2. These concepts supply core concepts for managing the evolution of the EA and are thus part of each EA management initiative. Although different organizations can use different terms for these concepts. Figure 5.9 displays the relationships between task and information model. We use the color-coding introduced in Section 4.3 to indicate MBB-related concepts (blue) and language-related concepts (green).

Another critical aspect of developing EA management functions using the method base is linked to the integration of different MBBs. Quality criteria **Q6** demands that an EA management function is implementable. Using building blocks to design an EA management function can lead to an inappropriate method description a) due to the selection of tasks that are not applicable in the given organizational context or b) due to unsatisfied information demands of participants, e.g. the use of information that has not been documented before.

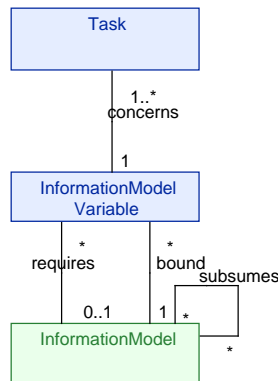


Figure 5.9.: Constituents of an MBB—tasks and information model

The former issue relates to an appropriate selection of MBBs and is discussed in more detail in Section 5.3. Later issue relates to information inconsistencies and is further investigated in the following.

Information to be processed by a task is gathered via notation viewpoints such that an arbitrary configuration of an MBB can fail to be implementable. If for instance a task is used that defines architecture standards based on the currently used technologies and no information on currently used technology is available, the resulting method description can be ‘inconsistent’. We revisit the concept of INFORMATION MODEL introduced above to avoid the design of such inconsistent EA management functions. Two particular sources of information inconsistencies exist, namely

- tasks that inform a user about a part of the information model, which has not previously been documented in another task.
- triggers that raise inconsistencies. Thereby different types of inconsistencies raise depending on the type of trigger. In the case of the
  - temporal triggers: inconsistencies arise, if information is e.g. used every half a year but maintained on a yearly basis,
  - event trigger: inconsistencies arise, if a task should be triggered by changes to information that is never maintained in another task.

Inconsistencies related to the information model arise from the fact that MBBs change certain characteristics of the EA information base or parts thereof. These characteristics are represented by **meta-attributes** and can be changed by tasks. The performed change is described in the **pre-conditions** and **post-conditions** of a task. Following example 5.9 illustrates this conception.



**Example 5.9: The concept of meta-attributes.** An MBB concerned with gathering EA-related information via interviews is completed when

the respective information has been gathered, i.e. changes the meta-attribute ‘documented’ of the associated information model to ‘true’.

Figure 5.10 provides the excerpt from our conceptual model detailing the relationship CONCERNS between TASK and INFORMATION MODEL VARIABLE with the concepts of PRE-CONDITION and POST-CONDITION that build on the concept of META-ATTRIBUTE.

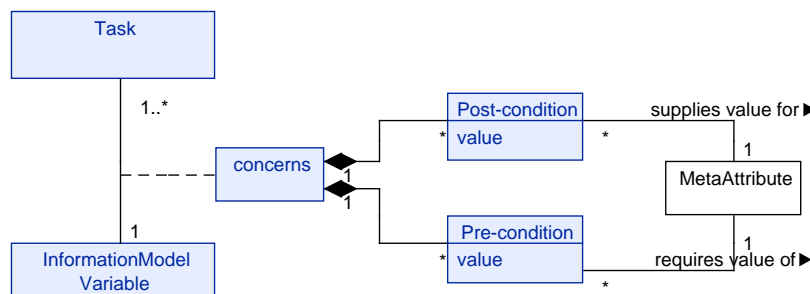


Figure 5.10.: Constituents of an MBB—pre-conditions and post-conditions

Each TASK operates on an INFORMATION MODEL VARIABLE and details the operation in terms of PRE-CONDITIONS and POST-CONDITIONS. These pre-conditions and post-conditions supply or require certain values for the META-ATTRIBUTE to hold. An exemplary pre-condition supplied as part of an information model variable would state that the task can only be executed, if information conforming to the given information model is documented. The same variable might state as post-condition that the documented information is cleared for communication. Any MBB specifies a dedicated set of pre-conditions and post-conditions. The conditions employ a dedicated terminology as specified by the META ATTRIBUTES. If more than one condition applies for an MBB, the single conditions are combined via an *AND*, i.e. a *logical conjunction* (cf. Carnap [Ca58, page 7]). The concept of pre- and post-conditions can be exemplified along the pattern for standard conformity management as presented by Ernst in [Er08, pages 7–14].

**Example 5.10: Pre-conditions and post-conditions.** To reduce heterogeneity and in particular the number of used programming languages, an enterprise wants to analyze the current conformity of business applications with standards. The analysis should be performed using a pattern-based evaluation method. A pre-condition for performing the analysis is that information about the currently used programming languages is documented (PROGRAMMINGLANGUAGE.DOCUMENTED). After the analysis is performed the post-condition of the task defines that the boolean attribute ISSTANDARDCONFORM of a business application is maintained.

The EA management function cannot operate isolated from other *enterprise-level management functions* (ELMF) that affect the evolution of the organization. Instead the EA management function influences the decision making process in these ELMFs, e.g by defining guidelines for the future evolution or by providing a holistic view on the EA. Thereto, EA management establishes quality gates in the ELMFs, embeds consulting tasks, or communicates EA-related information to ELMF stakeholders (cf. quality criterion **Q7**). Special MBBs take this role and may hence be not ‘part’ of the EA management function but have to be integrated into the processes of related ELMFs. This special type of MBBs is called **mixinMBB**. Exemplifying the concept of the **mixinMBB** a best practice for project portfolio management is sketched subsequently.



**Example 5.11: The concept of mixinMBB.** Project proposals in large enterprises typically have to undergo an approval process prior to receive a budget. This process is typically called *project portfolio management*, which aims at determining the optimal mix and sequencing of projects to best fit the organization’s overall objectives.

If a problem addressed by the EA management function is concerned with increasing homogeneity, the project proposals should be assessed during the project portfolio management from an EA perspective. Thereto, the proposals are assessed for compliance with architectural principles and standards or with respect to the envisioned target state of the EA. The project portfolio management process is thus enhanced with a project step in which an enterprise architect, who is familiar with the architecture principles, standards, or the target state of the EA assesses the proposed projects and provides a recommendation.



A **mixinMBB** must be embedded in another method description, i.e. an ELMF or another MBB. In Example 5.11 the assessment of proposed projects is only usable, if the results of the assessments are available to the responsible persons deciding on the project portfolio. A **mixinMBB** is never triggered separate from its embedding process. Therefore, each **mixinMBB** supplies an event trigger that is configured during the development method to fire with the post-condition of the preceding process step. Figure 5.11 introduces the concept of **MIXINMBB** to our conceptual model and relates it to the related method via the concept of **MIXINTARGETVARIABLE**.

Following OCL constraint ensures that **MIXINMBBs** are associated to correctly typed **TRIGGER VARIABLES**:

```
context TriggerVariable
inv: if mbb.ocIsTypeOf(MixinMBB) -> self.type = event
```

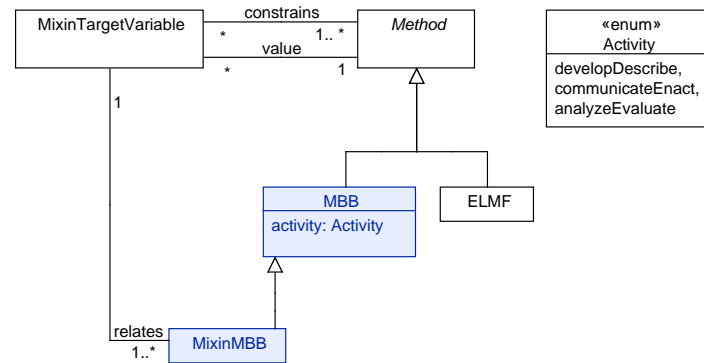


Figure 5.11.: Integrating MBBs with other enterprise-level management functions

Figure 5.12 shows the integrated conceptual model of BEAMS. The concepts representing methodical elements are highlighted in blue, concepts from EA modeling languages are shown in green, and concepts representing organization-specific customization during the development method are shown in white.

### 5.2.2. Rules and technique to characterize a situation

Environmental factors for an EA management function have to be determined during the characterize situation phase of the development method. Therefore, an enterprise architect has to specify related ELMFs that contribute to the EA management function, e.g. by representing information sources. The information demands of the EA management function are reflected in the **information model**. The concept of **meta-attribute** can be used to specify characteristics of this information model. In this vein, existing information that is re-used by the EA management function is classified as “documented”. Documented thereby defines that a dedicated process exists that ensures that the information is maintained. The referenced process is thereby either an already existing part of the EA management function or a related ELMF.

We provide a simple technique to support the definition of existing information sources. The **characterization technique** consists of two steps that are iteratively applied, namely

1. select concept from the information model,
2. select meta-attribute that applies for this concept,
3. define the ELMF that is responsible for maintaining the concept, and
4. update the organization-specific configuration accordingly.

Example 5.12 illustrates the application of the characterization technique along a fictitious organization called BS&M.



**Example 5.12: Applying the characterization technique.** At BS&M an EA management initiative is started. To ensure that information is not



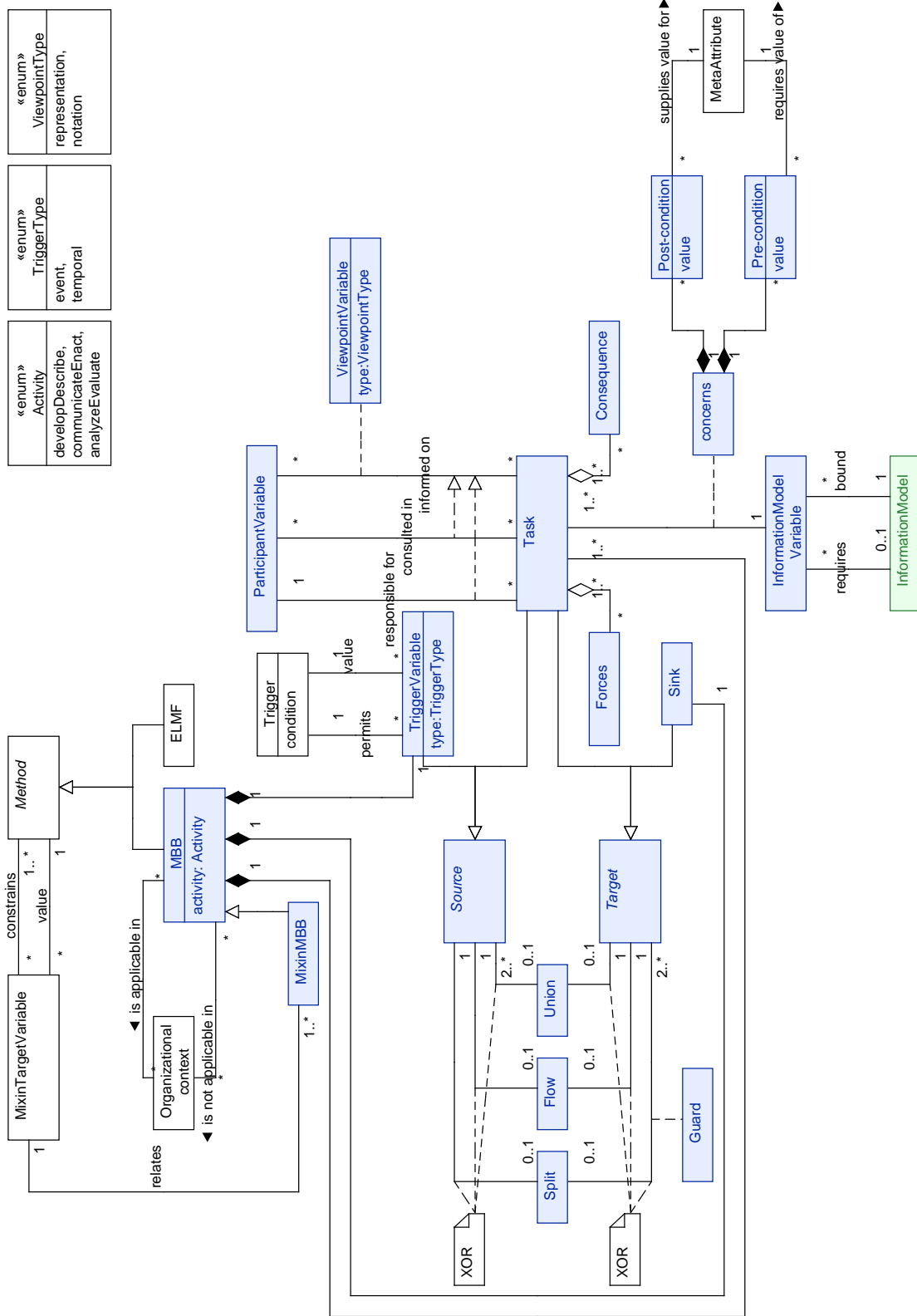


Figure 5.12.: The MBB constituents and their environmental factors as interlinked by the BEAMS method base

redundantly maintained, the enterprise architects arrange a kick-off meeting to which employees from the different departments are invited. During the kick-off meeting, the enterprise architects present the information demands for their EA management endeavor and ask for already existing information sources.

By iteratively applying the steps of the characterization technique the enterprise architects identify that information on BUSINESS APPLICATIONS and the using ORGANIZATIONAL UNITS is already available in a *configuration management data base* (CMDB) which is maintained by an *IT infrastructure library* (ITIL) initiative. The organization-specific configuration is updated accordingly:

- BUSINESSAPPLICATION.DOCUMENTED
- ORGANIZATIONALUNIT.DOCUMENTED
- ORGANIZATIONALUNITUSINGBUSINESSAPPLICATION.DOCUMENTED



### 5.2.3. Development method

Subsequently, we detail the phase **characterize situation** that represents the first part of our development method. We designate the involved participants and delineate the single steps to be performed. While we assume the enterprise architect to be the typical user of the method (see our discussion in Section 4) other stakeholders of the EA management initiative need to be consulted during this phase to identify the problems to be addressed. The phase characterize situation consists of three sub-activities, namely **determine organizational context**, **identify and operationalize EA-related problem**, and **specify existing information sources**. The output of the characterize situation phase that is stored in the **organization-specific configuration** is a set of defined organizational contexts, an actual problem to be pursued, and information on already existing EA-related content. Figure 5.13 shows a detailed activity diagram describing the single steps to be performed to achieve the aforementioned outcomes.

To develop an organization-specific EA management function, the enterprise architects have to characterize the situation in which the management function should be embedded in the step **determine organizational context**. Different factors and criteria influencing the applicability of an EA management function exist. To support the enterprise architects in characterizing the situation, a **catalog of organizational context** descriptions that impact the applicability of the MBBs in the method base is provided. The enterprise architects browse the catalog and select the organizational contexts that reflect the current situation in the organization. Output of the step is an organization-specific configuration containing a set of selected organizational contexts that describe the environment in which the EA management function should be embedded.

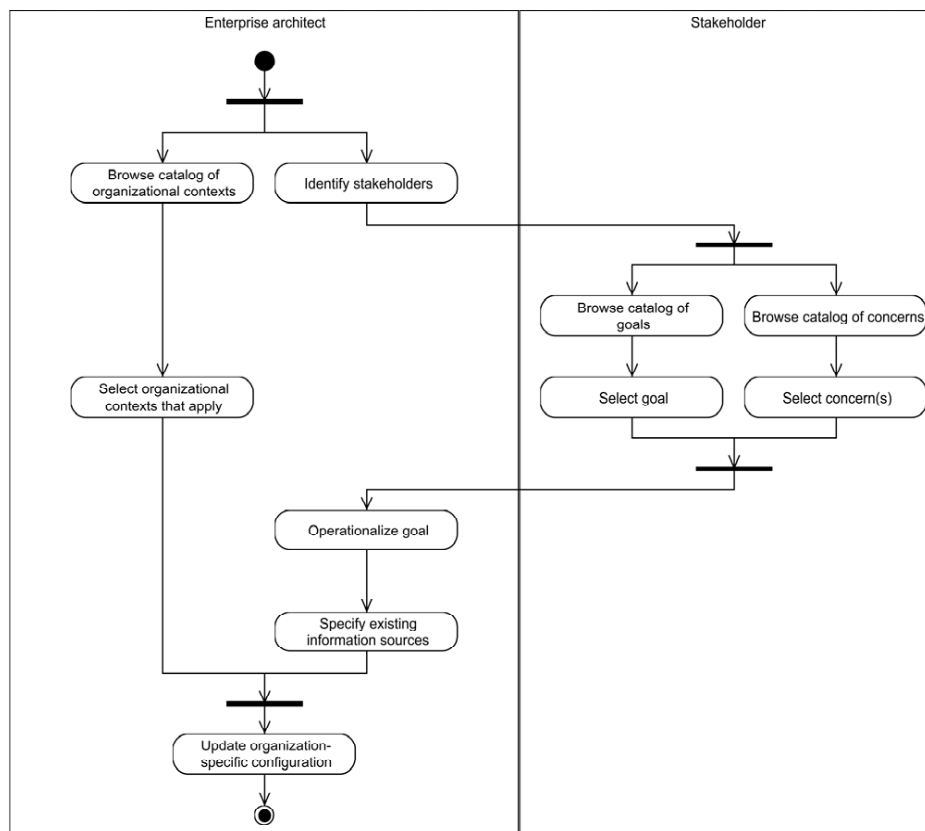


Figure 5.13.: Development method: characterize situation



**Example 5.13: Development method—select context.** In our fictitious example, we accompany the enterprise architects from a fictitious organization, namely the financial service provider BS&M through their first experiences with EA management. The situation at BS&M can be characterized as follows: Over the last years BS&M has been constantly growing resulting in a heterogeneous application landscape due to a rising number of business request to IT.

To cope with the historically grown and proliferating application landscape, an ITIL project was launched a year ago. Furthermore, the federated IT departments were centralized and a process for deciding on the project portfolio based on defined criteria as estimated project costs was set up to increase standardization of the provided IT solutions.

Browsing the catalog of organizational contexts the enterprise architects select the following characteristics that are subsequently stored in the configuration, namely

- the initiative can be characterized as “bottom-up initiative” as no official mandate from upper management exists,

- the organizational structure supplies a “centralized IT department”, and
- “office tools” should be used in the initiative as no dedicated tool support for EA management yet exists and no official budget is available for the initiative.

Besides the environment in which the EA management function should be embedded the enterprise architects have to **identify the EA-related problems** to be addressed. Therefore, stakeholders of the EA management initiative are identified and consulted. Typically the identified problems are described by the stakeholders on a rather abstract level. BEAMS provides a collection of typical EA management-related problems. This collection is organized in two catalogs, namely the catalog of goals defining ‘what’ should be achieved, and the catalog of concerns specifying ‘where’ the goals can be applied. Based on the combination of a selected goal and a concern, a problem is defined and an information model is determined<sup>3</sup>.

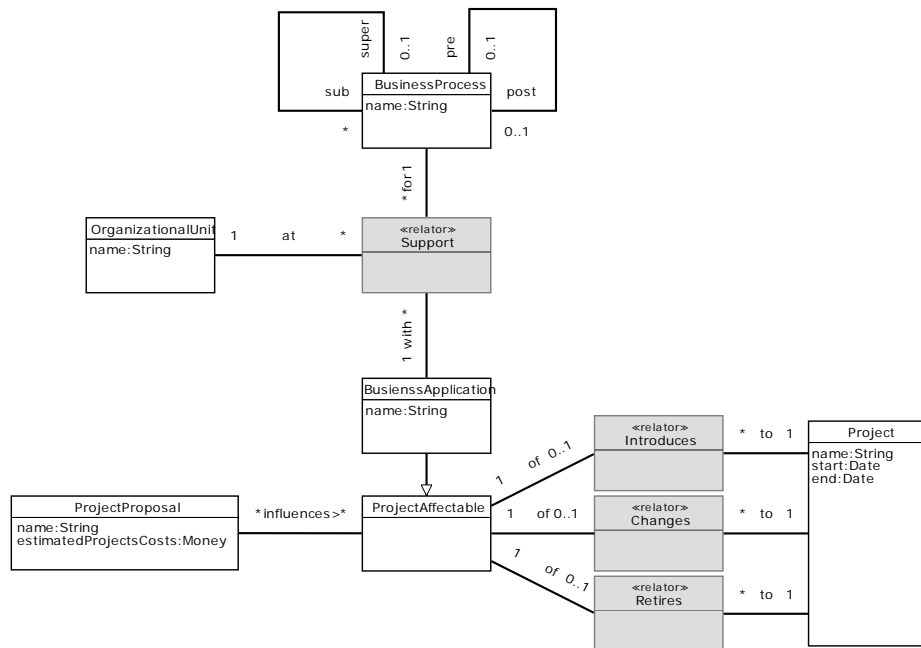
**Example 5.14: Development method—identify problem.** At BS&M the enterprise architects identify the project portfolio managers as potential stakeholder of the EA management initiative. During interviews these stakeholders expressed problems with determining the impact of planned projects onto the application landscape. In particular, the impact on the business support provided by the applications is of major interest as well as interdependencies between different projects.

Browsing the catalog of goals, the enterprise architects accordingly select the goal “increasing transparency”. Furthermore, the catalog of concerns is browsed to identify relevant elements of the EA on which the goal should be applied. The concern “business application supports business process at organizational unit” is selected, thereby introducing the corresponding concepts and relationships to the information model. Further the cross-cutting aspects “project changes architecture elements” and “project proposals affect architecture elements” are selected and applied onto the concept business application. Reflecting the problem statement of the stakeholders, the enterprise architects specify an information model based on the integration of the corresponding IBBs<sup>4</sup> as provided by BEAMS. The information model is stored in the organization-specific configuration.

---

<sup>3</sup>Due to the focus of this thesis on the method part of an EA management function, we abstain from further discussions on the identification and integration of information models representing goals and concerns. Further information on the language counterpart can be found in our publications concerning EA modeling languages [BMS10b, BMS10i].

<sup>4</sup>The concept of IBBs is introduced in Section 4.3.



An exemplary information model for the above described problem

To operationalize the goal, the enterprise architects decide to use the qualitative measure “stakeholder satisfaction”, which is proposed as an operationalization for the goal “increasing transparency” by the BEAMS catalog of goals.

Complementing the characterization of the situation, already existing information sources that contribute to the EA management function by providing required input, need to be specified by the enterprise architects in the step **specify existing information sources**. In this phase the **characterization technique** is applied.

**Example 5.15: Development method—specify information sources.** Revisiting the concepts from the information model, the enterprise architects of BS&M identify the ITIL CMDB (see example 5.12) and the project charter from the project portfolio management as information sources for their EA management initiative. Applying the techniques results in an update of the organization-specific configuration: the meta-attribute “documented” holds for the business applications, organizational units, and project proposals. Nevertheless, not all information demands described by the above information model are yet covered.

### 5.3. Configure MBBs

The second phase of the development method **configure MBBs** takes the organization-specific configuration as input and iteratively selects and configures MBBs to address the defined problem. Thereto, the MBBs as contained in the method base have to be configured to the organization-specific needs. Preparing the customization of the MBBs, we detail the variable concept in Section 5.3.1. We define different types of variables and outline how they are bound during the configure EA management function phase to ensure consistency of the configured method description. We supply consistency rules and techniques in Section 5.3.2. Finally, we supply the steps to be performed in the **configure EA management function** phase of the development method that build on the introduced variable concept and use the defined techniques.

#### 5.3.1. The variable concept

Four different types of variables are introduced in BEAMS: **trigger variables**, **participant variable**, **viewpoint variables**, and **information model variables**. An MBB in BEAMS represents an underspecified re-usable building block that must be complemented via configuration during the development method. We discuss how the variables are bound in applying the development method.

##### 5.3.1.1. Information model variable

The information model variable is a placeholder for the information model defining the syntax of an organization-specific EA modeling language. An information model can thereby refer to a goal, a concern, or an arbitrary combination thereof. In line with our discussions in [Bu07e], we define an information model as follows.

**Definition: Information model**

An information model specifies the syntax of an EA modeling language, i.e. it defines which concepts and relationships between concepts should be considered in describing the EA.

We use an I-pattern from the EA management pattern catalog to provide an example of an information model representing a concern.



**Example 5.16: Information model.**



I-pattern I-24 from [Bu08b, page 196]

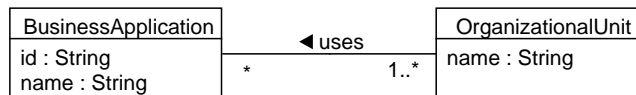
The ORGANIZATIONAL UNIT represents a subdivision of the organization according to its internal structure. A BUSINESS APPLICATION is a software system providing support for a business process. Represented by the relationship HOSTS a business application is hosted by a specific organizational unit that is responsible for its operation and maintenance.



An IBB as provided by BEAMS represents a best practice information model described using a defined notation. Different IBBs can be integrated with each other into a comprehensive information model. The IBBs, can relate to each other in different ways. Two IBBs can be linked via the relationships *overlaps*, *conflicts*, or *subsumes*. *Overlapping* denotes two IBBs that share at least one concept. Two IBBs *conflict* with each other, if they overlap and contain different understandings. One IBB ( $i_1$ ) *subsumes* another one ( $i_2$ ), if the subsumed IBB ( $i_2$ ) is completely overlapped by the subsuming one ( $i_1$ ).

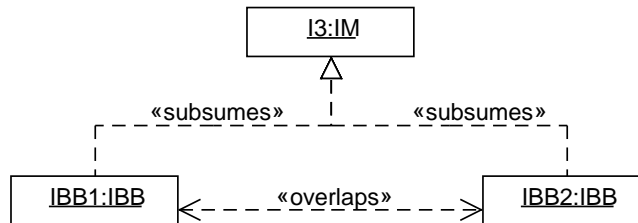


**Example 5.17: Subsumes relationship between information models.** The following IBB2 represents similar information as the I-pattern from Example 5.16 (IBB1). IBB1 represents a ‘hosting’ relationship, while IBB2 represents a ‘using’ relationship.



IBB2

IBB1 and IBB2 overlap as they share the concepts of BUSINESS APPLICATION and ORGANIZATIONAL UNIT. Under the assumption that the two IBBs are not conflicting, i.e. share a common semantics of the overlapping concepts, an *information model* (IM), can be created that subsumes IBB1 and IBB2.



The exemplary information models and their relationships



It must be noted that the **subsumes** relationship reads ‘inverse’ to classical inheritance-relationships of object-oriented modeling. Put it more simply, the specialized IBB subsumes

the more general. The subsumes relationship between information models is grounded in the *embedding* relationship, we established in [BKS10, page 86].

The information model variable in an MBB is bound during configuration to the actual information model on which the corresponding participants operate in the associated tasks. The information model thereby results from the integration of one or more IBBs (cf. Example 5.14). The excerpt from our conceptual model shown in Figure 5.14 delineates the relationships between TASKS, INFORMATION MODEL VARIABLE, and INFORMATION MODEL as well as additionally introduces the SUBSUMES relationships.

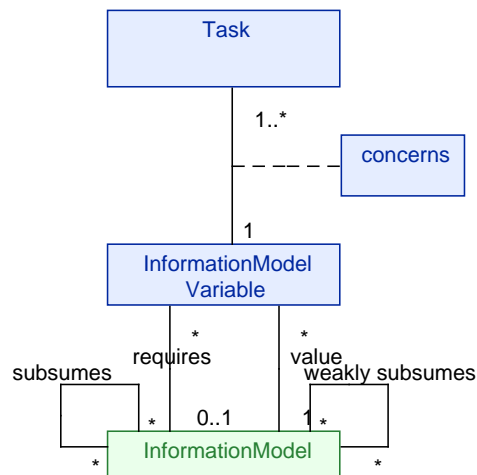


Figure 5.14.: Information model variable

### 5.3.1.2. Viewpoint variable

Each participant is supplied with a viewpoint for performing operations during task execution, i.e. to read or update EA information. The actual viewpoint to be used is specified during the development method and replaces the viewpoint variable. Depending on the quality of the operation two types of viewpoints are distinguished as introduced before:

- **representation function** that is limited to read-only access, i.e. the participants associated with a viewpoint of that type serve as information consumers or
- **notation function** that allows read and write access (creating, updating, deleting, and reading), i.e. the participants associated with a viewpoint of that type serve as information suppliers.

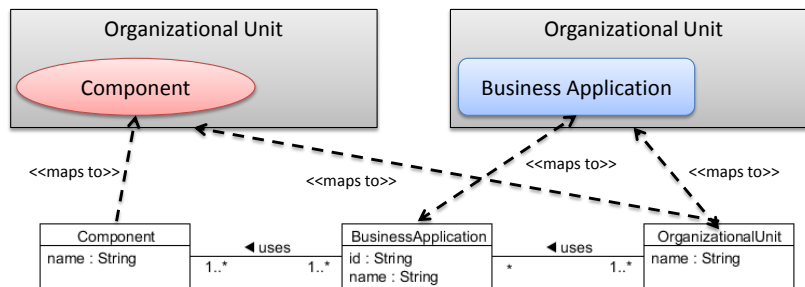
The viewpoint thereby supplies a graphical notation for the information model specifying the syntax and semantics. The notation function establishes a bijective mapping, i.e. a one-to-one relationship, between the elements defined by the information model and the graphical concepts specified by the *visualization model* [Bu07c, pages 5–6]. The representation function supplies a surjective function mapping information to graphical concepts. Thereby, aggregation as for instance calculating sums, or traversing relationships can be performed.





**Example 5.18: Representation vs. notation function.** An information model consisting of the concepts ORGANIZATIONAL UNIT, BUSINESS APPLICATION, and COMPONENT represents the basis for two different viewpoints.

One viewpoint (left side) groups the used components by the using organizational units thus traversing the associated business applications. While this viewpoint cannot be updated due to the missing information regarding business applications its use is limited to a representation function. The second viewpoint (right side) illustrates the business applications and the using organizational unit. Thus, it establishes a one-to-one relationship between the elements of the information model and the elements of the visualization model, which enables the use as notation function.



Based on the distinction between representation and notation function, we distinguish the following types of subsuming relationships between two information models:

- Subsuming in the context of a representation function: In a representation function, the concepts are accessed read-only. We define this type of subsuming as **read subsuming**. For two IBBs  $i_1$  and  $i_2$ , we denote the fact that  $i_1$  is read subsumed by  $i_2$  as  $i_1 \sqsubseteq i_2$ .
- Subsuming in the context of a notation function: In a notation function, the concepts are accessed via read and write. We define this type of subsuming as **update subsuming**. For two IBBs  $i_1$  and  $i_2$ , we denote the fact that  $i_1$  is update subsumed by  $i_2$  as  $i_1 < i_2$ . Thereby, update subsuming entails read subsuming.

In the following, we describe how the subsumes relationships apply to the different information models provided and consumed by the tasks in a sequence. In Figure 5.15 the operations (read or read & write) that are allowed during a task are defined by the type of viewpoint used by the associated participants.

The first task takes  $IM_0$  as input. During task execution a participant uses a representation function to access information. As the representation function does not supply information, the output of the task  $IM_1$  is identical to  $IM_0$ .

$$IM_1 = IM_0$$

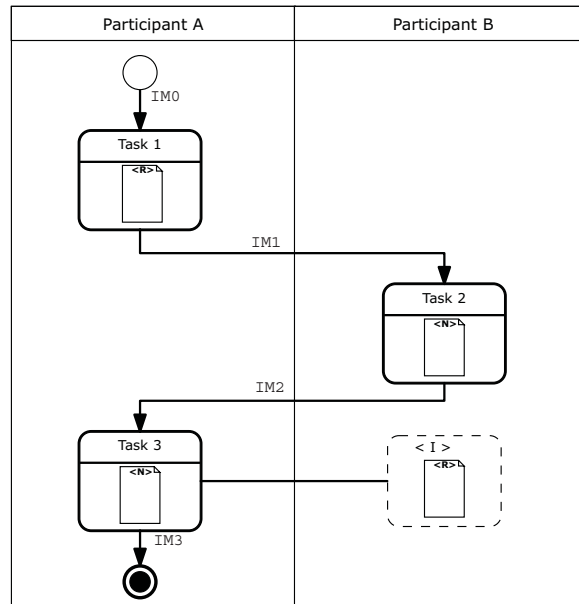


Figure 5.15.: Input and output information models of tasks

During task 2 a notation function is used to provide information such that the information models used as input and output of the further task sequence can be related as follows:

$$IM_2 <: IM_1$$

$$IM_3 <: IM_2$$

In line with the above discussion, our conceptual model of BEAMS can be extended with a relationship connecting the INFORMATION MODEL with the VIEWPOINT.

### 5.3.1.3. Participant variable

The **participant variable** is replaced during the configuration by an organization-specific role. BEAMS provides a **catalog of participants** that includes descriptions of single participants and delineate typical questions. If the participant description refers to a group, i.e. a committee or a board, recommendations for the staffing of the group can be given. The BEAMS conceptual model is accordingly updated with the concept PARTICIPANT that values the PARTICIPANT VARIABLE during configuration.

### 5.3.1.4. Trigger variable

The **trigger variable** is valued during the configuration to either a specific schedule, if a temporal trigger is considered, or is bound to an information model defining the concepts which initiate the execution. If a trigger type event is defined, the method base can supply restrictions of the admissible triggers for certain MBBs. This constraints are reflected in the augmented conceptual model of BEAMS in the PERMITS relationship. A TRIGGER VARIABLE

can specify that certain elements (cross-cutting aspects) are part of the trigger specification. Figure 5.16 provides the corresponding excerpt from the conceptual model.

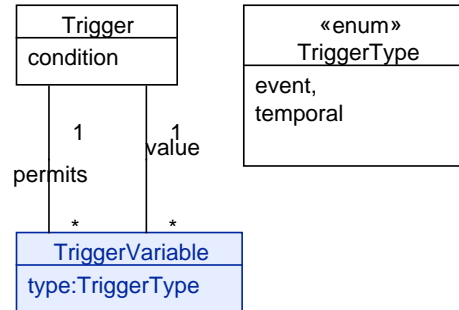


Figure 5.16.: Constituents of an MBB—trigger and trigger variables

Figure 5.17 presents the conceptual model of BEAMS. Therein, we employ the color-coding as introduced before: method-related parts are highlighted in blue, language-related parts are highlighted in green, and organization-specific concepts are not highlighted.

### 5.3.2. Rules and techniques for configuring MBBs

The integration approach of BEAMS enables re-use of existing EA management best practices but also provides challenges with respect to consistent integration of building blocks into a coherent method description for the EA management function. Two sources of inconsistencies exist that relate to the pre- and post-conditions and the triggers of an MBB. Subsequently, we define three inconsistencies that can arise during the integration of two MBBs and define respective rules to ensure a consistent method description. The rules are summarized textually and additional formalization using OCL constraints are given.

- Information deficiencies between tasks

Each task from a method prescription must be reachable, i.e. the pre-conditions of any task must be empty or a subset of the post-conditions of other tasks.

```

context Task
inv: Task.allInstances() -> forAll(t1,t2 |
    succ(t1) = t2 implies t2.preCondition SUBSET t1.postCondition)

```

- idle triggers

The information models specified by event triggers must be a subset of the post-conditions of the tasks contained in the current method description.

```

context TriggerVariable
inv: TriggerVariable.allInstances(type=event) -> forAll(t |
    self.Tasks -> exists(t' : Task | t.condition SUBSET t'.postCondition))

```

- misfit triggers:

If multiple temporal triggers are used that are associated to method descriptions of one problem, their schedules should be aligned. If the documentation of the current

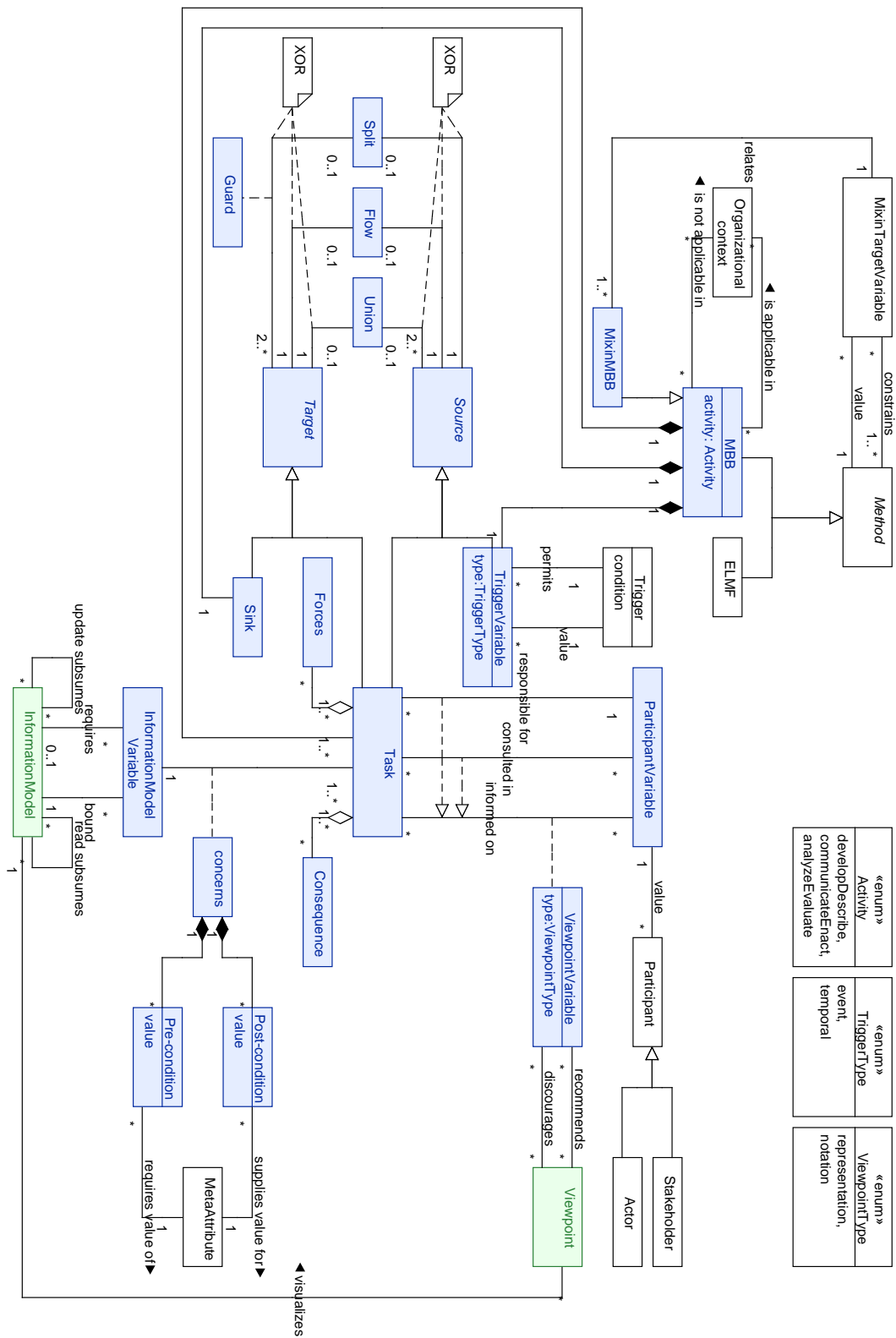


Figure 5.17: The conceptual model of BEAMS

state of the EA for instance is performed on a half-yearly basis, analyzing the standard conformity based on a quarterly schedule does not make sense.

Reflecting the two sources of inconsistencies triggers and conditions, we subsequently propose two techniques. The two techniques are based on dedicated rules to ensure information appropriateness for an integrated method description base on selected MBBs (see Section 5.3.2.1) and to support the selection of appropriate triggers (see Section 5.3.2.2).

### 5.3.2.1. Assessment technique

During the configure MBB phase of the development method a comprehensive method description is developed in a iterative manner. Thereto, MBBs are selected based on the organization-specific configuration and integrated into the set of previously selected and configured MBBs. The identification of admissible MBBs is thereby performed using **implicit relationships** between MBBs as introduced in Section 4.3. The MBB relationships **integrable**, **alternatives**, and **related** are subsequently detailed based on the above defined subsumes-relationship between the information models specified as part of the pre-conditions and post-conditions of tasks. Thereby, the MBBs are considered from a black-box perspective, i.e. the pre-condition of the first task in the sequence is used as pre-condition for the MBB and the post-conditions of all tasks contained in the sequence are respectively summed up to the post-condition. In this vein, the pre-condition of the example in Figure 5.15 would be that  $IM_0$  is documented and the post-condition would read as  $IM_1$ ,  $IM_2$ , and  $IM_3$  are documented. According to the subsumes-relationship introduced above,  $IM_3$  update subsumes  $IM_1$  such that the post-condition of the example MBB can be summarized as  $IM_3$  is documented.

From an inner perspective on MBBs, i.e. on task sequences, we can assume that the information model of the post-condition of the last task in the sequence always update subsumes the information model defined in the pre-condition of the first task. Put it more simply, we assume that the information is passed on from one task of the sequence to the next even, if the respective information is not used during the execution of the corresponding task. This assumption reflects the idea of a central repository as typically used in EA management endeavors. On the level of relating MBBs this assumption nevertheless does not hold as different MBBs are typically only related via exchanged information without a defined ordering.

The idea of a central repository is reflected during the development method by the **organization-specific configuration**. As part of the configuration also the set of fulfilled conditions, i.e. the post-conditions of the already selected, configured, and customized MBBs is maintained. A condition in the context of BEAMS relates to the concept of *assertion* in terms of Pries-Heje and Baskerville [PHB08]. It enables assessment of the suitability of an MBB by specifying requirements with respect to

- dedicated concepts to be contained in the associated information model,
- certain meta-attributes of the associated information model, and
- the temporal relation of the associated information model, i.e. the state of the EA which is covered,

or a combination of the above. Reflecting the problem-independent nature of MBBs, we restrict the specification of requirements regarding certain concepts to be covered by the information model to elements with cross-cutting aspects like projects, principles, or standards.

The meta-attributes are defined in the **catalog of meta-attributes** provided by BEAMS and are derived from the practice-proven method descriptions. The following meta-attributes are currently defined in BEAMS: (1) documented, (2) consistent, (3) approved, and (4) communicated. While the first meta-attribute represents a prerequisite for the other three, no further logic ordering of the other meta-attributes exists. An MBB also specifies the state of the EA to which it relates. Gathering information on EA elements automatically by crawler for instance has only proven to be applicable for the current state of the EA. Gathering information via interviews in contrast can be performed for the current as well as for the target state. The MBB as contained in the method base, i.e. in their ‘unconfigured state’ may specify more than one state in which they are applicable. After configuration during the development method, the method fragment<sup>5</sup> is linked to the actual information model that determines the state to which applies. In line with the above discussion we define a **condition** as follows.

**Definition: Condition**

A condition is a premise upon which the applicability of an MBB depends. A pre-condition specifies the essentials which must hold for an MBB to be applicable. The post-conditions delineate environmental requirements that hold after the MBB has been executed. A condition consists of an information model part, an optional temporal part, and a meta-attribute part.

