Ingo Arnold

Enterprise Architecture Function

A Pattern Language for Planning, Design and Execution

Springer

Enterprise Architecture Function

Ingo Arnold

Enterprise Architecture Function

A Pattern Language for Planning, Design and Execution

Springer

Ingo Arnold
Riehen, Switzerland

ISBN 978-3-030-84588-9 ISBN 978-3-030-84589-6 (eBook)
https://doi.org/10.1007/978-3-030-84589-6

This Springer imprint is published by the registered company Springer Nature Switzerland AG.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

For my family, my loved ones and for my wonderful sons Jonas and Luis who followed their father's path into computer science.

Preface

Architecture Has Become All-Pervasive in the Twenty-First Century

The topics of *architecture* in general and *enterprise architecture* in particular have become ubiquitous in the twenty-first century. The networked universal computing machine (*the world computer*) has directly or indirectly permeated virtually every aspect of our lives, transforming *the way we cooperate, partner, create, and run businesses and plan, implement, and provide goods and services. What were originally distinguished as* business *and* technology *are merging and constituting a common* business model *backbone that is neither distinguishable in form nor in essence.* At the same time, we find that dealing with, if not mastering, complexity has become the central challenge for modern enterprises. Successfully planning, designing, and operating business models today is increasingly synonymous with establishing and evolving their digital means and building materials. In summary, business ecosystems have gained significantly in potential through digital means on the one hand but have also become inherently more complex on the other—both in their breadth (*many touchpoints*) and in their depth (*complex touchpoints*). Architecture has become ever-present because of its enormous relevance for the enterprise-wide planning, construction, and operation of digital material constituting digitalized services and means.[1] Job titles on business cards include *enterprise*

[1] In terms of digital means, the industry delineates two terms that sound very similar but have important differences in their conception: *digitalization* and *digitization*. Gartner, for example, defines digitalization in its IT Glossary as follows: "Digitalization is the use of digital technologies to change a business model and provide new revenue and value-producing opportunities; it is the process of moving to a digital business." Similarly, definitions of digitization have been proposed, with Gartner defining the term in their IT Glossary as follows: "Digitization is the process of changing from analogue to digital form, also known as digital enablement. Said another way, digitization takes an analogue process and changes it to a digital form without any different-in-kind changes to the process itself" (Gartner Glossary 2020).

architecture, *architecture management*, *business*, and *domain architecture*, or *enterprise solution architecture*. Architects participate in steering board meetings, contribute to strategic decision-making, create roadmaps, identify consolidation opportunities, or identify systematic shortcomings from a perspective that goes beyond individual services. Although the term *enterprise architecture* is so commonly used, a closer look reveals that architects, strategists, portfolio managers, project managers, and other stakeholders lack a common understanding of its conception.

At the Same Time Architecture Is Interpreted in Lots of Different Ways

For some, *enterprise architecture* is an architecture super-discipline that encompasses all other disciplines, like domain, solution, security, or data architecture; for others, it is a framework or tool; for still others, it is part of the corporate police department trying to make the good work of others hurt; for many, *enterprise architecture* may be a binder of drawings geared to help a manager convince others that he is in charge and in control and that his yard is in beautiful order. In its practical use, the term *enterprise architecture* covers quite a wide field—that is, it is not uniformly defined or understood.

My Own Architecture Journey

I have spent my entire professional life in the architecture arena. While practically designing and delivering concrete solutions caught my initial attention, over time I broadened my perspective of architecture as a strategic planning and governance discipline. Ten years ago, I wrote a book on *solution architecture* with three friends (Vogel et al. 2011). The goal of the book was to strike a good balance between the aspects of holistic orientation, theoretical substance, and practical guidance—a consolidation of our architecture experiences at that time. When we thought about writing the book, the topic of solution architecture lacked a holistic framework that established a common, unified terminology and described architecture in a differentiated way. We had all been intuitively searching for a framework that covered the essential solution architecture dimensions for a long time. Throughout our professional lives and educational journeys, each of us had developed our architecture understanding from individual insights. With the book we wrote, we finally reached

the point where we reconciled our individual insights and finally merged them into a common architecture orientation framework that became the core of our book.