The pre- and post-conditions represent *assertions* as introduced as part of a design theory nexus instantiation in Section 2.3. Reflecting the triple as described above, each condition is described in a threepart style that can be specified using the *Backus-Naur Form* (cf. Knuth [Kn64]) as follows

```
<goal>                ::= <principle | standard | strategy | project | ...>
<infoModel>           ::= <concern | goal>
<temporal-relation> ::= <current | planned | target>
<meta-attribute>     ::= <documented | consistent | publicized | ...>
<condition>          ::= <infoModel>[.<temporal-relation>].<meta-attribute>
```

The concept of pre-conditions and post-conditions is also mirrored in the activity framework of an EA management function such that for each transition between the activities a set of minimal pre-conditions and post-conditions can be specified that must hold. Apart from the mixinMBBs, the MBBs of the develop & describe activity typically do not specify pre-conditions. At least one of their tasks uses a notation function to gather information according to the associated information model such that after the execution of the MBB the post-condition CONCERN.DOCUMENTED holds. The minimum post-condition of CONCERN.DOCUMENTED, might be extended by using a mixinMBB to ensure that the documented information model is additionally checked for consistency or that the documentation has been approved. The MBBs associated with the communicate & enact activity at minimum require a certain concern to be documented as pre-condition. For each MBB related to this activity at least one task exists that uses a representation function to communicate the information of the associated information model. Complementing, the analyze & evaluate activity requires the concern

---

<sup>5</sup>We use the term **method fragment** to distinguish between the unconfigured MBB and the configured one.

which serves as basis for the analysis to be documented. After the analysis is performed the operationalized goal should be documented, which represents the minimal post-condition of each MBB associated to the activity of analyze & evaluate. The classification rules defining minimal pre- and post-conditions for the different EA management activities are summarized in Table 5.3.

	pre-condition	post-condition
<b>develop &amp; describe</b>		concern.documented
<b>communicate &amp; enact</b>	concern.documented	concern.documented, concern.communicated
<b>analyze &amp; evaluate</b>	concern.documented	concern.documented, goal.documented

Table 5.3.: Relating activities and meta-attributes

Each EA management-related problem is typically solved by passing through the above described activities. Thus, the concern is documented in a first step (develop & describe), then communicated (communicate & enact), analyzed in terms of a goal which is documented (analyze & enact) and finally the analysis results might be communicated again (communicate & enact). In this vein, a typical solution cycle for a problem reads as follows

1. concern.documented,
2. concern.communicated,
3. goal.documented,
4. goal.communicated.

The aforementioned cycle might be detailed by incorporating further meta-attributes, as consistent and approved.

With the above understanding of pre-conditions and post-conditions, we revisit the rules defining the implicit relationships between MBBs to prepare our **assessment technique**. Taking into account the trifecta of a condition, we define that two MBBs, ( $MBB_1$  and  $MBB_2$ ) are

**integrable** if the pre-condition of  $MBB_2$  represents a subset of the post-condition of  $MBB_1$ , i.e.

- if the information model of the post-condition of  $M_1$  update subsumes the information model of the pre-condition of  $M_2$  ( $IBB_{M1} :> IBB_{M2}$ ) AND
- if the meta-attributes specified by the pre-condition of  $MBB_2$  are fulfilled by the meta-attributes specified by the post-condition of  $MBB_1$ , AND
- if the temporal states specified by the pre-condition of  $M_2$  and the temporal states specified by the post-conditions  $M_1$  share at least one state to which they can be applied.

**alternatives** if the pre-conditions and the post-conditions of both MBBs are the same, i.e.

- if the information model of the pre-condition of  $MBB_1$  and the information model of the pre-condition of  $MBB_2$  are equal AND if the information model of the post-

condition of  $MBB_1$  and the information model of the post-condition of  $MBB_2$  are equal, AND

- if the meta-attributes specified by the pre-condition of  $MBB_1$  are equal to the meta-attributes specified by the pre-condition of  $MBB_2$  and if the meta-attributes specified by the post-condition of  $MBB_1$  are equal to the meta-attributes specified by the post-condition of  $MBB_2$ , AND
- if both MBBs share at least one temporal state.

**related** if the pre-condition of  $MBB_1$  and  $MBB_2$  are equal but the post-conditions differ, i.e.

- if the information model of the pre-condition of  $MBB_1$  and the information model of the pre-condition of  $MBB_2$  are equal, AND
- if the meta-attributes specified by the pre-condition of  $MBB_1$  are equal to the meta-attributes specified by the pre-condition of  $MBB_2$ , AND
- both MBBs have at least one shared state to which they can be applied.

The **assessment technique** uses the above defined relationships to support the user of the development method in configuring or updating a coherent and consistent management function. Input for the technique is the organization-specific configuration, i.e. the

- EA management-problem(s) currently under investigation,
- the common set of post-conditions that hold with respect to the information model reflecting the problem(s) based on the already selected and configured MBBs, and
- the selected organizational context descriptions.

The organization-specific configuration represent the assertions against which the MBBs of the method base are assessed for suitability. In this vein, the **assessment technique** supports the user during the development method with identifying applicable MBBs as well as ineligible MBBs, or evolution paths. Thereby, the assessment technique makes use of the above defined implicit relationships between MBBs. The single steps performed for the above activities are subsequently detailed:

### Identify applicable MBBs

1. Identify applicable EA management activities based on the problem (information model) under investigation and the therefore already fulfilled meta-attributes.
2. Search method base for MBBs that are integrable with respect to the set of fulfilled conditions and the temporal relation.
3. Evaluated admissible MBBs according to the specified organizational context, i.e. for each selected organizational context in which an MBB is applicable the MBB is higher prioritized, MBBs that delineate not to be applicable in one of the selected organizational contexts are removed from the set.

### Identify ineligible MBBs

1. Investigate the situation defined in the organization-specific configuration with respect to changed organizational contexts and EA management-related problems which are no longer of interest. If no obsolete problem exists continue with step 4.



2. For each obsolete problem the associated method fragments from the method description are highlighted as “to be deleted”.
3. Identify method fragments from the remaining EA management function that are not longer admissible due to the changed set of fulfilled post-conditions and highlight impacted method fragments.
4. Identify and highlight method fragments that specify to not be applicable in the changed contexts.
5. Identify alternative MBBs that replace the highlighted method fragments.
6. Remove method fragments that are marked as “to be deleted”.

### Identify evolution paths

1. Search method base for related MBBs.
2. Evaluate MBBs according to the specified organizational context (see step 3 of identify applicable MBBs).



**Example 5.19: Assessment technique.** At BS&M enterprise architects design an EA management function using the BEAMS method base. The EA management-related problem to be addressed is labeled as “increasing transparency on the interplay of planned projects”. Therefore an area-of-interest, i.e. information model is defined which consists of the concepts BUSINESS PROCESS, ORGANIZATIONAL UNIT, BUSINESS APPLICATION, and PROJECT PROPOSAL.

The current organization-specific configuration reads as follows<sup>6</sup>:

- organizationalUnit.current.documented
- businessApplication.current.documented
- projectProposal.current.documented

By applying the assessment technique, we identify that only MBBs from the develop & describe activity are applicable as the current concern is not completely documented. After an MBB has been selected and configured to document business processes, the condition “businessProcess.current.documented” is added to the organization-specific configuration. Therefore, the current concern is completely documented. In the next selection step of the development method, the assessment technique is again applied and then returns mixinMBBs from the activity develop & describe activity (resulting in CONCERN.CURRENT.CONSISTENT or CONCERN.CURRENT.APPROVED) and MBBs from the communicate & enact, and analyze & evaluate activity whose pre-conditions are fulfilled.




---

<sup>6</sup>For reasons of brevity, we restrict the subsequent discussion to the class-level omitting discussions on relationships and attributes.

Using the assessment technique the user of the development method is supported in the selection of MBBs. The assessment technique can further be applied during evolution and adaptation of the EA management function (see Section 5.4) to identify method fragments that are not longer suitable. Thereby, the assessment technique can be used to identify possible adaptation and evolution paths.

### 5.3.2.2. Liveliness technique

With the above described technique, we can ensure that users of the development method design only consistent and coherent EA management function. Nevertheless, we cannot ensure ‘liveliness’ of the configured method as the single method fragments are not connected via control flow such that single method fragments might never be initiated. Initiation of an MBB is performed via specifying a trigger. Triggers can either be of the type temporal or of the type event. Temporal triggers ensure liveliness of the associated method descriptions. Relating different MBBs that are initiated by temporal triggers might nevertheless lead to discrepancies and friction.



**Example 5.20: Discrepancies in trigger configuration.** A banking company has established an EA management function that addresses its need for transparency regarding the application landscape and the therein used technology. To maintain the documentation and ensure actuality of the information, the enterprise architects send questionnaires to the application owners at the beginning of every fiscal year. The application owners, if necessary, update the contained information. The current state of the application landscape is reported to the upper management at every end of the fiscal year.

Due to a change in the upper management, the enterprise architects are demanded to provide an overview on the current landscape on a half-yearly basis.



The change of the communication process, e.g. the reporting, in the above example does not lead to an inconsistent method description. It however results in discrepancies regarding the interplay of the two processes of documentation and communication as the upper management would be provided with the same description twice, if the maintenance process is not updated accordingly.

Besides discrepancies, ‘idle’ processes can be defined. Idle processes refer to method fragments which are never initiated, as they define a triggering event that never occurs. Idle processes can only emerge, if event triggers are used.

**Example 5.21: Idle method configuration.** To increase homogeneity, an small-sized enterprise which has no defined project portfolio process decides to assess each project proposal for conformance with architectural standards. Therefore, a process is set up that is triggered, if a new project proposal is documented. The process defines that each project manager of a non-conformant project is consulted by an enterprise architect who discusses possible changes ensuring standard conformance.

Whereas the above example represents a way of enacting standards that is frequently used, a prerequisite of the enactment that ensures liveliness is the definition of a process documenting the project proposals. In cases like the one described above, this is typically performed by interlinking the step to the project portfolio management process.

Configured triggers of the type event can either refer to elements from the layers or ones representing cross-cutting aspects. If cross-cutting aspects are referenced in the trigger definition, the associated MBB typically represents a `mixinMBB`, i.e. has to be integrated into another ELMF, like project portfolio management, if projects or standards are concerned, or the strategies and goals management, if visions and goals, are considered. In cases where the trigger is configured to relate to other concepts, these concepts needs to be documented, changed, or updated by an already selected and configured MBB. In this vein, the information model used in the specification of the trigger must be update subsumed by the information model incorporated in the organization-specific configuration and the meta-attribute “documented” must apply for the information model fragment.

In line with the above discussions, we define a **liveliness technique** supporting the development method by guiding the selection of appropriate triggers. The liveliness technique can be employed during the development method to detect

**discrepancies in temporal trigger specifications** Starting from the MBB currently under investigation, the set of already configured MBBs incorporated in the organization-specific configuration is assessed to identifying predecessor MBBs (by reverse traversing of the integrable relationship) and thus builds a ‘dependency graph’ until no unfulfilled precondition is left. The dependency graph is subsequently searched for MBBs specifying temporal triggers which are compared to detect discrepancies. Thereby, a discrepancies is detected, if the frequency of an earlier MBB (predecessor) is less than the frequency of an MBB latter in the graph (successor).

**idle method configurations** Whenever the trigger of an MBB is configured during the development method the set of fulfilled conditions of the configuration is searched for concepts for which the meta-attribute ‘documented’ applies. The admissible specification of a trigger is then limited to these concepts.

The development method uses the **liveliness technique** to support the enterprise architect during the configuration of MBBs. Thereby we ensure that the resulting method description is a consistent and coherent one.

### 5.3.3. Development method

The phase **configure EA management function** of the development method represents an iterative activity consisting of two sub-activities. During the sub-activity **select MBB** the set of MBBs applicable in the current situation is determined using the assessment technique (cf. Section 5.3.2.1). Based on the output of the assessment technique the enterprise architect selects an MBB. The selected MBB is subsequently configured to the organization-specificities in the sub-activity **configure MBBs**. Therein, the enterprise architect is supported by applying the liveliness technique (cf. Section 5.3.2.2). The two sub-activities are iteratively performed until all EA management activities of the framework (cf. Figure 4.7) are covered. The output of the configured EA management function activity is a coherent and self-contained EA management function that addresses the so far defined set of problems and is stored in the organization-specific configuration. Figure 5.18 provides an overview on the activities of configure EA management function.

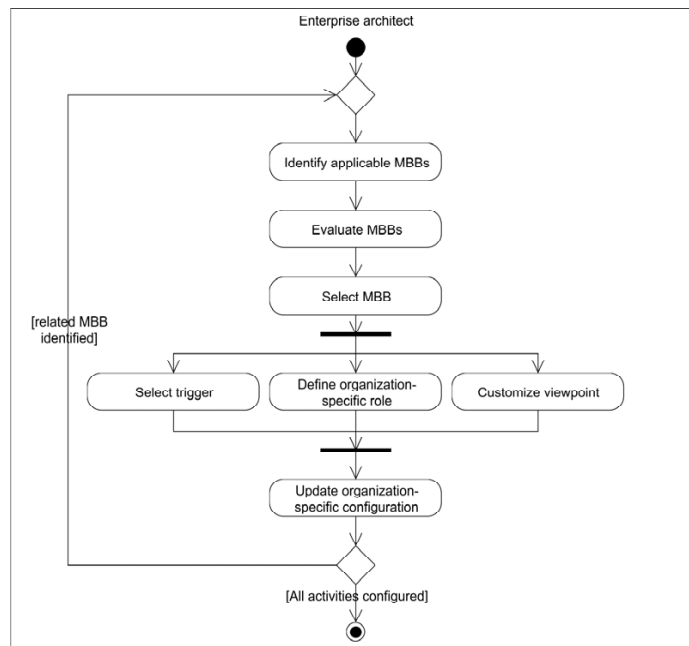


Figure 5.18.: Development method: configure EA management function

Entering the construction of the EA management function itself, the step **select MBB** is executed by the enterprise architects. The enterprise architects identify applicable MBBs by applying the assessment technique (cf. Section 5.3.2.1) and choose an admissible MBB from the set of appropriate MBBs. The choice is supported by taking into account the participants that must be involved in executing the tasks as well as the consequences of applying an MBB.



**Example 5.22: Development method—select MBB.** At BS&M, the input for assessment technique is the information stored in the configuration, namely

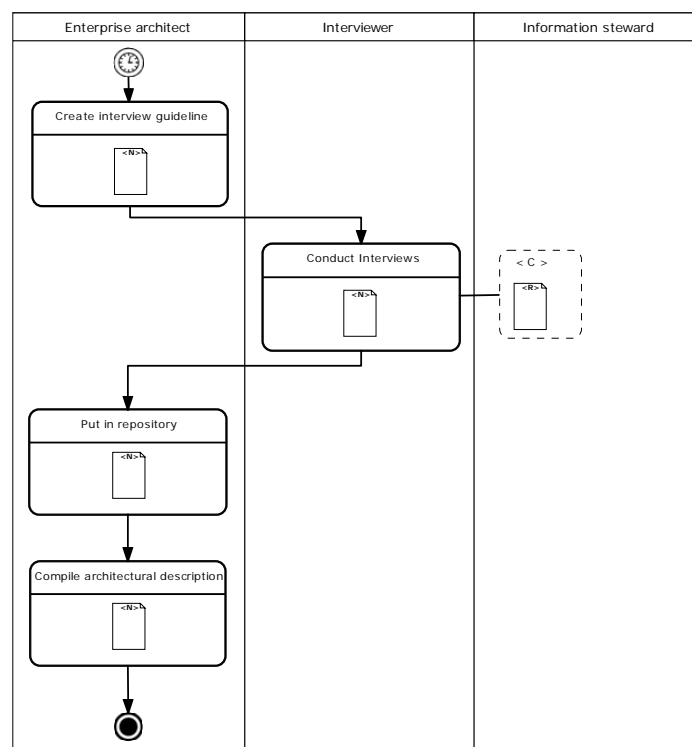
goal: increasing transparency, current state of the EA

context: bottom-up initiative, centralized IT department, office tools

conditions: none

Based on above criteria the enterprise architects identify applicable MBBs from the method base using the assessment technique. With respect to the current goal of the associated problem, the set of admissible MBBs can be limited to the ones associated with the activities “develop & describe” and “communicate & enact”. Taking further the empty set of fulfilled conditions into account, MBBs from the activity “communicate & enact” can be excluded, such that the following MBBs from the “develop & describe” activity are evaluated by the assessment technique. The subsequent fitting matrix illustrates the evaluation results.

	bottom-up initiative	centralized IT department	office tools
Describe by interview	●		●
Describe by questionnaire	●		
Describe by workshop			●



MBB describe by interview

The enterprise architects of BS&M decide to use the first MBB to gather the missing information on BUSINESS PROCESSES. Besides the evaluation results, the convincing argument therefore, was the possibility to indi-

vidually promote the EA management initiative at the different business departments in a face-to-face interview (consequence of the “describe by interview” MBB).



While above step already started shifting the process from an analytic to a constructive one, the step **configure MBBs** is clearly related to design and construction. Three parallel activities are performed during this step all relating to the customization of the selected MBB, namely

- the trigger of the MBB is detailed using the liveliness technique as proposed in Section 5.3.2.2 taking into account possible limitations that are already specified by the MBB (with respect to the type of trigger or contained concepts).
- the participant variables delineated by the MBB are replaced and detailed by an organization-specific role.
- for each involvement of a participant in a task the used viewpoints are defined. While the constraints provided by the type of viewpoint are accounted for, the recommendations and dissuasions can optionally be considered.

After the configuration, the customized method fragment is integrated into the set of already configured method fragments that represent the current status quo of the EA management function described by the organization-specific configuration.



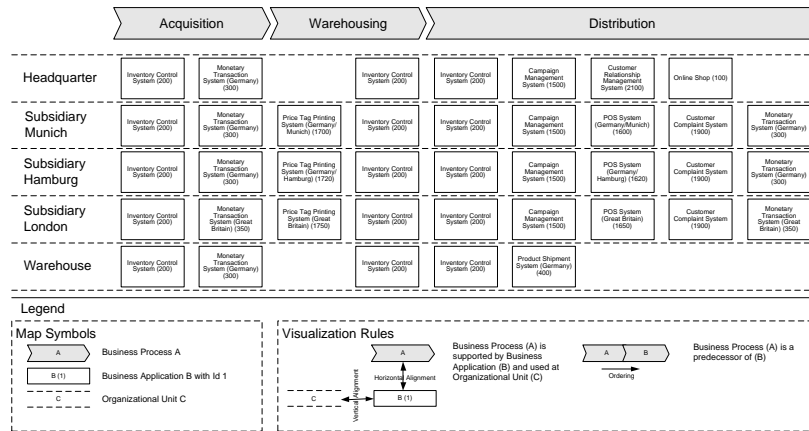
**Example 5.23: Development method—configure MBB.** At BS&M the enterprise architects configure the selected MBB as follows (for a graphical representation of the configured method fragment see Example 5.22):

The trigger is specified by the MBB to be of type “temporal”. Using the liveliness technique the enterprise architects of BS&M investigate that no ‘pre-decessor’ method exists such that no further constraints for the trigger selection exist. In line with the update schedule of the CMDb from the ITIL initiative, the enterprise architects decide to update the documentation of business processes on a yearly basis.

The participant variable INTERVIEWER is defined to be an enterprise architect to facilitate the promotion of the EA management initiative. Further, the process owners are identified as information stewards.

Complementing, the viewpoints used to involve the different participants are defined. Reflecting the absence of a dedicated tool support, typical office documents are used with one exception: the architectural description used in the last step is displayed in a so-called *process support map*, a matrix visualization that relates business processes, business applications, and organizational units.

## 5. BEAMS: Building Blocks for EA Management Solutions



Process support map

After the enterprise architects have finished customization of the selected MBB, it is integrated into the set of already defined method fragments of the EA management function and the organization-specific configuration is extended. Complementing the conditions on the information model are updated. If not all activities of the EA management function are yet covered, the development method continues with the identification of the next MBBs that are admissible by iteratively performing the steps select MBB and configure MBB. Otherwise, the enterprise architects can either start to characterize the next situation and problem to be addressed (configuration cycle) or continue the development method with the analysis of the EA management function.

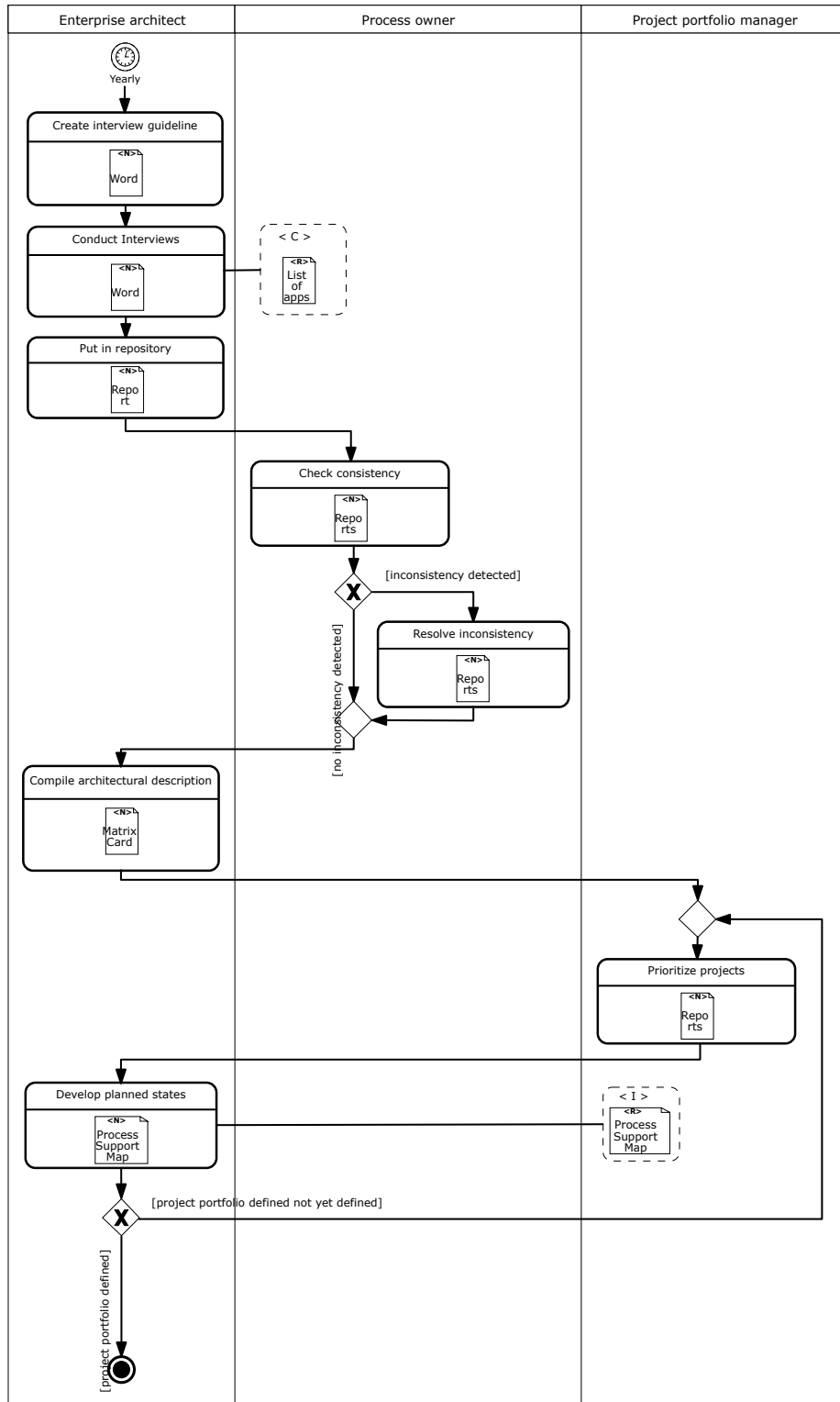
**Example 5.24: Development method—integrate MBB.** The enterprise architects from BS&M update their organization-specific configuration to on the one hand incorporate the configured method fragment and on the other hand extend the set of fulfilled conditions with the condition “concern.current.documented” as now methods have been delineated that document all concepts specified by the information model.

Based on the updated configuration a new set of admissible MBBs can be identified. The assessment technique now additionally returns MBBs from the communicate & enact activity as the minimum pre-condition concern.documented is fulfilled. Omitting the iterative steps, we subsequently present the resulting EA management function that addresses the problem of “increasing transparency on the interplay of planned projects”. The resulting EA management function was developed by customizing three MBBs (see Appendix A.1) from the BEAMS method base, namely

- describe by interview (develop & describe),
- ensure information consistency (develop & describe), and
- develop planned states of the EA (develop & describe).

## 5. BEAMS: Building Blocks for EA Management Solutions

The organization-specific EA management function is given subsequently.





## 5.4. Analyze EA management function

Since quick-wins and short-term benefits of EA management are rather sparse, the stringent implementation of an EA management function is not easy to ensure. A central challenge for enterprise architects is to ensure organizational implementability (cf. quality criteria **Q6** from Section 4.1.1). The third phase of the development method is concerned with analyzing the organizational implementability of an EA management function designed with BEAMS. Central thereto, is the distinction between stakeholders and actors we introduce in Section 5.4.1. Based on this distinction, we present two different techniques to analyze organizational implementability in Section 5.4.2. Finally, we discuss the application of the techniques during the analyze EA management function phase of the development method in Section 5.4.3.

### 5.4.1. Stakeholders and actors

Organizational implementability relates to the involved actors on the one hand, who are responsible for maintaining the architectural descriptions by providing accurate EA information, and the stakeholders typically representing the sponsors of the EA management initiative on the other hand, who need to be convinced of continuing their support. The fact that the so-called information providers, i.e. the actors, typically differ from the information consumers, i.e. the stakeholders, results in a lack of motivation and reluctant behavior on the part of the former since there is no obvious value of documenting, analyzing, and communicating the EA-related information. Thereby it is not unusual that the overall number of providers regarding a specific piece of EA information outweighs the one of actual end users or, even worse, there is an uncertainty about how many stakeholders are consuming a specific piece of EA information at all. In addition, the overall provisioning costs for such a piece are only transparent to the providers, who typically do not receive consumer's feedback.

Referring to the above distinction of participants in **stakeholders** and **actors**, we subsequently revise our preliminary definitions from 3.1.1 and Section 4.3. The revised definitions are thereby based on the different levels of involvement in the tasks of the EA management function as discussed above.

**Definition: Stakeholder**

A stakeholder is an individual or group of individuals who has an EA management-related problem. The involvement of a stakeholder in the execution of the EA management function is typically limited to being informed on the execution of tasks or the results achieved thereby.

**Definition: Actor**

An actor is an individual or group of individuals that participates in the execution of an EA management-related task. The participation thereby can either be of type responsible for or consulted in the execution.

The conceptual model presented in Figure 5.19 introduces the distinction of PARTICIPANTS in STAKEHOLDERS and ACTORS.

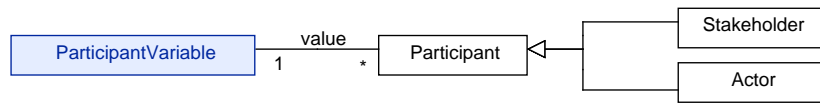


Figure 5.19.: Constituents of an MBB—Stakeholders and actors

### 5.4.2. Techniques for ensuring organizational implementability

Based on the above defined distinction between stakeholders and actors, we propose two analysis techniques investigating the organizational implementability. The single steps to be performed in applying the techniques are subsequently presented:

#### stakeholder involvement technique

1. For each problem from the organization-specific configuration the associated stakeholder and the respective method fragments of the EA management function are identified.
2. With the associated stakeholder at hand, the method fragments are searched for an “informed” involvement of the stakeholder.
3. If no informed involvement is found an exception is returned.

#### stakeholder-actor-dependency technique

1. The first problem from the organization-specific configuration is taken.
2. The associated stakeholder and method fragments are identified.
3. All actors involved in a task of the associated method fragments are included in a dependency graph such that an arrow indicates the dependence of the problem’s stakeholder from the respective actor.
4. Repeat steps 1–3 until all problems have been processed.
5. Return dependency graph.

### 5.4.3. Development method

The activity **analyze EA management function** investigates the organizational implementability in three steps addressing the aforementioned challenges. The three steps are illustrated in Figure 5.20. In the first step, the involvement of stakeholders in the EA management function is analyzed using the above presented techniques to ensure that the EA-related information representing or contributing to the solution of the stakeholder’s problem is communicated. Stakeholder-actor-dependencies are made transparent in the second step. By comparing the identified dependency structure and the organizational control structures different organizational intervention are proposed in the final step of the activity.

The utilization of the BEAMS development method ensures that roles and responsibilities are defined for each task, which are either already present in the associated organization or which have to be established in the course of the EA management initiative. In the step **analyze stakeholder involvement**, the enterprise architects ensure that the stakeholder associated

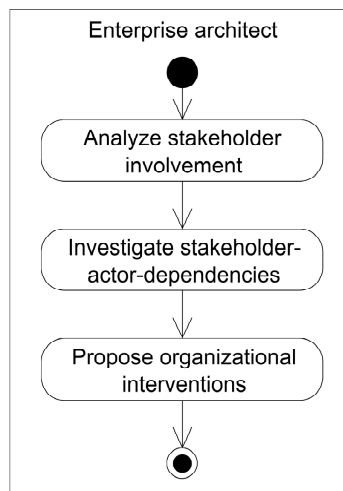


Figure 5.20.: Development method: analyze EA management function

with the current problem is at some point in the method informed on the achieved results. In this vein, the EA-related information representing the benefit for the stakeholder is made transparent. By applying the technique the enterprise architects identify, if the stakeholders are informed on the achieved results. If none such involvement can be identified, the user of the development method returns to the step “configure EA management function” of the development method and selects an admissible MBB from the communicate & enact activity.



**Example 5.25: Development method—analyze EA management function.** The organization-specific configuration of BS&M contains only one problem to be addressed for which the “project portfolio managers” represent the associated stakeholders. Analyzing the configured method with respect to an ‘informed-involvement’ of the project portfolio managers, the task “develop planned states” is identified where the stakeholders are informed on the compiled architectural descriptions via a “process support map”.

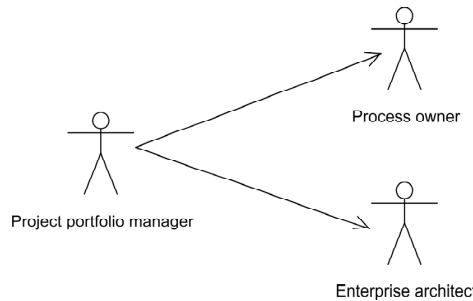


In the step **investigate stakeholder-actor-dependencies** the supply and demand of pieces of EA information is investigated and the dependencies between information providers and consumers are made explicit. Therefore, the technique is applied. The resulting dependency graph represents the input for the subsequent step.



**Example 5.26: Development method—dependency graph.** The dependency graph of the EA management function of BS&M shows the re-

relationships between the project portfolio manager as the stakeholder (information consumer) and the actors (information providers).



Dependency graph of the EA management function of BS&M



The final step, **propose organizational interventions** takes the dependency graph from the preceding step as input and analyzes which motivations for the information suppliers (actors) exist to contribute to the EA management function. Juxtaposing the information dependencies with the organizational control structures in the organization, the enterprise architect can investigate which dependencies are organizationally supported. In particular, the following relations between organizational control and information dependencies can exist:

**Line-of-control:** The information consumer is a (direct) superordinate of the information provider, thus being able to directly demand the information from the corresponding provider. In this case we speak about an alignment of organizational control and information dependencies such that organizational implementability can be regarded as assured.

**Tits-for-tats:** The information consumer can exchange information needed to address one problem with information provided for another. Such case applies, when circular information dependencies exist, or if such dependencies are mediated over organizational control relationships. In the latter case, an information consumer's subordinates would be responsible needed for providing information A to the (prospective) provider of the needed information B.

**Social competition:** The information consumer can raise peer-level competition between the information providers. Such situation exists, if the consumer is in a well-respected but not empowered role, being able to create transparency about information provision by the equally leveled information providers.

Revisiting the EA management function against the above provided classification of organizational interventions, the enterprise architects can decide, if their configuration appropriately addresses the aspects of organizational implementability.



**Example 5.27: Development method—organizational interventions.** The enterprise architects of BS&M revisit the stakeholder-actor-

dependency graph against the background of the control structures established by the organization. Even though no direct line-of-control can thereby be identified between the project portfolio managers (stakeholders) and the process owners (actors) which represent peer-level employees, the enterprise architects believe the organizational implementability to be assured due to the situation of “tits-for-tats” between the process owners. Due to the fundamental interest of the process owners in having an overview on the planned changes with respect to their process on the one hand and the fact that the process owners are keen to have “their” projects to be highly prioritized, the information provision can be regarded ensured.



### 5.5. Adapt and evolve the EA management function

The remaining quality criteria **Q8** (facilitate performance measurement) and **Q9** (being sustainable and adaptable) from Section 4.1.1 relate to evolutionary aspects of the EA management function. Measuring the performance therein is regarded a prerequisite to identify potentials for improvements, i.e. adapting the EA management function. Performance in the context of EA management can either be evaluated independent of the problems addressed, e.g. in terms of general stakeholder satisfaction (subjective assessments), or can be assessed depending on the problems and the ability to establish appropriate methods and means to address the problems (objective assessments). Explicit problem statements are a prerequisite for objective performance measurement. The coverage of the activities of the EA management function represents a performance criterion, e.g. actively communicating and enacting standard conformity can be regarded to represent a ‘better’ way than ‘just’ passively documenting the used technologies. The abstract goal “increasing transparency” is an exception from the aforementioned rule, as its realization builds on MBBs contributing to the activities develop & describe and communicate & enact. This can be explained by the fact that increasing transparency typically is not a goal in itself but is a general means that facilitates the achievement of other EA management goals.

Different reasons why the method representing the EA management function should be re-configured or re-designed exists. If the performance of the EA management function is continuously measured the need for re-design can be identified. Two directions of misalignment can thereby be distinguished, namely

**under performing** An under performing EA management function is characterized by decreasing interest in the architectural descriptions, incomplete tasks that are stopped prior to being successfully conducted, and unhappiness of the stakeholders with the achieved results, e.g. with respect to the actuality of information or the level of detail. An under performing EA management function can result from an incorrect characterization or a changed situation which is not yet reflected in the design of the EA management function.

**well performing** A well performing EA management function is characterized by a high interest of stakeholders in the architectural descriptions compiled as results of the EA management function, an increasing number of requests for additional problems to be addressed by the endeavor, and a high satisfaction of the stakeholders. The enterprise architects of well performing EA management functions are typically confronted with requests to raise the maturity level.

Addressing the need for re-designing the EA management function the activity **adapt and evolve EA management function** provides guidance to re-design and re-configure the EA management function. Therefore, the organization-specific configuration is taken as input and the **assessment** and **liveliness technique** presented in Section 5.3.2 are used to identify ineligible MBBS and re-design an underperforming management function, or to identify evolution paths in cases of a well-performing EA management function.



**Example 5.28: Development method—adapt EA management function.** To measure the performance of the EA management initiative, the enterprise architects of BS&M perform interviews on a yearly basis in which they ask for the stakeholder satisfaction with the achieved results (cf. operationalization of the goal “increase transparency”).

In the second year, the enterprise architects identify a decreasing stakeholder satisfaction, which they trace back to outdated information in architectural descriptions. Aside they experience troubles in conducting the interviews with the process owner which are no longer willing to spend time for an interview. Using the above described method, the enterprise architects identify the method fragment related to describing business processes via interviews to be no longer applicable owing to the expansion of BS&M that now operates world-wide. Due to the expenses for traveling, an interview-based method to gather information is no longer applicable. Searching the method base for an alternative MBB, the enterprise architects decide to augment the EA management function by replacing the “describe by interview” MBB with the alternative “describe by questionnaire” NBB.



In cases of well performing EA management functions, the enterprise architects are typically enabled to enhance the maturity level of the initiative. The development method of BEAMS supports evolution of the EA management function in a twofold way: first, the existing EA management function can be easily extended to incorporate newfound problems by performing the activities **characterize situation** and **configure EA management function**. Second, the method base and the linked MBBS can be used to derive evolution paths which provide more sophisticated methods to address an EA management-related goal or to increase the level of detail used in the architectural descriptions (assessment technique). supporting the evolution decision, different types of analyses can be performed based on the organization-specific configuration answering the following questions

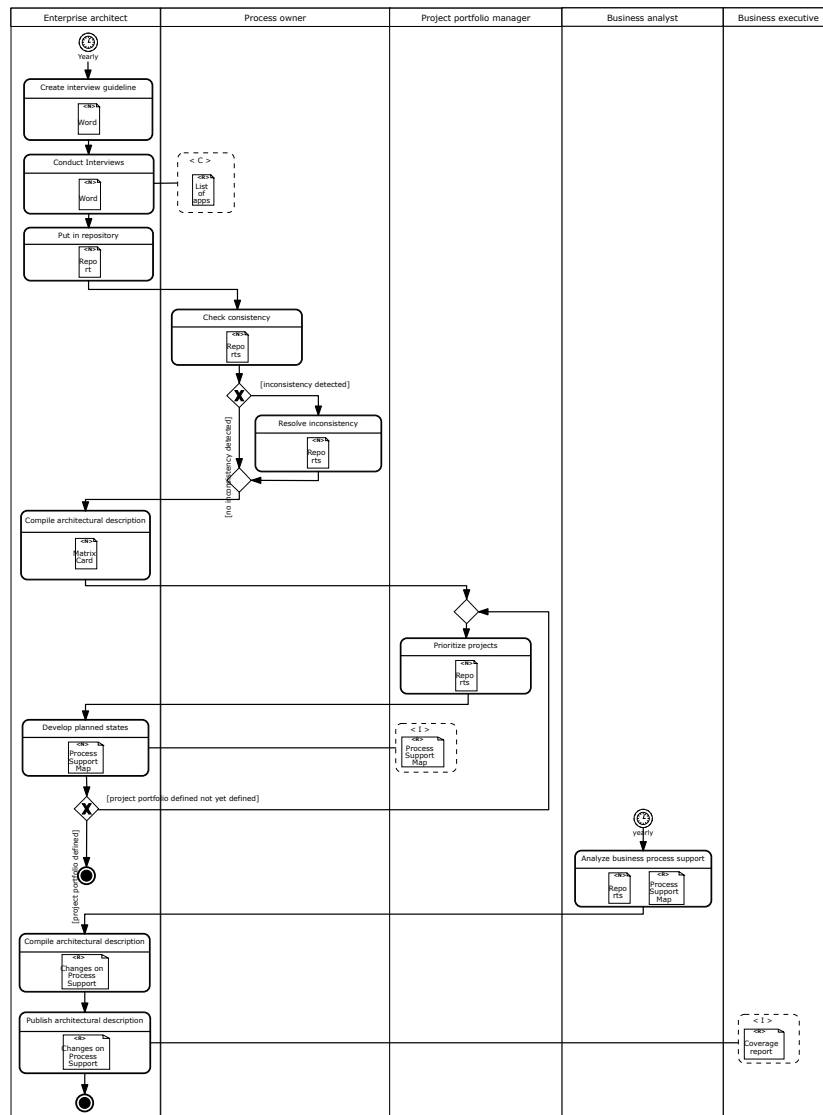
- Are all EA management activities covered for each problem of the initiative?
- In how many tasks is a participant involved (and in which way, i.e. responsible, consulted, informed)?
- Which participants are provided with which visualizations?
- Which information, e.g. on business processes, business applications, infrastructure elements, is used by whom, in which task, etc.?

Based on the answers to above questions, additional stakeholders and problems can be identified.



**Example 5.29: Development method—evolve EA management function.** At BS&M the EA management initiative was successfully implemented with a constant stakeholder satisfaction over the last years. The benefits of an up-to-date architectural description of the application landscape and the acceleration of the decisions in the project portfolio management have waken the business executives, which decided to enhance the set of problems to be addressed by the EA management endeavor. The enterprise architects of BS&M are requested to augment the EA management function to additionally address the goal “improve capability provision” in terms of analyzing the current application landscape with respect to the provided business support.

Using the development method and the assessment technique, the enterprise architects identify the MBB “single expert evaluation” from the BEAMS method base to be applicable in this context. The MBB is configured and integrated into the organization-specific EA management function. Analyzing the resulting EA management function the enterprise architects detect that the business executives have not been informed on achieved results. Therefore, an additional MBB from the communicate & enact activity (“publish architectural description”) is selected to ensure that the stakeholders are informed. The resulting EA management function is presented subsequently.



Re-designed EA management function of BS&M



## 5.6. Developing and maintaining the method base

Complementing the development method that specifies activities and steps to apply the method base, we subsequently discuss how the method base is developed and maintained. Figure 5.21 illustrates an UML activity diagram of the administration method to evolve the method base, which consists of six activities (including the identification of a new solution) that relate to the construction of a design theory nexus instantiation and the pattern-based theory building process as introduced in Section 2.3. The activity diagram is further complemented with the different parts of the method base that the output of the single activities



contributes to<sup>7</sup>. In the initial development phase of BEAMS, the administration method was iteratively executed to construct the first version of the method base. Starting with an empty method base, new solutions to EA-related problems were iteratively identified, restructured, and integrated. Subsequently, we delineate the steps performed in each activity and discuss their execution along an example from the initial set of practice-proven solutions.

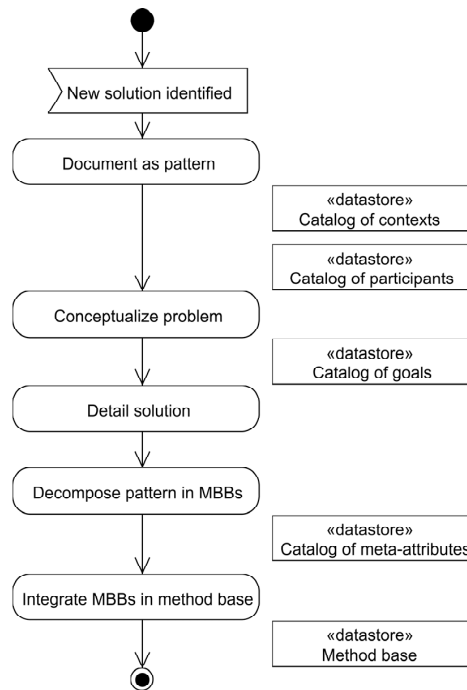


Figure 5.21.: Administration method—developing and maintaining the method base

New solutions for EA management-related problems can originate either from academic work or can be observed in practice. The set of solutions used to initialize the BEAMS method base represents a combination of both aforementioned origins but with a focus on proven practices. The initial input for the development of the method base is the collection of abstract best practices as contained in the EA management pattern catalog (cf. discussion in Section 4.2.4). The abstract method descriptions from the M-Patterns were complemented and enhanced with current best practices during a series of workshops conducted as part of an industry-funded research project with a focal points on methods for EA management. The project was called *enterprise architecture management method library* (EAMML)<sup>8</sup> and conducted between October 2009 and September 2010. The idea behind the EAMML project was to gather information on how organizations accomplish their EA management function and document the thereby identified best practices. In addition, the specific situations, i.e. the organizational contexts, that constrain the applicability of the best practices represented an intended outcome of the EAMML project. Participating in the research project were 32 partners from academia (e.g. University of Singapore, OFFIS, and TU Wien), consultants (e.g. act! consulting, Detecon, Price Waterhouse Coopers, and Steria Mummert Consulting), tool vendors (BOC

<sup>7</sup>To enhance readability of the diagram, we omitted the activities “browse catalog” and “update catalog” which connect the different data stores with the sequence of activities.