Ten years have passed since then, and I decided it was the right time to summarize my experience in the field of *enterprise architecture* and write the book you are holding in your hands. However, there is a difference between the situation of solution architecture in 2011 and the topic of enterprise architecture in 2021. In the area of solution architecture, a framework gap existed, and we suffered from lacking an orientation model that was timeless, agnostic to technology, and that provided a sustainable reference beyond temporary architecture trends. In the area of enterprise architecture, the situation today is almost exactly the other way around. Numerous enterprise architecture frameworks claim to cover the topic in whole or in part, and an enormous amount of effort has been and is being expended to not only create the corresponding paperwork but to build an entire industry on it.

Initially I Was Enthusiastic About Enterprise Architecture Frameworks. . .

When I started my journey into enterprise architecture, I found that there were already several frameworks. They included definitions of what architecture means, suggested view models, development processes, roles and responsibilities, artifacts, and nomenclatures for modeling and more. I considered myself fortunate to be able to build on proven practices, most of which were offered in a comprehensive, unique package. These frameworks impressed me a lot—not only but also because of their sheer size, emphasis on inherent structures, and the list of prominent companies that have contributed to them and claim to use them themselves. And I was not alone. Academic and non-academic publications, enterprise architecture experts and consultants, architecture tool and platform vendors, and practitioners in the field—basically everyone I met or everything I read (a few exceptions included)—sang one song in unison: "Enterprise architecture frameworks have everything you need—just do it."

Since an *enterprise architecture function*[2] has full responsibility for the governance and execution of the architecture within an enterprise, corresponding

[2] Enterprise architecture functions establish architecture capabilities organizationally—they are socio-technical systems that realize architecture disciplines by integrating their full breadth and depth into an enterprise-wide operating model.

frameworks are basically blueprints that identify all aspects that a well-designed architecture function generally needs to consider.

When I speak of *enterprise architecture frameworks* here, I am not thinking of Zachman,[3] *The Open Group Architecture Framework*[4] (TOGAF), or ArchiMate[5] in the narrowest possible sense—i.e., not in the sense that their cores are released by respective consortia and then eke out a quiet existence on bookshelves or on their official websites. A complementary (yet essential) addition to *naked* TOGAF or ArchiMate is a vast commercial ecosystem, an ecosystem that includes consulting services; tons of publications, books, tools, and platforms that feed respective tool vendors; and an architecture certification industry. In addition, there are countless architecture practitioners who have gone to the trouble of acquiring certifications—and who want something in return for the investment they have made. Finally, those practitioners whose frustration with using such frameworks increased over time also remain stabilizing factors. While frustrated practitioners, on the one hand, have painfully learned that an architecture framework is not "all you need" and that it is not enough to "just do it," these practitioners, on the other hand, have also learned how useful the authoritative power of dropping framework names can be in the pursuit of their own personal agendas.

. . .Until I Realized a Widely Spread Misconception

When I first started my enterprise architecture journey, I quickly climbed the learning curve thanks to these frameworks, delighted in what my colleagues had to say, and happily joined in the enterprise architecture song myself. I trusted in the wisdom of crowds. However, despite my initial enthusiasm, I painfully realized over the years that the claim of frameworks to be all you need is false. There were simply too many disappointments, frustrations, and merely declared successes (which were in fact failures) to continue to ignore the evidence of a significant gap between the

[3] The Zachman Framework proposes a basic enterprise architecture schema that provides a structured approach to holistically viewing and defining an enterprise's architecture. The basic scheme distinguishes two dimensions that introduce the intersection of two classifications. While the first classification distinguishes the primitive interrogatives what, how, when, who, where, and why, the second classification is derived from the philosophical concept of reification (i.e., from the process of transforming an abstract idea into its instantiation and vice versa).