<sup>8</sup>See <http://www.matthes.in.tum.de/wikis/sebis/eamml> (cited 2011-01-21) for details on the project.

and iteratec), and practitioners (e.g. Bausparkasse Schwäbisch Hall, Capgemini sd&m, Credit Suisse, E.ON, Munich Re Group, PostFinance, RTC, SEB, and Siemens). The research project was scheduled in six phases with dedicated work packages, namely

- develop an activity framework of the EA management function using an extensive literature review (see the EA management activity framework proposed in Section 4.3; the according literature analysis can be found in [BMS10j]),
- derive initial method fragments for the identified activities from the EA management pattern catalog,
- conduct on-site workshops with the above partners from academia and industry gathering information on how they perform the different activities and which context factors influence the applicability,
- identify and document the practice-proven methods, revise and extend the existing methods to the on-site feedback provided by the partners,
- perform remote reviews of the documented methods with the partners, and
- revise and finalize the method descriptions based on the provided feedback.

The EAMML research project resulted in a set of practice-proven methods for EA management. This set was used in the initial development of BEAMS. For each of these methods, the single activities of the administration method were executed. Subsequently, each activity is detailed and complemented by an expository example. We use an observed best practice that was repeatedly discussed during the workshops of the EAMML to exemplify the activities of the administration method. Whenever, we refer to one of the catalogs, we provide an overview on the actual content of the current version of the BEAMS method base and present parts in the text flow and in the appendix<sup>9</sup>.

In the first step of the administration method, the observed best practice is **documented as a pattern**. Going into detail, the observed solution is documented following a typical pattern form (cf. Section 4.2.4), thereby making explicit the problem addressed by the solution, the context describing the environmental factors in which the solution can be applied, and the consequences of applying the solution. Furthermore, the single steps to be performed for solving the problem are described delineating possible forces that influence the design of the solution.



**Example 5.30: Administration method—document as pattern.**

Several industry partners of the EAMML reported on a success story that they experienced during their work, which is documented in the form of a pattern in the following.

**Problem:** You feel the risk of an unmanaged application landscape, with a multitude of technologies. You are afraid that the cost of development,

---

<sup>9</sup>The whole method base of BEAMS can be accessed at <http://www.matthes.in.tum.de/wikis/beams/home> (cited 2011-01-21).

operation, and maintenance of new business applications steadily increases due to a plethora of different technologies used. You believe architectural standards will help to reduce risks and costs through homogenization?

**Context:** The EA management initiative has been launched by a group of new employees who have made positive experiences with EA management in their former organizations. Founded by these employees no ‘official mandate’ and thus budget for the initiative exists. Nevertheless the initiative is officially tolerated as pilot for EA management. The IT departments of the organization are decentralized which hampers the establishment of enterprise-wide standards.

**Solution:** Arrange a meeting of software architects to define a set of appropriate technologies that represent enterprise-wide standards. Ensure that developers with respective skillset for applying the standards are available. Update the documentation of the current application landscape by conducting interviews at the beginning of the fiscal year. Assess currently used business applications for compliance with these standards and identify the business agency with the highest standard conformity. Officially award the ‘best’ business agency.

**Forces:** Defining enterprise-wide standards, the currently used technologies can either be taken into account or standards can be defined from scratch. Awarding can either be done using a page on the intranet or via a mass mail to all employees.

**Consequences:** The defined standards have to be maintained and evolved to ensure appropriateness for the implementation of business demands.



Organizations are unique in their way of doing business, the lived culture, used tools, and many other aspects. These different influences are reflected in the context description of the documented patterns. Representing the output of the first activity of the administration method, the context description of the pattern under consideration is compared to the already existing context descriptions provided by the BEAMS **catalog of organizational contexts**. If no congruent counterpart can be identified, the catalog of contexts is extended. In the initial version of BEAMS, we identified four dimensions of organizational contexts, namely “background of the EA management initiative”, “organizational culture”, “tool support for EA management”, and “organization of the IT department”. The complete catalog of organizational contexts that are linked to at least one MBBs via an “is applicable” or “is not applicable” relationship (cf. Section 5.1) is given in Appendix A.2.



**Example 5.31: Administration method—derive context descriptions.** Revisiting the pattern presented before, the following three char-

acteristics from two dimensions were derived from the pattern's context description:

**Background of the initiative** Different reasons and backgrounds why an EA management endeavor is initiated exists, namely

**bottom-up initiatives** Bottom-up initiatives are characterized by a group of 'heroes' representing the drivers of EA management. In organizations with bottom-up initiatives no dedicated mission and thus support from the upper management exists. Therefore, a bottom-up initiative is characterized by concealing the initiative behind project or program names, no dedicated budget for the initiative, and a limitation in scope. Bottom-up initiatives typically depend on the social and professional skills of the 'heroes'.

**pilot initiatives** Pilot initiatives are typically set up, if doubts and disbelieves in the benefits of EA management exist. Pilots are typically carried out hand in hand with bigger transformation projects and are characterized by a defined point in time where a decision about the continuation of the project is made, a limited scope and reach in terms of goals pursued, as well as the absence of a dedicated tool support.

**Organization of the IT department** Modern organizations have grown historically and exhibit different ways of how the IT department as a cross-function delivering capabilities used throughout the enterprise can be organized:

**decentralized IT department** In organizations with decentralized IT departments, solution delivery is aligned with the line of business and the IT managers report to the business executives. In this vein, coordination between the different local IT departments is aggravated which may result in heterogeneous solutions. In contrast, the specialized needs of the business agencies are addressed and the IT staff is controlled by the business agency which fosters business knowledge in the IT staff.



In line with our administration method, the problem addressed by the pattern is conceptualized in the activity **conceptualize problem** to the abstract goal and the concern on which the goal applies. Therefore, the BEAMS **catalog of goals** is parsed to identify a congruent counterpart. If no counterpart can be identified, the catalog is augmented to contain the newfound goal. Similarly, the concern to which the pattern applies is conceptualized, compared, and if necessary incorporated into the list of contained concerns (either relating to the cross-cutting aspects or the layers cf. Figure 1.1). The goals covered in the initial version of the BEAMS **catalog of goals** are

- ensure compliance,
- foster innovation,

- improve capability provision,
- improve project execution,
- increase disaster tolerance,
- increase homogeneity,
- increase management satisfaction,
- reduce operating cost, and
- reduce security breaches.



**Example 5.32: Administration method—derive goals and problems.** Parsing the problem of the pattern description, terms referring to an abstract goal can be easily identified as “homogenization”. The abstract goal of our example can therefore be mapped to the abstract goal “increasing homogeneity”.

In a similar vein, terms relating to concepts of the EA can be identified as “business applications” and “technology”. A cross-cutting aspect “standard” is also emphasized in the problem description.



In the **detail solution** activity, the notation for MBB description (see Section 5.1.2) is used to detail the solution provided by the pattern. Output of this activity is a detailed description of the solution using the language for describing MBBs as presented in Section 5.1.2. Thereto, the pattern’s solution section is revisited with the help of the observed cases to identify the single constituents of the MBB description as follows:

- The steps describing the solution are used to derive the tasks of the MBB description.
- For each task exactly one responsible participant is defined. If more than one participant is responsible for executing an identified tasks, this task needs to be detailed into fine grainer tasks for which a responsible participant can be defined. Further participants can be involved in the execution of a task via the relationships “informed” or “consulted”.
- For each relationship between a participant and a task at least the type of viewpoint is defined. Further best-practice recommendations or dissuasions regarding the viewpoint used can be specified textually, if available. In case a participant is only informed the type of viewpoint is defined to be of representation.
- Tasks for which a defined ordering exists are related using the language notation. Thereby, splits are complemented with a condition. For all tasks that do not have a preceding task, a trigger is defined. If provided by the pattern, the trigger type is detailed. For each task that does not specify a successive tasks, a sink is modeled.
- Forces are connected to the tasks they affect, i.e. for which they describe techniques to be selected. If more than one task is affected by the force, the force is further detailed.

The identified tasks are thereby described independent from the problem to be addressed, e.g. a step relating to the documentation of business applications is abstracted to a task for documenting EA-related concepts. Nevertheless, cross-cutting aspects can be referred to during tasks description, e.g. a step concerned with setting standards for business processes is abstracted to a task setting standards for EA-related concepts.

The BEAMS **catalog of participants** is used to ensure a consistent naming of participants. If a participant can be related to the skillset of an already existing participant, the participant is renamed, otherwise the BEAMS catalog of participants is extended. An excerpt of the current BEAMS catalog of participants that maps the participants identified in the above example to their counterparts is given in Appendix A.3. The catalog entries of the most prominent participant is exemplarily given below.

**Enterprise architect** Enterprise architects manage and lead the EA program, i.e. are the persons responsible for developing, communicating, and analyzing architectural states of the EA. They are in charge of the EA management function in terms of designing, implementing, and adapting the corresponding tasks. Furthermore, they oversee the EA budget (if available), support the introduction and establishment of EA management-related tasks, and increase the use of architectural descriptions and tools by promoting EA management. Typical questions an enterprise architect is concerned with are

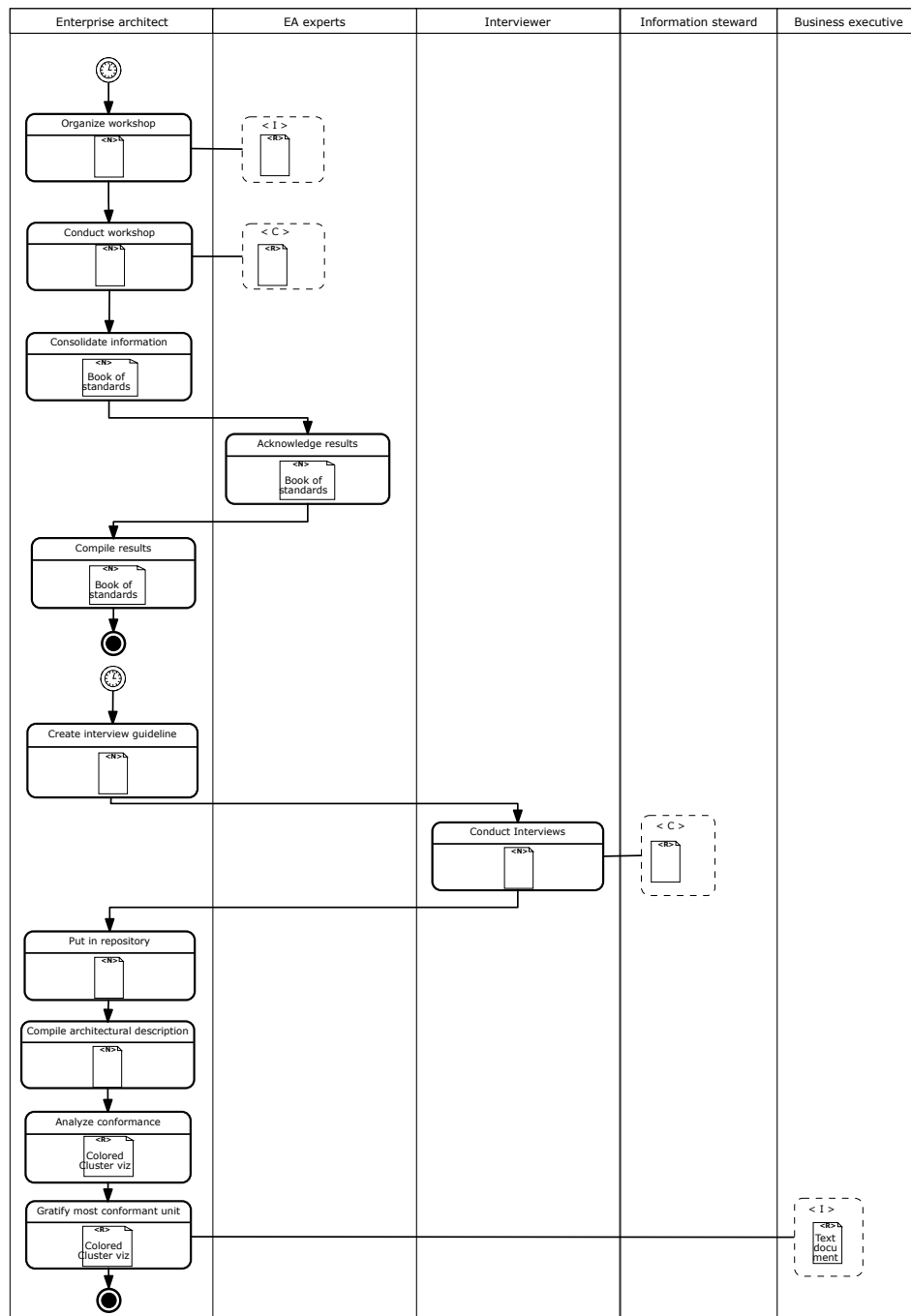
- How to promote the EA management function and its benefits?
- How to foster and improve communication between business and IT?
- How to best address the organization-specific problems from an organization-wide perspective?
- How to balance local and global interests?
- How can potentials for improvements in the EA be identified?
- How should the EA evolve to best fit the objectives and strategies of the organization?
- How to identify stakeholders of the EA management function?

The identified forces represent free choices for the actual implementation of a task. Put it more simply a force does only have local impact on a task, i.e. does not change the participants involved or pre- and post-conditions of a task. Each time a task is executed the responsible actor can decide which force (technique) to use.



**Example 5.33: Administration method—detail method description.** The pattern for increasing homogenization was initially discussed in the workshops of the EAMML and further detailed during the review phases. Based on the input of several industry partners that use the solution, the pattern was revised. Subsequently, the findings are illustrated using the modeling language. The model is further complemented with a textual description.

## 5. BEAMS: Building Blocks for EA Management Solutions



Four participants involved in executing the solution are identified. Comparing the skillset of the participants with the entries in the BEAMS **catalog of participants** the following mapping can be established (if deviating original names exists, these are given in brackets) “enterprise architect”, “EA expert” (technology expert), “interviewer” (working student), “information steward” (application owner), “business executive” (business unit manager). The viewpoints used by the participants during the conduction

of a task are specified to be of type notation and representation. Further recommendations or dissuasions for a certain viewpoint are documented, see e.g. the type “colored cluster visualization” for the task “analyze conformance”.

Complementing information, e.g. forces for certain tasks are provided textually in the associated description. The concept of force can be exemplified along the task “gratify most conformant unit”. Forces of this tasks that have been gathered during the EAMML workshop are a gratification send by mail, thus ensuring that everybody has been informed, or publishing the gratification on the intranet, thus giving it a longer-lasting nature.

For some tasks identified in the analysis of the pattern solution a logic ordering of the steps can be defined, resulting in a solution consisting of two disconnected task sequences. For each of these sequences a trigger and sink are defined. In the using enterprises both sequences of tasks were initiated based on a defined schedule such that the type of trigger can in both cases be defined to be of type “temporal”.



In the step **decompose pattern in MBBs**, the description of the pattern’s solution is decomposed into re-usable and modular method fragments that correspond to the main activities of the EA management function. Typically the tasks described by the solution can be easily classified as belonging to **develop & describe**, **communicate & enact**, or **analyze & evaluate**. For each task sequence, the pre-conditions that must hold to execute the first task and the post-conditions that are assumed to hold after the execution of the last task are defined. In line with the contexts, goals, and participants, the conditions are specified using a common set of meta-attributes. The conditions thereby are concern-independent referencing only properties of the concern or making assumptions on the coverage of cross-cutting aspects. The BEAMS **catalog of meta-attributes** is used to ensure a consistent terminology. If the set of meta-attributes is extended, the logical sequence of meta-attributes and the minimal set of pre- and post-conditions as relating to the activities of the EA management function needs to be updated.



**Example 5.34: Administration method—decompose patterns.**

Based on the logic ordering of steps identified above and the main activities of an EA management function, the solution provided by the pattern can be distinguished into four task sequences that relate to (1) ‘define standards’, (2) ‘describe current state’, (3) ‘analyze current state’, and (4) ‘communicating and enacting’ the defined standards. These sequences relate to the activity framework of BEAMS as follows, (1) and (2) can be related to the develop & describe activity, (3) can be classified as belonging to analyze & evaluate, and (4) can be evaluated as belonging to the communicate & enact activity. For each trigger and each sink that starts or



accordingly ends a sequence of tasks, the following pre- and post-conditions can be defined based on the pattern description.

	pre-condition	post-condition
(1)		concern.current.documented
(2)		concern.current.documented
(3)	concern.current.documented	concern.current.communicated, goal.current.documented
(2)	concern.current.communicated goal.current.documented	concern.current.communicated, goal.current.communicated



Representing the final activity of our administration method, for each identified MBB a ‘record sheet’ is compiled in the activity **integrate MBBs in method base**. The record sheet complements the model of the MBB as introduced before with additional textual information, i.e. the EA management activity to which the MBB belongs, constraints with respect to the states the MBB can be applied, i.e. the temporal-dependency as specified by the pre- and post-conditions, the pre- and post-conditions, the organizational context in which the MBB can be applied and the ones where it cannot be applied, and a recommendation if available for the specification of the trigger. If the MBB is a **mixinMBB**, the related **ELMF** is complementingly. In addition, each record sheet has a textual description that delineates the forces of single tasks and possible consequences of the applying the MBB.

The “describe by interview” MBB, which was derived from the “documenting applications” part of our pattern solution is used subsequently to exemplify the structure of an MBB as documented in the BEAMS method base. Further MBBs, especially the MBBs “multi expert evaluation”, “single expert evaluation”, and “officially gratify standard conformance” which resulted from our pattern, can be found in Appendix A.1.

### Describe by interview

<b>Activity</b>	develop & describe
<b>Participating actors</b>	enterprise architect, interviewer, information steward
<b>States</b>	current state, target state
<b>Pre-condition</b>	
<b>Post-condition</b>	concern.documented
<b>Organizational context</b>	
<b>applicable</b>	bottom-up initiative, office tools
<b>not applicable</b>	
<b>Trigger</b>	yearly, half yearly
<b>ELMF</b>	

Table 5.4.: MBB describe by interview

This building block employs an interview-based technique to create the architecture documentation. Thereby, an **INTERVIEWER** and an **INFORMATION STEWARD** meet and conduct

an interview based on a guideline compiled by the ENTERPRISE ARCHITECT. During the execution of the interview, the gathered information is documented by the INTERVIEWER. After the completion of all interviews, the ENTERPRISE ARCHITECT transcribes the gathered information into the repository and compiles an architectural description.

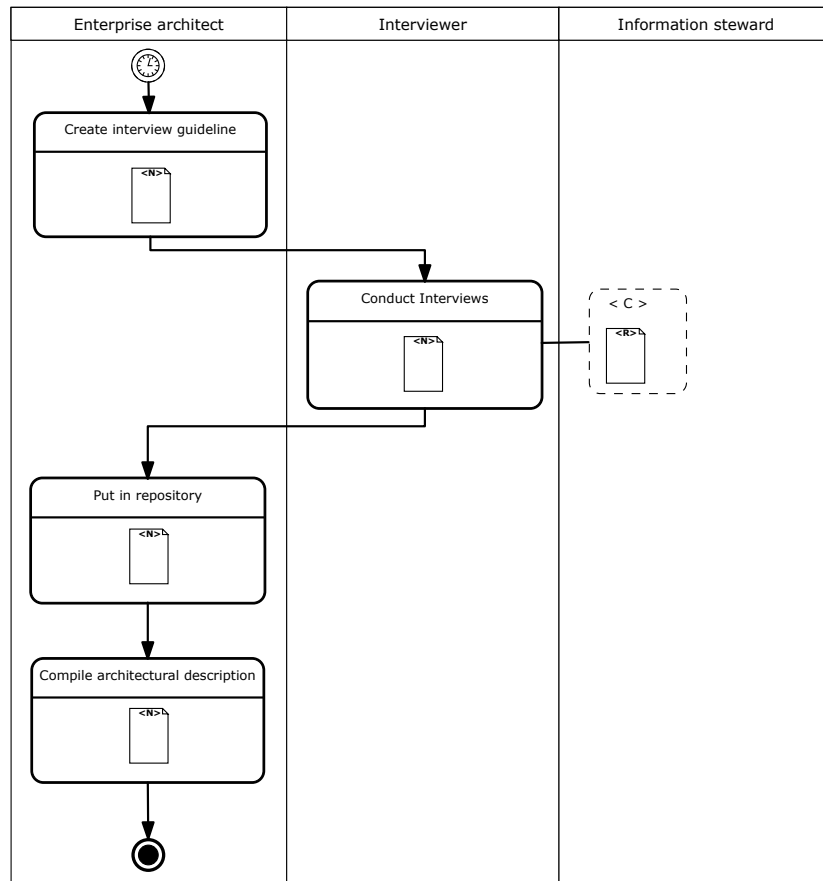


Figure 5.22.: MBB describe by interview

### Forces

“Telephone” vs. “face-to-face interview”: While a face-to-face interview typically leads to higher acceptance rate and receives more attention from the information stewards, it can be more time-consuming and cost-expensive especially in globally spread organizations.

### Consequences

- This building block follows the pull mechanism principle in execution. Thus, it might be more conveniently in use for the information stewards, while in contrast being inevitably connected to much higher effort, as the conduction requires the development of a respective guideline for the interview as well as requires a high expenditure of time from the interviewer for preparing, organizing, and conducting the interview.
- Furthermore, time has to be spend in integrating the answers, which might result in inconsistencies regarding syntactical correctness of an architectural description on the

one hand and inconsistencies between different descriptions with a common sub-area of interest on the other hand.

- By applying this method, the interviewer can promote the EA management initiative during the interview. This is especially helpful, if the topic of EA management is newly introduced or EA management should be promoted to the target community.

## 5.7. Summary

In this chapter, we described the *building blocks for EA management solutions* (BEAMS). BEAMS supplies a **development method** to design organization-specific EA management functions using a method base of proven best practices. The central concept of BEAMS, the **method base** and the therein interlinked *method building blocks* (MBBs) are described in this chapter. The BEAMS method base represents a design theory nexus instantiation as introduced in Section 2.2.3. Similarly, we understand the contained MBBs as design theories. In addition to the development method, we present the methods to apply and administer the method base. **MBBs** represent modular practice-proven solutions that can be flexibly combined and integrated into a consistent and coherent EA management function that accounts for the specific situation of an associated organization. This situation is determined by the specific EA management-related problem and the constraining organizational contexts. We lay the groundwork for BEAMS in this chapter by defining a modeling language for MBBs, and delineating how the MBBs contained in the method base are selected, configured, and integrated into a consistent and coherent EA management function. We discuss how the method base was initially developed by presenting the **administration method** to maintain and evolve of the method base.

The development method consists of four phases: characterize situation, configure EA management function, analyze EA management function, as well as adapt and evolve the EA management function. Figure 5.23 illustrates the first three phases. The development method represents an iterative process to design organization-specific EA management functions taking one specific situation at a time and developing an appropriate EA management function in a step-wise manner.

Laying the groundwork for the development method, we present a modeling language for describing MBBs as contained in the method base in Section 5.1. Thereto, we revisit the quality criteria for a ‘good’ EA management function as discussed in Section 4.1 and derive the relevant concepts of our language. We further consider general requirements for modeling languages as proposed by Frank in [Fr09]. The single constituents of an MBB are introduced in a stepwise manner and integrated into a conceptual model representing the basis for BEAMS. Complementing the syntax and semantics definition of our language to model MBBs, we propose a graphical notation. Thereto, prominent languages for process modeling are revisited based on the elicited requirements and a notation for modeling MBBs that builds on the *business process model and notation* (BPMN) is presented.

Complementing the perspective on MBBs as problem independent method descriptions, Section 5.2 presents the environmental concepts of MBBs. These concepts enable the organization of the method base, i.e. represent the general, context, and solution assertions as discussed

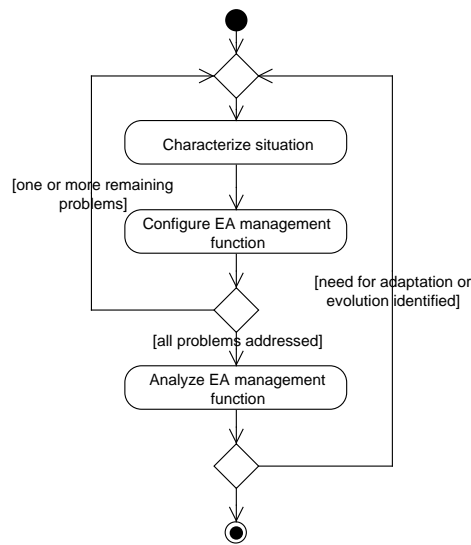


Figure 5.23.: Activity diagram illustrating the development method

in Section 2.3. Distinguishing between static relationships and links that can be dynamically established during configuration, we introduce the concepts of the organizational context for which an MBB has proven to be applicable or not applicable and the EA management-related problem to be addressed. Further revisiting the quality criteria, we introduce the concept of **mixinMBBs** which enables interlinking of the EA management function with other enterprise-level management functions. In addition, we present a technique to specify existing information sources that contribute to EA management.

In Section 5.3 the **variable** concept is introduced. Variables are on the one hand used to interlink the method-related elements of the EA management function with the language counterpart and to on the other hand enable organization-specific configuration of MBBs. During configuration of MBBs, we must ensure that the resulting method description is a consistent one. Two sources of inconsistency exist, i.e. information deficiencies and idle or misfit triggers. We introduce the concept of **pre-conditions** and **post-conditions** of MBBs to enable a consistent integration. Conditions specify certain characteristics for the associated information model that must hold to execute an MBB or that must hold after the MBB was executed. We introduce a defined set of meta-attributes defining a set of characteristics of an associated information model. Based on the understanding of pre- and post-conditions we detail the implicit relationships between MBBs. Using these implicit relationships we detail consistency rules (general assertions as discussed in Section 2.3). Finally, we introduce two different techniques that relate to the aforementioned sources of inconsistency, namely an **assessment technique** and a **liveliness technique**. The techniques build on **implicit relationships** between MBBs and the condition concept to assess compliance with the consistency rules.

Complementing the preparation of the development method, we introduce a distinction between **stakeholder** and **actor** in Section 5.4. Based on this distinction, we present two techniques to analyze the organizational implementability of an EA management function designed using the BEAMS method base. The stakeholder-involvement technique investigates, if the designed method informs the stakeholder of the problems on the achieved results and

the stakeholder-actor-dependency technique makes dependencies between information supplier and information consumer explicit. Different organizational interventions are supplied that can be applied based on the resulting dependency matrix.

The different phases of the development method are detailed in a stepwise manner in Section 5.2 to Section 5.5. Thereby, we provide an overview on the single steps to be performed and detail when to use which of the supplied techniques. Furthermore, the concept of an organization-specific configuration which contains the output of the single phases and reflects the idea of a central repository typically used during EA management endeavors is introduced. The theoretic exposition of the development method is complemented with an example. Finally, we discuss in Section 5.6 how the method base was initially developed using the results from an industry funded research project. The **administration method** specifies how the method base can be maintained and evolved by reflecting the approach to construct a design theory nexus instantiation and the pattern-based theory building process introduced in Section 2.3.



---

## Developing a Prototypic Toolset: The BEAMS Configurator

---

Theory-based design and development of an EA management function is a complex and error-prone process for which technical support is helpful. As Brinkkemper stated in [Br96, page 275] “the application of information systems development methods makes no sense without a proper automated support tool”. With respect to the context of situational method engineering where single method fragments are combined into a comprehensive method description, he further states that such tool functionalities should be extended with “consistency rules [which] are automatically checked and guarded”. The development method as well as the administration method presented in Chapter 5 are supported by a corresponding toolset, the **BEAMS configurator**. This configurator builds around the method base, i.e., a design theory nexus instantiation on design theories for EA management. As Pries-Heje and Baskerville describe in [PHB08] a tool-support complementing a nexus instantiation has to provide capabilities of *decision support systems* (DSS). Such systems support human decision makers by providing them information, processing of data in models, and focusing on effectiveness rather than efficiency in decision making [Sp93, page 3]. Further, the toolset has to provide *knowledge management* (KM) functionalities for administrating the method base. Therefore, especially functionalities for supporting the preservation, development, and distribution of knowledge, according to the KM cycle of Probst [Pr98] are needed (see Section 3.1.4). In the following, we describe the use cases for a BEAMS configurator in Section 6.1. We design such configurator building on the services of enterprise 2.0 tools (cf. Büchner et al. in [BMN09]) and present a prototypic implementation of the BEAMS configurator using the open source web-based platform *Tricia* [In10] and the BPMN modeling platform *Oryx* [DOW08a, DOW08b].

The BEAMS configurator is no EA management tool. Such tool is nevertheless required to implement a method configuration developed using the BEAMS approach. In [Ma08] we analyzed different EA management tools and showed that the tools greatly differ with respect to the provided flexibility. In [Bu08a, pages 20-21], we distinguish three characteristics for EA management tools, namely

**metamodel driven** tools provide maximum flexibility concerning adaptations of the information model incorporated in the tool and with respect to the used method. Therefore, strong metamodeling capabilities are incorporated in the underlying repositories.

**methodology driven** tools typically provide a comprehensive predefined information model together with a set of predefined visualizations, and methods for documentation, communication, and analysis. In these tools, minor adaptations with respect to the languages and methods are possible.

**process driven** tools which provide maximum guidance for EA management and can be seen as an extension to the methodology driven ones. They complement the predefined languages with defined workflows. Thereby, the whole EA management process is defined, providing procedures and specifying activities that have to be conducted to perform EA management.

A method configuration resulting from the development method can be implemented by the flexible *metamodel driven* tools. *Methodology driven* tools can also be suitable for implementing a method configuration, as long as the tool's predefined methodology is similar to the configured method. In addition, wiki-based systems can be used to implement the configured EA management methods. Wiki-based systems have proven to be useful in the context of architecture documentation (cf. Bachmann and Merson in [BM05]). In [Bu09h] and [Bu11c] we present a "lightweight" approach to EA documentation and analysis using wikis. Hence, wiki-based systems focus on text-based documentation of EA-related information and typically do not provide methodical support by workflows.

### 6.1. Use cases of the BEAMS configurator

In this section we describe the use cases supported by the BEAMS configurator. The BEAMS configurator supplies two core functionalities: Firstly, it supports the enterprise architect in developing organization-specific EA management functions, i.e., it supports the **development method**. Secondly, it provides mechanisms for the administrators that maintain and evolve the method base, i.e., execute the **administration method**. Central element of the configurator is the method base containing best-practices to design EA management functions. This method base is stored in the configurator and is made accessible to the users of the tool. The activities of the development method, namely **characterize situation**, **configure EA management function**, and **analyze EA management function** form three groups of use cases that the configurator has to support. In addition, the following use cases are supported:

**UC01** Browse BEAMS method base The users can search and browse through the different catalogs of BEAMS:

**UC01a** Browse catalog of organizational contexts The users can find an organizational context by name or by full-text content and can view its description.

**UC01b** Browse catalog of goals The users can find a goal by name or by full-text content and can read through the goal description.



**UC01c Browse catalog of participants** The users can find the participants defined in BEAMS and can view their definitions.

**UC01d Browse catalog of MBBs** The users can find and access MBBs, i.e., can view their descriptions in diagrams and text.

**UC02 Access configuration** The users can access the current organization-specific configuration:

**UC02a Browse configuration** The users can see the constituents of the current configuration, i.e., can view the specified organizational contexts and goals as well as the selected MBBs and their configurations.

**UC02b Export configuration** The users can export the organization-specific configuration in a defined format either for importing into a configurable EA management tool or for communicating the method's structure.

In Figure 6.1 we describe the use cases and their interplay in a UML 2.0 *use case diagram* [Ob10e, pages 603–621].

Subsequently, we briefly describe the use cases reifying the development method:

**UC03 Select context** Based on the catalog of organizational context, the users select the contexts that apply to the organization under consideration. By selection, the corresponding context is added to the organization-specific configuration.

**UC04 Select problem** The users select an EA management-related problem that a stakeholder would like to be addressed by the EA management function. Thereto, the users have to perform different sub-activities as follows:

**UC04a Specify stakeholder** The users specify the stakeholder, whose problem is to be described subsequently.

**UC04b Select goal** The users select the problem's constituting goal from the catalog of goals.

**UC04c Select concern** The users select the problem's concern from the catalog of concerns.

**UC05 Select MBB** The users select an admissible MBB from the catalog of MBBs. The configurator therefore filters the catalog via the technique for assessing whether an MBB is admissible in the organizational context as described in the organization-specific configuration.

**UC06 Customize MBB** The users bind the different variables specified in the MBB to corresponding values. The information model variable is automatically bound to the information model representing the selected problem. The customized MBB is integrate to the method of the organization-specific configuration. The following sub-activities have to be performed:

**UC06a Customize trigger** The users supply a trigger for the MBB's trigger variable. The configurator supports the selection by background analyses of the trigger specification and determines, whether the trigger can be fired or is idle.

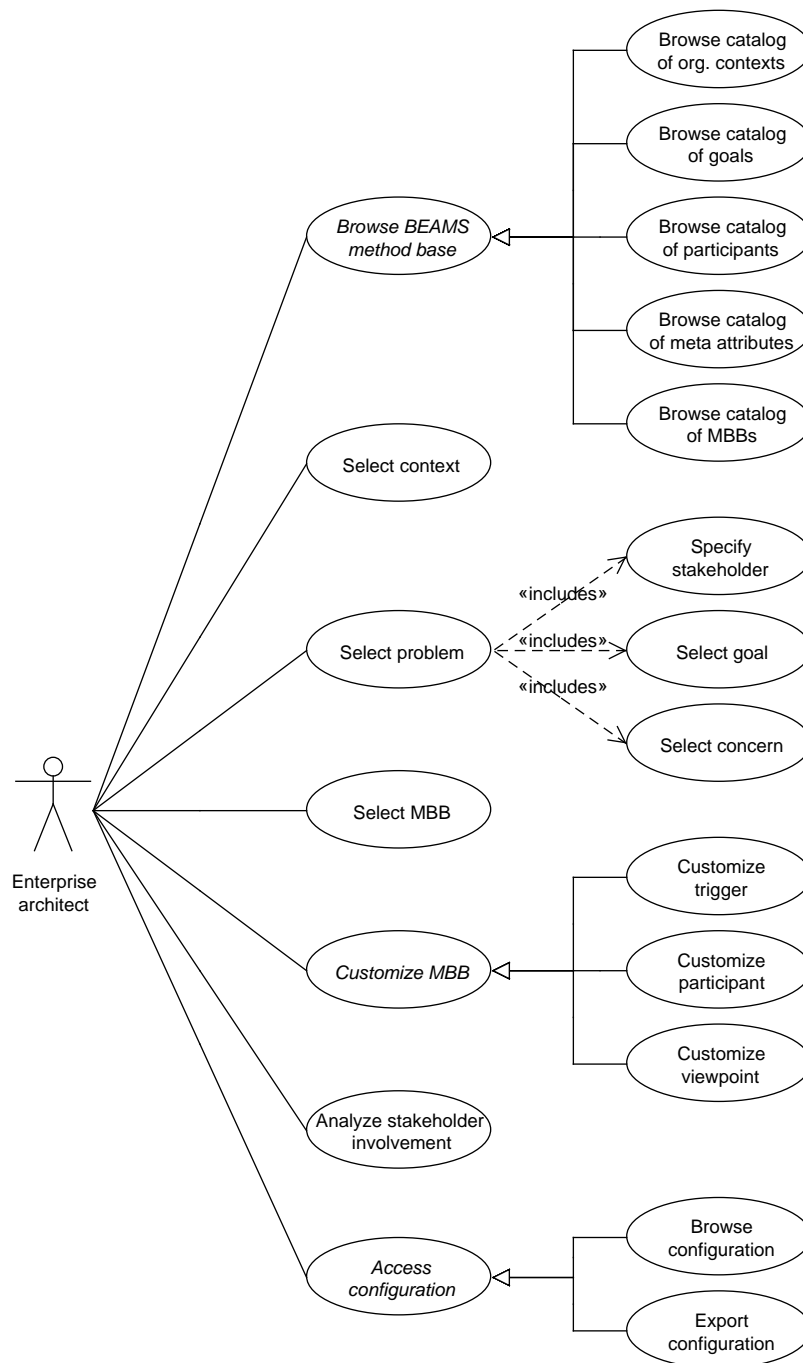


Figure 6.1.: Use case diagram illustrating the development of an organization-specific EA management function

**UC06b Customize participant** The users bind the participant variables of the MBB to organizational roles. Newly added roles are further stored in the organization-

specific configuration and used to recommend roles, during the next execution of UC06b.

**UC06c Customize viewpoint** The users develop a viewpoint for supplying or requesting information from a method’s participant. The configurator ensures that a viewpoint of an appropriate type (notation or representation) is supplied.

**UC07 Analyze stakeholder involvement** The users analyze the organization-specific configuration and determine, which tasks are executed to satisfy the information demands of which stakeholders. Further, the analysis determines which actors have to perform these tasks to provide the relevant information.

Above use cases describe the tool’s functionality from the perspective of the enterprise architect using the method base to develop an organization-specific EA management function. The BEAMS configurator further provides support for the administration of the method base. The use case diagram in Figure 6.2 details the required functionality from the perspective of the administrator.

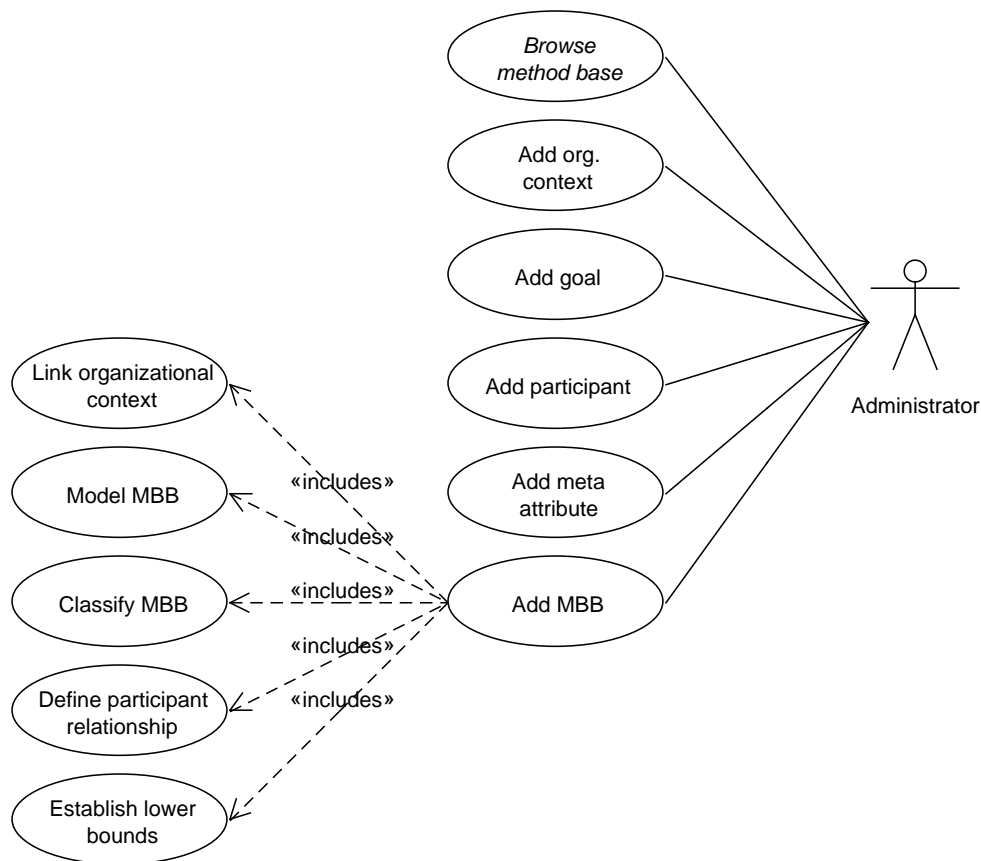


Figure 6.2.: Use case illustrating the administration of the method base

Subsequently, we textually describe the supported use cases available to the administrators of the method base (except for the already explained **UC01** browse method base).

- UC08 Add organizational context** The administrators define a new organizational context and supply a description of the corresponding environmental factors.
- UC09 Add goal** The administrators define a new goal together with a corresponding operationalization thereof via an information model.
- UC10 Add participant** The administrators describe a new type of participant involved in at least one task.
- UC11 Add meta attribute** The administrators describe a new meta attribute and specify how this meta attribute relates to the already existing meta attributes.
- UC12 Add MBB** The administrators add a new MBB consisting of different relevant descriptions, which are created in the sub-activities:
  - UC12a Link organizational context** The administrators describe under which organizational contexts the MBB is applicable and denote contexts, which are detrimental to the MBB's successful completion.
  - UC12b Model MBB** The administrators create a graphical specification of the MBB's tasks, flows, and variables using the graphical notation as presented in Section 5.1.2.
  - UC12c Classify MBB** The administrators specify the EA management activity to which the MBB contributes, delineate relationships to other enterprise-level management functions, and designate the EA states, onto which the MBB can be applied.
  - UC12d Define participant relationship** The administrators link the MBB's participant designators to the participants in the corresponding catalog.
  - UC12e Establish lower bounds** For any variable in the MBB model, the administrators supply lower bounds, if necessary. Thereby, relationships to triggers, information model building blocks, and participants are established.

With respect to the evolutionary nature of the method base, especially use case **UC01** is subject to an additional requirement. The administrators must have the option to browse and view not only the latest version of the contents of the method base, but also historized versions thereof. As already discussed at the beginning of this section and in Section 2.3 the BEAMS configurator is not an EA management tool, but a tool for supporting the meta-activity of designing an EA management function. A variety of tools providing support for executing EA management already exist (cf. [Ma08, pages 353-354]). Therefore, tool support for the execution of an EA management function itself is not in the scope of this thesis. Another use case is concerned with measuring the performance of the EA management function. As the BEAMS configurator is no EA management tool, tool-based support for performance measurement is not part of the BEAMS configurator.

### 6.2. Design of the BEAMS configurator

The use cases delineate the functionalities that the **BEAMS configurator** has to support. Central to the design of such configurator is the realization of the method base. This method base contains textual content describing the participants, goals, and organizational contexts.

Further, the MBBs provide combined content with both a textual part and a structured part in form of a diagrammatic method description. The different types of contents are interlinked with each other and are accessed by different ways of searching. The decision support characteristic of the tool-support for the development method is complemented with knowledge management characteristics needed for the administration method. These characteristics are both covered by enterprise 2.0 tools. In the following, we explore how typical services offered by enterprise 2.0 tools can be used to realize the use cases delineated in Section 6.1. In our analysis, we build on a service catalog<sup>1</sup> described by Büchner et al. in [BMN09]. In the following the fourteen core categories of services are described and the use cases supported by the particular service category are given in brackets:

**Authoring** targets the collaborative, web-based creation and manipulation of content (UC08–UC12). An exemplary capability provided by enterprise 2.0 tools in this context is structuring of content by templates (UC08–UC11). Furthermore, support for tables, images, and other media objects is provided (UC12), as well as functionalities to export content (UC02a).

**Link management** reflects services to handle references to content, e.g. wiki pages, files, or images (UC08–UC12). Supporting functionalities for link management are e.g. automatic propagation of link updates or labeling of invalid links.

**Personalization** provides services to present content in a user-specific form or to present user-specific content (UC02).

**Revision management** contains functionalities for tracing changes and evolution of content (UC01), e.g. by version control, audit trails, restoring old versions, or restoring.

**Search** centers around services to find content (UC01). Typical services provided by enterprise 2.0 tools are sorting of search results, full-text search over different content, storing of searches, and highlighting of search results (UC07). In addition, detailed searches are frequently provided which enable filtering of the search results or supporting text search features as AND, OR, NOT operators or wildcards (UC01).

**Tagging** supports the development and establishment of a bottom-up categorization system. Different services provided by enterprise 2.0 tools are private tags which are only visible for the creator (UC02), support for tag creation, and tag support for all content types.

For the development method especially the searching and the tagging services are of interest. These services support the identification and selection of admissible MBBs. The enterprise architects use personal tags to link customized versions of an MBB to the original MBB. Specialized search mechanisms can be used to implement our assessment and liveliness techniques. For the administration method, the services for authoring and link management are relevant. These services can be used to on the one hand enable collaborative development and evolution of MBBs, contexts, goals, and concerns, as well as to support the organization, i.e., interconnection, of the method base on the other hand.

---

<sup>1</sup>A detailed description of the service catalog used to evaluate different enterprise 2.0 tools can be found at <http://wwwmatthes.in.tum.de/wikis/enterprise-2-0-tool-survey-2010/home> (cited 2011-02-14)

### 6.3. Prototypic implementation of the BEAMS configurator

Our prototypic implementation of the BEAMS configurator is based on the open-source framework Tricia [In10]. This framework enables the development of enterprise 2.0 tools by providing extension points and a plugin mechanism following a data model-driven-approach. A broad range of the services of enterprise 2.0 tools is delivered out-of-the-box by Tricia and existing plugins (cf. Büchner et al. [BMN09]). Among these services is a wiki plugin building the foundation of our method base. For the prototypic implementation of the BEAMS configurator, we further use Oryx<sup>2</sup> an open and extensible web-based modeling framework originally developed for business process modeling [DOW08a]. The plugin-based extension mechanism of ORYX and the “stencil technology” allow to extend the framework to support other domain-specific modeling languages [DOW08a, pages 383–384].

Subsequently, we illustrate how the use cases identified in Section 6.1 are supported by our prototypic implementation. In particular, we revisit how the enterprise 2.0 services identified by Büchner et al. in [BMN09] are used to realize the prototypic **BEAMS configurator** and illustrate the realization of our prototype by screenshots. Our BEAMS configurator builds on the Tricia core and different plugins:

**core** which provides basic abstractions required by all applications of the domain, e.g. user profiles, groups, memberships, login, and registration procedures [BMN10],

**wiki plugin** that supports storing, accessing, and manipulating content in wiki pages and wikis [BMN10],

**hybrid wiki plugin** that provides an *open templating mechanism* that enables to store structured (key value pairs) and unstructured information in a wiki page [Bu11c, pages 140–147], and

**model editor plugin** that enables creation, manipulation, and storage of models using the Oryx model editor and predefined *StencilSets* [Di10].

#### 6.3.1. Browse and administrate the method base

The wiki plugin of Tricia provides a *what you see is what you get* (WYSIWYG)-editor that can be used to manage the content of the method base, e.g. to create wiki pages describing a specific organizational context or defining a participant (UC08–UC12). Figures can be included in pages to provide overviews and entry points for the different catalogs (UC01). Wiki pages are further used to document MBBs. Therein, the following constituents of an MBB have to be presented:

1. a model that describes the method fragment using the BPMN-like notation (cf. Section 5.1.2),
2. structured information to define the concepts linked to the MBB in the method base, and
3. unstructured information, e.g. additional textual description of an MBB.

---

<sup>2</sup>For more details on the Oryx project see <http://bpt.hpi.uni-potsdam.de/Oryx/> (cited 2011-02-04).

Tricia supports to store and to browse wiki pages and thereby MBBs by the three plugins as introduced above. Structured information is stored using the Tricia hybrid wiki plugin in open templates [Bu11c, pages 139–146]. Thereby, a wiki page is equipped with key-value-pairs that relate to the type of the page expressed via a tag, e.g. “mbb”. Each type is associated to a certain template that defines the admissible keys. For the type “mbb” the following keys are defined: ELMF, participants, states, name, activity, pre-condition, post-condition, organizational context (applicable in and not applicable in), and trigger. The values of the attributes can contain hyperlinks to the wiki pages describing related elements. Empty values are permitted for the keys ELMF, pre- and post-condition, organizational context, and trigger. The graphical model of the method is created using the model editor plugin developed by Dierl in [Di10]. This plugin enables the creation, manipulation, and representation of models in Tricia. We provide a dedicated *StencilSet* that incorporates the syntax and notation of our modeling language (cf. Section 5.1) to support modeling (UC12b) and customization of MBBs (UC06). Figure 6.3 illustrates the realization of our BEAMS configurator to define and customize MBBs.

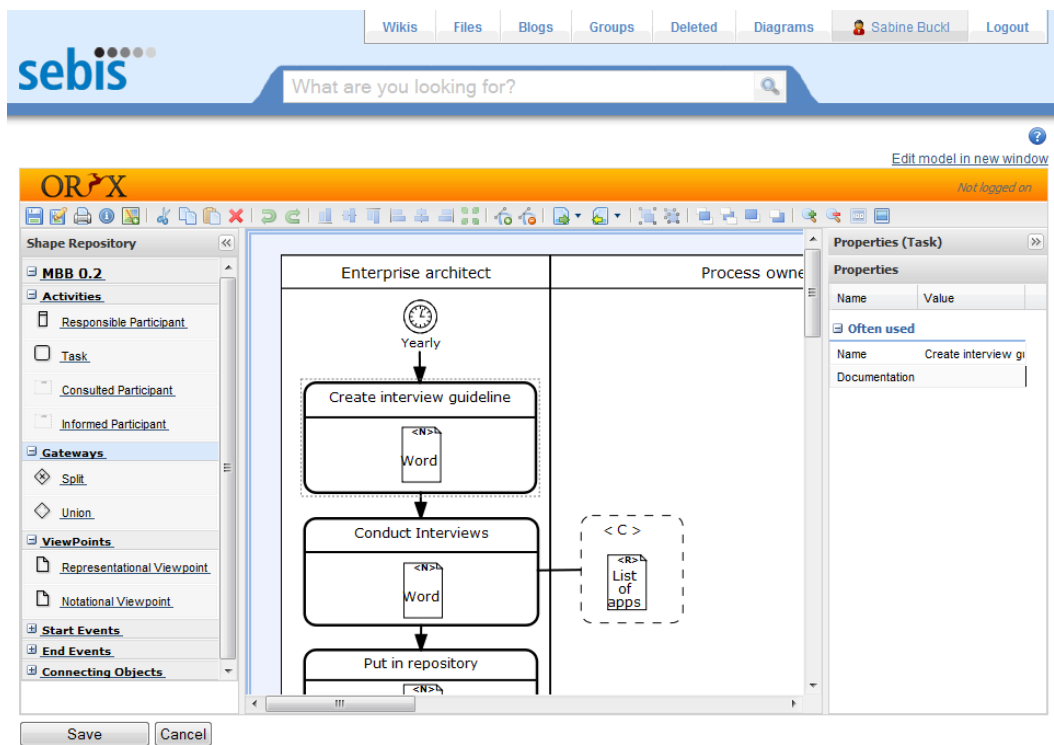


Figure 6.3.: BEAMS configurator: editor for modeling MBBs

The model editor plugin allows to embed models in a wiki page. Besides structured information and a model a wiki page can also contain unstructured content. The unstructured content is used to textually summarize the MBB and for describing the forces as well as consequences associated with the MBB. Figure 6.4 displays a screenshot from the BEAMS method base<sup>3</sup> and highlights the different parts of an MBB description. The value of the trigger attribute is left

<sup>3</sup>The BEAMS method base is as available at [http://wwwmatthes.in.tum.de/wikis/beams/home\(2011-02-07\)](http://wwwmatthes.in.tum.de/wikis/beams/home(2011-02-07)).

empty as no recommendations or restrictions for the trigger are defined by the MBB. Figure 6.4 additionally illustrates the *back referencing* feature of the Tricia hybrid wiki plugin [Bullc, page 145]. In the template section “references”, all “incoming links”, i.e., wiki pages that contain references to the current page, are listed.

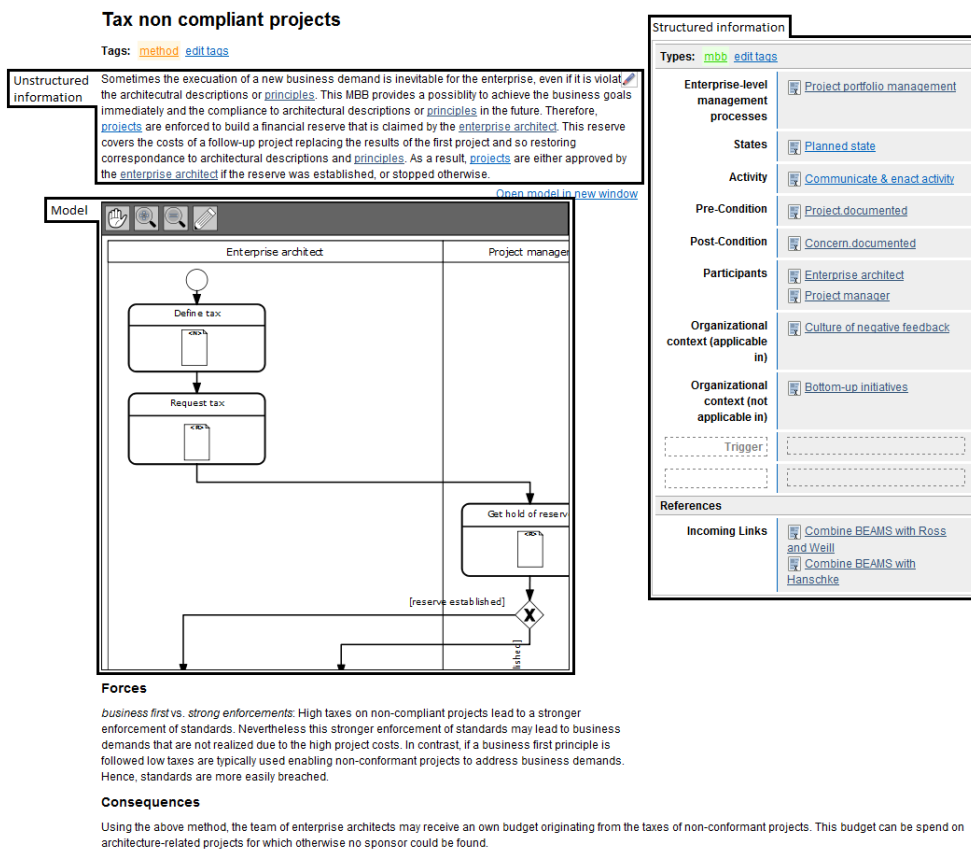


Figure 6.4.: Method base: a wiki page describing MBBs

### 6.3.2. Develop the EA management function

The search services of Tricia can be used to create a personalized dashboard that lists the selected organizational contexts, problems, participants, etc. and the configured method description (UC02). Figure 6.5 illustrates such a dashboard for the fictitious example created in Chapter 5.

The user is supported by our BEAMS configurator during the configuration of MBBs in different ways (UC06). During customization of MBBs, the model editor ensures that only valid MBB descriptions are created. In addition, the auto completion and suggestion services of Tricia support the user during renaming activities as well as in establishing links in the method base (UC12a, UC12d, UC12e). The integration of different MBBs (UC06) is supported by a dedicated plugin that takes an enumeration of MBBs as input and provides one method



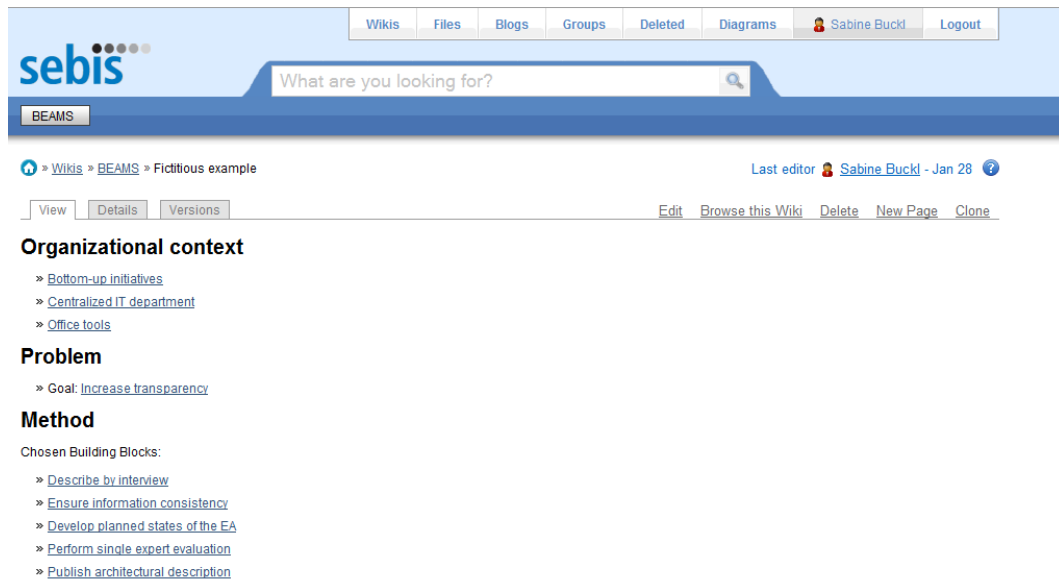


Figure 6.5.: BEAMS configurator: dashboard of the fictitious example

description as output. The result from integrating the MBBs specified in the fictitious example (cf. Figure 6.5) is illustrated in Figure 6.6.

Advanced search operations, e.g. AND, OR, and NOT operators, filtering, sorting, and highlighting of search results are in particular interesting for our BEAMS configurator. The Tricia platform provides full-text search over all content also including the Oryx-based MBB models. These search capabilities can be used to identify admissible MBBs based on the organization-specific configuration (UC03), i.e., the selected contexts as well as already configured MBBs. Thereby, the platform realizes the assessment technique for prioritizing the MBBs. The results of such search performed for our fictitious example from Chapter 5 is displayed in Figure 6.7. The search service can additionally be used to search the organization-specific configuration, e.g. to identify method fragments that are impacted by a changed organizational context (UC02a).

A dedicated plugin of our BEAMS configurator analyzes the stakeholder involvement in an organization-specific configuration. The plugin searches the graphical model description of a selected MBB and identifies the occurrences of a dedicated stakeholders, yielding one of the following three results:

**participant not found** if the stakeholder is not represented in the method,

**participant not informed** if the stakeholder is represented in the method but is not involved in any task, and

**participant informed** if the stakeholder participates in the method, i.e., is either responsible, consulted, or informed in at least one task.

Based on the search services of the Tricia toolset further plugins can be implemented that realize the liveness techniques for the trigger specification or that analyze stakeholder-actor-dependencies.

## 6. Developing a Prototypic Toolset: The BEAMS Configurator

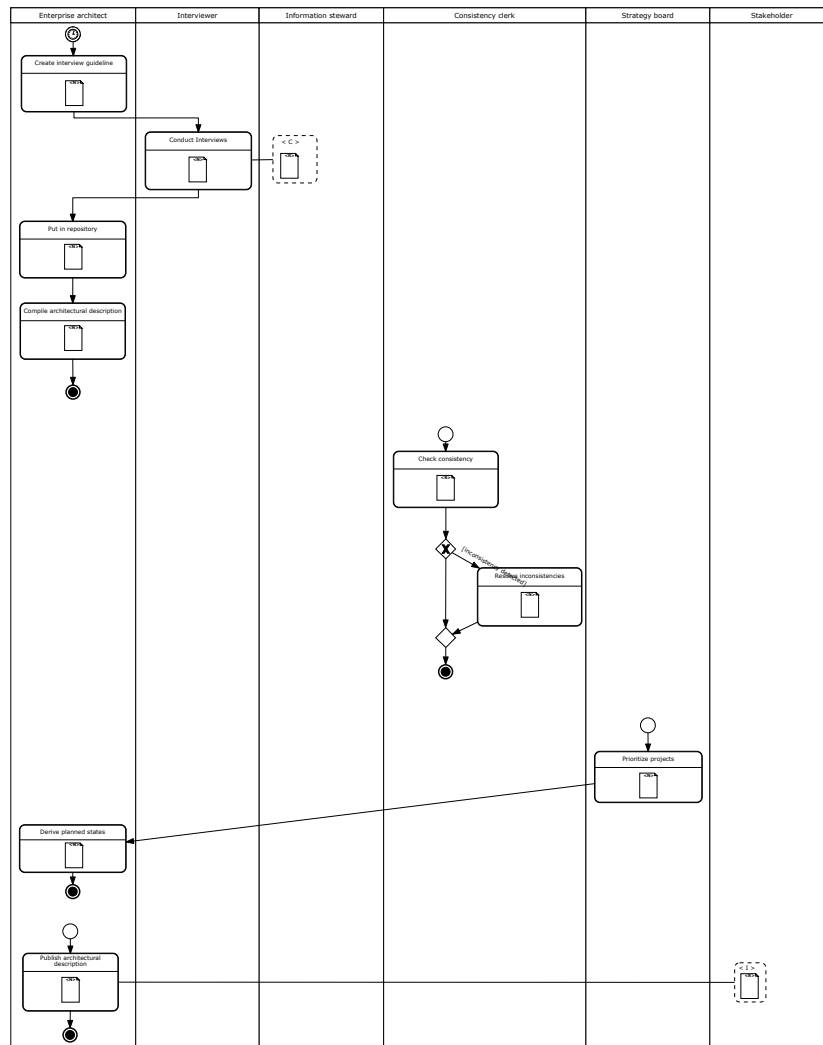


Figure 6.6.: Not customized method resulting from MBB integration

**Contents with tags 'fictitious example' and 'method' in space 'BEAMS'**

▼ Last modification: Any Date

▼ Content type: Wiki Page (3)

▼ Space: BEAMS (3)

▼ Embedded Function: insertModel (3)

▼ Type Tags: mbb (3)

▼ Special: Contains Invalid Links

Search for:  Go sort by: Relevance ▼ Tag Filter: ▼ Attribute Filter: ▼

required tags: fictitious example method

clipboard

excluded tags

Attribute: Pre-Condition Value contains:

Add additional filter

**Describe by interview**

This building block employs an interview-based technique to create the architecture documentation. Thereby, an interviewer and an information steward meet and conduct an interview. During the execution of the interview, the gathered information is documented by the interviewer. After the completion of all interviews, the...

BEAMS | [Last edited by Thomas Dierl, Jan 31]

[fictitious example](#) [method](#) [mbb](#) [edit tags](#)

**Ensure information consistency**

This building block describes an optional part of the develop & describe activity which can be used to ensure consistency of the gathered information and therefore the architectural description developed. Thereby, the consistency of the architectural description is ensured by a consistency clerk. He or she analyzes whether...

BEAMS | [Last edited by Sabine Buckl, Jan 18]

[fictitious example](#) [method](#) [mbb](#) [edit tags](#)

Figure 6.7.: BEAMS configurator: results from applying the assessment technique

### 6.3.3. Exporting the EA management function

The Tricia platform provides a functionality to export the content of wiki pages in *Adobe*<sup>®</sup> *Portable Document Format* (PDF) [Ad06]. This export contains both the unstructured and the structured information supplied on the page. The model editor plugin supports exporting the model in *JavaScript Object Notation* (JSON)<sup>4</sup>, *PDF*, and *Scalable Vector Graphics* (SVG) [W308] formats. Further, the Tricia platform provides a “clone” functionality. Cloning of wiki pages or wikis can be used to export the organization-specific configuration to another wiki that can subsequently be used as lightweight EA management tool in the sense of [Bu11c] (UC08).

According to the service catalog of Büchner et al. [BMS09b] enterprise 2.0 tools provide collaboration support by access control for different content objects, feedback mechanisms, social networking via user groups, or awareness via notifications. The Tricia toolset, on which the BEAMS configurator is based, provides different functionalities as e.g. access control, version control, and feedback mechanisms that can be used to facilitate the evolution and administration of the method base.

## 6.4. Summary and conclusion

In this chapter we present the **BEAMS configurator**, a prototypic tool support for BEAMS that realizes the nexus-bases IS generator as discussed in our research method (cf. Section 2.3). The BEAMS configurator is no EA management tool but incorporates functionalities of DSS and KM tools to support an enterprise architect in using the BEAMS method base. The BEAMS configurator helps the enterprise architect to apply our development method and the techniques presented in Chapter 5 to design an organization-specific EA management function. The prototypic implementation demonstrates feasibility of the development method and shows the inner consistency of the method base.

In Section 6.1 we prepare the implementation and derive use cases for the BEAMS configurator from the perspective of its users. We distinguish between use cases supporting the development method and ones supporting the administration method. Two auxiliary use cases are relevant to both methods and are hence presented separately. Each of the twelve use cases is textually described and its implications on the method base or the organization-specific configuration are discussed.

The design of the prototypic implementation is presented in Section 6.2. We decide to realize the BEAMS configurator on the basis of an enterprise 2.0 tool. This decision is justified by an analysis showing that the use cases can be supported by typical services of such tools as identified by Büchner et al. in [BMN09]. Table 6.1 provides an overview how the services of enterprise 2.0 tools relate to the use cases of the BEAMS configurator.

The functionalities to collaborative knowledge exchange (*authoring*) and the searching functionalities are important for our BEAMS configurator. Use case **UC06** configure MBB is not directly supported by a typical enterprise 2.0 service. We decide to develop a corresponding functionality based on a web-based modeling tool Oryx [DOW08a, DOW08b].

---

<sup>4</sup>For more information see [www.json.org](http://www.json.org) (cited 2011-02-07).

	Authoring	Link management	Personalization	Revision management	Search	Tagging
UC01				●	●	
UC02	●		●			●
UC03					●	
UC04					●	
UC05					●	
UC06						
UC07					●	
UC08	●	●				
UC09	●	●				
UC10	●	●				
UC11	●	●			●	
UC12	●	●				

Table 6.1.: Realizing use cases of the BEAMS configurator by enterprise 2.0 services

In addition to the theoretic discussions, we present a prototypic implementation of the BEAMS configurator based on the open-source web-collaboration platform Tricia [In10] and the web-based modeling platform Oryx in Section 6.3. Both platforms are integrated to provide

- support for the modeling language for MBBs by the *model editor* plugin [Di10],
- support for storing structured and unstructured information by the *hybrid wiki* plugin [Bu11c], and
- support for the assessment technique and analyses by specialized plugins.

The prototypic implementation of the BEAMS configurator does not provide step-by-step support for developing organization-specific EA management functions, but demonstrates the feasibility of the method, and the incorporated techniques, as well as the consistency of the method base. To provide comprehensive support for the enterprise architect, we currently develop a wizard-based support that guides the user through the steps of the development method using the plugins presented above. An application of the BEAMS configurator in practice is presented in Chapter 7. One extensions of the prototypic implementation could be the export of executable workflows for EA management tools, e.g. using *business process execution language* (BPEL) [OA07]. Other extensions could graphically represent the analysis results, e.g. using the *system cartography tool* [Bu07a], or provide more sophisticated collaboration support, e.g. via automated notification.

---

## Evaluating and Applying BEAMS

---

In the last step of our research method presented in Section 2.3 We evaluate BEAMS. Evaluation is an essential part of any design-oriented research (cf. Walls et al. [WWES92, page 41], Hevner et al. [He04, page 85–87], as well as Verschuren and Hartog [VH05]). The objective of evaluation in the context of design science research is to demonstrate the utility of the designed artifact and the provided innovation. A variety of different approaches to evaluate design-oriented research results exist (cf. Hevner et al. [He04, page 86] or Wilde and Hess [WH07]). Researchers as Fettke and Loos in [FL03] or Siau and Rossi in [SR08] advocate for a multi-perspective evaluation. Thereby, different characteristics of the design artifact can be evaluated with appropriate methods, which hence enhances validity of the evaluation of the design artifact as a whole. We use a threefold approach to evaluate BEAMS in this chapter and evaluate BEAMS firstly from a theoretic perspective, secondly against the state-of-the-art, and thirdly in practice using observational case studies.

Rigorously conducted research according to Becker [Be10, page 16] requires an evaluation of the design artifact against the objectives of the research program. In Section 7.1 we refer to the quality criteria for a ‘good’ EA management function as discussed in Section 4.1.1 to perform a theoretic evaluation of our development method. We argue that any EA management function designed using our development method fulfills these quality criteria ‘per design’. Furthermore, we revisit the requirements for the development method from Section 4.1.2 and discuss how they are satisfied by BEAMS. Finally, we apply the general characteristics of design science research according to Hevner et al. [He04] to understand the quality of our research outcome.

A core quality criteria of design science research according to Verschuren and Hartog [VH05, pages 749] is the innovation of the design artifact. Section 7.2 demonstrates that BEAMS is an innovative design artifact. We compare BEAMS with prevalent EA management approaches. Using the analysis framework presented in Section 3.2, we show that BEAMS on the one hand builds on the current state-of-the-art and on the other hand supplies an innovative ‘meta-

process' that facilitates its adaptation and evolution. We further demonstrate the utility of BEAMS by assessing the development method against the background of a fictitious case presented in literature. We show that skilled enterprise architects who use the development method, design a similar EA management function as provided by the literature example. We illustrate that BEAMS can be used in combination with other approaches and detail how it can be used to complement the ADM of TOGAF.

An empirical evaluation of BEAMS can be performed by applying the development method in practice. Section 7.3 presents the findings from two case studies in which the development method was applied in real world settings. We discuss how BEAMS was used and outline the application of the development method in cooperation with industry partners to (re-)design an organization-specific EA management function. The case studies originate from the financial industry and the public sector. We report the findings gained during the application and discuss the utility of the development method in such cases.

### 7.1. Theoretic evaluation against the characteristics

During the theoretic evaluation, we evaluate BEAMS from the utilitarian perspective and investigate how it facilitates the design of 'good' EA management functions. In line with the procedure proposed in the research method in 2.3, we evaluate the development method by revisiting the quality criteria, requirements, and the general research guidelines as discussed in Section 4.1. The development method and the underlying method base are thereby subsequently subjected to an argumentative evaluation process.

We used the quality criteria **Q1** to **Q9** in Section 5 to guide the design of BEAMS and its single constituents. The MBBs contained in the method base are interlinked with the organizational contexts in which they have proven to be applicable or in which they have proven not to be applicable. In the development method, suitable MBBs are selected based on a characterization of the situation such that **Q1** is fulfilled for the resulting EA management function. The **characterize situation** phase of the development method further identifies the stakeholders and their specific problems, which are used as input to the configuration (**Q2**). The configuration of MBBs as part of the development method is completed, if MBBs belonging to the develop & describe, communicate & enact, and analyze & evaluate activity are chosen. At this point, methods to document, analyze, and communicate (**Q3**) the selected area-of-interest are defined. During the configuration of a single MBB, the enterprise architect has to specify when the method fragment is executed (trigger) and who is responsible for executing the single tasks. Therefore, quality criterion **Q4** is fulfilled for the resulting EA management function. The EA artifacts used in the different tasks to gather and provide architecture information have to be defined during the configuration of an MBB. Whereas the area-of-interest is already defined in the **characterize situation** phase, the corresponding visualization is specified by selecting a respective viewpoint. Viewpoint and area-of-interest together define an EA modeling language used to create EA artifacts (**Q5**). The development method uses the assessment technique to ensure that only admissible MBBs can be selected for integration. Similarly, the assessment technique allows to identify inconsistencies in a method description. Executability of the EA management function is ensured by the techniques for investigating organizational implementability (**Q6**). Further, the method base provides the MIXINMBB concept to define

links to other ELMFs (**Q7**). The operationalization of the EA management-related problems defined in the **characterize situation** phase of the development method supports performance measurement and enables adaptation of the EA management function by an explicit linking to the associated method fragments. Such that quality criteria **Q8** can be regarded fulfilled, if a respective operationalization of the problem into measurable questions is found. The implicit relationships between MBBs as maintained by the method base can be used to derive possible evolution paths such that **Q9** is fulfilled.

In Section 4.1.2, we also elicited requirements with respect to BEAMS. Therein, requirement **R0** is fulfilled by design, i.e., quality criteria **Q1** to **Q9** are satisfied ‘per design’ using the techniques of BEAMS. BEAMS is a comprehensive approach to design EA management functions that requires familiarization with the underlying terminology and conceptualization. The administration method that maintains and evolves the method base specifies a procedure to re-organize, structure, and integrate practice-proven solutions to extend the method base (**R4**). The method base has been initially developed using the administration method and a set of practice-proven solutions thereby satisfying requirement **R2**. The development method guides the user during the (re-)design of an organization-specific EA management function in the development method by dedicated techniques that ensure that the resulting method description is a consistent and coherent one (**R3**). In Section 6 we discuss how services provided by enterprise 2.0 tools can be used to develop tool support for BEAMS. Based on these results we discuss a prototypic implementation of the BEAMS configurator (**R5**). The two remaining requirements (**R1**) and (**R6**) require more in-depth discussions. We will come back to them in Sections 7.3 and 7.2.2, respectively.

The final part of our theoretic evaluation is an assessment of the quality of our research outcome with respect to the general design research guidelines proposed by Hevner et al. [He04]:

**Design as an artifact** The contribution of this thesis is BEAMS, an approach to design organization-specific EA management functions. In this thesis, we develop *constructs* and *models* to link MBBs to organizational contexts. We discuss how to configure the method descriptions to organization-specific needs, e.g. roles, viewpoints, and triggers. We present the development *method* that supports users to select, configure, and integrate MBBs. Finally, we introduce a prototypic toolset, the BEAMS configurator, which *instantiates* our contribution.

**Problem relevance** A plurality of approaches to develop EA management functions has been proposed by researchers and practitioners (cf. Section 3). These approaches nevertheless do not account for the organization-specificity of EA management. BEAMS is an innovative approach to design organization-specific EA management functions. It incorporates existing best practice knowledge and provides sophisticated techniques to ensure organizational implementability of the developed EA management function.

**Design evaluation** The evaluation of the development method pursues a threefold approach. First, we assess the quality of BEAMS with respect to the defined requirements using argumentative evidence (cf. Section 7.1). Second, the innovation of BEAMS is evaluated against the state-of-the-art (cf. Section 7.2), and third, we use observational case studies to show the utility of BEAMS. Latter evaluations do not include an evaluation of the long-term utility of the resulting EA management functions, as the necessary period of observation is beyond the scope of a single doctoral thesis. Nevertheless, we performed

a subjective assessment with respect to the utility of the resulting EA management functions by challenging its practicability by experienced practitioners (cf. Section 7.3).

**Research contributions** BEAMS contributes new *foundations* for developing organization-specific EA management functions. Furthermore, the research approach taken in this thesis (cf. Section 2.3) contributes to the development of *research methodologies*. It presents a systematic research method for pattern-based theory building in research environments with high practitioner involvement.

**Research rigor** The development of BEAMS is performed according to the research method outlined in Section 2.3. In addition, the development method and the underlying method base are developed based on theoretical foundations of the discipline and its literature (cf. Section 3.3 and Section 4.2). The utility of BEAMS is complementingly evaluated from a theoretic and practical perspective in this section.

**Design as a search process** The central contribution of the thesis, the BEAMS development method is designed based on the findings of a previous artifact, the EA management pattern catalog (cf. Section 4.2.4). Furthermore, the underlying method base of organized practice-proven solutions can be iteratively enhanced and improved if new solutions are identified. In this vein, this thesis reflects the principle of *design as a search process* in a twofold way, namely on the instance and meta-level.

**Communication of research** The findings of this thesis, or preliminary stages thereof, are communicated in various ways. Different scientific papers have been presented and discussed in front of the scientific communities of computer science (cf. [BMS10c, BMS10j, BMS10a]) and information systems (cf. [Bu10a, BMS10k, BMS11, Bu11a]). BEAMS was further presented to students as part of the lecture *software engineering for business applications—master course: enterprise architecture management*<sup>1</sup> and to the community of EA management practitioners in different presentations and workshops (e.g. *EAMKON2010* and *EAMKON2011*<sup>2</sup>).

## 7.2. Comparing BEAMS with the state-of-the-art

The theoretic evaluation assesses the quality of BEAMS from a utilitarian perspective. We subsequently provide an evaluation against the state-of-the-art. Thereby, we show that BEAMS represents an innovative artifact, which covers more relevant aspects of EA management than prevalent approaches. BEAMS provides means to account for organization-specific aspects and supplies techniques to ensure organizational implementability of the resulting EA management function. Such means are not present in current EA management approaches.

As basis for our comparison, we classify BEAMS along the analysis framework from Section 3.2. Table 7.1 summarizes the classification, which is subsequently detailed.

With the concept of MIXINMBBs BEAMS provides dedicated techniques to ensure a bidirectional integration of the EA management function with other ELMFs. The MBBs contained in the method base reflect the activities **develop & describe**, **communicate & enact**,

---

<sup>1</sup>See <http://www.matthes.in.tum.de/wikis/sebis/vorlesung-eam> (cited 2011-02-11) for more details.

<sup>2</sup>See <http://www.eamkon.de/> (cited 2011-02-14) for more details.



INTEGRATION	unidirectional		bidirectional		
DEVELOP & DESCRIBE	current	planned	target	principle	question
COMMUNICATE & ENACT	current	planned	target	principle	question
ANALYZE & EVALUATE	current	planned	target	delta analysis	
CONFIGURE TO	organizational context		scope and reach		
ADAPT TO	organizational context		scope and reach		

Table 7.1.: Method classification for BEAMS

as well as **analyze & evaluate**. Nevertheless, best practice solutions with respect to questions operationalizing goals are currently scarce in practice and therefore not yet reflected in the method base. With the administration method at hand, newfound solutions providing questions can be integrated into the method base. The development approach used by BEAMS explicitly accounts for the aspect of configurability and adaptation concerning both organizational contexts as well as scope and reach of the endeavor.

We subsequently revisit a fictitious case from literature to demonstrate how BEAMS is applied by a skilled enterprise architect. Therein, we show that the resulting EA management function is similar to the one proposed in literature (cf. Section 7.2.1). In Section 7.2.2, we further outline how BEAMS can be used to complement other approaches by using the example of the ADM of TOGAF.

### 7.2.1. Revisiting a fictitious case from literature

We subsequently sketch a fictitious case from literature to assess the utility of BEAMS. Fictitious cases in literature typically strongly reflect the general approach presented by their authors such that they can be regarded as ‘objective assessment’. We show that a familiarized user can achieve similar results by applying our BEAMS approach. We choose the case of *ACME Energy* presented by Johnson and Ekstedt in [JE07, pages 293–306] to demonstrate the applicability of our development method. Thereto, we present excerpts from the case description below, extract the required input information for our development method, and discuss results from applying the method.

The situation at *ACME Energy* is described in [JE07, pages 293–296] as follows: *ACME Energy* is one of Europe’s largest energy utilities. The organization is globally distributed with a shared service business group providing services that are used across different business units. The benefits of an EA management function are known at *ACME Energy* and the issue receives top management attention. Furthermore, a dedicated modeling application is used to support the EA management endeavor. This tool provides flexible modeling capabilities and serves as a central repository for EA-related data.

Based on the above excerpt from Johnson and Ekstedt [JE07], we assume that the following organizational contexts apply:

- top-down initiative
- federated IT department
- specialized EA management tool

The goal pursued by the EA management endeavor of *ACME Energy* is described as “the building of flexible information systems” [JE07, page 297]. The goal is further detailed to the criterion of *maintainability* and different influencing factors thereof. We abstain from discussing the model-related aspects of the fictitious example and focus on the method-related part in the following. At *ACME Energy* a general process to document and maintain the current and target state of the EA is defined, such that a high level overview of both states is available. *ACME Energy* wants to establish a detailed planning for their EA evolution.

In the terminology of BEAMS the EA management-problem of *ACME Energy* relates to the abstract goal “improve capability provision” and a concern targeting planned states of the EA. With the characterization of the situation at hand, we iteratively search the BEAMS method base for admissible MBBs. Table 7.2 presents the results of applying the assessment technique. Thereby, a (●) indicates that the MBB is applicable in the respective organizational context.

MBB	top-down initia- tive	federated department	IT	specialized EA man- agement tool
Describe by cen- tral repository		●		●
Describe by ques- tionnaire	●			●
Describe by work- shop	●			
Describe by inter- view				

Table 7.2.: *ACME Energy*: results from applying the assessment technique

According to the fictitious case described in [JE07, page 299], the *business information officer* (BIO) “sent out her enterprise architects to collect the information [...] by identifying the main information sources”. Although no detailed description how this collection is performed is given, the employee “Mr. Andersson” is mentioned, who provides the information. For collecting the information, any of above MBBs can be used. We consistently assume that Mr. Andersson is interviewed. After deciding how to develop the two planned states of the EA, called *scenarios* in Johnson and Eksted [JE07], we apply the assessment technique to select the next MBB. The result of applying the assessment technique is shown in Table 7.3.

MBB	top-down initia- tive	federated department	IT	specialized EA man- agement tool
Publish architec- tural description	●			●
Quantitative assessment				●
Approve architec- ture description	●			
Ensure informa- tion consistency				

Table 7.3.: *ACME Energy*: results from applying the assessment technique (second iteration)

At *ACME Energy*, the BIO decides to perform a quantitative assessment of the two scenarios using *Bayesian analysis* [JE07, page 303]. The results of the assessment are communicated, i.e., the BIO is responsible for updating the intranet web pages [JE07, page 296]. This reflects the activity of **communicate & enact**, for which an MBB to publish architectural descriptions on the intranet is chosen (cf. Section A.1).

We above sketched that a skilled enterprise architect using the development method, can come up with a similar method for the EA management function as provided by a fictitious case from literature. At some points, we had to assume which MBB in detail would have been chosen. In contrast, at other points in the design of the EA management function the BEAMS framework only provided abstract method descriptions. This can be exemplified along the quantitative assessment for which BEAMS only details how a quantitative assessment generally takes place and who should be involved. BEAMS does not propose the techniques of Bayesian analysis (cf. Johnson and Ekstedt [JE07, page 303–306]). Summarizing the findings, we can state that BEAMS provides a general framework and detailed method descriptions for designing EA management functions incorporating best practices as proposed in literature but can also be used in combination with other approaches as we will illustrate subsequently.

### 7.2.2. Combining BEAMS with other approaches

TOGAF is a widely used and accepted EA management framework (cf. Section 3.3.5). On more than 800 pages, the framework discusses various aspects of EA management and provides the ADM as central methodical contribution. The embracingness of the textual description of TOGAF that needs to be investigated, makes it hard to find the relevant contributions. In addition, the method descriptions provided by TOGAF's ADM remain on a rather abstract level such that an organization willing to establish an EA management function based on TOGAF is left alone with the organization-specific design of the single phases.

In [Bu11a], we detail how the ADM of TOGAF can be complemented with the MBBs as provided by BEAMS. Figure 7.1 details a mapping between the single phases of the TOGAF ADM and MBBs as provided by BEAMS that can be used to detail the single phases based on the specific situation of the organization.

A detailed description how BEAMS and TOGAF can be successfully combined is given in [Bu11a]. The theoretic exposition is complemented by an application example using the frequently used problem “increasing homogeneity of the application landscape”.

## 7.3. Observational case studies

We subsequently present two case studies that result from projects with partnering enterprises. In the case studies, the development method of BEAMS was used to (re-)design an organization-specific EA management function. The main objective of the presented case studies is to demonstrate applicability and utility of our development method. We additionally report on the lessons learned and identify potentials for improvement. The partnering organizations in which the case studies were conducted, belong to different industry sectors,

## 7. Evaluating and Applying BEAMS

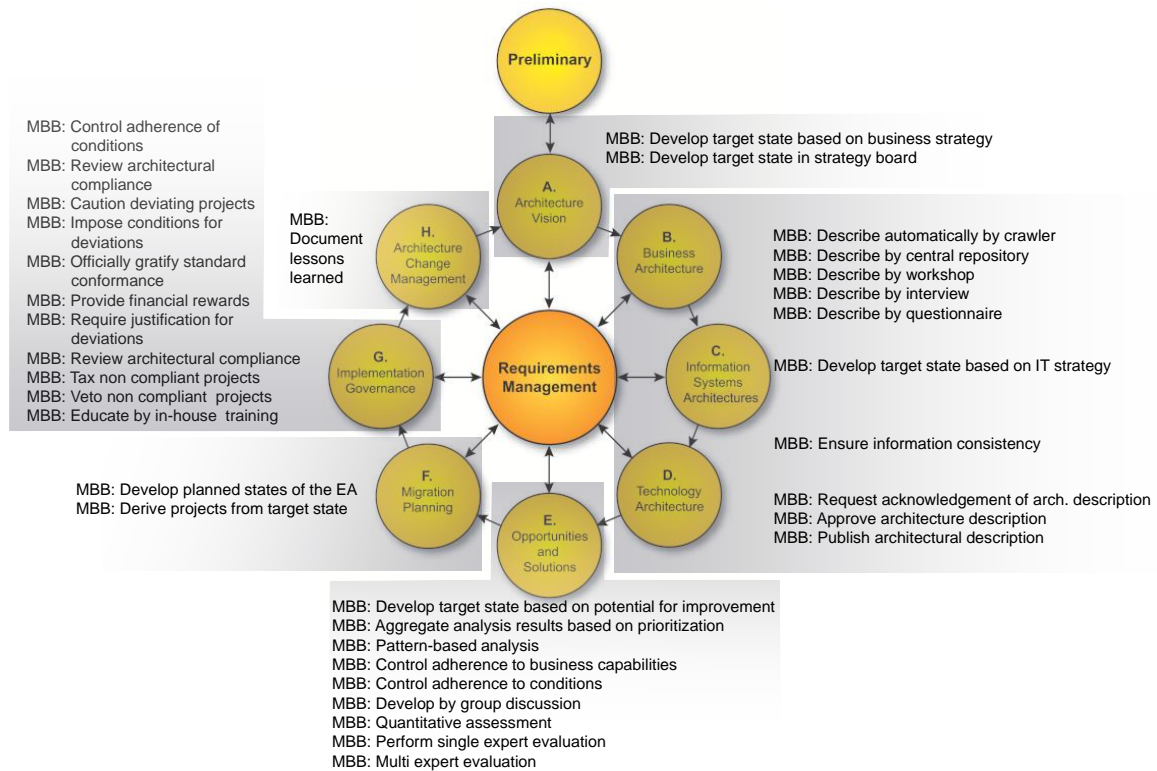


Figure 7.1.: Combining BEAMS with TOGAF's ADM

differ with respect to the focus of the EA management initiatives, as well as concerning the 'EA management-history' of the organizations.