[4] The Open Group Architecture Framework (TOGAF). TOGAF proposes an approach to design, plan, implement, and govern enterprise architecture. It describes a generic method for developing architectures. TOGAF suggests a common vocabulary, a generic information model, an adaptable role model, general architecture artifacts, and tooling.

[5] ArchiMate is an open and independent modeling standard for enterprise architectures that supports the description, analysis, and presentation of architecture within and between architectures. ArchiMate is an open group standard and is based on fundamental architectural concepts defined by the IEEE (i.e., IEEE 1471).

frameworks themselves and their successful operationalization in a specific business context.

Do not get me wrong. Enterprise architecture frameworks are extremely useful in many ways. They do a good job of aggregating architecture best practices, generic knowledge, and reference material and making them compactly available to the architecture community. The fact that frameworks exist and are widely accepted sends a silent, albeit clear, message to all: do not reinvent the wheel where proven practices exist. While it is naïve to recklessly ignore best practices, believing that an architecture practice or framework is in itself a *solution* that is fit for purpose in a concrete environment without adaptation is similarly naïve.

When I reached the peak of my frustration curve, I took stock of *what* I saw going wrong frequently when using architecture frameworks. Next, I analyzed *why* this was happening. At a very high level, the *what* can be traced to one, surprisingly common, misconception: the extremely unfounded expectation that a generic solution will completely satisfy the requirements of a specific problem. Generic solutions such as architecture frameworks propose generic best practices, techniques, and other responses to appropriately generic problems. However, concrete problems must be addressed by concrete solutions. Where concrete problems differ, concrete solutions differ as well.

When I wanted to understand *why* this is such a widespread misconception, I found it due to a paradox: The impressive amount of material that frameworks contain, their apparent completeness, supposed comprehensiveness, as well as their structuredness, loudly confirmed by a huge commercial ecosystem, give the strong impression that this is *all you ever need* and that you *do not need to do or adapt anything else*. I have seen architects with years of experience naively (i.e., 1:1) adopt generic framework practices for their concrete solutions. The real paradox is that the richer the framework, the worse it is applied. It is the heightened version of the well-known "a fool with a tool is still a fool" aphorism: "a fool with a tool is an armed fool"—armed in the sense that the fool has successfully immunized himself against the insight of being a fool in the first place.

Problems in Operationalizing Enterprise Architecture Frameworks

Obviously, the naive use of generic reference models almost inevitably leads to suboptimal concrete solutions. To better understand this phenomenon, I have held generic frameworks against concrete architecture function designs over the past 10 years and analyzed frequently observed, or experienced, shortcomings or even complete failures. Here, it did not matter whether it was an initial function design or an evolution to improve its fitness. Based on the analysis, I narrowed down those aspects that disproportionately determine success versus failure. It is imperative that

these be designed correctly and therefore deserve our undivided attention—so let me give you specifics on some of the misconceptions.

For example, when adopting frameworks, architecture functions do not consider the complete service lifecycle as equally relevant. Enterprise architecture frameworks tend to overemphasize planning aspects (*plan*) while at the same time underemphasizing their contribution to continuous service improvement (*build* and *run*). Another faulty notion to which architecture functions succumb under overwhelming framework impression is to focus too much on the internal makeup of a function (*white-box perspective*) without considering the expectations of their actual stakeholders (*black-box perspective*). Frameworks also largely conceal their concrete establishment in the form of an organizational capability. As a result, they lack references to *evolutionary fitness* as an important design maxim for architecture functions. This in turn leads to regular organization restarts, which undermines trust and acceptance. To give another example, frameworks seem taxonomically complete at first glance. Accordingly, architecture functions often expend little or no effort in creating a company-specific, stringent, and unambiguous conceptual as well as terminological foundation. This inevitably creates a relevant gap in understanding since the central conceptions of each business are simultaneously subsets and supersets of a respective framework glossary. Architecture functions pay the price for this bitterly and yet at the same time almost unnoticeably: in the form of inefficient, ineffective, and mostly never-ending debates, which in turn lead to suboptimal decisions and their protracted consequences. Furthermore, the frameworks often lack references to differentiations that are important in practice. For example, many frameworks do not give any indication of the *architecture levels*[6] to be differentiated in a company or do not adequately separate the *perform mandates* of an architecture organization from its *governance*[7] *mandates*. This, in turn, leads to imprecise organizational boundaries. As a final example of the inadequate adoption of frameworks by architecture functions, consider the inconsistent and disconnected definition of practices, like *architecture methods*, *view models*, *patterns*, *principles*, and *roadmaps*. These are often implemented incompletely, vaguely, and thus inconsistently and are isolated from each other, which leaves their potentials insufficiently exploited.