The case studies were conducted as parts of two student projects. The first case study was performed as part of a student internship. The second case study was conducted as part of a so-called *guided research*<sup>3</sup> and performed by a single student. We supervised both student projects and supported the execution of the single steps of the development method by participating in meetings, answering questions, and reviewing preliminary results. Due to the implicit participation in the project, we subsequently report on feedback from two different perspectives:

- the students take a 'consultancy perspective' and need a detailed understanding of BEAMS, and
- the industry partners take a 'customer perspective' and provide feedback on the designed EA management functions.

<sup>3</sup>For more details see [https://drehscheibe.in.tum.de/myintum/kurs\\_verwaltung/cm.html?id=IN2169&lang=en](https://drehscheibe.in.tum.de/myintum/kurs_verwaltung/cm.html?id=IN2169&lang=en) (cited 2011-02-09).

Both perspectives contributed to our evaluation and the identification of potentials for improvement.

The use cases are subsequently documented using a unified structure to enhance readability. The structure consists of six parts, namely

**organization** a short outline of the organization, describing the industry sector and the history of the EA management initiative,

**context of the case study** the context in which the case study takes place, e.g. an initiative or program,

**course of action** an overview on the course of action of the case study, including time schedule, project duration, members of the team, etc., and

**results** the results of the case study, i.e., excerpts from the resulting BEAMS method description.

The section concludes with findings that summarize the feedback of the industry partners and students on the overall utility of BEAMS.

### 7.3.1. A case from the financial industry

The first case was conducted in an internship between April 2010 and October 2010 at an international acting IT provider of a financial institution (FI). As the IT service provider has a longer history in the context of EA management, the case study is an ex post evaluation in which BEAMS is used to re-design the EA management function.

#### **Organization**

The organization under consideration is the IT provider of an internationally operating German bank. In 1996 the topic of EA management had its first occurrence in the organization<sup>4</sup> under the label of enterprise-wide data modeling. The need for transparency arose in the context of a merger. Prior to the merger both companies independently conducted enterprise-wide data modeling endeavors. After the merger, the enterprise-wide data models were maintained, although a change in the focus as well as concerning the scope took place. In certain parts of the enterprise the focus shifted towards business process centric modeling, while other parts continued to do pure data modeling. In the year 2002 the term EA management makes its first appearance, when a project was launched to increase the business IT alignment. Therefore, architectural information from different parts of the organization was consolidated and used to identify fields for action. In order to assess the advances made in this field, a similar project was launched in the year 2005. The take-over by an international banking company at the end of 2005 changed the overall make-up of the organization significantly. In particular, the IT departments of the formerly independent organizations, as well as the IT assets developed, operated, and managed by them, were to undergo extensive changes leading to an increased centralization of structures and finally to the outsourcing of the IT provider in 2009.

#### **Context of the case study**

The objective of the program in which the case study took place was *improved support for project portfolio management decisions*. The IT service provider receives an annual budget

---

<sup>4</sup>At this time the IT provider was still part of the financial institution. The carve-out took place in 2009.

to cover the expenses for maintaining the IT service portfolio and for conducting projects to correspond to the business demands of the financial institution. Projects raise two different kinds of costs—development costs and operating costs. The project portfolio management process should be supported by the EA management function to balance the money spent to develop and maintain the IT service portfolio.

### **Course of action**

The project was initiated by us in March 2010. Five master students from TU München formed the core team of the project together with one representative of the IT service provider. The master students were prepared for the project by a lecture in which BEAMS was presented<sup>5</sup>. The students were accompanied by us in the initial workshop and during the presentation of the results in the final workshop. The students executed the development method by iteratively performing the following steps, namely

- arrange meetings with the industry partner,
- conduct workshops to
  - present and discuss a draft for the EA management function<sup>6</sup>,
  - gather input for the next step of the development method, and
- develop a draft for the EA management function including alternatives.

During the internship regularly meetings on a weekly-basis were performed in which the students had to report on their current results and the next steps to be performed. We also discussed open questions with the students. After the internships both the students and the representative from the industry partner provided feedback about their subjective impressions on the development method and the achieved results.

### **Results**

In the **characterize the situation** activity, the students identified the following stakeholders, organizational contexts, and goals<sup>7</sup>:

- stakeholders and associated goals:
  - IT board, reduce operating costs
  - enterprise architect, increase homogeneity
- organizational contexts:
  - top-down initiative
  - culture of negative feedback
  - specialized tools

---

<sup>5</sup>For an overview on the lecture see <http://wwwmatthes.in.tum.de/wikis/sebis/vorlesung-eam> (cited 2011-02-09)

<sup>6</sup>This step is omitted in the first iteration.

<sup>7</sup>As this thesis focuses on the method-part of the EA management function, we only provide an overview on the concern's concepts that are of relevance for the presentation of the case study.

- concerns:
  - demand
  - project proposal with the attributes cost, priority, and isCompliant
  - standards

In several iterations of the **configure EA management function** activity, the students developed an EA management function as described in Figure 7.2. The resulting method description contains five MBBs. Table 7.4 lists the MBBs as well as the organizational contexts and states for which they are applicable. Furthermore, the ELMF in which the respective MBB has to be integrated, if existing, is given.

MBB	Organizational contexts	States	ELMFs
<b>Describe by interview</b>	bottom-up, office tools, pilot initiative	current state, target state	
<b>Describe by central repository</b>	specialized EA management tools	current state, target state	
<b>Perform single expert evaluation</b>		currents state, planned state, target state	
<b>Review architectural compliance</b>	culture of negative feedback	planned state	project portfolio management
<b>Require justification for deviation</b>	culture of negative feedback	planned state	project portfolio management

Table 7.4.: Case study FI: selected MBBs

Table 7.5 provides an overview on the pre-conditions and post-conditions of the configured method fragments of the EA management function. After the configuration, the pre- and post-conditions of the configured method fragments do not longer reference the information variables but refer to concepts from the underlying information model.

The following participants are involved in the illustrated part of the EA management function. The original names of the participants as contained in the BEAMS catalog of participants are given in brackets.

Enterprise architecture managers (enterprise architect) manage and lead the EA program, select and implement the EA management function and furthermore identify standards. They are responsible for discussing and deciding on the compliance of the project proposals from a holistic perspective. Therefore, they take the enterprise strategy and architectural principles into account.

Business relations managers (interviewer) act on the side of the project portfolio management and are in direct contact with clients. They collect clients' demands for services and perform a first evaluation of the demand with respect to the business impact.

## 7. Evaluating and Applying BEAMS

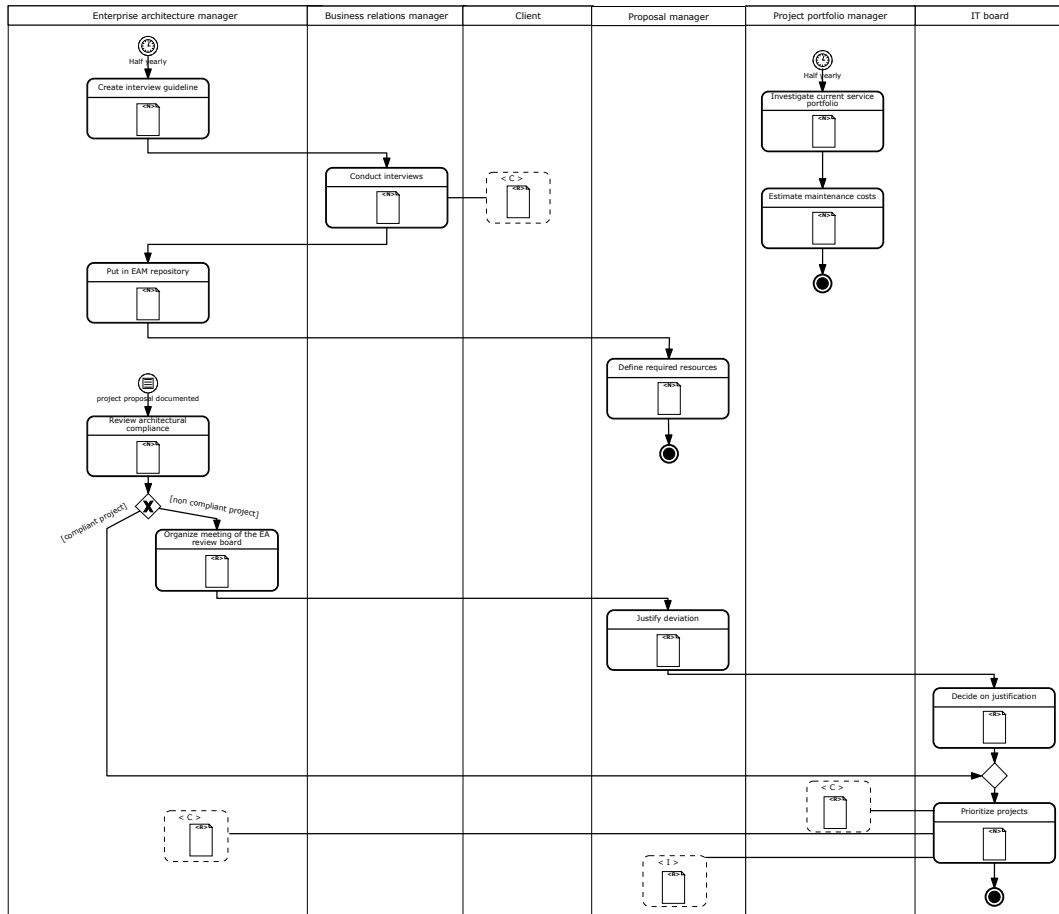


Figure 7.2.: Case study FI: excerpt from the method description

MBB	Pre-conditions	Post-conditions
<b>Describe by interview</b>		demand.documented
<b>Describe by central repository</b>		projectProposal.documented
<b>Perform single expert evaluation</b>	projectProposal.documented	projectProposal.documented, cost.documented
<b>Review architectural compliance</b>	projectProposal.documented, standard.documented	projectProposal.documented, standard.documented, compliance.documented
<b>Require justification for deviation</b>	compliance.documented	priority.documented

Table 7.5.: Case study FI: pre-conditions and post-conditions of the selected MBBs

Clients (information steward) represent the business agencies from the banking company that are the owners of the demands which are reflected in the project proposals.

Proposal managers (information steward) work side-by-side with the developers to create a



project proposal, realizing customers' demands. They plan the resources which are necessary for the project realization.

Project portfolio managers (EA expert) are in charge of the organization's project portfolio management. They assess project proposals, prepare their budgeting, and provide information to support the decision on the project portfolio.

IT board (EA review board) consists of several participants from the IT departments and the business agencies. The IT board identifies those project proposals which should be accomplished and assigns budget. Hence the IT board is responsible for prioritizing projects.

In the **analyze EA management function** activity, the students assessed the organization-specific configuration with respect to the stakeholder involvement and the stakeholder-actor-dependency to ensure organizational implementability. The method description in Figure 7.2 shows full stakeholder involvement, as both enterprise architects and the IT board are involved in the final step "prioritize projects". Figure 7.3 further provides the analysis results from investigating stakeholder-actor-dependencies.

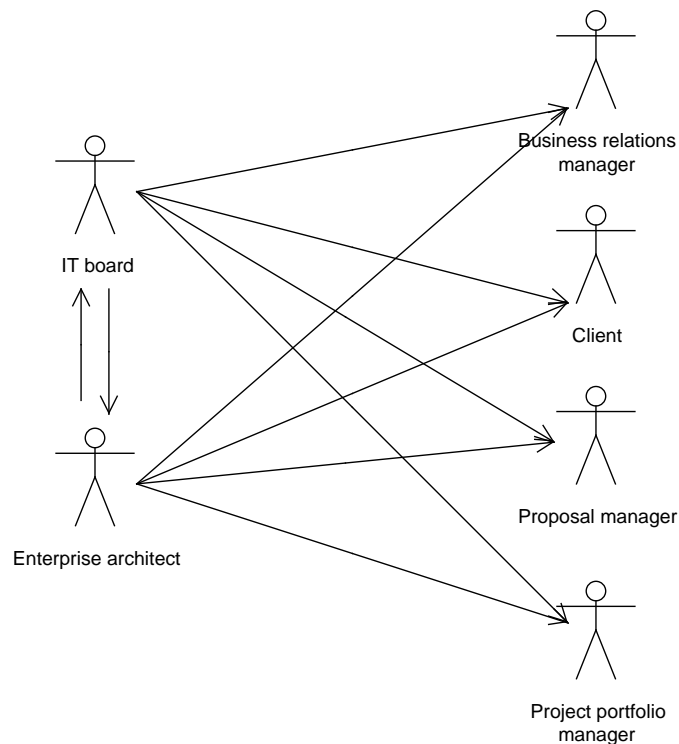


Figure 7.3.: Case study FI: stakeholder-actor-dependencies

Due to the similar problems of the IT board and the enterprise architect they depend on the same information providers. Although the enterprise architect and the IT board do not stand in a direct line-of-control with the actors they depend on, the organizational implementability is ensured due to the official mandate of the EA management initiative.

### 7.3.2. A case from the public sector

Our second case was performed in a “guided research” by a student between October 2010 and January 2011 at the central IT department of a registered association of research institutes from the public sector (PS). In this case study the development method of BEAMS was used to design an EA management function from scratch.

#### Organization

The non-governmental and non-profit association of German research institutes has a history of over 100 years. The different research institutes conduct basic research in the natural sciences, life sciences, social sciences, as well as arts and humanities. The association has a historically grown heterogeneous EA. In particular the IT infrastructure is diverse. To attract leading researchers, the association follows the principle of ‘academic freedom’, i.e., the researchers define their research and the associated infrastructure in terms of staffing or IT systems. Administration services are provided by the central administrative headquarter, which has recently established the position of an “enterprise architect” to enable a managed evolution of the provided business and IT architecture as well as ensure that synergies are taken advantage of.

#### Context of the case study

The objective of the establishment of the position of an enterprise architect is to reduce heterogeneity of the processes as well as the supporting IT. Synergy effects are expected, e.g. due to a reduction of maintenance costs and limitation of expert knowledge that needs to be objected. Furthermore, the research institutes should be supported in decisions regarding infrastructure changes by defined standards. The establishment of an EA management function in the association is challenged by the principle of “academic freedom”.

#### Course of action

The project was initiated in October 2010. A master student from TU München and the enterprise architect of the association form the core team of the project. The master student was prepared for the project by her work as student assistant at sebis. The duties and responsibilities during her work at sebis centered around maintaining content in the BEAMS method base. We supervised the student during the project and reviewed preliminary results. Furthermore, we took part in the workshops conducted with the industry partner and participated in discussions. Alike the first case study, the project was guided by a series of workshops (see Section 7.3.1).

During the guided research we reviewed the student’s preparations for the workshops as well as the workshop results. The student worked autonomous between the workshops and only occasionally called for feedback and assistance. After the guided research we performed a feedback round and interviewed the representative of the industry partner as well as the student to receive a subjective impression on the development method and the achieved results.

#### Results

In the **characterize situation** activity, the student identified the following stakeholders, organizational contexts, and goals:

- stakeholders and associated goals:
  - Project manager, increase homogeneity
  - Upper management, increase transparency

- organizational contexts:
  - pilot initiative
  - centralized IT department
  - office tools
- concerns:
  - business applications support business process at organizational unit (summarized as ‘concern’ in the following)
  - attribute isStandard for business applications and technologies
  - conformity of an application landscape

In several iterations of the activity **configure EA management function**, the student developed the EA management function as shown in Figure 7.4. The resulting method description builds on six MBBs (cf. Table B.1). Table 7.6 provides an overview on the pre- and post-conditions of the configured method fragments.

MBB	Pre-conditions	Post-conditions
<b>Describe by interview</b>		concern.current.documented
<b>Ensure information consistency</b>	concern.current.documented	concern.current.consistent
<b>Multi expert evaluation</b>		standards.documented
<b>Request acknowledgment for architecture description</b>	concern.current.documented, standards.documented	concern.current.approved, standards.approved
<b>Pattern-based analysis</b>	concern.current.documented, standards.documented	conformity.documented
<b>Publish architectural description</b>	conformity.documented	conformity.published

Table 7.6.: Case study PS: pre-conditions and post-conditions of the selected MBBs

In the **analyze EA management function** activity, the student investigates the organization-specific configuration with respect to the stakeholder involvement and the stakeholder-actor-dependencies. The method as described in Figure 7.4 involves all relevant stakeholders, as they are informed on achieved results regarding their associated problem. The upper management is provided with an overview on the current state of the EA and the project managers are informed on the analysis results with respect to conformity of the provided services. Figure 7.5 further provides the analysis results from investigating stakeholder-actor-dependencies.

With the upper management as stakeholder for the problem **increase transparency** on the current application landscape, the stakeholder-actor dependencies mirror the lines of control in the organization. Thereto, the organization implementability is assumed to be ensured.

## 7. Evaluating and Applying BEAMS

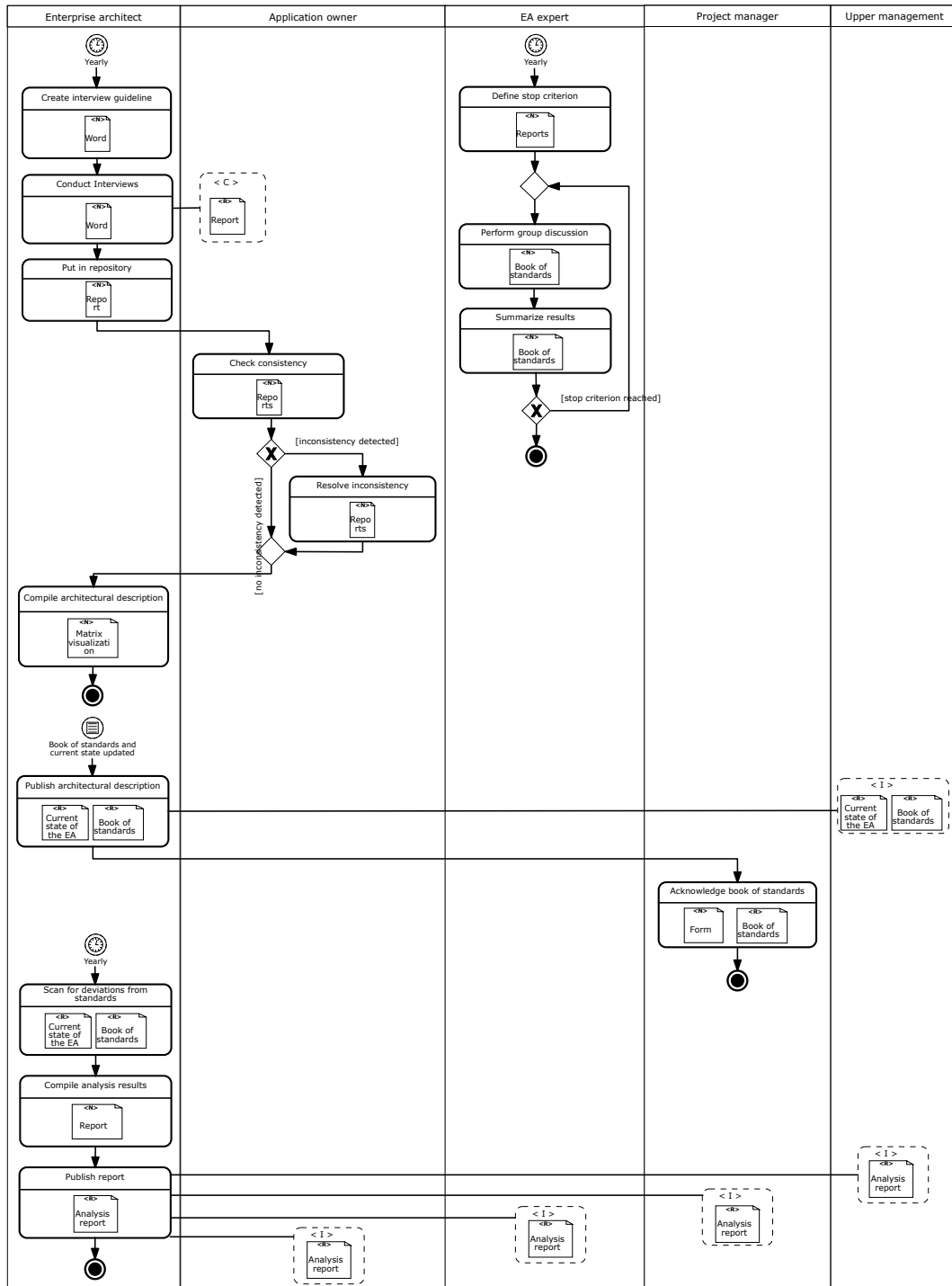


Figure 7.4.: Case study PS: excerpt from the method description

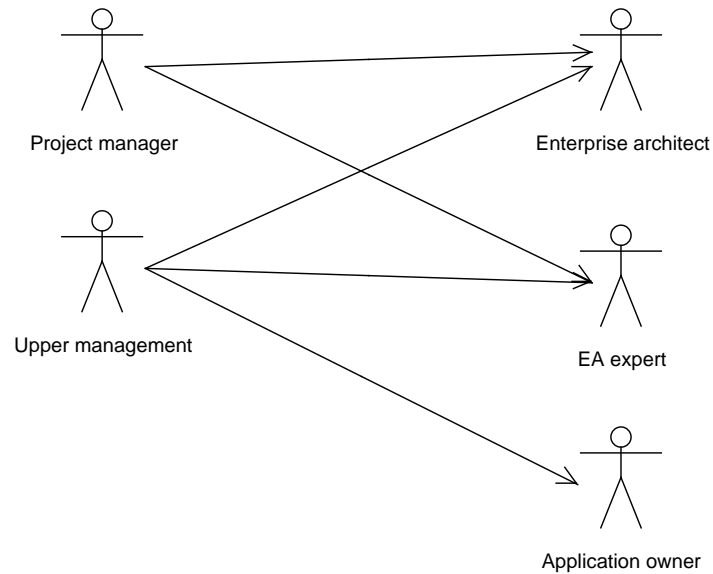


Figure 7.5.: Case study PS: stakeholder-actor-dependencies

### 7.3.3. Findings from the case studies

After the completion of the student projects, we requested feedback from the members of the project teams. The students' feedback targets BEAMS from the 'perspective of consultants' that apply the development method. The feedback of the industry partners' representatives is provided from a 'customer perspective' and target the practicability as well as usability of the method's output.

The students in the first case study experienced difficulties in searching the method base because of its availability in printed form only. In response to this feedback, we compiled the prototypic tool support described in Chapter 6. The tool support and in particular the online available method base was used in the second case study. With the tool-support at hand, the student provided positive feedback on the usability and applicability of the development method. The graphical notation for describing the EA management function was showed to be beneficial. Nevertheless, the students mentioned the overall complexity of BEAMS, which requires familiarization. After a first phase the students from both case studies reported that they could flexibly respond to emerging or changed EA management-related problems of the industry partners.

The industry partners' feedback centers around the results of the development method. The students acted as consultants and hence the industry partners did not have to familiarize with the details of BEAMS. The industry partners in particularly stated the following three benefits from applying the BEAMS approach, namely

- the re-use of best practice knowledge during the design of the EA management function,
- a comprehensive description of the EA management function, and
- transparency with respect to the stakeholder-actor dependencies.

We have to remark that the provided feedback was gathered right after the projects were finished. Therefore, the feedback does not provide statements with respect to the actual performance of the developed EA management functions. The industry partners however challenged the proposed MBBs during the **configure EA management function** phase for practicability against the background of their personal experience.

### 7.4. Critical reflection of the evaluation

In this section we completed the final activity of our research method as presented in Section 2.3. The evaluation presented in this chapter is based on a threefold approach. The structure of this section reflects an evaluation

- against the general assertions, i.e., the characteristics of the EA management function,
- with respect to innovation by revisiting the state-of-the-art, and
- concerning utility by performing observational case studies.

In Section 7.1 the context, goal, solution, and general assertions are subjected to an argumentative evaluation process. Therefore, we discuss how an EA management function developed using BEAMS is a ‘good’ one with respect to the quality criteria from Section 4.1. Furthermore, we revisit the general design research guidelines as proposed by Hevner and showed that our research was conducted accordingly.

We showed that BEAMS represents an innovative artifact by revisiting the state-of-the-art in Section 7.2. Thereby, we used the analysis framework presented in Section 3.2 to classify our approach. Complementing, we discussed a fictitious case from literature and demonstrated that based on the described situation, the application of BEAMS results in a similar design of the EA management function as proposed in literature. Further, we showed that the development method can be used for integration with other EA management approaches along the example of the ADM of TOGAF.

In Section 7.3 we used case studies to complement the theoretic evaluation and demonstrate the utility of BEAMS. The two case studies performed from April 2010 to January 2011 apply the development method *ex post*, to improve an existing EA management function, and *ex ante* to establish an EA management function. Thus, the presented cases reflect the two different ways of evaluation in practice as discussed in our research method in Section 2.3. During the case studies, the method base was used

- for nexus-driven problem elicitation (**characterize situation**),
- for nexus-driven artifact design (**configure EA management function**), and
- for nexus-driven artifact evaluation.

The final part of the evaluation, the nexus-driven artifact evaluation was performed by challenging the design of the resulting EA management function with respect to practicability. Therefore, the industry partners were asked to provide a subjective feedback. A long-term evaluation could provide further insights into the suitability of the established management function. Such evaluation nevertheless requires a time frame of at least a year in order to

ensure execution of each part of the EA management function for at least two times. Due to the required time frame such an evaluation is not performed in the course of this thesis. A long term evaluation could contribute to the detection of anomalies in the design. Based on the procedure presented in Section 2.3 such anomalies would lead to an inspection and revision of the assertions in the BEAMS method base.





---

## Critical Reflection and Outlook

---

The expected benefit of EA management depends on the design of an organization-specific management function. ‘Organization-specificity’ in this thesis means that the EA management function is confined to the relevant parts of the EA, supplies appropriate methods for developing, enacting, and analyzing different states of the EA, and is configurable and adaptable to co-evolve with the organization. The EA management functions proposed by prevalent EA management approaches fail to provide full scale organization-specificity. Practitioners seeking to apply and adopt these approaches experience several difficulties in doing so. This thesis solves the aforementioned problem by presenting a systematic method to develop EA management functions based on practice-proven solutions. Recalling the research objective as stated in the introduction, we present the results of this thesis, provide a critical reflection on the findings, and conclude with suggestions for future research.

### 8.1. Summary of results

In Chapter 1 we discuss the experienced gap and defined the research objective of this thesis as follows

***Research objective:*** Develop a method for designing and re-designing organization-specific EA management functions based on best-practice knowledge documented in current EA management approaches.

This research objective was subdivided into five research questions (cf. Section 1.1). The five research questions guide the design and structure of this thesis. Subsequently, we summarize the contributions of the single chapters of this thesis and discuss the achieved results in answering the research questions.

In Chapter 2 we described the theoretic underpinnings of the research presented in this thesis. Along the framework of Becker and Niehaves [BN07] we discussed the epistemological, ontological, and linguistic positions that guided our research. Based on the work of Walls et al. [WWES92], Gregor and Jones [GJ07], as well as Schermann et al. [SBK07b], we developed a framework for design theories that accounts for the characteristics of the EA management domain. We further revisited the idea of a design theory nexus as proposed by Pries-Heje and Baskerville [PHB08] to combine competing design theories. We introduced our understanding of patterns as preliminary artifacts for developing design theories and presented our approach to pattern-based theory building. We finally provided an answer to research question 1 and presented the research method used in this thesis. The research method details how design theories can be developed in rigorous research in close interaction with an industry partner. Thereto, we delineate how design contributes to theory building and theory building justifies design by using the concept of patterns.

In Chapter 3 we explored the problem domain and existing knowledge base of prominent EA management approaches. Thereto, we introduced a common terminology to describe the problem and context domain based on the general model theory of Stachowiak [St73] and the ISO Standard 42010 [In07]. We further developed an analysis framework based on kernel theories from the problem domain. Thereto, we revisited the design perspective of Simon in [Si96], the management cycle of Deming [De82] as well as Shewhart [Sh86], the knowledge management of Probst [Pr98], and the viable system model of Beer [Be79, Be81, Be85]. Following the guidelines for literature reviews as proposed by Webster and Watson in [WW02], we revisited 15 prominent approaches to design EA management functions. The results of the state-of-the-art review supported the need for a systematic development method. Answering research question 2 we elicited three main activities of the EA management function and identified the organizational context and EA management-related problem as typical configuration aspects influencing the design of an EA management function.

In Chapter 4 we elicited requirements for the development of organization-specific EA management functions to answer research question 3. Thereto, we described quality criteria of a ‘good’ EA management function and requirements on the meta-level, i.e., regarding the development method and the method base from a utilitarian perspective. Further, we discussed contributing theories for the solution domain as method engineering (cf. [He93, Gu94, Br96]), situational method engineering (cf. Harmsen [Ha97]),  $i^*$  (cf. Yu [Yu96]), and our previous work on patterns for EA management [Bu07d, Bu08b, Er10]. In response to the identified requirements, we sketched the core principles of *building blocks for EA management solutions* (BEAMS). With BEAMS we augmented the conception of a design theory nexus instantiation for designing EA management functions.

In Chapter 5 we present the core contribution of this thesis and the central component of BEAMS: the development method to design organization-specific EA management functions using a method base. This method is designed to satisfy the corresponding requirements and to facilitate the creation of EA management functions that fulfill the described quality criteria. Driven by the requirement elicitation from the preceding section, we presented the development method to design organization-specific EA management functions. The development method consists of three main phases: characterize situation, configure EA management function, and

analyze EA management function. We delineated these phases in a stepwise manner. First, we introduced the concepts and techniques relevant for the respective phase to answer research question 4. Thereby, the following concepts deserve special attention, namely

- the **variable** concept to enable organization-specific configuration of MBBs,
- the concept of **pre-condition** and **post-condition** supports consistent integration of different MBBs,
- the **mixinMBB** concept permits interlinking of the EA management function with other enterprise-level management functions, and
- the concepts of **stakeholders** and **actors** that enable analyses concerning the organizational implementability of the resulting management function.

Secondly, we detailed the steps of the development method referencing the concepts and techniques introduced before. In addition, we introduced the administration method to maintain and evolve the method base. This method grounds on the idea of pattern-based theory building. With the development method and the administration method at hand, we answered the first part of research question 5.

Responding to the second part of our final research question 5, we discussed a prototypic tool support for BEAMS in Chapter 6. Thereto, we elicited use cases from the perspective of a user of the development method as well as from the perspective of an administrator maintaining the method base. Based on the required functionality, we revisited typical services provided by enterprise 2.0 tools as discussed by Büchner et al. in [BMN09] and showed that these services can be used to realize the BEAMS configurator. Therefore, we used existing open-source tools to answer research question 5. We delineate the design for our toolset and discuss a prototypic implementation based on the web-based content management system *tricia* [Bü07b, In10] and the open source modeling platform *Oryx* [DOW08a, DOW08b].

In Chapter 7 we evaluate BEAMS from different perspectives. Firstly, we assessed the quality of BEAMS from a theoretic perspective. We revisited the quality criteria for ‘good’ EA management functions, the requirements for our design method, and the general guidelines for design research as presented by Hevner in [He04] to assess the quality of BEAMS using argumentative evidence. Secondly, we assessed the innovation of our designed artifact by evaluating BEAMS against the state-of-the-art. Using the analysis framework from Section 3.2, we classified our contribution. Further, we used a fictitious case derived from the state-of-the-art literature to show that a skilled enterprise architect can re-develop an EA management function using BEAMS. Thirdly, we showed that the method can be applied in practice by presenting two case studies in which the development method was applied in cooperation with industry partners from the public sector and the financial industry.

## 8.2. Critical reflection and future research

BEAMS is an innovative framework that supports the design of organization-specific EA management functions based on practice-proven-solutions. BEAMS provides methods and techniques to ensure organizational implementability of the EA management function. In

addition, we demonstrated the usability of BEAMS by applying the development method in practice. In summary, we identify the following three main contributions that represent the outcome of our research:

- the **development method** to design organization-specific EA management functions,
- a **method base** of practice proven method building blocks, and
- the **BEAMS configurator** a prototypic toolset supporting BEAMS.

These contributions provide an innovation in the field of EA management and lay the groundwork for further research in the field. In the following we discuss potential future research topics based on BEAMS.

The development method of BEAMS is influenced by the contributing theory of situational method engineering as presented in Section 4.2.2. Further related disciplines that could contribute to more sophisticated techniques to the development method are organizational theory, product line engineering, or general development models. Especially the step of analyzing the resulting EA management function can be enhanced to incorporate further techniques. In addition, further applications of the development method in practice are of interest. Long-term observations of organizations that applied the development method could provide further information on the applicability of the development method as well as the adaptability of the designed EA management function. Thus a larger set of case studies would open a research field in which organization-specific configurations of different organizations could be investigated for e.g. specificities of industry sectors, needs for changes in the method building blocks, or emerging new building blocks.

The current version of the method base incorporates the best practices as described in the EA management pattern catalog, as found in literature, and the practice-proven solutions gathered during a project with over 30 industry partners. Future research topics concerned with the method base center around its evolution and maintenance. A topic for which currently BEAMS lacks support are “questions”, i.e., methods supporting the quantitative assessment of architectures or simulation with respect to planned and target EAs (cf. Section 7.2). Different approaches to perform such assessments have been proposed (cf. Johnson and Eksted [JE07] and our approach in [Bu08e]) but are currently not reflected in BEAMS. First steps towards a future evolution of the method base are already performed by establishing an online community of practitioners around the method base. In addition researchers can be invited to discuss, enhance, and maintain the currently available method base.

Related with the above discussion on support for a BEAMS community is the tool support. In this thesis we discussed a prototypic design of the BEAMS configurator, which leaves open space for future research. Enterprise architects as intended users of the development method would benefit from a wizard-based support for the execution of the development method and the application of the techniques. Further an automated support for notifications on new method building blocks as well as the possibility to share experiences in applying method building blocks are potential future contributions to the BEAMS configurator. In addition, further export functionalities of the designed EA management function to dedicated EA management tools could enhance the benefit of BEAMS especially from a practitioners’ perspective.

## A.1. Method building blocks

In the following exemplary MBBs from BEAMS are presented. To access the whole method base please visit <http://wwwmatthes.in.tum.de/wikis/beams/mbb> (cited 2011-02-14).

### Develop planned states of the EA

<b>Activity</b>	develop & describe
<b>Participating actors</b>	strategy board, enterprise architect
<b>States</b>	planned state
<b>Pre-condition</b>	concern.current.documented, project.documented
<b>Post-condition</b>	concern.planned.documented
<b>Organizational context</b>	
<b>applicable</b>	bottom-up initiatives, centralized IT department, federated IT department
<b>not applicable</b>	
<b>Trigger</b>	yearly, half yearly
<b>ELMF</b>	

Table A.1.: MBB develop planned states of the EA

Besides a top-down approach planned states can be derived from ongoing projects and project proposals. The amount of proposals from business and IT as well as the required time and

budget for realizing these projects typically outranges the available resources. Therefore, organizations have to decide which projects best align with their intended target state and architectural principles. To get an overview on the projects' impact and effect on the overall architecture, the ENTERPRISE ARCHITECT derives planned states based on the available projects. These projects have been prioritized in advanced by the STRATEGY BOARD members.

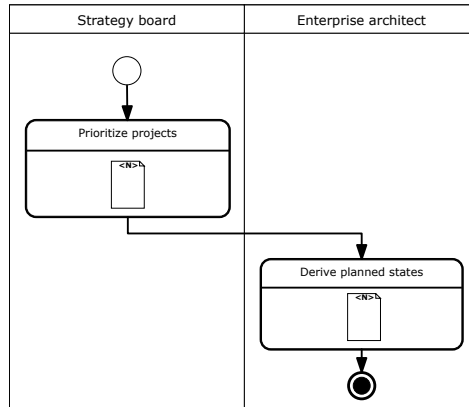


Figure A.1.: MBB develop planned states of the EA

## Force

- “Single” vs. “multiple criteria”: The project proposals can be evaluated and prioritized according to one or more criteria. While taking multiple criteria into account, e.g. expected return on investment, strategic impact, or risks, allows a more thoroughly decision, it is time-consuming and may result in a complex evaluation and decision process which lacks transparency and traceability. By contrast, a single criterion may not be sufficient to cover all aspects that should be considered in the decision process.
- “Run” vs. “change project prioritization”: Projects can be distinguished in run and change projects. A run project assures the operation continuity, e.g. upgrading software due to its end of lifetime or changed legal regulations (‘run the business’). A change project might be initiated by the demand for a new and innovative capability (‘change the business’). If resources and budget are limited, it is common to decide during the prioritization task that run projects are realized, while change projects are postponed. In that case, change projects won’t be executed and the organization won’t be able to react on external changes, e.g. IT innovations or new business opportunities.
- “Single” vs. “multiple scenarios”: The organization has to decide, if for a given point in time one planned state is sufficient, or if more than one scenario is needed. Having multiple scenarios enables the enterprise architect to present different possibilities for a planned state to the strategy board or to develop fallback strategies for risky projects. By contrast, developing just a single scenario requires less effort and enables to concentrate on a straight way to the planned state without distractions.

### **Consequences**

The use of this building block enables the ENTERPRISE ARCHITECT to focus more on actual business demands than on the abstract visions. Nevertheless, this approach bears the risk that the envisioned target state is not considered enough.

As a consequence, criterion or criteria to prioritize the proposed projects have to be defined. Therefore, the goals of the EA management endeavor and their operationalization to metrics might be used.

**Ensure information consistency**

<b>Activity</b>	develop & describe
<b>Participating actors</b>	consistency clerk
<b>States</b>	current state, planned state, target state
<b>Pre-condition</b>	concern.documented
<b>Post-condition</b>	concern.consistent
<b>Organizational context</b>	
<b>applicable</b>	
<b>not applicable</b>	
<b>Trigger</b>	event trigger
<b>ELMF</b>	

Table A.2.: MBB ensure consistency

This building block describes an optional part of the develop & describe activity which can be used to ensure consistency of the gathered information and therefore the architectural description developed. Thereby, the consistency of the architectural description is ensured by a CONSISTENCY CLERK. He or she analyzes whether the description complies to the underlying EA information model or violates constraints that are imposed by the model. Further, the CONSISTENCY CLERK tries to resolve inconsistencies or contradictions between the different fragments of the description, e.g. if one gathered information piece indicates that IT support is provided for a certain capability, while another one indicates that no IT support exists.

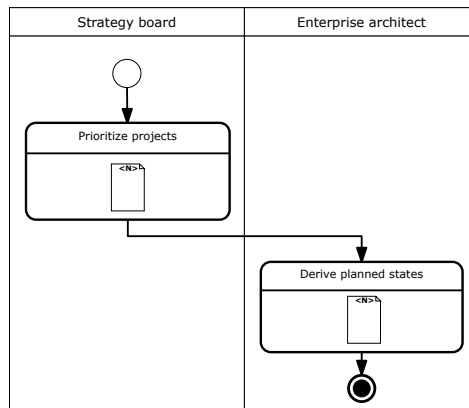


Figure A.2.: MBB ensure information consistency

**Force**

- “High quality” vs. “low costs”: In this respect, a CONSISTENCY CLERK with more information about the architecture is likely to discover more intricate inconsistencies as a clerk that stays to a basic checking. A more thorough and detailed checking however results in higher efforts, overall costs, and time delays regarding the completion of architectural descriptions.



- “Coverage” vs. “low costs”: While a consistency check on all information is sensible, a cost-benefit ratio should be taken into account. To decide on the question, if consistency checking should be applied to an area of interest or not, additionally the intended usage of the corresponding part should be taken into account. If the documentation provides input to expert-based analyses, the effort spent on consistency checking may well be reduced potentially to a point, where explicit checking is omitted. This can be justified with the implicit consistency checking that will take place, when experts discuss the architecture. The application of more formal analyses techniques, as e.g. metrics or rule-based analyses, in contrast calls for an explicit consistency checking especially with respect to the underlying assumptions of the formal analysis technique. This is necessary, as a formal technique might produce misleading or incorrect results, if applied to inconsistent information or architectural documentations, where the technique’s underlying assumptions do not hold.
- “Automated” vs. “manual consistency checking”: While automated consistency checking always ensures consistent and up-to-date information, it may hamper the maintenance of information as it complicates the information gathering process, e.g. certain information cannot be documented without updating other parts as well. Manual consistency checking at defined times may lead to inconsistent information at certain points in time, but contrastingly ensures that update of information is possible anytime.
- “Tool based” vs. “expert-based”: Consistency checking can either be performed by an expert or based on constraints incorporated into a tool for EA management. Especially, in the latter case, the consistency checking might be automated to a certain extend and therefore reduce overall costs and time expenditures.

## Consequences

Consistency checking is on the one hand an additional and time-consuming task that can on the other hand greatly improve the utility of the architectural description. To achieve an increased level of consistency, an experienced CONSISTENCY CLERK must perform checking activities on the description or incorporate constraints into a tool for EA management, which may automate the consistency checking to a certain extend.

### Publish architectural description

<b>Activity</b>	communicate & enact
<b>Participating actors</b>	enterprise architect, stakeholder
<b>States</b>	current state, planned state, target state
<b>Pre-condition</b>	concern.documented
<b>Post-condition</b>	concern.communicated
<b>Organizational context</b>	
<b>applicable</b>	specialized EA management tools, enterprise 2.0 tools (wikis), office tools
<b>not applicable</b>	
<b>Trigger</b>	Event trigger
<b>ELMF</b>	

Table A.3.: MBB Ensure Consistency

This building block contains the steps to make architectural descriptions and architectural principles available, i.e. if an architecture description is introduced or changed. The ENTERPRISE ARCHITECT has to publicize it and ensure that the architecture description is accessible, e.g. via mail or file share, for all relevant stakeholders.

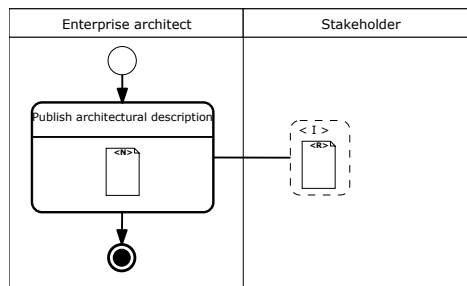


Figure A.3.: MBB Publish architectural Description

### Force

- “Pull” vs. “push”: Publishing architectural descriptions can either be performed using a pull or a push mechanism. While in the former case, the respective stakeholders have to search and look up the information, e.g. on a wiki-page, the intranet, or the EA management repository, in the latter case the stakeholders are actively informed, e.g. via e-mail, rss-feeds, or paper circuits.

## Officially gratify standard conformance

<b>Activity</b>	communicate & enact
<b>Participating actors</b>	enterprise architect, business executive
<b>States</b>	planned state
<b>Pre-condition</b>	concern.documented
<b>Post-condition</b>	
<b>Organizational context</b>	
<b>applicable</b>	culture with positive feedback
<b>not applicable</b>	
<b>Trigger</b>	yearly, half yearly
<b>ELMF</b>	

Table A.4.: MBB officially gratify standard conformance

To increase the overall conformance to architectural descriptions this building block can be used. An ENTERPRISE ARCHITECT, thereby, prepares gratifications, e.g. on the intranet homepage or at a company internal event, where the business departments with the highest degree of conformity are announced.

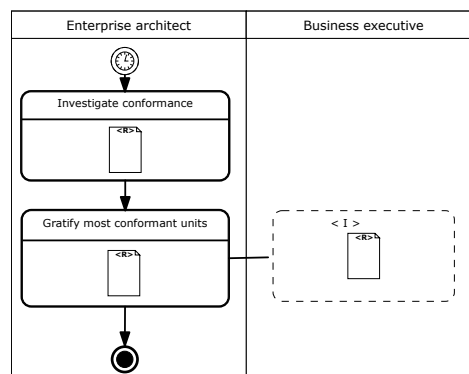


Figure A.4.: MBB Officially gratify standard conformance

## Force

- “Intranet” vs. “Event”: The gratification can be announce via the intranet systems as a news entry or on a dedicated site, but also on company-wide events where the conformant units are publicly announced.
- “Single” vs. “Multiple units”: There can either be only one unit declared as most conformant or a number of most conformant units.

### **Consequences**

While the application of this building block may lead to a ‘social competition’ between the business units, we also experienced a higher amount of error-prone data, i.e. units providing manipulated data.

## Perform single expert evaluation

<b>Activity</b>	analyze & evaluate
<b>Participating actors</b>	EA expert
<b>States</b>	current state, planned state, target state
<b>Pre-condition</b>	concern.documented
<b>Post-condition</b>	
<b>Organizational context</b>	
<b>applicable</b>	
<b>not applicable</b>	
<b>Trigger</b>	
<b>ELMF</b>	

Table A.5.: MBB perform single expert evaluation

This MBB describes an analysis technique where an EA EXPERT performs an individual analysis based on the provided input and her/his experiences.

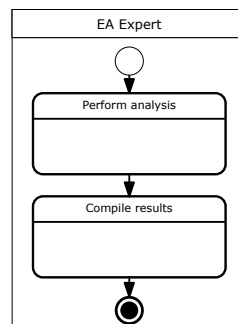


Figure A.5.: MBB Perform single expert evaluation

## Consequences

Expert-based analyses allow for a maximum flexibility with respect to the analyzed architectures. Thereby, especially the drawback of prerequisite assumptions of metrics-based analyses is remedied. In contrast, expert-based analysis are intrinsically subjective and may encounter acceptance problems, as the analyses are mostly not transparent. It is advisable to provide a rationale along every analysis result, shortly explaining the conclusions drawn by the expert to overcome the difficulty.

## Multi expert evaluation

<b>Activity</b>	analyze & evaluate
<b>Participating actors</b>	EA expert
<b>States</b>	current state, planned state, target state
<b>Pre-condition</b>	concern.documented
<b>Post-condition</b>	
<b>Organizational context</b>	
<b>applicable</b>	
<b>not applicable</b>	
<b>Trigger</b>	
<b>ELMF</b>	

Table A.6.: MBB multi expert evaluation

Following an expert-based approach, this MBB proposes a multi expert evaluation (similar to the Delphi method). The results are based on the experts' experience on architectural interrelationships. Thus, the MBB requires the involvement of multiple EA EXPERTS who discuss their estimation in a panel. The discussion includes multiple iterations. Each iteration starts with an anonymous summary of the experts opinions from the previous round as well as reasons they provided for their judgments. The experts are thereby encouraged to refine their answers in the light of the replies of the other members. A predefined stop criterion, e.g. the number of rounds or the achievement of consensus, terminates the process.

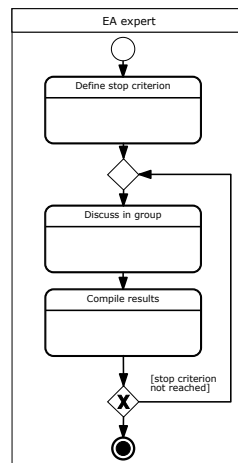


Figure A.6.: MBB Multi expert evaluation

### Force

- “Individual analysis results” vs. “aggregated analysis results”: Concerning the overall outcome of a group discussion, two different options can be taken into account. On the one hand, each EA analyst participating in the discussion can give his/her individual

analysis result, leading to an issue of aggregating the different results. On the other hand, the group can give one committed analysis statement as result. While the latter option helps to avoid potentially critical aggregation tasks, a consensus building in the analyst group may be time consuming and lead to overly mitigated outcomes, if strongly different opinions had been argued during the discussion.

### **Consequences**

Subjectivity of analyses can be partially avoided, if multiple experts are involved in performing analyses. Performing a group discussion, e.g. according to the Delphi method, may additionally help to prevent extreme positions and analysis results. Nevertheless, this method is not without subtle problems. Different social competencies might lead to degenerate discussions, where a ‘strong’ and eloquent EA EXPERT overrules his peers or argues them into a biased outcome.

## A.2. Catalog of organizational contexts

The **BEAMS catalog of organizational contexts** contains the following entries:

**Organizational culture** The EA management function has to account for ‘political’ subtleties that ground in the organizational culture. The organizational culture relates to the attitudes, experiences, and value of an organization determined by the people of the organization. Aspects that can be considered in specifying the organizational culture are:

**open culture** In an open culture disseminating information on the EA and the respective transparency is perceived positively. Similarly, change is interpreted to be positive. Therefore, results of the EA management function, i.e. EA-related artifacts or parts thereof, should be made visible immediately for a wide range of people. In this vein, visibility of the EA management initiative and transparency of the achieved results, e.g. different architectural states, are given to foster common understanding, to support the decision making by facilitating consensus, and to enable traceability of decisions.

**‘political’ culture** In an organization with a highly political environment, dedicated methods can be applied to ensure confidentiality of sensitive information, i.e. future plans of the EA. In such environments transparency besides its benefits can lead to political resistance as e.g. people loose responsibilities and therefore influence. Furthermore, change is typically negatively associated.

**culture of negative feedback** In organizations with a culture of negative feedback, typically a strong mode of enactment or enforcement is used. This enactment is performed via control and if necessary disciplines and punishments.

**culture of positive feedback** In organizations with a culture of positive feedback incentives are used to motivate people. The culture of positive feedback typically represents a weak mode of enactment as people can decide freely.

**Tool support** Today’s organizations are complex and highly interconnected systems whose EA is described using hundreds or even thousands of elements, e.g. applications, processes, organizational units, etc. Therefore, tool support is required to manage the overall complexity. Different ways of providing tool support for the area of EA management exist, which may all have their dedicated influences on the design of an EA management function:

**EA management tools** A plethora of different tools specialized for the area of EA management exists [se05, Ma08]. While these tool differ widely regarding the predefined information model, capabilities to specify and execute workflows, as well as the viewpoints provided, they enable decentralized information gathering and typically support a light-weight access via web interfaces. Nevertheless, the use of specialized EA management tools typically requires training of the intended users as they cannot be used without prior familiarization. Furthermore, EA management tools require a high investment.



**Enterprise 2.0 tools (wikis)** Enterprise 2.0 tools, in particular wikis, can be used as central repository for EA-related information. Wikis are easy to use and provide access via an web-interface. In addition, they employ techniques to foster collaborative work, e.g. versioning, roles and rights management, as well as concurrent editing. In contrast, wikis usually store only unstructured information and do not provide capabilities to generate visualizations from content. A multitude of open-source wikis which require only investments in terms of installation, configuration, and maintenance exist.

**Office tools** Frequently used in the context of EA management are tools from the office family, especially spreadsheet applications and presentation programs. Although these tools do neither offer access control nor collaboration support, they benefit from the fact that employees are typically familiar with these tools and that no initial investments have to be made as they are usually already available in the organizations.

**Background of the initiative** Different reasons and backgrounds while an EA management endeavor is initiated exists, namely

**bottom-up initiatives** Bottom-up initiatives are characterized by a group of ‘heroes’ representing the drivers of EA management. In organizations with bottom-up initiatives no dedicated mission and thus support from the upper management exists. Therefore, an bottom-up initiative is characterized by concealing the initiative behind project or program names, no dedicated budget for the initiative, and a limitation in scope. Bottom-up initiatives typically strongly depend on the social and professional skills of the ‘heroes’.

**top-down initiatives** Top-down initiatives are initiated by the upper management, which passes a board decision regarding the context of EA management and decides on the establishment of dedicated roles and responsibilities. Furthermore, top-down initiatives are characterized by a dedicated budget for the initiative, the establishment of control mechanisms, and the possibility to employ enforcement mechanisms.

**pilot initiatives** Pilot initiatives are typically set up, if doubts and disbelieves in the benefits of EA management exist. Pilots are typically carried out hand in hand with bigger transformation projects and are characterized by a defined point in time where a decision about the continuation of the project is made, a limited scope and reach in terms of goals pursued, and the absence of a dedicated tool support.

**Organization of the IT department** Modern organizations have grown historically and exhibit different ways of how the IT department as a cross-function delivering capabilities used throughout the enterprise can be organized:

**centralized IT department** Organizations that employ a centralized IT department have a defined entry point for all IT-related demands. In this vein, responsibilities for solution delivery, developing, and implementing IT for all business agencies is controlled by a central authority. While this organization is economical from the standpoint of skills and overhead, the flexibility to deliver customized solutions to fit specialized business agencies demands and the ability to foster business knowledge in the IT staff is limited.

**decentralized IT department** In organizations with decentralized IT departments, solution delivery is aligned with the line of business and the IT managers report to the business agency directors. In this vein, coordination between the different departments is aggravated which may result in heterogeneous solutions. In contrast, the specialized needs of the business agencies are addressed and the IT staff is controlled by the business agency which fosters business knowledge in the IT staff.

**federated IT department** Organizations that employ a federated IT agency possess an organization-wide IT unit (at the CIO's office) which has primary responsibility for the evolution of the EA. In addition, each business agency has an associated IT department. The directors of the federated IT departments report to the business agency directors and the organization-wide IT unit. While benefitting from the balance between effectively aligning the IT with the individual needs of the business departments and the organization-wide view, the federated organization challenges with the complexity of coordination, the high administrative costs, and the problem of dual reporting.

### A.3. Catalog of participants

Two exemplary participant descriptions from the BEAMS catalog of participants are given in the following.

**Business executives** Business executives are persons responsible for running an organization or a business department of an organization. Business executives are responsible for managing their business department in an effective and efficient way. This includes the identification of critical dependencies between business and IT as well as assessing the current performance and identifying potentials for improvement. Typical questions a business architect is concerned with are

- How can the supported IT functionality be analyzed?
- Which future scenarios are supported by current IT solutions?
- How can the quality of IT support be improved?
- How can the current performance of the department be improved?
- How are decisions supported by the EA management function?
- What is required to comply with EA management decision making?

**Enterprise architect** Enterprise architects manage and lead the EA program, i.e. are the persons responsible for developing, communicating, and analyzing architectural states of the EA. They are in charge of the EA management function in terms of designing, implementing, and adapting the corresponding tasks. Furthermore, they oversee the EA budget (if available), support the introduction and establishment of EA management-related tasks, and increase the use of architectural descriptions and tools by convincing other members of the organization of the benefits of EA management. Typical questions an enterprise architect is concerned with are

- How to promote the EA management function and its benefits?
- How to foster and improve communication between business and IT?
- How to best address the organization-specific problems from an organization-wide perspective?
- How to balance local and global interests?
- How can potentials for improvements in the EA be identified?
- How should the EA evolve to best fit the objectives and strategies of the organization?
- How to identify stakeholders of the EA management function?

## APPENDIX B

---

A case study from the public sector

---

### **B.1. Selected and configured MBBs**

<b>MBB</b>	<b>Organizational contexts (applicable in)</b>	<b>Organizational contexts (not applicable in)</b>	<b>States</b>
<b>Describe by interview</b>	bottom-up, office tools, pilot initiative		current state, target state
<b>Ensure information consistency</b>		specialized EA management tools	current state, planned state
<b>Multi expert evaluation</b>			current state, planned state, target state
<b>Request acknowledgment for architecture description</b>	political culture		current state, planned state, target state
<b>Pattern-based analysis</b>			current state, planned state, target state
<b>Publish architectural description</b>	Specialized EA management tools, enterprise 2.0 tools, office tools		current state, planned state, target state
<b>MBB</b>	<b>ELMFs</b>	<b>Pre-conditions</b>	<b>Post-conditions</b>
<b>Describe by interview</b>			concern.documented
<b>Ensure information consistency</b>	EA management function	concern.documented	concern.consistent
<b>Multi expert evaluation</b>		concern.documented	goal.documented
<b>Request acknowledgment for architecture description</b>		concern.documented	concern.approved
<b>Pattern-based analysis</b>		concern.documented	goal.documented
<b>Publish architectural description</b>		concern.documented	concern.published

Table B.1.: The selected MBBs for the case study from the public sector



- [Ab07] Abdulhadi, S.: *i\* Guide*. [http://istar.rwth-aachen.de/tiki-index.php?page\\_ref\\_id=67](http://istar.rwth-aachen.de/tiki-index.php?page_ref_id=67) (cited 2011-02-14). 2007.
- [ACB89] Alavi, M.; Carlson, P.; Brooke, G.: *The ecology of MIS research: a twenty year status review*. In *ICIS '89: Proceedings of the tenth international conference on Information Systems*. pages 363–375. New York, NY, USA. 1989. ACM.
- [Ad06] Adobe Systems Incorporated: *PDF Reference – sixth edition: Adobe® Portable Document Format, Version 1.7, November 2006*. 2006.
- [Åg07] Ågerfalk, P. J.; Brinkkemper, S.; Gonzalez-Perez, C.; Henderson-Sellers, B.; Karlsson, F.; Kelly, S.; Ralyté, J.: *Modularization Constructs in Method Engineering: Towards Common Ground?* In (Ralyté, J.; Brinkkemper, S.; Henderson-Sellers, B., Ed.): *Situational Method Engineering*. pages 359–368. Springer. 2007.
- [AG10] Aier, S.; Gleichauf, B.: *Application of Enterprise Models for Engineering Enterprise Transformation*. *Enterprise Modelling and Information System Architectures*. 5:56–72. 2010.
- [Ai08a] Aier, S.; Kurpjuweit, S.; Riege, C.; Saat, J.: *Stakeholderorientierte Dokumentation und Analyse der Unternehmensarchitektur*. In (Hegering, H.-G.; Lehmann, A.; Ohlbach, H. J.; Scheideler, C., Ed.): *GI Jahrestagung (2)*. volume 134 of *LNI*. pages 559–565. Bonn, Germany. 2008. Gesellschaft für Informatik.
- [Ai08b] Aier, S.; Kurpjuweit, S.; Schmitz, O.; Schulz, J.; Thomas, A.; Winter, R.: *An Engineering Approach to Enterprise Architecture Design and its Application at a Financial Service Provider*. In (Loos, P.; Nüttgens, M.; Turowski, K.; Werth, D., Ed.): *Modellierung betrieblicher Informationssysteme (MobIS 2008)*. pages 115–130. 2008.
- [Ai09a] Aier, S.; Buckl, S.; Franke, U.; Gleichauf, B.; Johnson, P.; Närman, P.; Schweda, C. M. et al.: *A Survival Analysis of Application Life Spans based on Enterprise Architecture Models*. In *3<sup>rd</sup> International Workshop on Enterprise Modelling and Information Systems Architectures*. pages 141–154. Ulm, Germany. 2009.

- [Ai09b] Aier, S.; Kurpjuweit, S.; Saat, J.; Winter, R.: *Business Engineering Navigator – A "Business to IT" Approach to Enterprise Architecture Management*. In (Bernard, S.; Doucet, G.; Götze, J.; Saha, P., Ed.): *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance*. pages 77–98. Bloomington. 2009. Author House.
- [Ai09c] Aier, S.; Kurpjuweit, S.; Saat, J.; Winter, R.: *Enterprise Architecture Design as an Engineering Discipline*. *AIS Transactions on Enterprise Systems*. 1:36–43. 2009.
- [AK99] Applegate, L. M.; King, J. L.: *Rigor and Relevance – Careers on the Line*. *MIS Quarterly*. 23(1):17–28. 1999.
- [Ak04] van Aken, J. E.: *Management Research Based on the Paradigm of the Design Sciences: The Quest for Field-Tested and Grounded Technological Rules*. *Journal of Management Studies*. 41(2):219–246. 2004.
- [Ak05] van Aken, J. E.: *Management Research as a Design Science: Articulating the Research Products of Mode 2 Knowledge Production in Management*. *British Journal of Management*. 16(1):19–36. 2005.
- [Al72] Alexander, C.: *Notes on the Synthesis of Form*. Harvard University Press. Cambridge, MA, USA. 1972.
- [Al77] Alexander, C.; Ishikawa, S.; Silverstein, M.; Jacobson, M.; Fiksdahl-King, I.; Angel, S.: *A Pattern Language*. Oxford University Press. New York, NY, USA. 1977.
- [Al79a] Albrecht, A. J.: *Measuring Application Development Productivity*. In *Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium*. pages 83–92. Armonk, NY, USA. October 1979. IBM Cooperation.
- [Al79b] Alexander, C.: *The Timeless Way of Building*. Oxford University Press. New York. 1979.
- [Am98] Ambler, S. W.: *Process Patterns: Building Large-Scale Systems Using Object Technology*. SIGS Press. New York, NY, USA. 1998.
- [AMT08] Aveiro, D.; Mendes, J. a.; Tribolet, J.: *Organizational modeling with a semantic wiki*. In *Proceedings of the 2008 ACM symposium on Applied computing*. SAC '08. pages 592–593. New York, NY, USA. 2008. ACM.
- [Ar07] Arbab, F.; Boer, F. S. d.; Bonsangue, M. M.; Lankhorst, M. M.; Proper, E. H.; Torre, L. W. N. v. d.: *Integrating Architectural Models - Symbolic, Semantic and Subjective Models in Enterprise Architecture*. *Enterprise Modelling and Information Systems Architectures*. 2(1):40–57. 2007.
- [ARW08a] Aier, S.; Riege, C.; Winter, R.: *Classification of Enterprise Architecture Scenarios – An Exploratory Analysis*. *Enterprise Modelling and Information Systems Architectures*. 3:14–23. 2008.
- [ARW08b] Aier, S.; Riege, C.; Winter, R.: *Unternehmensarchitektur – Literaturüberblick Stand der Praxis*. *Wirtschaftsinformatik*. 50(4):292–304. 2008.



- 
- [AS07] Aier, S.; Schönherr, M.: *Integrating an Enterprise Architecture Using Domain Clustering*. In (Lankhorst, M. M.; Johnson, P., Ed.): *Second Workshop on Trends in Enterprise Architecture Research*. pages 23–30. 2007.
- [AS09] Aier, S.; Schelp, J.: *A Reassessment of Enterprise Architecture Implementation*. In *Trends in Enterprise Architecture Research*. pages 53–68. 2009.
- [AST10a] Aveiro, D.; Silva, A. R.; Tribolet, J. M.: *Extending the Design and Engineering Methodology for Organizations with the Generation Operationalization and Discontinuation Organization*. In *DESRIST 2010*. pages 226–241. 2010.
- [AST10b] Aveiro, D.; Silva, A. R.; Tribolet, J. M.: *Towards a GOD-theory for organizational engineering: continuously modeling the continuous (re)generation, operation and deletion of the enterprise*. In *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing*. pages 150–157. New York, NY, USA. 2010. ACM.
- [Av10] Aveiro, D.: *G.O.D. (Generation, Operationalization & Discontinuation) and Control (sub)organizations: a DEMO-based approach for continuous real-time management of organizational change caused by exceptions*. PhD thesis. Instituto Superior Técnico, Universidade Técnica de Lisboa. Lisbon, PT. 2010.
- [AW09] Aier, S.; Winter, R.: *Virtual Decoupling for IT/Business Alignment – Conceptual Foundations, Architecture Design and Implementation Example*. *Business & Information Systems Engineering*. 1(2):150–163. 2009.
- [AWH91] Avison, D.; Wood-Harper, A.: *Information systems development research: an exploration of ideas in practice*. *Comput. J.* 34:98–112. 1991.
- [Ba90] Barley, S. R.: *The Alignment of Technology and Structure through Roles and Networks*. *Administrative Science Quarterly*. 35(1):61–103. 1990.
- [Ba96] Baumgarten, B.: *Petri-Netze. Grundlagen und Anwendungen*. Spektrum Akademischer Verlag. 2<sup>nd</sup> edition. 1996. 3827401755.
- [BB03] Bennet, A.; Bennet, D.: *The partnership between Organizational Learning and Knowledge Management*. In (Holsapple, C. W., Ed.): *Handbook on knowledge management 1. Knowledge Matters*. pages 439–455. Berlin, Germany. 2003. Springer.
- [BB04] Brocke, J. v.; Buddendick, C.: *Konstruktionstechniken für die Referenzmodellierung – Systematisierung, Sprachgestaltung und Werkzeugunterstützung*. In (Becker, J.; Delfmann, P., Ed.): *Referenzmodellierung – Grundlagen, Techniken und domänenbezogene Anwendung*. pages 19–50. Heidelberg, Germany. 2004. Physica-Verlag.
- [BC96] Brocklesby, J.; Cummings, S.: *Designing a Viable Organization Structure*. *Long Range Planning*. 29(1):49–57. 1996.
- [BCR94] Basili, V. R.; Caldiera, G.; Rombach, H. D.: *The Goal Question Metric Approach*. Wiley. New York. 1994.
- [BDK07] Becker, J.; Delfmann, P.; Knackstedt, R.: *Adaptive Reference Modeling. Integrating Configurative and Generic Adaptation Techniques for Information Models*. In

- (Becker, J.; Delfmann, P., Ed.): *Reference Modeling – Efficient Information Systems Design Through Reuse of Information Models*. pages 23–49. Berlin, Germany. 2007. Physica.
- [Be79] Beer, S.: *The Heart of Enterprise*. John Wiley. New York, NY, USA. 1979.
- [Be81] Beer, S.: *Brain of the Firm*. John Wiley. New York, NY, USA. 2<sup>nd</sup> edition. 1981.
- [Be85] Beer, S.: *Diagnosing The System for Organisations*. John Wiley. New York, NY, USA. 1985.
- [Be95] Bem, D. J.: *Writing a Review Article for Psychological Bulletin*. *Psychological Bulletin*. 118(2):172–177. 1995.
- [Be03] Becker, J.; Holten, R.; Knackstedt, R.; Niehaves, B.: *Forschungsmethodische Positionierung in der Wirtschaftsinformatik – epistemologische, ontologische und linguistische Leitfragen* –. Technical report. Münster University, Germany. 2003.
- [Be05] Bernard, S. A.: *An Introduction to Enterprise Architecture*. Authorhouse. Bloomington, USA. 2<sup>nd</sup> edition. 2005.
- [Be09a] Becker, J.; Niehaves, B.; Olbrich, S.; Pfeiffer, D.: *Forschungsmethodik einer Integrationsdisziplin – Eine Fortführung und Ergänzung zu Lutz Heinrichs "Beitrag zur Geschichte der Wirtschaftsinformatik" aus gestaltungsorientierter Perspektive*. In (Becker, J.; Krcmar, H.; Niehaves, B., Ed.): *Wissenschaftstheorie und gestaltungsorientierte Wirtschaftsinformatik*. pages 1–22. Heidelberg, Germany. 2009. Physica-Verlag.
- [Be09b] Bender, G.: *Designing a Stakeholder-Specific Enterprise Architecture Management based on Patterns*. Master thesis. Fakultät für Informatik, Technische Universität München. 2009.
- [Be10] Becker, J.: *Prozess der Gestaltungsorientierten Wirtschaftsinformatik*. In (Österle, H.; Winter, R.; Brenner, W., Ed.): *Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz*. pages 13–17. St. Gallen, Switzerland. 2010. infowerk ag.
- [BES10a] Buckl, S.; Ernst, A.; Schweda, C. M.: *Enterprise Architecture Management – Chancen und Herausforderungen*. *ERP-Management*. 1:34–39. 2010.
- [BES10b] Buckl, S.; Ernst, A. M.; Schweda, C. M.: *Teaching Enterprise Architecture Management with student mini-projects*. In (Engels, G.; Luckey, M.; Pretschner, A.; Reussner, R., Ed.): *2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010)*. pages 309–320. Paderborn, Germany. 2010.
- [BG09] Bernard, S. A.; Grasso, J.: *Enterprise Architecture Formalization and Auditing*. In (Doucet, G.; Götze, J.; Saha, P.; Bernard, S., Ed.): *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance*. Bloomington, USA. 2009. AuthorHouse.
- [BHS07] Buschmann, F.; Henney, K.; Schmidt, D. C.: *Pattern Oriented Software Architecture Volume 5: On Patterns and Pattern Languages*. John Wiley & Sons, Inc. Weinheim, Germany. 2007.

- 
- [Bi98] Bird, G. B.: *The business benefit of standards*. *StandardView*. 6:76–80. 1998.
- [BJ87] Brooks Jr., F. P.: *No Silver Bullet – Essence and Accidents of Software Engineering*. *Computer*. 20:10–19. April 1987.
- [BK04] Banker, R. D.; Kauffman, R. J.: *The Evolution of Research on Information Systems: A Fiftieth-Year Survey on Literature in Management Science*. *Management Science*. 50(3):281–298. 2004.
- [BKS10] Buckl, S.; Krell, S.; Schweda, C. M.: *A formal approach to Architectural Descriptions – Refining the ISO Standard 42010*. In (Aalst, W.; Mylopoulos, J.; Sadeh, N. M.; Shaw, M. J.; Szyperski, C.; Albani, A.; Dietz, J. L. G., Ed.): *Advances in Enterprise Engineering IV*. volume 49 of *Lecture Notes in Business Information Processing*. pages 77–91. Springer Berlin Heidelberg. 2010.
- [BM79] Burrell, G.; Morgan, G.: *Sociological Paradigms and Organizational Analysis*. Heinemann Educational Publishers. London, UK. 1979.
- [BM02] Baskerville, R.; Myers, M. D.: *Information Systems as a Reference Discipline*. *MIS Quarterly*. 26(1). 2002.
- [BM05] Bachmann, F.; Merson, P.: *Experience Using the Web-Based Tool Wiki for Architecture Documentation*. Technical report. Software Engineering Institute, Carnegie Mellon University. 2005.
- [BMN09] Büchner, T.; Matthes, F.; Neubert, C.: *A Concept and Service based Analysis of Commercial and Open Source Enterprise 2.0 Tools*. In (Fred, A.; Dietz, J.; Liu, K.; Filipe, J., Ed.): *KMIS*. pages 37–45. 2009.
- [BMN10] Büchner, T.; Matthes, F.; Neubert, C.: *Data Model Driven Implementation of Web Cooperation Systems with Tricia*. In (Dearle, A.; Zicari, R., Ed.): *Objects and Databases*. *Lecture Notes in Computer Science*. pages 70–84. Springer. Berlin, Heidelberg, Germany. 2010.
- [BMS09a] Buckl, S.; Matthes, F.; Schweda, C. M.: *Classifying Enterprise Architecture Analysis Approaches*. In (Poler, R.; van Sinderen, M.; Sanchis, R., Ed.): *Enterprise Interoperability – Second IFIP WG 5.8 International Workshop, IWEI 2009, Valencia, Spain, October 13-14, 2009. Proceedings*. pages 66–79. Berlin, Heidelberg, Germany. 2009. Springer.
- [BMS09b] Buckl, S.; Matthes, F.; Schweda, C. M.: *Future Research Topics in Enterprise Architecture Management – A Knowledge Management Perspective*. In (Aier, S.; Schelp, J.; Schönherr, M., Ed.): *4<sup>th</sup> Workshop on Trends in Enterprise Architecture Research (TEAR 2009)*. pages 5–20. 2009.
- [BMS09c] Buckl, S.; Matthes, F.; Schweda, C. M.: *A Viable System Perspective on Enterprise Architecture Management*. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11-14 October 2009*. pages 1483–1488. IEEE. 2009.
- [BMS10a] Buckl, S.; Matthes, F.; Schweda, C.: *A situated approach to enterprise architecture management*. In *IEEE International Conference on Systems, Man and Cybernetics (SMC 2010)*. pages 587–592. IEEE. 2010.

- [BMS10b] Buckl, S.; Matthes, F.; Schweda, C. M.: *Conceptual Models for Cross-Cutting Aspects in Enterprise Architecture Modeling*. In *14<sup>th</sup> IEEE International EDOC Conference Workshops (EDOCW)*. pages 245–252. 2010.
- [BMS10c] Buckl, S.; Matthes, F.; Schweda, C. M.: *A Design Theory Nexus for Situational Enterprise Architecture Management – Approach and Application Example*. In *14<sup>th</sup> IEEE International EDOC Conference (EDOC 2010)*. Vitoria, Brazil. 2010.
- [BMS10d] Buckl, S.; Matthes, F.; Schweda, C. M.: *From EA management patterns towards a prescriptive theory for designing enterprise-specific EA management functions – Outline of a research stream*. In (Schumann, M.; Kolbe, L. M.; Breitner, M. H.; Freirichs, A., Ed.): *Multikonferenz Wirtschaftsinformatik (MKWI2010)*. pages 67–78. Göttingen, Germany. 2010.
- [BMS10e] Buckl, S.; Matthes, F.; Schweda, C. M.: *Future Research Topic in Enterprise Architecture Management – A Knowledge Management Perspective*. *Journal of Enterprise Architecture (JEA)*. 6(3):16–27. 2010.
- [BMS10f] Buckl, S.; Matthes, F.; Schweda, C. M.: *Interrelating Concerns in EA Documentation – Towards a Conceptual Framework of Relationships*. In (Engels, G.; Luckey, M.; Pretschner, A.; Reussner, R., Ed.): *2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010)*. pages 243–252. Paderborn, Germany. 2010.
- [BMS10g] Buckl, S.; Matthes, F.; Schweda, C. M.: *A Meta-Language for EA Information Modeling – State-of-the-art and Requirements Elicitation*. In (Bider, I.; Halpin, T.; Krogstie, J.; Nurcan, S.; Proper, E.; Schmidt, R.; Ukor, R., Ed.): *Enterprise, Business-Process and Information Systems Modeling*. Lecture Notes in Business Information Systems. pages 169–181. Springer. 2010.
- [BMS10h] Buckl, S.; Matthes, F.; Schweda, C. M.: *A Modeling Language for Describing EA Management Methods*. In (Esswein, W.; Turowski, K.; Jührisch, M., Ed.): *Modellierung betrieblicher Informationssysteme (MobIS2010) – Modellgestütztes Management, 15.-17.September 2010, Dresden*. Bonn, Germany. 2010. Gesellschaft für Informatik (GI).
- [BMS10i] Buckl, S.; Matthes, F.; Schweda, C. M.: *A technique for Annotating EA Information Models*. In (Barjis, J., Ed.): *Enterprise and Organizational Modeling and Simulation: 6<sup>th</sup> International Workshop, EOMAS 2010 held at CAiSE 2010, Hammamet, Tunisia, June 2010 Selected Papers*. Lecture Notes in Business Information Systems. pages 113–127. Berlin, Heidelberg, Germany. 2010. Springer.
- [BMS10j] Buckl, S.; Matthes, F.; Schweda, C. M.: *Towards a Method Framework for Enterprise Architecture Management – A Literature Analysis from a Viable System Perspective*. In *5<sup>th</sup> International Workshop on Business/IT Alignment and Interoperability (BUSITAL 2010)*. 2010.
- [BMS10k] Buckl, S.; Matthes, F.; Schweda, C. M.: *Utilizing Patterns in Developing Design Theories*. In *2010 International Conference on Information Systems (ICIS 2010)*. 2010.

- [BMS10] Buckl, S.; Matthes, F.; Schweda, C. M.: *Vergangenheit, Gegenwart und Zukunft von EAM*. In (Keuntje, J. H.; Barkow, R., Ed.): *Enterprise Architecture Management in der Praxis – Wandel, Komplexität und IT-Kosten im Unternehmen beherrschen*. pages 377–415. Düsseldorf, Germany. 2010. Symposium.
- [BMS11] Buckl, S.; Matthes, F.; Schweda, C. M.: *A Method Base for Enterprise Architecture Management*. In (Ralyte, J.; Mirbel, I.; Deneckere, R., Ed.): *IFIP Working Conference on Method Engineering (ME 2011)*. Paris, France. 2011. Springer.
- [BN94] Bernus, P.; Nemes, L.: *A Framework to Define Generic Enterprise Reference Architecture and Methodology*. In *Proceedings of the International Conference on Automation, Robotics and Computer Vision (ICARCV'94), Singapore, November 10–12*. pages 88–92. 1994.
- [BN96] Bernus, P.; Nemes, L.: *A Framework to Define Generic Enterprise Reference Architecture and Methodology*. *Proceedings of the International Conference on Automation, Robotics and Computer Vision (ICARCV'94), Singapore, November 10–12*. 9:179–191. 1996.
- [BN07] Becker, J.; Niehaves, B.: *Epistemological perspectives on IS research: a framework for analysing and systematizing epistemological assumptions*. *Information Systems Journal*. 17:197–214. 2007.
- [BNS03] Bernus, P.; Nemes, L.; Schmidt, G.: *Handbook on Enterprise Architecture*. Springer. Berlin, Heidelberg, Germany. 2003.
- [Bo05] de Boer, F. S.; Bonsangue, M. M.; Jacob, J.; Stam, A.; van der Torre, L. W. N.: *Enterprise Architecture Analysis with XML*. In *38<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2005)*. volume 8. page 222b. Los Alamitos, CA, USA. 2005. IEEE Computer Society Press.
- [BP93] Britton, G. A.; Parker, J.: *An Explication of the Viable System Model for Project Management*. *Systems Practice*. 6(1):21–51. 1993.
- [Br96] Brinkkemper, S.: *Method engineering: engineering of information systems development methods and tools*. *Information and Software Technology*. 38(4):275–280. 1996.
- [Br97] Bradner, S.: *Key words for use in RFCs to Indicate Requirement Levels*. <http://www.ietf.org/rfc/rfc2119.txt> (last accessed May, 22<sup>nd</sup> 2009). 1997.
- [Br05] Braun, C.; Wortmann, F.; Hafner, M.; Winter, R.: *Method Construction – A Core Approach to Organizational Engineering*. In *Proceedings of the 2005 ACM symposium on Applied computing*. SAC '05. pages 1295–1299. New York, NY, USA. 2005. ACM.
- [Br07] Braun, C.: *Modellierung der Unternehmensarchitektur – Weiterentwicklung einer bestehenden Methode und deren Abbildung in einem Meta-Modellierungswerkzeug*. PhD thesis. Universität St.Gallen. 2007.
- [Br10] Bruls, W. A. G.; Steenbergen, M. v.; Foorthuis, R. M.; Boos, R.; Brinkkemper, S.: *Domain Architectures as an Instrument to Refine Enterprise Architecture*. *Communications of the Associations for Information Systems*. 27:517–540. 2010.

- [BRS95] Becker, J.; Rosemann, M.; Schütte, R.: *Grundsätze ordnungsmäßiger Modellierung*. *Wirtschaftsinformatik*. 37(5):435–445. 1995.
- [BRU00] Becker, J.; Rosemann, M.; von Uthmann, C.: *Guidelines of Business Process Modeling*. In (van der Aalst, W.; Desel, J.; Oberweis, A., Ed.): *Business Process Management*. volume 1806 of *Lecture notes in computer science*. pages 30–49. Springer. Berlin, Germany. 2000.
- [BS06] van Berg, M. d.; van Steenberghe, M.: *Building an Enterprise Architecture Practice – Tools, Tips, Best Practices, Ready-to-Use Insights*. Springer. Dordrecht, The Netherlands. 2006.
- [BS11] Buckl, S.; Schweda, C. M.: *On the state-of-the-art in EA management literature*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2011.
- [Bu96] Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P.; Stal, M.: *Pattern-oriented software architecture: a system of patterns*. John Wiley & Sons, Inc. New York, NY, USA. 1996.
- [Bu06] Bucher, T.; Fischer, R.; Kurpjuweit, S.; Winter, R.: *Analysis and Application Scenarios of Enterprise Architecture: An Exploratory Study*. In *10<sup>th</sup> IEEE International EDOC Conference (EDOC 2006)*. Washington, DC, USA. 2006. IEEE Computer Society.
- [Bu07a] Bucher, T.; Klesse, M.; Kurpjuweit, S.; Winter, R.: *Situational Method Engineering – On the Differentiation of “Context” and “Project Type”*. In (Ralyé, J.; Brinkkemper, S.; Henderson-Sellers, B., Ed.): *Situational Method Engineering: Fundamentals and Experiences. Proceedings of the IFIP WG 8.1 Working Conference, 12-14 September 2007, Geneva, Switzerland*. pages 33–48. Berlin, Germany. 2007. Springer.
- [Bü07b] Büchner, T.: *Introspektive modellgetriebene Softwareentwicklung*. PhD thesis. Fakultät für Informatik, Technische Universität München. 2007.
- [Bu07c] Buckl, S.; Ernst, A. M.; Lankes, J.; Matthes, F.; Schweda, C. M.; Wittenburg, A.: *Generating Visualizations of Enterprise Architectures using Model Transformation (extended version)*. *Enterprise Modelling and Information Systems Architectures – An International Journal*. 2(2):3–13. 2007.
- [Bu07d] Buckl, S.; Ernst, A. M.; Lankes, J.; Schneider, K.; Schweda, C. M.: *A pattern based Approach for constructing Enterprise Architecture Management Information Models*. In (Oberweis, A.; Weinhardt, C.; Gimpel, H.; Koschmider, A.; Pankratius, V.; Schnizler, Ed.): *Wirtschaftsinformatik 2007*. pages 145–162. Karlsruhe, Germany. 2007. Universitätsverlag Karlsruhe.
- [Bu07e] Buckl, S.; Ernst, A. M.; Lankes, J.; Schweda, C. M.; Wittenburg, A.: *Generating Visualizations of Enterprise Architectures using Model Transformation*. In (Reichert, M.; Strecker, S.; Turowski, K., Ed.): *2<sup>nd</sup> International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007)*. pages 33–46. Bonn, Germany. 2007. Gesellschaft für Informatik.

- 
- [Bu08a] Buckl, S.; Dierl, T.; Matthes, F.; Ramacher, R.; Schweda, C. M.: *Current and Future Tool Support for EA Management*. In (Steffens, U.; Addicks, J.; Streekmann, N., Ed.): *MDD, SOA und IT-Management (MSI 2008)*. Berlin, Germany. 2008. GITO-Verlag.
- [Bu08b] Buckl, S.; Ernst, A. M.; Lankes, J.; Matthes, F.: *Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008)*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2008.
- [Bu08c] Buckl, S.; Ernst, A. M.; Lankes, J.; Matthes, F.; Schweda, C. M.: *Enterprise Architecture Management Patterns – Exemplifying the Approach*. In *12<sup>th</sup> IEEE International EDOC Conference (EDOC 2008)*. Munich, Germany. 2008. IEEE Computer Society.
- [Bu08d] Buckl, S.; Ernst, A. M.; Matthes, F.; Schweda, C. M.: *An Information Model for Landscape Management – Discussing Temporality Aspects*. In (Johnson, P.; Schelp, J.; Aier, S., Ed.): *Pre-Proceedings of the 3<sup>rd</sup> Workshop on Trends in Enterprise Architecture Research*. pages 63–77. Sydney, Australia. 2008.
- [Bu08e] Buckl, S.; Matthes, F.; Renz, W.; Schweda, C. M.; Sudeikat, J.: *Towards simulation-supported enterprise architecture management*. In (Loos, P.; Nüttgens, M.; Turowski, K.; Werth, D., Ed.): *Modellierung betrieblicher Informationssysteme (MobIS2008)*. pages 131–145. 2008.
- [Bu09a] Buckl, S.; Ernst, A. M.; Kopper, H.; Marliani, R.; Matthes, F.; Petschownik, P.; Schweda, C. M.: *EAM Pattern for Consolidations after Mergers*. In *SE 2009 – Workshopband*. pages 67–78. Kaiserslautern, Germany. 2009.
- [Bu09b] Buckl, S.; Ernst, A. M.; Lankes, J.; Matthes, F.; Schweda, C. M.: *State of the Art in Enterprise Architecture Management 2009*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2009.
- [Bu09c] Buckl, S.; Ernst, A. M.; Matthes, F.; Ramacher, R.; Schweda, C. M.: *Using Enterprise Architecture Management Patterns to complement TOGAF*. In *13<sup>th</sup> IEEE International EDOC Conference (EDOC 2009)*. Auckland, New Zealand. 2009. IEEE Computer Society.
- [Bu09d] Buckl, S.; Ernst, A. M.; Matthes, F.; Schweda, C. M.: *Enterprise Architecture Management Pattern for EA Visioning*. In *Proceedings of EuroPLOP 2009*. 2009.
- [Bu09e] Buckl, S.; Ernst, A. M.; Matthes, F.; Schweda, C. M.: *How to make your enterprise architecture management endeavor fail!* In *Pattern Languages of Programs 2009 (PLOP 2009), Chicago*. 2009.
- [Bu09f] Buckl, S.; Ernst, A. M.; Matthes, F.; Schweda, C. M.: *An Information Model for Managed Application Landscape Evolution*. *Journal of Enterprise Architecture (JEA)*. 5(1):12–26. 2009.
- [Bu09g] Buckl, S.; Franke, U.; Holschke, O.; Matthes, F.; Schweda, C. M.; Sommestad, T.; Ullberg, J.: *A Pattern-based Approach to Quantitative Enterprise Architecture Analysis*. In *15<sup>th</sup> Americas Conference on Information Systems (AMCIS)*. San Francisco, CA, USA. 2009.