Aim of This Book

The blurred line between critical and semi-critical success factors in frameworks showed that we do not need larger or more frameworks but that the central gap appears where the concrete design of enterprise architecture functions is concerned. For this reason, I have placed the architecture function at the center of consideration.

[6] See the *Architecture Level* pattern in this book's pattern catalog (Chap. 6).

[7] See the *Architecture Governance* pattern in this book's pattern catalog.

In other words, I make it a first-class citizen and view the topics of architecture and frameworks through the ocular of their organizational incarnation.

Specifically, my book proposes how to design an architecture function so that it ultimately meets the expectations of the business for effectively developing, delivering, as well as operating digitalized services. My book will further outline what roles, services and processes, elaboration and reference methods, disciplines, or artifacts an architecture function is responsible for. It will make tangible to you the overall value proposition of an architecture function, as well as introduce an enterprise operating model into which architecture integrates as an organizational capability. In doing so, I do not reduce the term *architecture function* to the realm of IT but remain largely agnostic in the objective as well as normative portions of my book—agnostic regarding the nature of the contributions (e.g., business versus IT), the process models and attitudes employed (e.g., agile, waterfall, DevOps), the size of the enterprise, and other determining factors.

When an enterprise architecture framework is a blueprint for the subject of enterprise architecture, then my book is a blueprint for an architecture function enabling a digitalized business in the twenty-first century.

In any book, beyond *what* is to be described, the way in which (*how*) it presents its content and offers guidance and orientation accordingly is significant. For my book, the challenge was, on the one hand, to capture the full breadth and depth of the topic of architecture function design and, on the other hand, not to reduce the multifaceted nature of the theme to an oversimplified formula or a monodimensional set of function building blocks. In addition to typical framework content, establishing organizational capabilities is not only about the methods, roles, activities, artifacts, and responsibilities of an architecture discipline but also about its capacity, funding, mandate, engagement model, or adequate organizational embedding in a surrounding operating model, to name just a few examples. Because my book, unlike frameworks, claims to literally generate an architecture function design tailored to its context, I decided to use a pattern language regarding the *how* of my book. I deliberately chose a pattern language and patterns because they have a high degree of familiarity among architects, scale well, but also have the flexibility to accommodate design-determining factors of very different kinds in a single design proposal.

I encourage all readers to follow the navigational structures of my book closely. These structures allow you to fully grasp an architecture function in its holism and plasticity and thus to plan, design, and ultimately operate it accordingly. You will further round out and condense your understanding of what is presented in a pattern by following its references to associated patterns. In this way, you iteratively increase your understanding of architecture functions in general and develop a fully customized design for your own architecture function in particular.

Overall, this book is an expression of my desire for a work that meaningfully structures the design of an enterprise architecture function while providing hands-on guidance for practitioners. In particular, the book is independent of any particular enterprise architecture framework, mindset, tool, or platform—thus timeless in that regard. It belongs to that group of foundational works that provide you with a stable

and future-proof reference that transcends current or future trends in enterprise architecture schools. Writing this book demanded an intensive and in-depth examination of the subject of architecture beyond the usually isolated considerations of individual aspects. During the time I planned, designed, and wrote this book, I learned a great deal and steadily broadened my own perspective and experience. On the one hand, I drew on my own experience; on the other hand, I discussed with many enterprise architects and with colleagues in my network who held and still hold senior architecture positions in a variety of multinational companies and across many industries. I also had challenging conversations with students and people freshly entering the field of architecture. All these exchanges helped me to look at the subject from many different, new, and fresh angles. As a result of these fruitful debates, I gained valuable knowledge and a deeper understanding regarding the design and operation of an enterprise architecture function.

What you hold in your hands is my approach of organizing and explaining architecture from the perspective of an architecture function, putting it on a solid conceptual foundation and merging it into a pattern language. I sincerely hope that this book will help you build and evolve your own specific architecture function, customized to the needs of your business. You are most welcome to share your experiences, successes, as well as failures, or just questions, with me. Please let me know where my book has been of great help to you. But please also be sure to let me know where you have unanswered design questions even after reading the book. I am extremely curious to hear from you about how you fared on your own personal architecture function journey.

Riehen, Switzerland Ingo Arnold

References

Vogel, Oliver, Ingo Arnold, Arif Chugtai, Timo Kehrer, *Software Architecture: A Comprehensive Framework and Guide for Practitioners*, Springer-Verlag, Berlin, 2011

Gartner, *Gartner Glossary*, https://www.gartner.com/en/information-technology/glossary, 2020

Contents

1 Introduction 1
- 1.1 Starting Position 1
- 1.2 Aims of the Book 10
- 1.3 Architecture 14
- 1.4 Enterprise Architecture 18
- 1.5 Enterprise Architecture Function 20
- 1.6 Pattern Language 21
- 1.7 Reader Guide 23
  - 1.7.1 Book Architecture 23
  - 1.7.2 Target Audience 26
  - 1.7.3 Chapter Overview 28
  - 1.7.4 Chapters in Detail 28

Further Reading 30

2 Architecture Function Pattern Language 31
- 2.1 Overview 31
- 2.2 Pattern 32
- 2.3 Pattern Catalog 36
- 2.4 Pattern Language 36
- 2.5 Architecture Function Pattern Language 41
  - 2.5.1 Architecture Function Pattern Topology 43
  - 2.5.2 Architecture Function Pattern Catalog 47
  - 2.5.3 Architecture Function Pattern Ontology 49
- 2.6 Architecture Function Pattern Language Adoption 50

Further Reading 54

3 Architecture Function: Context 57
- 3.1 Overview 57
- 3.2 Business Model 59
- 3.3 Operating Model 60
- 3.4 Value Chain 62

3.5 Organization . . . 64
3.6 Digitalization . . . 68
3.7 Service . . . 74
3.8 Enterprise . . . 75
3.8.1 Enterprise Organization . . . 76
3.8.2 Enterprise Value Chain . . . 77
Further Reading . . . 92

4 Architecture Function: Challenge . . . 95
4.1 Overview . . . 95
4.2 Architecture Function . . . 97
4.2.1 Architecture Function Vision and Mission . . . 98
4.2.2 Architecture Function Organization . . . 99
4.2.3 Architecture Function Engagement Model . . . 100
4.2.4 Architecture Function Communication . . . 101
4.2.5 Architecture Function Governance . . . 102
4.2.6 Architecture Function Roadmap . . . 103
4.2.7 Architecture Function Apparatus . . . 103
4.3 Architecture Function Services . . . 104
4.3.1 All Value Streams . . . 106
4.3.2 Service Planning Value Stream . . . 109
4.3.3 Service Building Value Stream . . . 114
4.3.4 Service Running Value Stream . . . 116
4.4 Architecture Function Qualities . . . 118
4.4.1 Architecture Function Usability . . . 119
4.4.2 Architecture Function Elasticity . . . 120
4.4.3 Architecture Function Evolvability . . . 121
4.4.4 Architecture Function Reliability . . . 122
Further Reading . . . 123