- [Bu09h] Buckl, S.; Matthes, F.; Neubert, C.; Schweda, C. M.: *A Wiki-based Approach to Enterprise Architecture Documentation and Analysis*. In *The 17<sup>th</sup> European Conference on Information Systems (ECIS) – Information Systems in a Globalizing World: Challenges, Ethics and Practices, 8.–10. June 2009, Verona, Italy*. pages 2192–2204. Verona, Italy. 2009.
- [Bu09i] Buckl, S.; Matthes, F.; Schulz, C.; Schweda, C. M.: *Teaching Enterprise Architecture Management – A Practical Experience*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2009.
- [Bu10a] Buckl, S.; Dierl, T.; Matthes, F.; Schweda, C. M.: *Building Blocks for Enterprise Architecture Management Solutions*. In (Harmsen, F. e. a., Ed.): *Practice-Driven Research on Enterprise Transformation, second working conference, PRET 2010, Delft*. pages 17–46. Berlin, Heidelberg, Germany. 2010. Springer.
- [Bu10b] Buckl, S.; Matthes, F.; Roth, S.; Schulz, C.; Schweda, C. M.: *A Conceptual Framework for Enterprise Architecture Design*. In (Aalst, W.; Mylopoulos, J.; Sadeh, N. M.; Shaw, M. J.; Szyperski, C.; Proper, E.; Lankhorst, M. M. et al., Ed.): *Trends in Enterprise Architecture Research*. volume 70 of *Lecture Notes in Business Information Processing*. pages 44–56. Springer Berlin Heidelberg. 2010.
- [Bu10c] Buckl, S.; Matthes, F.; Schulz, C.; Schweda, C. M.: *Exemplifying a Framework for Interrelating Enterprise Architecture Concerns*. In (Sicilia, M.-A.; Kop, C.; Sartori, F., Ed.): *Ontology, Conceptualization and Epistemology for Information Systems, Software Engineering and Service Science: 4<sup>th</sup> International Workshop, ONTOSE 2010, held at CAiSE 2010 Hammamet, Tunisia, June 2010, Revised Selected Papers*. Lecture Notes in Business Information Systems. pages 33–46. Berlin, Heidelberg, Germany. 2010. Springer.
- [Bu11a] Buckl, S.; Dierl, T.; Matthes, F.; Schweda, C. M.: *Complementing The Open Group Architecture Framework with Best Practice Solution Building Blocks*. In *44<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2011)*. Kauai, Hawaii, USA. 2011.
- [Bu11b] Buckl, S.; Matthes, F.; Monahov, I.; Schweda, C. M.: *Modeling Enterprise Architecture Transformations*. In *3<sup>rd</sup> International IFIP WG5.8 Working Conference on Enterprise Interoperability (IWEI 2011)*. Stockholm, Sweden. 2011.
- [Bu11c] Buckl, S.; Matthes, F.; Neubert, C.; Schweda, C. M.: *A Lightweight Approach to Enterprise Architecture Modeling and Documentation*. In (Soffer, P.; Proper, E., Ed.): *Information Systems Evolution – CAiSE Forum 2010, Hammamet, Tunisia, June 2010, Selected Extended Papers*. pages 136–149. Berlin, Heidelberg, Germany. 2011. Springer.
- [BV03] Birkhölzer, T.; Vaupel, J.: *IT-Architekturen – Planung, Integration, Wartung*. VDE Verlag. Berlin, Germany. 2003.
- [BW96] Benbasat, I.; Weber, R.: *Research Commentary: Rethinking "Diversity" in Information System Research*. *Information Systems Research*. 7(4):389–399. 1996.
- [BW06] Balabko, P.; Wegmann, A.: *Systemic classification of concern-based design methods in the context of enterprise architecture*. *Information Systems Frontiers*. 8(2):115–131. 2006.



- 
- [BZ99] Benbasat, I.; Zmud, R. W.: *Empirical Research in Information Systems – The Practice of Relevance*. *MIS Quarterly*. 23(1):3–16. 1999.
- [Ca58] Carnap, R.: *Introduction to Symbolic Logic and Its Applications*. Dover Publications. 1<sup>st</sup> edition. 1958.
- [Ca07] Carlsson, S. A.: *Developing Knowledge Through IS Design Science Research: For Whom, What Type of Knowledge, and How?* *Scandinavian J. Inf. Systems*. 19(2):75 – 86. 2007.
- [CH04a] Chen, W.; Hirschheim, R.: *A paradigmatic and methodological examination of information systems research from 1991 to 2001*. *Information Systems Journal*. 14(2):197–235. 2004.
- [CH04b] Coplien, J. O.; Harrison, N. B.: *Organizational Patterns of Agile Software Development*. Prentice Hall PTR. Indianapolis, IN, USA. 2004.
- [Ch10] Chair for Informatics 19 (sebis), Technische Universität München: *EAM Pattern Catalog Wiki*. <http://eampc-wiki.systemcartography.info> (cited 2011-02-14). 2010.
- [Co96] Coplien, J. O.: *Software Patterns: Management Briefs*. Cambridge University Press. Cambridge, UK. 1996. 188484250X.
- [CST09] Caetano, A.; Silva, A. R.; Tribolet, J.: *A role-based enterprise architecture framework*. In *Proceedings of the 2009 ACM symposium on Applied Computing*. SAC '09. pages 253–258. New York, NY, USA. 2009. ACM.
- [CST10] Caetano, A.; Silva, A. R.; Tribolet, J.: *Business Process Decomposition – An Approach Based on the Principle of Separation of Concerns*. *Enterprise Modeling and Information Systems Architectures*. 5(1):44–58. 2010.
- [De82] Deming, E. W.: *Out of the Crisis*. Massachusetts Institute of Technology. Cambridge, MA, USA. 1982.
- [De08] DeMarco, T.; Hruschka, P.; Lister, T.; Robertson, S.; Robertson, J.; McMenamin, S.: *Adrenaline Junkies and Template Zombies – Understanding Patterns of Project Behavior*. Dorset House. New York, NY, USA. 2008.
- [De09a] Department of Defense (DoD) USA: *DoD Architecture Framework Version 2.0: Volume 1: Introduction, Overview, and Concepts – Manager’s Guide*. <http://www.defenselink.mil/cio-nii/docs/DoDAF%20V2%20-%20Volume%201.pdf> (cited 2011-02-14). 2009.
- [De09b] Department of Defense (DoD) USA: *DoD Architecture Framework Version 2.0: Volume 2: Architectural Data and Models – Architect’s Guide*. <http://www.defenselink.mil/cio-nii/docs/DoDAF%20V2%20-%20Volume%202.pdf> (cited 2011-02-14). 2009.
- [De09c] Department of Defense (DoD) USA: *DoD Architecture Framework Version 2.0: Volume 3: DoDAF Meta-model Physical Exchange Specification – Developer’s Guide*. <http://www.defenselink.mil/cio-nii/docs/DoDAF%20V2%20-%20Volume%203.pdf> (cited 2011-02-14). 2009.

- [Di05] Dietz, J. L.: *A World Ontology Specification Language*. In *On the Move to Meaningful Internet Systems 2005: OTM Workshops*. pages 688–699. 2005.
- [Di06] Dietz, J. L.: *Enterprise Ontology*. Springer. Heidelberg, Germany. 2006.
- [Di10] Dierl, T.: *Design and prototypic implementation of a web-based data and process model configurator*. Master’s thesis. Fakultät für Informatik, Technische Universität München. 2010.
- [Do08] Doucet, G.; Götze, J.; Saha, P.; Bernard, S. A.: *Coherency Management: Using Enterprise Architecture for Alignment, Agility, and Assurance*. *Journal on Enterprise Architecture*. 4(2). 2008.
- [Do09a] Doucet, G.; Götze, J.; Saha, P.; Bernard, S. A.: *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance*. AuthorHouse. Bloomington, USA. 2009.
- [Do09b] Doucet, G.; Götze, J.; Saha, P.; Bernard, S. A.: *Commencing the Journey: Realizing Coherency Management*. In (Doucet, G.; Götze, J.; Saha, P.; Bernard, S., Ed.): *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance*. Bloomington, USA. 2009. AuthorHouse.
- [Do09c] Doucet, G.; Götze, J.; Saha, P.; Bernard, S. A.: *Introduction to Coherency Management: The Transformation of Enterprise Architecture*. In (Doucet, G.; Götze, J.; Saha, P.; Bernard, S., Ed.): *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance*. Bloomington, USA. 2009. AuthorHouse.
- [DOW08a] Decker, G.; Overdick, H.; Weske, M.: *Oryx – An Open Modeling Platform for the BPM Community*. In (Dumas, M.; Reichert, M.; Shan, M.-C., Ed.): *Business Process Management, 6th International Conference, BPM 2008, Milan, Italy, September 2-4, 2008. Proceedings*. volume 5240 of *Lecture Notes in Computer Science*. pages 382–385. Springer. 2008.
- [DOW08b] Decker, G.; Overdick, H.; Weske, M.: *Oryx – Sharing Conceptual Models on the Web*. In *Demo Session of the 27<sup>th</sup> International Conference on Conceptual Modeling (ER)*. pages 536–537. Barcelona, Spain. 2008. LNCS 5231.
- [DP00] Davenport, T. H.; Prusak, L.: *Working Knowledge: how organizations manage what they know*. Harvard Business School Press. Boston, USA. 2<sup>nd</sup> edition. 2000.
- [Dr06] Drucker, P. F.: *The Practice of Management*. Harper Paperbacks. Oxford, UK. Reissue edition. 2006.
- [Du78] Dubin, R.: *Theory Building – revised edition*. Free Press. London, UK. 1978.
- [DW99] D’Souza, D. F.; Wills, A. C.: *Objects, components, and frameworks with UML: the catalysis approach*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA. 1999.
- [ED09] Ettema, R.; Dietz, J. L.: *ArchiMate and DEMO – Mates to Date?* In *Advances in Enterprise Engineering III*. pages 172–186. Albani, Antonia and Barjis, Joseph and Dietz, Jan L.G. 2009.

- 
- [EG07] Eisenhardt, K. M.; Graebner, M. E.: *Theory building from cases: Opportunities and challenges*. *Academy of Management Journal*. 50(1):25–32. 2007.
- [EH89] Espejo, R.; Harnden, R.: *The Viable System Model: Interpretations and Applications of Stafford Beer's VSM*. John Wiley. Chichester, UK. 1989.
- [Ek04] Ekstedt, M.; Johnson, P.; Lindström, Å.; Gammelgård, M.; Johansson, E.; Plazaola, L.; Silva, E. et al.: *Consistent Enterprise Software System Architecture for the CIO – A Utility-Cost Based Approach*. In *37<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2004)*. 2004.
- [Ek09] Ekstedt, M.; Franke, U.; Johnson, P.; Lagerström, R.; Sommestad, T.; Ullberg, J.; Buschle, M.: *A Tool for Enterprise Architecture Analysis of Maintainability*. In (Winter, A.; Ferenc, R.; Knodel, J., Ed.): *13th European Conference on Software Maintenance and Reengineering, CSMR 2009*. pages 327–328. Kaiserslautern, Germany. 2009.
- [Er06] Ernst, A. M.; Lankes, J.; Schweda, C. M.; Wittenburg, A.: *Using Model Transformation for Generating Visualizations from Repository Contents – An Application to Software Cartography*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2006.
- [Er08] Ernst, A. M.: *Enterprise Architecture Management Patterns*. In *PLoP 08: Proceedings of the Pattern Languages of Programs Conference 2008*. Nashville, USA. 2008.
- [Er10] Ernst, A. M.: *A Pattern-Based Approach to Enterprise Architecture Management*. PhD thesis. Technische Universität München. München, Germany. 2010.
- [Ex09] Executive Office of the President of the United States: *Improving Agency Performance Using Information and Information Technology – Enterprise Architecture Assessment Framework v3.1*. [http://www.whitehouse.gov/sites/default/files/omb/assets/fea\\_docs/OMB\\_EA\\_Assessment\\_Framework\\_v3\\_1\\_June\\_2009.pdf](http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/OMB_EA_Assessment_Framework_v3_1_June_2009.pdf) (last accessed 2010-11-11). 2009.
- [Fe94] Ferstl, O. K.; Sinz, E. J.; Amberg, M.; Hagemann, U.; Malischewski, C.: *Tool-Based Business Process Modeling Using the SOM Approach*. In *GI Jahrestagung*. pages 430–436. 1994.
- [Fe06] Fettke, P.: *State-of-the-Art des State-of-the-Art – Eine Untersuchung der Forschungsmethode “Review” innerhalb der Wirtschaftsinformatik*. *WIRTSCHAFTSINFORMATIK*. 48(74):257–266. 2006.
- [Fi08] Fischer, R.: *Organisation der Unternehmensarchitektur – Entwicklung der aufbau- und ablauforganisatorischen Strukturen unter besonderer Berücksichtigung des Gestaltungsziels Konsistenzerhaltung*. PhD thesis. Universität St.Gallen. 2008.
- [FL03] Fettke, P.; Loos, P.: *Multiperspective Evaluation of Reference Models—Towards a Framework*. In *Conceptual Modeling for Novel Application Domains*. volume 2814 of *Lecture Notes in Computer Science*. pages 80–91. Springer Berlin / Heidelberg. 2003.
- [Fo96] von Foerster, H.: *Wissen und Gewissen – Versuch einer Brücke*. Suhrkamp. Frankfurt a. M., Germany. 1996.

- [Fr94] Frank, U.: *MEMO: A Tool Supported Methodology for Analyzing and (Re-)Designing Business Information Systems*. In (Ege, R.; Singh, M.; Meyer, B., Ed.): *Technology of Object-Oriented Languages and Systems*. pages 367–380. 1994.
- [Fr98a] Frank, U.: *The MEMO Meta-Metamodel*. Technical Report 9. Arbeitsberichte des Instituts für Wirtschaftsinformatik Koblenz. 1998.
- [Fr98b] Frank, U.: *The MEMO Object Modelling Language (MEMO-OML)*. Technical Report 10. Arbeitsberichte des Instituts für Wirtschaftsinformatik Koblenz. 1998.
- [Fr99] Frank, U.: *Memo: Visual Languages for Enterprise Modelling*. Technical Report 18. Arbeitsberichte des Instituts für Wirtschaftsinformatik Koblenz. 1999.
- [Fr00] Freestone, R.: *Urban Planning in a Changing World: The Twentieth Century Experience (Studies in History, Planning, and the Environment)*. Spon Press. London, UK. 2000.
- [Fr02] Frank, U.: *Multi-perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages*. In *35<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2002)*. pages 1258–1267. Washington, DC, USA. 2002.
- [Fr03] Frank, U.: *Einige Gründe für die Wiederbelebung der Wissenschaftstheorie. Die Betriebswirtschaftslehre*. 63(3):278–292. 2003.
- [Fr08] Frank, U.; Heise, D.; Kattenstroth, H.; Schauer, H.: *Designing and Utilising Business Indicator Systems within Enterprise Models – Outline of a Method*. In *Modellierung betrieblicher Informationssysteme (MobIS 2008) – Modellierung zwischen SOA und Compliance Management 27.-28. November 2008*. Saarbrücken, Germany. 2008.
- [Fr09] Frank, U.: *The MEMO Meta Modelling Language (MML) and Language Architecture (ICB-Research Report)*. Technical report. Institut für Informatik und Wirtschaftsinformatik. Duisburg-Essen, Germany. 2009.
- [FRO03] Fitzgerald, B.; Russo, N. L.; O’Kane, T.: *Software development method tailoring at Motorola*. *Commun. ACM*. 46:64–70. April 2003.
- [FS95] Ferstl, O. K.; Sinz, E. J.: *Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen*. *Wirtschaftsinformatik*. 37(3):209–220. 1995.
- [FS97] Ferstl, O. K.; Sinz, E. J.: *Modeling of Business Systems Using the Semantic Object Model (SOM) – A Methodological Framework*. pages 339–358. Springer. 1997.
- [FS05] Ferstl, O. K.; Sinz, E. J.: *Modeling of Business Systems Using SOM*. In (P. Bernus et al., Ed.): *Handbook on Architectures of Information Systems*. Berlin, Germany. 2005. Springer.
- [FS08] Ferstl, O. K.; Sinz, E. J.: *SOM*. In (Kurbel, K.; Becker, J.; Gronau, N.; Sinz, E.; Suhl, L., Ed.): *Enzyklopädie der Wirtschaftsinformatik – Online Lexikon*. 2008.
- [Ga75] Gadamer, H.-G.: *Wahrheit und Methode – Grundzüge einer philosophischen Hermeneutik*. J.C.B. Mohr. Tübingen, Germany. 3<sup>rd</sup> edition. 1975.

- 
- [Ga94] Gamma, E.; Helm, R.; Johnson, R. E.; Vlissides, J. M.: *Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Professional Computing Series)*. Addison-Wesley Professional. Munich, Germany. 1994.
- [Ga07] Gallupe, R. B.: *The Tyranny of Methodologies in Information Systems Research*. *SIGMIS Database*. 38:20–28. 2007.
- [Ge09] Gehlert, A.; Schermann, M.; Pohl, K.; Krcmar, H.: *Towards a Research Method for Theory-driven Design Research*. In (Hansen, H. R.; Karagiannis, D.; Fill, H. G., Ed.): *Business Services: Konzepte, Technologien, Anwendungen, 9. Internationale Tagung Wirtschaftsinformatik*. volume 1. pages 441–450. Wien, Austria. 2009. Österreichische Computer Gesellschaft.
- [Gi10] Giachetti, R. E.: *Design of Enterprise Systems – Theory, Architecture, and Methods*. CRC Press. Boca Raton, FL, USA. 2010.
- [GJ07] Gregor, S.; Jones, D.: *The anatomy of a design theory*. *Journal of the Association of Information Systems*. 8(5):312–335. 2007.
- [Gl89] Glasersfeld, E.: *Cognition, Construction of Knowledge, and Teaching*. 1989.
- [GLS06] Gammelgård, M.; Lindström, Å.; Simonsson, M.: *A reference model for IT management responsibilities*. In *10<sup>th</sup> IEEE International EDOC Conference Workshops (EDOCW)*. pages 26–33. Washington, DC, USA. 2006. IEEE Computer Society.
- [Gr06] Gregor, S.: *The nature of theory in information systems*. *MIS Quarterly*. 30(3):491–506. 2006.
- [Gu94] Gutzwiller, T. A.: *Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen*. PhD thesis. Universität St.Gallen. 1994.
- [Gu05] Guizzardi, G.: *Ontological foundations for structural conceptual models*. PhD thesis. CTIT, Centre for Telematics and Information Technology. Enschede, The Netherlands. 2005.
- [GW06] Gericke, A.; Winter, R.: *Situational change Engineering in Healthcare*. In (Stormer, H.; Meier, A.; Schumacher, M., Ed.): *ECEH*. pages 227–238. Bonn, Germany. 2006. Gesellschaft für Informatik.
- [Ha97] Harmsen, A. F.: *Situational Method Engineering*. PhD thesis. University of Twente. Twente, The Netherlands. 1997.
- [Ha05] Harmon, K.: *The "Systems" Nature of Enterprise Architecture*. In *IEEE International Conference on Systems, Man and Cybernetics (SMC 2005)*. pages 78–58. 2005.
- [Ha10] Hanschke, I.: *Strategic IT Management – A Toolkit for Enterprise Architecture Management*. Springer. Berlin, Germany. 2010.
- [HBO94] Harmsen, F.; Brinkkemper, S.; Oei, J.: *Situational method engineering for informational system project approaches*. In *Proceedings of the IFIP WG8.1 Working Conference on Methods and Associated Tools for the Information Systems Life Cycle*. pages 169–194. New York, NY, USA. 1994. Elsevier Science Inc.

- [He93] Heym, M.: *Method-Engineering – Spezifikation und Integration von Entwicklungsmethoden für Informationssysteme*. PhD thesis. St. Gallen, Switzerland. 1993.
- [He04] Hevner, A. R.; March, S. T.; Park, J.; Ram, S.: *Design Science in Information Systems Research*. *MIS Quarterly*. 28(1):75–105. 2004.
- [He05] Heinrich, L. J.: *Forschungsmethodik einer Integrationsdisziplin – Ein Beitrag zur Geschichte der Wirtschaftsinformatik*. *NTM International Journal of History and NTM International Journal of History and Ethics of Natural Sciences, Technology and Medicine*. 13(2):104–117. 2005.
- [He08] Heise, D.; Strecker, S.; Frank, U.; Jung, J.: *Erweiterung einer Unternehmensmodellierungsmethode zur Unterstützung des IT-Controllings*. In (Bichler, M.; Hess, T.; Krcmar, H.; Lechner, U.; Matthes, F.; Picot, A.; Speitkamp, B. et al., Ed.): *Multikonferenz Wirtschaftsinformatik*. GITO-Verlag, Berlin. 2008.
- [HK03] Hirschheim, R.; Klein, H. K.: *Crisis in the IS Field? A Critical Reflection on the State of the Discipline*. *Journal of the Association for Information Systems*. 4(1):237–293. 2003.
- [HKL95] Hirschheim, R.; Klein, H. K.; Lytinen, K.: *Information Systems Development and Data Modeling – Conceptual and Philosophical Foundations*. Cambridge University Press. Cambridge, UK. 1995.
- [Ho10] ter Hofstede, A. H.; van der Aalst, W. M.; Adams, M.; Russel, N.: *Modern Business Process Automation – YAWL and its Support Environment*. Springer. Berlin, Germany. 2010.
- [HP04] Hirvonen, A. P.; Pulkkinen, M.: *A Practical Approach to EA Planning and Development: the EA Management Grid*. In (Abramowicz, W., Ed.): *Proceedings of 7th International Conference on Business Information Systems*. pages 284–302. Poznan, PL. 2004.
- [HPK09] Harmsen, A. F.; Proper, E. H.; Kok, N.: *Informed Governance of Enterprise Transformations*. In (Proper, H. E.; Harmsen, F.; Dietz, J. L., Ed.): *Advances in Enterprise Engineering II*. pages 155–180. Berlin, Heidelberg, Germany. 2009. Springer.
- [HS95] Henderson-Sellers, B.: *Who Needs An OO Methodology Anyway? Guest editorial for Journal of Object-Oriented Programming*. 8(6):6–8. 1995.
- [HSR10] Henderson-Sellers, B.; Ralyé, J.: *Situational Method Engineering: State-of-the-Art Review*. *Journal of Universal Computer Science*. 16(3):424–478. 2010.
- [HV93] Henderson, J. C.; Venkatraman, N.: *Strategic alignment: leveraging information technology for transforming organizations*. *IBM Systems Journal*. 32(1):472–484. 1993.
- [HV97] ter Hofstede, A. H.; Verhoef, T.: *On the feasibility of situational method engineering*. *Inf. Syst.* 22:401–422. 1997.

- 
- [HW05] Hafner, M.; Winter, R.: *Vorgehensmodell für das Management der unternehmensweiten Applikationsarchitektur*. In (Ferstl, O. K.; Sinz, E. J.; Eckert, S.; Isselhorst, T., Ed.): *Wirtschaftsinformatik*. pages 627–646. Heidelberg, Germany. 2005. Physica-Verlag.
- [HW08] Hafner, M.; Winter, R.: *Processes for Enterprise Application Architecture Management*. In *41<sup>st</sup> Hawaii International Conference on System Sciences (HICSS 2008)*. page 396. Los Alamitos, CA, USA. 2008. IEEE Computer Society.
- [IE91] IEEE: *IEEE Standard Taxonomy for Software Engineering Standards, Std. 1002-1987*. In (Institute of Electrical and Electronics Engineers, Ed.): *IEEE Software Engineering Standards Collection, Spring 1991 Edition*. New York, NY, USA. 1991. IEEE Computer Society.
- [IE00] IEEE: *IEEE Std 1471-2000 for Recommended Practice for Architectural Description of Software-Intensive Systems*. 2000.
- [IF99] IFIP-IFAC Task Force on Architecture for Enterprise Integration: *Geram: Generalised Enterprise Reference Architecture and Methodology – version 1.6.3*. 1999.
- [IF03] IFIP-IFAC Task Force on Architecture for Enterprise Integration: *GERAM: The Generalised Enterprise Reference Architecture and Methodology*. In (Bernus, P.; Nemes, L.; Schmidt, G., Ed.): *Handbook on Enterprise Architecture*. pages 21–63. Berlin, Heidelberg, Germany. 2003. Springer.
- [IJ06] Iacob, M.-E.; Jonkers, H.: *Quantitative Analysis of Enterprise Architectures*. In (Konstantas, D.; Bourrières, J.-P.; Léonard, M.; Boudjlida, N., Ed.): *Interoperability of Enterprise Software and Applications*. pages 239–252. Geneva, Switzerland. 2006. Springer.
- [IL08] Isomäki, H.; Liimatainen, K.: *Challenges of Government Enterprise Architecture Work – Stakeholders’ Views*. In (Wimmer, M.; Scholl, H. J.; Ferro, E., Ed.): *Electronic Government, 7th International Conference*. pages 364–374. Turin, Italy. 2008. Springer.
- [In96] International Organization for Standardization: *ISO/IEC 10746 Reference Model of Open Distributed Processing (RM-ODP)*. 1996.
- [In99] International Organization for Standardization: *ISO 15704 (1999) ISO/DIS 15704: Industrial automation systems – Requirements for enterprise reference architectures and methodologies. ISO/TC 184/SC5/WG1*. 1999.
- [In06] International Organization for Standardization: *ISO 19439:2006 Enterprise integration – Framework for enterprise modelling*. 2006.
- [In07] International Organization for Standardization: *ISO/IEC 42010:2007 Systems and software engineering – Recommended practice for architectural description of software-intensive systems*. 2007.
- [In08] International Telecommunication Union: *ITU-T Z.151: User requirements notation (URN) – Language definition*. 2008.

- [In10] InfoAsset: *Tricia – Open Source Web Collaboration and Knowledge Management Software*. <http://www.infoasset.de/wikis/infoasset/home> (last accessed September, 3<sup>rd</sup> 2010). 2010.
- [IT09] IT Governance Institute: *Framework Control Objectives Management Guidelines Maturity Models*. <http://www.isaca.org/Knowledge-Center/cobit> (cited 2011-02-14). 2009.
- [Ja94] Jayaratna, N.: *Understanding and Evaluating Methodologies: NIMSAD, a Systematic Framework*. McGraw-Hill, Inc. New York, NY, USA. 1994.
- [Ja01] Jackson, R.: *Plato – A Beginner’s Guide*. Hoder & Stroughton. London, UK. 2001.
- [Ja02] James, G. A.: *Best practices for selecting enterprise architects*. Research note. tactical guidelines, tg-17-4980. Gartner, Inc. 2002.
- [JE07] Johnson, P.; Ekstedt, M.: *Enterprise Architecture – Models and Analyses for Information Systems Decision Making*. Studentlitteratur. Pozkal, Poland. 2007.
- [JNL06] Johnson, P.; Nordström, L.; Lagerström, R.: *Formalizing Analysis of Enterprise Architecture*. In *Enterprise Interoperability*. pages 35–44. Bordeaux, France. 2006. Springer.
- [Jo03] Jonkers, H.; van Burren, R.; Arbab, F.; de Boer, F.; Bonsangue, M. M.; Bosma, H.; ter Doest, H. et al.: *Towards a language for coherent enterprise architecture descriptions*. In *7<sup>th</sup> IEEE International EDOC Conference (EDOC 2003)*. Brisbane, Australia. 2003. IEEE Computer Society.
- [Jo04a] Johnson, P.; Ekstedt, M.; Silva, E.; Plazola, L.: *Enterprise Architecture for CIO Decision-Making: On the Importance of Theory*. In *Proceedings of the Second Annual Conference on Systems Engineering Research*. pages 1–10. 2004.
- [Jo04b] Jonkers, H.; Lankhorst, M. M.; Buuren, R. v.; Bonsangue, M. M.; Torre, L. W. N. v. d.: *Concepts for Modelling Enterprise Architectures*. *International Journal of Cooperative Information Systems*. 13:257–287. 2004.
- [Jo06] Jonkers, H.; Lankhorst, M. M.; ter Doest, H.; Arbab, F.; Bosma, H.; Wieringa, R.: *Enterprise architecture: Management tool and blueprint for the organisation*. *Information Systems Frontiers*. 8:63–66. 2006.
- [Jo07] Johnson, P.; Lagerström, R.; Närman, P.; Simonsson, M.: *Enterprise architecture analysis with extended influence diagrams*. *Information Systems Frontiers*. 9:163–180. 2007.
- [Jo09] Josey, A. e. a.: *TOGAF Version 9 – A Pocket Guide*. Van Haren Publishing. Wilco, Amersfoort, Netherlands. 2<sup>nd</sup> edition. 2009.
- [Jo10] Jonkers, H.; van den Berg, H.; Iacob, M.-E.; Quartel, D.: *ArchiMate Extension for Modeling TOGAF’s Implementation and Migration phases*. [http://www.bizzdesign.com/index.php/component/docman/doc\\_download/18-archimate-extension-for-modeling-togafs-implementation-and-migration-phases](http://www.bizzdesign.com/index.php/component/docman/doc_download/18-archimate-extension-for-modeling-togafs-implementation-and-migration-phases). 2010.



- 
- [Ju07] Jung, J.: *Entwurf einer Sprache für die Modellierung von Ressourcen im Kontext der Geschäftsprozessmodellierung*. PhD thesis. Universität Duisburg-Essen. Berlin, Germany. 2007.
- [Ka08] Kant, I.: *Critique of Pure Reason*. Penguin Classics. New York, NY, USA. 2008.
- [Ke07] Keller, W.: *IT-Unternehmensarchitektur*. dpunkt.verlag. Heidelberg, Germany. 2007.
- [Ki99] Kirch, J.: *Prozessorientiertes Management von Client-Server-Systemen*. PhD thesis. University of Wiesbaden. 1999.
- [Ki08] Kirchner, L.: *Eine Methode zur Unterstützung des IT-Managements im Rahmen der Unternehmensmodellierung*. PhD thesis. Universität Duisburg-Essen. Berlin, Germany. 2008.
- [KL67] Kamlah, W.; Lorenzen, P.: *Logische Propädeutik: Vorschule des vernünftigen Redens*. Metzler. Stuttgart, Germany. 2<sup>nd</sup> edition. 1967.
- [Kn64] Knuth, D. E.: *Backus Normal Form vs. Backus Naur Form*. *Commun. ACM*. 7:735–736. 1964.
- [KNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: *Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)"*. Technical Report Heft 89. Institut für Wirtschaftsinformatik (IWi), Universität des Saarlandes. 1992.
- [KO55] Koontz, H.; O'Donnell, C.: *Principles of Management – An Analysis of Managerial Functions*. McGraw-Hill. New York, USA. 1955.
- [Kü04] Kühn, H.: *Methodenintegration im Business Engineering*. PhD thesis. Universität Wien. 2004.
- [Ku09] Kurpjuweit, S.: *Stakeholder-orientierte Modellierung und Analyse der Unternehmensarchitektur*. PhD thesis. Universität St.Gallen. 2009.
- [KUJ09] Källgren, A.; Ullberg, J.; Johnson, P.: *A Method for Constructing a Company Specific Enterprise Architecture Model Framework*. In (Kim, H.-K.; Lee, R. Y., Ed.): *10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing, SNPD 2009*. pages 346–351. Daegu, Korea. 2009.
- [KW92] Kumar, K.; Welke, R. J.: *Methodology Engineering: A Proposal for Situation-specific Methodology Construction*. In (Cotterman, W. W.; Senn, J. A., Ed.): *Challenges and Strategies for Research in Systems Development*. pages 257–270. West Sussex, England. 1992. John Wiley.
- [KW07] Kurpjuweit, S.; Winter, R.: *Viewpoint-based Meta Model Engineering*. In (Reichert, M.; Strecker, S.; Turowski, K., Ed.): *2<sup>nd</sup> International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007)*. LNI. pages 143–161. Bonn, Germany. 2007. Gesellschaft für Informatik.
- [KW09] Kurpjuweit, S.; Winter, R.: *Concern-oriented business architecture engineering*. In *SAC'09: Proceedings of the 2009 ACM symposium on Applied Computing*. pages 265–272. New York, NY, USA. 2009. ACM.

- [La04] Lankhorst, M. M.; van Buuren, R.; van Leeuwen, D.; Jonkers, H.; ter Doest, H.: *Enterprise architecture modelling-the issue of integration*. *Adv. Eng. Inform.* 18(4):205–216. 2004.
- [La05] Lankhorst, M. M.: *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer. Berlin, Heidelberg, Germany. 2005.
- [La06] Lange, C.: *Entwicklung und Stand der Disziplinen Wirtschaftsinformatik und Information Systems – Interpretative Auswertung von Interviews: Teil III – Ergebnisse zur Wirtschaftsinformatik, ICB-Research Report No.4*. Technical report. Universität Duisburg-Essen. Duisburg, Essen, Germany. 2006.
- [La07] Lageström, R.: *Analyzing System Maintainability using Enterprise Architecture Models*. *Journal of Enterprise Architecture*. 3:33–42. 2007.
- [La08] Lagerström, R.; Chenine, M.; Johnson, P.; Franke, U.: *Probabilistic Metamodel Merging*. In (Bellahsene, Z.; Woo, C.; Hunt, E.; Franch, X.; Coletta, R., Ed.): *CAiSE Forum*. pages 25–28. Montpellier, France. 2008.
- [La09a] Lankhorst, M. M.: *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer. Berlin, Heidelberg, Germany. 2<sup>nd</sup> edition. 2009.
- [La09b] Lau, A.; Fischer, T.; Buckl, S.; Ernst, A. M.; Matthes, F.; Schweda, C. M.: *EA Management Patterns for Smart Networks*. In *SE 2009 – Workshopband*. pages 79–90. 2009.
- [La09c] Op ’t Land, M.; Proper, E.; Waage, M.; Cloo, J.; Steghuis, C.: *Enterprise Architecture – Creating Value by Informed Governance*. Springer. Berlin, Heidelberg. 2009.
- [LD08] Land, M. O.; Dietz, J. L.: *Enterprise ontology based splitting and contracting of organizations*. In *SAC ’08: Proceedings of the 2008 ACM symposium on Applied computing*. pages 524–531. New York, NY, USA. 2008. ACM.
- [Le99] Lee, A. S.: *Rigor and Relevance in MIS Research – Beyond the Approach of Positivism Alone*. *MIS Quarterly*. 23(1):29–34. 1999.
- [LHS08] Liimatainen, K.; Heikkilä, J.; Seppänen, V.: *A Framework for Evaluating Compliance of Public Service Development Programs with Government Enterprise Architecture*. In *Proceedings of the 2nd European Conference on Information Management and Evaluation*. pages 269–276. London, UK. 2008.
- [LJ08] Lagerström, R.; Johnson, P.: *Using Architectural Models to Predict the Maintainability of Enterprise Systems*. In *12th European Conference on Software Maintenance and Reengineering*. pages 248–252. Athens, Greece. 2008. IEEE.
- [LK04] Lyytinen, K.; King, J. L.: *Nothing At The Center? Academic Legitimacy in the Information Systems Field*. *Journal of the Association for Information Systems*. 5(6):220–246. 2004.
- [LKL10] Lucke, C.; Krell, S.; Lechner, U.: *Critical Issues in Enterprise Architecting — A Literature Review*. In *Proceedings of the Sixteenth Americas Conference on Information Systems (AMCIS 2010)*. Lima, Peru. 2010.

- 
- [LLO93] Luftman, J. N.; Lewis, P. R.; Oldach, S. H.: *Transforming the enterprise: The alignment of business and information technology strategies*. *IBM Systems Journal*. 32(1):198–221. 1993.
- [Lo87] Lorenzen, P.: *Constructive Philosophy*. University of Massachusetts Press. Amherst, MA, USA. 1987.
- [Lo01] Losee, J.: *A Historical Introduction to the Philosophy of Science*. Oxford University Press. New York, NY, USA. 4<sup>th</sup> edition. 2001.
- [Lu03] Luftman, J. N.: *Competing in the Information Age – Align in the Sand*. Oxford University Press. New York, NY, USA. 2<sup>nd</sup> edition. 2003.
- [Lu09] Luijpers, J.: *White Paper: Project Start Architecture – Version 1.0*. Technical report. Sogeti Nederland B.V. Diemen, The Netherlands. 2009.
- [LVP07] Leppänen, M.; Valtonen, K.; Pulkkinen, M.: *Towards a Contingency Framework for Engineering and Enterprise Architecture Planning Method*. In *30<sup>th</sup> Information Systems Research Seminar in Scandinavia (IRIS)*. pages 1–20. 2007.
- [LW04a] Langenberg, K.; Wegmann, A.: *Enterprise Architecture: What Aspect is Current Research Targeting?* Technical report. Laboratory of Systemic Modeling, Ecole Polytechnique Fédérale de Lausanne. Lausanne, Switzerland. 2004.
- [LW04b] Lê, L.-S.; Wegmann, A.: *Meta-model for Object-Oriented Hierarchical Systems*. Technical report. Laboratory of Systemic Modeling, Ecole Polytechnique Fédérale de Lausanne. Lausanne, Switzerland. 2004.
- [LW05] Lê, L.-S.; Wegmann, A.: *Definition of an Object-Oriented Modeling Language for Enterprise Architecture*. In *38<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2005)*. pages 179–188. 2005.
- [LW06] Lê, L.-S.; Wegmann, A.: *SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture*. In *39<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2006)*. 2006.
- [Ma95] Mayer, R.; Menzel, C.; Painter, M.; deWitte, P.; Blinn, T.; Perakath, B.: *Information Integration for Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report*. [http://www.idef.com/pdf/Idef3\\_fn.pdf](http://www.idef.com/pdf/Idef3_fn.pdf) (last accessed May, 22nd 2009). 1995.
- [Ma08] Matthes, F.; Buckl, S.; Leitel, J.; Schweda, C. M.: *Enterprise Architecture Management Tool Survey 2008*. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2008.
- [MaZT07] Magalhães, R.; Zacharias, M.; Tribolet, J. M.: *Making Sense of Enterprise Architectures as Tools of Organizational Self-Awareness (OSA)*. In *2<sup>nd</sup> Workshop on Trends in Enterprise Architecture Research (TEAR 2007)*. 2007.
- [MD97] Meszaros, G.; Doble, J.: *A Pattern Language for Pattern Writing*. pages 529–574. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA. 1997.
- [ME02] META Group: *Enterprise Architecture Desk Reference*. Stamford, CT, USA, 2002.

- [MMG02] Markus, M. L.; Majchrzak, A.; Gasser, L.: *A design theory for systems that support emergent knowledge processes*. *MIS Quarterly*. 26(3):179–212. 2002.
- [Mo03] Monod, E.: *A Copernican Revolution in IS: using Kant's critique of pure reason for describing epistemological trends in IS*. In *AMCIS 2003: Proceedings of the 9<sup>th</sup> Americas Conference on Information Systems*. Tampa, FL, USA. 2003.
- [Mo09] Moser, C.; Junginger, S.; Brückmann, M.; Schöne, K.-M.: *Some Process Patterns for Enterprise Architecture Management*. In (Münch, J.; Liggesmeyer, P., Ed.): *Software Engineering 2009 – Workshopband*. pages 19–30. Bonn, Germany. 2009. Lecture Notes in Informatics (LNI).
- [MS95] March, S. T.; Smith, G. F.: *Design and natural science research on information technology*. *Decis. Support Syst.* 15(4):251–266. 1995.
- [MWF08] Murer, S.; Worms, C.; Furrer, F. J.: *Managed Evolution*. *Informatik Spektrum*. 31(6):537–547. 2008.
- [My92] Mylopoulos, J.: *Conceptual modeling and Telos*. pages 49–68. Wiley. New York, USA. 1992.
- [My11] Mykhashchuk, M.; Buckl, S.; Dierl, T.; Schweda, C. M.: *Charting the landscape of enterprise architecture management – An extensive literature analysis*. In *Wirtschaftsinformatik*. Zürich, Swiss. 2011.
- [MZT10] Marques, J. a.; Zacarias, M.; Tribolet, J.: *A Bottom-Up Competency Modeling Approach*. In (Aalst, W.; Mylopoulos, J.; Sadeh, N. M.; Shaw, M. J.; Szyperki, C.; Albani, A.; Dietz, J. L. G., Ed.): *Advances in Enterprise Engineering IV*. volume 49 of *Lecture Notes in Business Information Processing*. pages 50–64. Springer Berlin Heidelberg. 2010.
- [Na03] National Association of State Chief Information Officers (NASCIO): *NASCIO Enterprise Architecture Maturity Model – Version 1.3*. <http://www.nascio.org/publications/documents/NASCIO-EAMM.pdf> (last accessed 2010-11-11). 2003.
- [NA07] NATO: *NATO Architecture Framework Version 3*. [http://www.nhq3s.nato.int/ARCHITECTURE/\\_docs/NAF\\_v3/ANNEX1.pdf](http://www.nhq3s.nato.int/ARCHITECTURE/_docs/NAF_v3/ANNEX1.pdf) (cited 2011-02-14). 2007.
- [Nä08] Närman, P.; Schönherr, M.; Johnson, P.; Ekstedt, M.; Chenine, M.: *Using Enterprise Architecture Models for System Quality Analysis*. In *12<sup>th</sup> IEEE International EDOC Conference (EDOC 2008)*. pages 14–23. Washington, DC, USA. 2008. IEEE Computer Society.
- [NCP91] Nunamaker, J. F.; Chen, M.; Purdin, T. D. M.: *Systems development in information systems research*. *J. Manage. Inf. Syst.* 7(3):89–106. 91.
- [Ne00] Neilson, R.: *Strategic Scenario Planning at CA International*. *Knowledge Management Review*. 2000.
- [Ni06a] Niehaves, B.: *Epistemological Perspectives on Multi-Method Information System Research*. In *The Reflective Designer – Designing IT-Consulting Processes*. 2006.
- [Ni06b] Niehaves, B.: *The Reflective Designer – Designing IT-Consulting Processes*. Phd thesis. Westfälische Wilhelms-Universität Münster. 2006.

- 
- [Ni06c] Niemann, K. D.: *From Enterprise Architecture to IT Governance – Elements of Effective IT Management*. Vieweg+Teubner. Wiesbaden, Germany. 2006.
- [No98] Noble, J.: *Classifying Relationships between Object-Oriented Design Patterns*. In *Australian Software Engineering Conference (ASWEC)*. pages 98–107. Los Alamitos, CA, USA. 1998. IEEE Computer Society.
- [No03] Noran, O.: *A Mapping of Individual Architecture Frameworks (GRAI, PERA, C4ISR, CIMOSA, Zachman, ARIS) onto GERAM*. In (Bernus, P.; Nemes, L.; Schmidt, G., Ed.): *Handbook on Enterprise Architecture*. pages 65–210. Berlin, Heidelberg, Germany. 2003. Springer.
- [OA07] OASIS: *Web Services Business Process Execution Language Version 2.0, OASIS Standard, 11 April 2007*. <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf> (cited 2011-02-07). 2007.
- [OA09] OASIS WSBPEL Technical Committee: *WS Business Process Execution Language Version 2.0*. 2009.
- [OB01] Orlikowski, W. J.; Barley, S. R.: *Technology and Institutions: What Can Research on Information Technology and Research on Organizations Learn from Each Other?* *MIS Quarterly*. 25(2):145–165. 2001.
- [Ob10a] Object Management Group (OMG): *Business Motivation Model 1.1*. <http://www.omg.org/spec/BMM/1.1/> (cited 2011-02-14). 2010.
- [Ob10b] Object Management Group (OMG): *Business Process Model and Notation (BPMN) – Version 2.0*. 2010.
- [Ob10c] Object Management Group (OMG): *Object Constraint Language (OCL) Available Specification, version 2.2 (formal/2010-02-01)*. 2010.
- [Ob10d] Object Management Group (OMG): *OMG Unified Modeling Language<sup>TM</sup> (OMG UML), Infrastructure – Version 2.3 (formal/2010-05-03)*. 2010.
- [Ob10e] Object Management Group (OMG): *OMG Unified Modeling Language<sup>TM</sup> (OMG UML), Superstructure – Version 2.3 (formal/2010-05-05)*. 2010.
- [Of09] Office of Government Commerce (OGC): *Managing Successful Projects with PRINCE2*. The Stationery Office. Norwich, UK. 2009.
- [Ös07] Österle, H.; Winter, R.; Hoening, F.; Kurpjuweit, S.; Osl, P.: *Business Engineering: Core-Business-Metamodell*. *WisU – Das Wirtschaftsstudium*. 36(2):191–194. 2007.
- [ÖW03] Österle, H.; Winter, R.: *Business Engineering – Auf dem Weg zum Unternehmen des Informationszeitalters*. Springer. Berlin, Germany. 2<sup>nd</sup> edition. 2003.
- [ÖWe10] Österle, H.; Winter, R.; (eds.), W. B.: *Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz*. infowerk ag. St. Gallen, Switzerland. 2010.
- [PH05] Pulkkinen, M.; Hirvonen, A. P.: *EA Planning, Development and Management Process for Agile Enterprise Development*. In *38th Hawaii International Conference on System Sciences*. Big Island, HI, USA. 2005. IEEE Computer Society.

- [PHB08] Pries-Heje, J.; Baskerville, R.: *The Design Theory Nexus*. *MIS Quarterly*. 32(4):731–755. 2008.
- [PNL07] Pulkkinen, M.; Naumenko, A.; Luostarinen, K.: *Managing information security in a business network of machinery maintenance services business - Enterprise architecture as a coordination tool*. *J. Syst. Softw.* 80(10):1607–1620. 2007.
- [Po80] Popper, K.: *The Logic of Scientific Discovery*. Unwin Hyman. London, UK. 1980.
- [Po96] Pollio, V.: *Vitruvii De Architectura Libri Decem – Zehn Bücher über Architektur – Übersetzt und mit Anmerkungen versehen von Curt Fensterbusch*. Primus-Verlag. Darmstadt, Germany. 5<sup>th</sup> edition. 1996.
- [Pr98] Probst, G. J. B.: *Practical Knowledge Management: A Model that Works*. Arthur D Little *PRISM*. 1998.
- [PS06] Pearlson, K. E.; Saunders, C. S.: *Managing and Using Information Systems*. Wiley & Sons. Hoboken, NJ, USA. 3<sup>rd</sup> edition. 2006.
- [Pu06] Pulkkinen, M.: *Systemic Management of Architectural Decisions in Enterprise Architecture Planning. Four Dimensions and Three Abstraction Levels*. In *39<sup>th</sup> Hawaii International Conference on System Sciences (HICSS 2006)*. volume 8. page 179c. 2006.
- [QEJ10] Quartel, D.; Engelsman, W.; Jonkers, H.: *ArchiMate Extension for Modeling and Managing Motivation, Principles and Requirements in TOGAF*. [http://www.bizzdesign.com/index.php/component/docman/doc\\_download/16-extending-ea-modelling-with-business-goals-and-requirements](http://www.bizzdesign.com/index.php/component/docman/doc_download/16-extending-ea-modelling-with-business-goals-and-requirements). 2010.
- [Ra02] Ralyté, J.: *Requirements Definition for the Situational Method Engineering*. In *Proceedings of the IFIP TC8 / WG8.1 Working Conference on Engineering Information Systems in the Internet Context*. pages 127–152. Deventer, The Netherlands. 2002. Kluwer, B.V.
- [RA09] Riege, C.; Aier, S.: *A Contingency Approach to Enterprise Architecture Method Engineering*. In *Service-Oriented Computing – ICSOC 2008 Workshops*. pages 388–399. 2009.
- [RB06] Ross, J. W.; Beath, C. M.: *Sustainable IT Outsourcing Success: Let Enterprise Architecture be your Guide*. *MIS Quarterly Executive*. 5(4):181–192. 12 2006.
- [RBW03] Rychkova, I.; Balabko, P.; Wegmann, A.: *Operational ASM Semantics behind Graphical SEAM Notation*. In *DAIS/FMOODS Ph.D. workshop*. pages 10–19. 2003.
- [RDR03] Ralyté, J.; Deneckère, R.; Rolland, C.: *Towards a generic model for situational method engineering*. In *Proceedings of the 15th international conference on Advanced information systems engineering*. CAiSE'03. pages 95–110. Berlin, Heidelberg. 2003. Springer-Verlag.
- [Re99] Rechtin, E.: *Systems Architecting of Organizations – Why Eagles can't swim*. CRC Press LLC. New York, NY, USA. 1999.

- [Re05a] Recker, J.: *Conceptual Model Evaluation – Towards more Paradigmatic Rigor*. In (Halpin, T.; Siau, K.; Krogstie, J., Ed.): *Proceedings of the Workshop on Evaluating Modeling Methods for Systems Analysis and Design (EMMSAD'05), held in conjunction with the 17th Conference on Advanced Information Systems (CAiSE'05), Porto, Portugal, EU*. pages 569–580. FEUP, Porto, Portugal. 2005.
- [Re05b] Recker, J.: *Evaluation of Conceptual Modeling Languages – An Epistemological Discussion*. In *Proceedings of the 11<sup>th</sup> Americas Conference on Information Systems (AmCIS 2005), August 11-14, 2005*. Omaha, NE, USA. 2005.
- [Re09] Regev, G.; Hayard, O.; Gause, D. C.; Wegmann, A.: *Modeling Service-Level Requirements: A Constancy Perspective*. In *RE 2009, 17th IEEE International Requirements Engineering Conference*. pages 231–236. 2009.
- [RH06] Reussner, R.; Hasselbring, W.: *Handbuch der Software-Architektur*. Dpunkt Verlag, Heidelberg, Germany. 2006.
- [RNE09] Raderius, J.; Närman, P.; Ekstedt, M.: *Assessing System Availability Using an Enterprise Architecture Analysis Approach*. In *Service-Oriented Computing — IC-SOC 2008 Workshops: IC-SOC 2008 International Workshops, Sydney, Australia, December 1st, 2008, Revised Selected Papers*. pages 351–362. Berlin, Heidelberg. 2009. Springer-Verlag.
- [Ro03] Ross, J. W.: *Creating a Strategic IT Architecture Competency: Learning in Stages*. *MIS Quarterly Executive*. 2(1). 2003.
- [RP96] Rolland, C.; Prakash, N.: *A proposal for context-specific method engineering*. In *Proceedings of the IFIP TC8, WG8.1/8.2 working conference on method engineering on Method engineering: principles of method construction and tool support: principles of method construction and tool support*. pages 191–208. London, UK, UK. 1996. Chapman & Hall, Ltd.
- [RPR98] Rolland, C.; Plihon, V.; Ralyté, J.: *Specifying the Reuse Context of Scenario Method Chunks*. In *Proceedings of the 10th International Conference on Advanced Information Systems Engineering*. pages 191–218. London, UK. 1998. Springer-Verlag.
- [RR01] Ralyté, J.; Rolland, C.: *An Assembly Process Model for Method Engineering*. In *Proceedings of the 13th International Conference on Advanced Information Systems Engineering*. CAiSE '01. pages 267–283. London, UK. 2001. Springer-Verlag.
- [RS03] Rossi, M.; Sein, M. K.: *Design Research workshop: A proactive research approach*. Presentation delivered at IRIS 26, August 9–12, 2003 <http://www.cis.gsu.edu/~emonod/epistemology/Sein%20and%20Rossi%20-%20design%20research%20-%20IRIS.pdf> (cited 2011-02-14). 2003.
- [RSV08] van der Raadt, B.; Schouten, S.; van Vliet, H.: *Stakeholder Perception of Enterprise Architecture*. In (Morrison, R.; Balasubramaniam, D.; Falkner, K. E., Ed.): *ECISA*. volume 5292 of *Lecture Notes in Computer Science*. pages 19–34. Berlin, Heidelberg, Germany. 2008. Springer.
- [RV08] van Raadt, B. d.; van Vliet, H.: *Designing the Enterprise Architecture Function*. In (Steffen Becker, F. P.; Reussner, R., Ed.): *4<sup>th</sup> International Conference on the*

- Quality of Software Architectures (QoSA2008)*. volume 5281 of *Lecture Notes in Computer Science*. pages 103–118. Karlsruhe, Germany. 2008. Springer.
- [RW73] Rittel, H. W. J.; Webber, M. M.: *Dilemmas in a general theory of planning*. *Policy Sciences*. 4(2):155–169. June 1973.
- [RW06] Rychkova, I.; Wegmann, A.: *A Method for Functional Alignment Verification in Hierarchical Enterprise Models*. In *BUSITAL*. 2006.
- [RW07] Rychkova, I.; Wegmann, A.: *Formal Semantics for Property-Property Relations in SEAM Visual Language: Towards Simulation and Analysis of Visual Specifications*. In *MSVVEIS*. pages 138–147. 2007.
- [RWR06] Ross, J. W.; Weill, P.; Robertson, D. C.: *Enterprise Architecture as Strategy*. Harvard Business School Press. Boston, MA, USA. 2006.
- [Sa93] Saeki, M.; Iguchi, K.; Wen-yin, K.; Shinohara, M.: *A Meta-Model for Representing Software Specification & Design Methods*. In *Proceedings of the IFIP WG8.1 Working Conference on Information System Development Process*. pages 149–166. Amsterdam, The Netherlands. 1993. North-Holland Publishing Co.
- [SB93] van Slooten, K.; Brinkkemper, S.: *A Method Engineering Approach to Information Systems Development*. In *Proceedings of the IFIP WG8.1 Working Conference on Information System Development Process*. pages 167–186. Amsterdam, The Netherlands. 1993. North-Holland Publishing Co.
- [SB09] Steenbergen, M. v.; Brinkkemper, S.: *The Architectural Dilemma: Division of Work vs. Knowledge Integration*. In *Proceedings of the 4<sup>th</sup> International Workshop on Business/IT Alignment and Interoperability (BUSITAL'09)*. Amsterdam, The Netherlands. 2009. Springer.
- [SBB07a] Steenbergen, M. v.; Berg, M. v. d.; Brinkkemper, S.: *An Instrument for the Development of the Enterprise Architecture Practice*. In (Cardoso, J.; Cordeiro, J.; Filipe, J., Ed.): *ICEIS 2007 - Proceedings of the Ninth International Conference on Enterprise Information Systems, Volume EIS, Funchal, Madeira, Portugal, June 12-16, 2007*. pages 14–22. 2007.
- [SBB07b] van Steenbergen, M.; van den Berg, M.; Brinkkemper, S.: *An Instrument for the Development of the Enterprise Architecture Practice*. In (Cardoso, J.; Cordeiro, J.; Filipe, J., Ed.): *ICEIS 2007 – Proceedings of the Ninth International Conference on Enterprise Information Systems, Volume EIS, Funchal, Madeira, Portugal, June 12-16, 2007 (3)*. pages 14–22. 2007.
- [SBK07a] Schermann, M.; Böhmman, T.; Krcmar, H.: *Fostering the Evaluation of Reference Models: Application and Extension of the Concept of IS Design Theories*. In (Oberweis, A.; Weinhardt, C.; Gimpel, H.; Koschmider, A.; Pankratius, V.; Schnizler, B., Ed.): *Wirtschaftsinformatik (2)*. pages 181–198. Karlsruhe, Germany. 2007. Universitaetsverlag Karlsruhe.
- [SBK07b] Schermann, M.; Böhmman, T.; Krcmar, H.: *A Pattern-based Approach for Constructing Design Theories with Conceptual Models*. In *ECIS2007: Proceedings of the European Conference on Information Systems*. pages 1368–1379. 2007.



- 
- [Sc98] Schütte, R.: *Grundsätze ordnungsmäßiger Referenzmodellierung – Konstruktion konfigurations- und anpassungsorientierter Modelle*. Gabler. Wiesbaden, Germany. 1998.
- [Sc99] Schütte, R.: *Basispositionen der Wirtschaftsinformatik – ein gemässigt konstruktivistisches Programm*. In (Becker, J.; König, W.; Schütte, R., W. O.; Zelewski, S., Ed.): *Wirtschaftsinformatik und Wissenschaftstheorie – Bestandsaufnahme und Perspektiven*. pages 211–241. Wiesbaden, Germany. 1999. Gabler.
- [Sc01] Scheer, A.-W.: *ARIS – Modellierungsmethoden, Metamodelle, Anwendungen*. Springer. Berlin, Germany. 4<sup>th</sup> edition. 2001.
- [Sc02] Scheer, A.-W.: *ARIS – Vom Geschäftsprozess zum Anwendungssystem*. Springer. Berlin, Germany. 4<sup>th</sup> edition. 2002.
- [Sc04] Schekkerman, J.: *Enterprise Architecture Score Card*. Technical report. Institute For Enterprise Architecture Development. 2004.
- [Sc06a] Schekkerman, J.: *Enterprise Architecture Deliverables Guide*. Technical report. Institute For Enterprise Architecture Development. 2006.
- [Sc06b] Schekkerman, J.: *Extended Enterprise Architecture Framework Essential Guide*. Technical report. Institute For Enterprise Architecture Development. 2006.
- [Sc06c] Schekkerman, J.: *Extended Enterprise Architecture Maturity Model Support Guide*. Technical report. Institute For Enterprise Architecture Development. 2006.
- [Sc06d] Schekkerman, J.: *How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework*. Trafford Publishing. Victoria, BC, Canada. 2006.
- [Sc08a] Schekkerman, J.: *Enterprise Architecture Good Practices Guide – How to Manage the Enterprise Architecture Practice*. Trafford Publishing. Victoria, BC, Canada. 2008.
- [Sc08b] Schönherr, M.: *Towards a common terminology in the discipline of Enterprise Architecture*. In (Aier, S.; Johnson, P.; Schelp, J., Ed.): *Pre-Proceedings of the 3<sup>rd</sup> Workshop on Trends in Enterprise Architecture Research*. pages 107–123. Sydney, Australia. 2008.
- [Sc10] Schekkerman, J.: *STREAM – A Successful and Pragmatic ‘Managed Diversity’ Enterprise Architecture Approach*. 2010.
- [se05] sebis: *Enterprise Architecture Management Tool Survey 2005*. Technical report. Chair for Informatics 19 (sebis), Technische Universität München. Munich, Germany. 2005.
- [Se11] Sein, M. K.; Henfridsson, O.; Puraro, S.; Rossi, M.; Lindgren, R.: *Action Design Research*. *MIS Quarterly*. 35(2). 2011.
- [SG89] Star, S. L.; Griesemer, J. R.: *Institutional Ecology: “Translations” and Coherence: Amateurs and Professionals in Berkeley’s Museum of Vertebrate Zoology*. *Social Studies of Science*. Vol. 19(3):pages 387–420. 1989.

- [Sh86] Shewhart, W. A.: *Statistical Method from the Viewpoint of Quality Control*. Dover Publication. New York, NY, USA. 1986.
- [Sh90] Shaw, M.: *Prospects for an Engineering Discipline of Software*. *IEEE Softw.* 7(6):15–24. 1990.
- [SH93] Spewak, S. H.; Hill, S. C.: *Enterprise Architecture Planning – Developing a Blueprint for Data, Applications, and Technology*. John Wiley & Sons. New York, USA. 1993.
- [SHL09] Seppänen, V.; Heikkilä, J.; Liimatainen, K.: *Key Issues in EA-Implementation: Case Study of Two Finnish Government Agencies*. In (Hofreiter, B.; Werthner, H., Ed.): *2009 IEEE Conference on Commerce and Enterprise Computing*. pages 114–120. Vienna, Austria. 2009. IEEE Computer Society.
- [Si96] Simon, H. A.: *The Sciences of the Artificial*. MIT Press. Cambridge, Massachusetts, USA. 3<sup>rd</sup> edition. 1996.
- [SK04] Schwarzer, B.; Krcmar, H.: *Wirtschaftsinformatik. Grundzüge der betrieblichen Datenverarbeitung*. Schäffer - Poeschel Verlag Stuttgart. 3<sup>rd</sup> edition. 2004. 3791021710.
- [Sp93] Sprague, Jr., R. H.: *A framework for the development of decision support systems*. In (Sprague, Jr., R. H.; Watson, H. J., Ed.): *Decision support systems (3rd ed.)*. pages 3–28. Prentice-Hall, Inc. Upper Saddle River, NJ, USA. 1993.
- [SR98] Schütte, R.; Rotthowe, T.: *The Guidelines of Modeling – An Approach to Enhance the Quality in Information Models*. In (Ling, T. W.; Ram, S.; Lee, M. L., Ed.): *Conceptual Modeling – ER’98*. pages 240–254. Berlin, Germany. 1998. Springer.
- [SR07] Strano, C.; Rehmani, Q.: *The role of the enterprise architect*. *Information Systems and E-Business Management*. 5(4):379–396. 2007.
- [SR08] Siau, K.; Rossi, M.: *Evaluation Techniques for Systems Analysis and Design Modelling Methods—A Review and Comparative Analysis*. *Information Systems Journal*. pages 1365–2575. 2008.
- [SS07] Schelp, J.; Stutz, M.: *A Balanced Scorecard Approach to Measure the Value of Enterprise Architecture*. In (Lankhorst, M. M.; Johnson, P., Ed.): *Second Workshop on Trends in Enterprise Architecture Research*. pages 5–11. 2007.
- [SSW04] Sinclair, J.; Simon, J.; Wilkes, R.: *Research Directions in MIS: An Assessment of Current Status*. In *Proceedings of the 10th Americas Conference on Information Systems (AMCIS 2004)*. pages 4243–4265. New York, USA. 2004.
- [St73] Stachowiak, H.: *Allgemeine Modelltheorie*. Springer-Verlag. Wien, Austria. 1973.
- [St98] Stack, G. J.: *Materialism*. In (Craig, E., Ed.): *Routledge Encyclopedia of Philosophy*. pages 170–171. New York, USA. 1998. Routledge.
- [St05] Steenbergen, M. v.: *Architecture Maturity Matrix DYA – Version 1.0*. Technical report. Sogeti Nederland B.V. Diemen, The Netherlands. 2005.
- [St10a] Steenbergen, M. v.; Bos, R.; Brinkkemper, S.; Weerd, I. v. d.; Bekkers, W.: *The Design of Focus Area Maturity Models*. In (Winter, R.; Zhao, J.; Aier, S., Ed.):

- Global Perspectives on Design Science Research*. volume 6105 of *Lecture Notes in Computer Science*. pages 317–332. Springer Berlin / Heidelberg. 2010.
- [St10b] Steenbergen, M. v.; Schipper, J.; Bos, R.; Brinkkemper, S.: *The Dynamic Architecture Maturity Matrix: Instrument Analysis and Refinement*. In (Dan, A.; Gittler, F.; Toumani, F., Ed.): *ICSOC/ServiceWave 2009*. pages 48–61. Berlin, Heidelberg, Germany. 2010. Springer.
- [St10c] Struck, V.; Buckl, S.; Matthes, F.; Schweda, C. M.: *Enterprise Architecture Management from a knowledge management perspective – Results from an empirical study*. In *5<sup>th</sup> Mediterranean Conference on Information Systems (MCIS2010)*, Tel Aviv. 2010.
- [SW08] Schelp, J.; Winter, R.: *On the Interplay of Design Research and Behavioral Research – A Language Community Perspective*. In (Vaishanvi, V.; Baskerville, R., Ed.): *Proceedings of the Third International Conference on Design Science Research in Information Systems and Technology (DESRIST2008), May 7-9, 2008, Westin, Buckhead, Atlanta, Georgia, USA*. pages 79–92. Georgia State University, Atlanta, Georgia, USA. 2008.
- [SW09] Schelp, J.; Winter, R.: *Language communities in enterprise architecture research*. In *DESRIST '09: Proceedings of the 4<sup>th</sup> International Conference on Design Science Research in Information Systems and Technology*. pages 1–10. New York, NY, USA. 2009. ACM.
- [SZ92] Sowa, J. F.; Zachman, J. A.: *Extending and Formalizing the Framework for Information Systems Architecture*. *IBM Systems Journal*. 31(3):590–616. 1992.
- [Sz09] Szyszka, B.: *Analysis and Classification of Maturity Models in Enterprise Architecture Management*. Bachelor's thesis. Fakultät für Informatik, Technische Universität München. 2009.
- [Ta44] Tarski, A.: *The Semantic Conception of Truth: and the Foundations of Semantics*. *Philosophy and Phenomenological Research*. 4(3):341–376. 1944.
- [Ta90] Takeda, H.; Veerkamp, P.; Tomiyama, T.; Yoshikawa, H.: *Modeling design processes*. *AI Mag*. 11(4):37–48. 1990.
- [Th09a] The Open Group: *TOGAF “Enterprise Edition” Version 9*. <http://www.togaf.org> (cited 2011-02-14). 2009.
- [Th09b] The Open Group: *TOGAF Version 9 – A Manual*. Van Haren Publishing. 9th edition. 2009. 978-90-8753-230-7.
- [TR07] Tasharofi, S.; Ramsin, R.: *Process Patterns for Agile Methodologies*. In (Ralyté, J.; Brinkkemper, S.; Henderson-Sellers, B., Ed.): *Situational Method Engineering: Fundamentals and Experiences*. volume 244 of *IFIP International Federation for Information Processing*. pages 222–237. Springer Boston. 2007.
- [UK72] Ulrich, H.; Krieg, W.: *Das St. Galler Management-Modell*. Paul Haupt. Bern, Suisse. 1972.
- [Un03] United States General Accounting Office: *Information Technology – A Framework for Assessing and Improving Enterprise Architecture Management (Version 1.1)*. 2003. GAO-03-584G.

- [Un07] United States Department of Commerce: *Enterprise Architecture Capability Maturity Model – Version 1.2*. [http://ocio.os.doc.gov/ITPolicyandPrograms/Enterprise\\_Architecture/PROD01\\_004935](http://ocio.os.doc.gov/ITPolicyandPrograms/Enterprise_Architecture/PROD01_004935) (last accessed 2010-11-11). 2007.
- [Va01] Vasconcelos, A.; Caetano, A.; Neves, J.; Sinogas, P.; Mendes, R.; Tribolet, J. M.: *A Framework for Modeling Strategy, Business Processes and Information Systems*. In *5<sup>th</sup> IEEE International EDOC Conference (EDOC 2001)*. pages 69–81. IEEE Computer Society. 2001.
- [Va04] Vasconcelos, A.; Pereira, C. M.; Sousa, P.; Tribolet, J. M.: *Open Issues On Information System Architecture*. In *Proceedings of the 6<sup>th</sup> International Conference on Enterprise Information Systems (ICEIS 2004)*. 2004.
- [Ve06a] Venable, J. R.: *A Framework for Design Science Research Activities*. In (Khosrow-Pour, M., Ed.): *Emerging Trends and Challenges in Information Technology Management*. pages 184–187. London, UK. 2006. Idea Group Inc.
- [Ve06b] Venable, J. R.: *The Role of Theory and Theorising in Design Science Research*. In *Design Science Research in Information Systems and Technology*. 2006.
- [VH05] Verschuren, P.; Hartog, R.: *Evaluation in Design-Oriented Research. Quality and Quantity*. 39(6):733–762. December 2005.
- [VK04] Vaishnavi, V. K.; Kuechler, W. J.: *Design Research in Information Systems*. -. 2004.
- [VRG02] Vessey, I.; Ramesh, V.; Glass, R. L.: *Research in Information Systems: An Empirical Study of Diversity in the Discipline and Its Journals. Journal of Management Information Systems*. 19(2):129–174. 2002.
- [VS08] Valtonen, K.; Seppänen, V.: *Adaptation and adoption of Finnish government EA method – Preliminary guidelines for method users*. Ministry of Finance. 2008.
- [VSL09] Valtonen, K.; Seppänen, V.; Leppänen, M.: *Government Enterprise Architecture Grid Adaptation in Finland*. In *Hawaii International Conference on System Sciences*. Los Alamitos, CA, USA. 2009. IEEE Computer Society.
- [VST03] Vasconcelos, A.; Sousa, P.; Tribolet, J. M.: *Information System Architectures: Representation, Planning and Evaluation. Journal of Systemics, Cybernetics and Informatics*. 1(6):78–84. 2003.
- [VST05] Vasconcelos, A.; Sousa, P.; Tribolet, J. M.: *Information System Architecture Evaluation: From Software to Enterprise Level Approaches*. In *12<sup>th</sup> European Conference On Information Technology Evaluation (ECITE 2005)*. 2005.
- [VST07] Vasconcelos, A.; Sousa, P.; Tribolet, J. M.: *Information System Architecture Metrics: an Enterprise Engineering Evaluation Approach. The Electronic Journal Information Systems Evaluation*. 10(1):91–122. 2007.
- [VST08] Vasconcelos, A.; Sousa, P.; Tribolet, J. M.: *Enterprise Architecture Analysis - An Information System Evaluation Approach. Enterprise Modelling and Information Systems Architectures*. 3(2):31–53. 2008.

- 
- [W308] W3C: *Scalable Vector Graphics (SVG) Tiny 1.2 Specification – W3C Recommendation 22 December 2008*. <http://www.w3.org/TR/SVGTiny12/> (cited 2011-07-02). 2008.
- [Wa05] Wagter, R.; van den Berg, M.; Luijpers, J.; van Steenberg, M.: *Dynamic Enterprise Architecture: How to Make IT Work*. John Wiley. 2005.
- [WBV07] van de Weerd, I.; Brinkkemper, S.; Versendaal, J.: *Concepts for Incremental Method Evolution: Empirical Exploration and Validation in Requirements Management*. In (Krogstie, J.; Opdahl, A.; Sindre, G., Ed.): *Advanced Information Systems Engineering*. Lecture Notes in Computer Science. pages 469–484. Springer Berlin / Heidelberg. 2007.
- [We02] Wegmann, A.: *The Systemic Enterprise Architecture Methodology (SEAM)*. Technical report. EPFL. 2002.
- [We03] Wegmann, A.: *On the Systemic Enterprise Architecture Methodology (SEAM)*. In *SEAM*. Published at the International Conference on Enterprise Information Systems 2003 (ICEIS 2003). pages 483–490. 2003.
- [We06] van de Weerd, I. v. d.; Brinkkemper, S.; Souer, J.; Versendaal, J.: *A Situational Implementation Method for Web-based Content Management System-applications: Method Engineering and Validation in Practice*. *Software Process: Improvement and Practice*. 11(5):521–538. 2006.
- [We07a] Wegmann, A.; Lê, L.-S.; Regev, G.; Wood, B.: *Enterprise modeling using the foundation concepts of the RM-ODP ISO/ITU standard*. *Inf. Syst. E-Business Management*. 5(4):397–413. 2007.
- [We07b] Wegmann, A.; Regev, G.; Rychkova, I.; Lê, L.-S.; Julia, P.: *Business and IT Alignment with SEAM for Enterprise Architecture*. In *11<sup>th</sup> IEEE International EDOC Conference (EDOC 2007)*. pages 111–121. 2007.
- [We08] Wegmann, A.; Kotsalainen, A.; Matthey, L.; Regev, G.; Giannattasio, A.: *Augmenting the Zachman Enterprise Architecture Framework with a Systemic Conceptualization*. In *12<sup>th</sup> IEEE International EDOC Conference (EDOC 2008)*. pages 3–13. 2008.
- [WF06] Winter, R.; Fischer, R.: *Essential Layers, Artifacts, and Dependencies of Enterprise Architecture*. In *1<sup>st</sup> Workshop on Trends in Enterprise Architecture Research (TEAR 2006)*. page 30. Washington, DC, USA. 2006. IEEE Computer Society.
- [WF07] Winter, R.; Fischer, R.: *Essential Layers, Artifacts, and Dependencies of Enterprise Architecture*. *Journal of Enterprise Architecture*. 3(2):7–18. 2007.
- [WH07] Wilde, T.; Hess, T.: *Forschungsmethoden der Wirtschaftsinformatik*. *WIRTSCHAFTSINFORMATIK*. 49(4):280–287. 2007.
- [Wi07] Wittenburg, A.; Matthes, F.; Fischer, F.; Hallermeier, T.: *Building an integrated IT governance platform at the BMW Group*. *International Journal Business Process Integration and Management*. 2(4). 2007.

- [Wi08a] Winter, R.: *Business Engineering – Betriebswirtschaftliche Konstruktionslehre und ihre Anwendung in der Informationslogistik*. In (Dinter, B.; Winter, R., Ed.): *Integrierte Informationslogistik*. pages 17–38. Berlin, Heidelberg, Germany. 2008. Springer.
- [Wi08b] Winter, R.: *Design science research in Europe*. *European Journal of Information Systems*. 17:470–475. 2008.
- [Wi09a] Winter, R.; Krcmar, H.; Sinz, E. J.; Zelewski, S.; Hevner, A. R.: *What in Fact is Fundamental Research in Business and Information Systems Engineering*. *Business & Information Systems Engineering*. 1(2):192–199. 2009.
- [Wi09b] Winter, R.; vom Brocke, J.; Fettke, P.; Loos, P.; Junginger, S.; Moser, C.; Keller, W. et al.: *Patterns in Business and Information Systems Engineering*. *Business & Information Systems Engineering*. pages 468–474. 2009.
- [Wi10] Winter, K.; Buckl, S.; Matthes, F.; Schweda, C. M.: *Investigating the state-of-the-art in enterprise architecture management method in literature and practice*. In *Proceedings of the 5th Mediterranean Conference on Information Systems*. 2010.
- [Wo10] Wout, J. v.; Waage, M.; Hartman, H.; Stahlecker, M.; Hofman, A.: *The Integrated Architecture Framework Explained*. Springer. 2010.
- [WR04] Weill, P.; Ross, J. W.: *IT Governance – How Top Performers Manage IT Decision Rights for Superior Results*. Harvard Business School Press. Boston, Massachusetts, USA. 2004.
- [WS08] Winter, R.; Schelp, J.: *Enterprise architecture governance: the need for a business-to-IT approach*. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*. pages 548–552. New York, NY, USA. 2008. ACM.
- [WW02] Webster, J.; Watson, R. T.: *Analyzing the Past to Prepare for the Future: Writing a Literature Review*. *MIS Quarterly*. 26(2):xiii–xxiii. 2002.
- [WWES92] Walls, J. G.; Widmeyer, G. R.; El Sawy, O. A.: *Building an Information System Design Theory for Vigilant EIS*. *INFORMATION SYSTEMS RESEARCH*. 3(1):36–59. 1992.
- [WWES04] Walls, J. G.; Widmeyer, G. R.; El Sawy, O. A.: *Assessing Information System Design Theory in Perspective: How Useful Was Our 1992 Rendition?* *Journal of Information Technology Theory and Practice*. 6(2):43–58. 2004.
- [YM94] Yu, E. S. K.; Mylopoulos, J.: *From E-R to "A-R" – Modelling Strategic Actor Relationships for Business Process Reengineering*. In (Loucopoulos, P., Ed.): *Entity-Relationship Approach (ER'94) – Business Modelling and Re-Engineering (Proceedings of 13th Int. Conf. on the Entity-Relationship Approach, Manchester, U.K., December 1994)*. pages 548–565. Lecture Notes in Computer Science n0. 881, Springer. 1994.
- [YSD06] Yu, E. S. K.; Strohmaier, M.; Deng, X.: *Exploring Intentional Modeling and Analysis for Enterprise Architecture*. In *1<sup>st</sup> Workshop on Trends in Enterprise Architecture Research (TEAR 2006)*. page 32. Hong Kong. 2006. IEEE Computer Society Press.

- [Yu96] Yu, E. S. K.: *Modelling strategic relationships for process reengineering*. PhD thesis. University of Toronto. Toronto, Ont., Canada. 1996.
- [Za87] Zachman, J. A.: *A framework for information systems architecture*. *IBM Syst. J.* 26(3):276–292. 1987.
- [Za09] Zachman Institute for Framework Advancement: *The Zachman Enterprise Framework*. <http://www.zachmaninternational.com/index.php/the-zachman-framework> (cited 2011-02-14). 2009.
- [Za10] Zacarias, M.; Pinto, H. S.; Magalhães, R.; Tribolet, J.: *A 'context-aware' and agent-centric perspective for the alignment between individuals and organizations*. *Inf. Syst.* 35:441–466. 2010.
- [Ze08] Zelewski, S.: *Erkenntnisinstrumente der Betriebswirtschaftslehre*. In (Corsten, H.; Reiß, M., Ed.): *Betriebswirtschaftslehre – Band 1*. München, Germany. 2008. Oldenbourg.





ACMM	Architecture Capability Maturity Model, page 121
ADM	architecture development method, page 83
ADOben	business engineering navigator, page 109
ARIS	Architecture of Integrated Information Systems, page 76
ASMs	abstract state machines, page 96
ATAM	architecture trade-off analysis method, page 93
BEAMS	building blocks for EA management solutions, page 115
BIO	business information officer, page 246
BMM	Business Motivation Model, page 174
BPEL	business process execution language, page 168
BPEL	business process execution language, page 240
BPMN	business process model and notation, page 168
CISR	Center for Information Systems Research, page 90
CMDB	configuration management data base, page 182
CobIT	Control Objectives for Information and Related Technology, page 165
CONOPS	Concept of Operations, page 103
DoDAF	Department of Defense Architecture Framework, page 72

DSS	decision support system, page 38
DYAMM	Dynamic Architecture Maturity Matrix, page 121
E2A	Extended Enterprise Architecture, page 87
E2AMM	Extended Enterprise Architecture Maturity Model, page 87
EA	enterprise architecture, page 1
EAAF	Enterprise Architecture Assessment Framework, page 121
EACMM	Enterprise Architecture Capability Maturity Model, page 121
EAMM	Enterprise Architecture Maturity Model, page 121
EAMML	enterprise architecture management method library, page 213
EAMPC	EA management pattern catalog, page 141
EAMPC	enterprise architecture management pattern catalog, page 5
EAMTS 2008	Enterprise Architecture Management Tool Survey 2008, page 117
EAP	Enterprise Architecture Program, page 87
EEM	Enterprise Engineering Methodology, page 78
eEPC	extended event driven process chains, page 168
EETs	Enterprise Engineering Tools, page 79
ELMF	enterprise-level management functions, page 179
EMLs	Enterprise Modeling Languages, page 78
EMOs	Enterprise Modules, page 79
EMs	Enterprise Models, page 79
EOS	Enterprise Operational Systems, page 80
EPC	event-driven process chain, page 76
GEMCs	Generic Enterprise Modeling Concepts, page 78
GERA	Generalised Enterprise Reference Architecture, page 78
GERAM	Generalised Enterprise Reference Architecture and Methodology, page 78
GoM	guidelines for modeling, page 128

---

I-Pattern	Information model pattern, page 142
IBB	information model building block, page 146
IDEF3	integrated definition for process description capture method, page 168
IEC	International Electrotechnical Commission, page 54
IEEE	Institute of Electrical and Electronics Engineers, page 54
IFAC	International Federation of Automatic Control, page 78
IFEAD	Institute For Enterprise Architecture Developments, page 87
IFIP	International Federation of Information Processing, page 78
IM	information model, page 187
IS	information systems, page 1
ISO	International Organization for Standardization, page 54
ITIL	IT infrastructure library, page 182
JSON	JavaScript Object Notation, page 239
KM	knowledge management, page 61
KPIs	key performance indicators, page 2
LBB	language building block, page 144
M-Pattern	Methodology pattern, page 142
MBB	method building block, page 144
MIT	Massachusetts Institute of Technology, page 90
NAF	NATO Architecture Framework, page 72
OCL	Object Constraint Language, page 162
OMG	Object Management Group, page 169
PDCA cycle	plan, do, check, and act, page 60
PDF	Adobe® Portable Document Format, page 239
PEMs	Partial Enterprise Models, page 79
RACI	responsible, accountable, consulted, and informed, page 165

## Bibliography

---

SD	strategic dependency, page 139
SEAM	systemic enterprise architecture methodology, page 95
sebis	software engineering for business information systems, page 5
SOM	Semantic Object Model, page 81
SR	strategic rationale, page 139
STREAM	Speedy, Traceable, Result-driven Enterprise Architecture Management, page 89
SVG	Scalable Vector Graphics, page 239
TAFIM	Technical Architecture Framework for Information Management, page 83
TOGAF	The Open Group Architecture Framework, page 83
UML	Unified Modeling Language, page 162
UML	unified modeling language, page 168
URN	User requirements notation, page 139
V-Pattern	Viewpoint pattern, page 142
VBB	viewpoint building block, page 146
VSM	viable systems model, page 64
WYSIWYG	what you see is what you get, page 234
YAWL	yet another workflow language, page 168

- actor, 149, 158, 205, 224, 263
- administration method, 8, 9, 50, 159, 161, 223, 225, 228
  - conceptualize problem, 216
  - decompose pattern, 220
  - detail solution, 217
  - integrate MBBs, 221
- analyze & evaluate, 149, 151, 220, 245
- analyze stakeholder involvement, 206
- application & information layer, 2
- ArchiMate, 97
- architectural description, 54, 57
- ARIS, 76
- assertion
  - for context and goal, 44
  - for solution, 44
  - general, 44
- BEAMS, 9
  - configurator, 8–10, 227, 232, 234, 239, 264
- building block, 5, 7, 144
- building blocks for EA management solutions, 9
- business & organization layer, 2
- business capabilities, 2
- business services, 2
- catalog
  - of goals, 216
  - of meta-attributes, 194, 220
  - of organizational contexts, 182, 215, 276
  - of participants, 190, 218, 219
- communicate & enact, 149, 151, 220, 244, 247
- concern, 56, 122
- configure & adapt, 149, 152
- control flow, 162, 170
  - guard, 162
  - split, 162, 170
  - union, 162, 170
- cross-cutting aspect, 176
- current state, 68–70, 150
- customization approach, 4
- definition, 13
- delta analysis, 70
- design artifact, 34, 47
- design rationale, 46
- design theory, 25, 26
  - artifact evolution, 34
  - context domain, 26
  - expository instantiation, 46
  - principles of adaptation, 33
  - principles of implementation, 47
  - problem domain, 26
  - solution domain, 26
  - testable implementation hypotheses, 47
  - testable solution model hypotheses, 32
- design theory nexus, 35
- design theory nexus instantiation, 25, 35, 50
- develop & describe, 149, 151, 220, 244

- development method, 3, 8–10, 50, 155, 158, 159, 173, 223, 228, 264
  - adapt and evolve EA management function, 161, 210
  - analyze EA management function, 156, 158, 161, 206, 228, 253, 255
  - characterize situation, 156, 158, 161, 173, 182, 210, 228, 242, 243, 250, 254, 258
  - configure EA management function, 156, 158, 161, 186, 200, 210, 228, 251, 255
  - configure MBBs, 186, 200, 202, 258
  - determine organizational context, 182
  - identify EA-related problem, 182, 184
  - select MBB, 200
  - specify information sources, 182, 185
- development process, 34
- EA management, 152
- EA management pattern
  - consequence, 215
  - context, 215
  - force, 215
  - problem, 214
  - solution, 215
- EA management governance, 66
- EA management pattern, 141
- EA model, 55
- enterprise architect, 128
- enterprise architecture, 2, 54, 72
- epistemology, 19
- GERAM, 78
- goal, 122, 174
- greenfield approach, 4
- guard, 170
- identify applicable MBBs, 196
- identify evolution paths, 197
- identify ineligible MBBs, 196
- implicit relationship, 224
- increase transparency, 255
- information model, 59, 180, 186
  - variable, 186
- information sink, 162, 163, 170
- information source, 162
- information system generator, 37
- infrastructure & data layer, 2
- infrastructure services, 2
- integration approach, 4
- investigate stakeholder-actor-dependencies, 207
- linguistically compatible, 32
- linguistically diverse, 32
- management function, 4
- mental conceptualization, 59
- meta-attribute, 177, 180
- meta-conceptualization, 34, 50
- method, 3, 24, 135
- method base, 8, 9, 146, 157, 159, 223, 264
- method building block, 155, 158, 161, 223
  - alternative, 154, 155, 193
  - condition, 194
  - consequence, 164
  - force, 164
  - integrable, 154, 155, 193
  - related, 154, 155, 193
  - relationship, 173, 193
  - relationships, 154
- method engineering, 136
- method fragment, 194
- method fragments, 137
- mixinMBB, 179, 224, 263
- model, 56
- modeling language, 3, 9, 24
- nexus-based IS generator, 38, 45, 50
- notation, 166, 167, 170
  - function, 188
- organization-specific configuration, 161, 182, 193
- organizational context, 158, 176
- participant, 146, 165, 169
  - consulted, 165, 169
  - informed, 165, 169
  - responsible, 165, 169
  - variable, 147, 186, 190
- pattern, 29
  - based documentation, 214
- planned state, 68–70, 150
- post-condition, 154, 158, 177, 224, 263

- 
- pre-condition, 154, 158, 177, 224, 263
  - principles, 68, 69
  - principles & standards, 2
  - problem, 122, 158, 174, 175
  - propose organizational interventions, 208
  - questions, 69
  - questions & KPIs, 2
  - representation, 166, 167, 170
    - function, 188
  - research design, 16
  - research method
    - application & evaluation, 11
    - eliciting method requirements, 43
    - exploring the problem and context domain, 42
    - formulating assertions, 43, 44
    - investigating solution models, 43
    - meta-conceptualization the solution domain, 43, 44
    - meta-conceptualizing the domains, 42, 43
    - nexus instantiation, 11
    - nexus-driven artifact design, 46
    - nexus-driven artifact evaluation, 47
    - nexus-driven evaluation, 46
    - nexus-driven problem elicitation, 46
    - problem diagnosis, 11
    - structuring the domains, 42
  - SEAM, 95
  - situational method engineering, 137
  - SOM, 81
  - stakeholder, 56, 149, 158, 205, 224, 263
  - strategies & projects, 2
  - subsumes, 187
    - read, 189
    - update, 189
  - target state, 68–70, 150
  - task, 135, 146, 148, 162, 169
  - technique
    - assessment, 195, 196, 210, 224
    - characterization, 180, 185
    - liveliness, 199, 210, 224
    - stakeholder involvement, 206
    - stakeholder-actor-dependency, 206
  - theoretical evaluation, 45
  - theory, 25
  - TOGAF, 83
  - trigger, 163, 170
    - event, 163, 170
    - temporal, 163, 170
    - variable, 186, 190
  - variable, 9, 147, 158, 224, 263
    - information model, 148
    - trigger, 148
    - viewpoint, 147
  - view, 57
  - viewpoint, 56, 167, 170
    - variable, 186
  - visions & goals, 2
  - Zachman Framework, 74