5 Architecture Function: Constitution . . . 125
5.1 Overview . . . 125
5.2 Architecture Function . . . 127
5.2.1 Architecture Function Vision and Mission . . . 128
5.2.2 Architecture Function Organization . . . 131
5.2.3 Architecture Function Engagement Model . . . 135
5.2.4 Architecture Function Communication . . . 137
5.2.5 Architecture Function Governance . . . 138
5.2.6 Architecture Function Roadmap . . . 140
5.2.7 Architecture Function Apparatus . . . 142
5.3 Architecture Function Services . . . 143
5.3.1 All Value Streams . . . 145
5.3.2 Service Planning Value Stream . . . 152
5.3.3 Service Building Value Stream . . . 173
5.3.4 Service Running Value Stream . . . 181

5.4 Architecture Function Qualities 184
5.4.1 Architecture Function Usability 185
5.4.2 Architecture Function Elasticity 187
5.4.3 Architecture Function Evolvability 188
5.4.4 Architecture Function Reliability 190
Further Reading 192

6 Pattern Catalog 193
6.1 Overview 193
6.2 Architecture Organization 195
6.2.1 Organizational Replication 195
6.2.2 Learning Organization 203
6.2.3 Architecture Maturity 207
6.2.4 Architecture Capacity 218
6.2.5 Architecture Role 222
6.2.6 Architecture Ownership 231
6.2.7 Architecture Funding 237
6.2.8 Architecture Sourcing 241
6.3 Architecture Engagement Model 249
6.3.1 Architecture SPOC 250
6.4 Architecture Discipline 254
6.4.1 Enterprise Architecture Discipline 254
6.4.2 Domain Architecture Discipline 260
6.4.3 Solution Architecture Discipline 264
6.5 Architecture Governance 267
6.5.1 Architecture Governance 268
6.5.2 Managed Architecture Evolution 274
6.5.3 Architecture Policy 281
6.5.4 Architecture Calendar 286
6.5.5 Domain-Organization Agnosticism 290
6.5.6 Decommissioning Reward 294
6.5.7 Architecture Decision 299
6.5.8 Architecture Traceability 304
6.6 Architecture Communication 308
6.6.1 Architecture Language 309
6.6.2 Domain Taxonomy 318
6.6.3 Architecture Innovation 325
6.7 Architecture Objective 329
6.7.1 Need to Know 329
6.7.2 Architecture Mandate 334
6.8 Architecture Asset 341
6.8.1 Architecture Demarcation 342
6.8.2 Architecture Asset 346
6.8.3 Landscape Asset 353

6.8.4 Off-the-Shelf Solution 362
6.9 Architecture Elaboration 366
6.9.1 Architecture Significance 367
6.9.2 Architecture Level 373
6.9.3 Architecture Condition 377
6.9.4 Architecture Alternative 385
6.9.5 Architecture Approach 393
6.9.6 Business Versus Technical 400
6.9.7 Architecture Artifact 405
6.9.8 Baseline Architecture Versus Target Architecture 411
6.10 Architecture Apparatus 415
6.10.1 Your Own Architecture Methodology 416
6.10.2 Architecture Methodology Adapter 424
6.10.3 Your Own Architecture View Model 430
6.10.4 Domain Architecture Methodology 444
6.10.5 Solution Architecture Methodology 450
6.10.6 Architecture Assessment Methodology 462
6.10.7 Reference Architecture Methodology 470
6.10.8 Architecture Roadmap Methodology 479
6.10.9 Architecture Pattern Methodology 489
6.10.10 Architecture Principle 501
6.10.11 Your own Architecture Platform 504
Further Reading 513

Index 517

About the Author

Ingo Arnold shaped the *enterprise architecture function* of a global Fortune 50s in the pharmaceutical industry and held a variety of architecture management positions over the course of 25 years. In addition to his roles in industry, Ingo is an assistant professor at universities in Switzerland and formerly in Germany, sharing his insights and experiences with several generations of computer science students. As the author of books on solution architecture, Ingo covers virtually the entire architecture spectrum of modern enterprises. Finally, Ingo is a well-known speaker at conferences, where he gives talks to international audiences on topics such as architecture management, governance, enterprise, solution, and security architecture.

List of Abbreviations

AADL	Architecture Analysis and Design Language
ABB	Architecture Building Block
ACID	Atomicity, Concurrency, Isolation, Durability
ADL	Architecture Description Language
ATAM	Architecture Trade-off Analysis Method
ATM	Automatic Teller Machine
BLOB	Binary Large Object
BPMN	Business Process Model and Notation
CAPEX	Capital Expenditure
CMM	Capability Maturity Model
CMS/CMDB	Configuration Management System or Database
COBIT	Control Objectives for Information and Related Technologies
CoP	Community of Practice
CSI	Continual Service Improvement
DoD	Definition of Done
DSL	Domain-Specific Language
EAM	Enterprise Architecture Management
ERP	Enterprise Resource Planning
GIOP	General Inter-ORB Protocol
I18N	Internationalization
IAM	Identity and Access Management
IEEE	Institute of Electrical and Electronics Engineers
IoC	Inversion of Control
IoT	Internet of Things
IT4IT	IT for IT
ITIL	Information Technology Infrastructure Library
KPI	Key Performance Indicator
MDA	Model-Driven Architecture
MIB	Management Information Base
ML	Machine Learning
NFR	Non-functional Requirement

OODBMS	Object-Oriented Database Management Systems
Open CA	Open Certified Architect
OPEX	Operational Expenditure
ORM	Object-Relational Mapping
PaaS	Platform as a Service
QAS	Quality Attribute Scenarios
RACI	Responsible, Accountable, Consulted, Informed
RDBMS	Relational Database Management System
RMI	Remote Method Invocation
ROI	Return on Invest
RPC	Remote Procedure Call
SaaS	Software as a Service
SEI	Software Engineering Institute
SLA	Service-Level Agreement
SPOC	Single Point of Contact
SWOT	Strengths, Weaknesses, Opportunities, Threats
TCO	Total Cost of Ownership
TOGAF	The Open Group Architecture Framework
UML	Unified Modeling Language

Chapter 1
Introduction

Abstract This chapter positions the topic of enterprise architecture and focuses on the architecture function as an organizational capability. It explains the starting position and the goals of this book and introduces key concepts that are further explored throughout it. The chapter concludes with an overview of the intended audience, the internal architecture of the book itself, and the contributions of each chapter. After reading this chapter, you will know how enterprise architecture, domain architecture, and solution architecture are differentiated from each other and how they holistically combine in an architecture function to form an overall organizational capability. You will know the intent of this book and how to use it to critically reflect on the architecture function in your organization, plan its construction or renewal, design and implement it, and ultimately operate it effectively.

1.1 Starting Position

Business in the Twenty-First Century

We find ourselves in extremely turbulent times of fundamental transformations encompassing all habitats and areas of life. In the twenty-first century, humanity is confronted on the one hand with the consequences of its rapid growth and its short-sighted overharvesting of the globe. It is therefore facing Herculean tasks in areas such as food, water, climate, energy, or biodiversity as well as crises and coordination processes around these, which demand a dizzying speed of adaptation from their societies. At the same time, we have at our disposal impressive technologies and the innovative power of a global knowledge community that has reached a level of organization and interconnectedness that is unique in the history of mankind.

Beyond impressive innovations in areas such as bio- and neurotechnology, energy, aerospace, or agrotechnology, the universal computing machine (i.e., the computer in all its shapes, sizes, and areas of application) connected to the world-wide web has become a fundamental game changer, catalyzing virtually all other technical and non-technical areas of life.

The networked universal computing machine (i.e., the world computer) has directly or indirectly penetrated virtually all of our spheres of life and work and, in

I. Arnold, *Enterprise Architecture Function*,
https://doi.org/10.1007/978-3-030-84589-6_1

just a few decades, has very fundamentally changed how we network socially, acquire and impart knowledge, optimize ourselves medically, plan vacations, travel and orient ourselves in the world, organize mobility, meet new partners, entertain ourselves, or participate in social discourse and political processes. And, of course, the world computer, and the plethora of specific applications and information we constructed on its basis, has changed the way we cooperate, join forces, start and run businesses, and plan, create, and provide goods and services. At the same time, the world computer has changed how we demand goods and services and use them in cumulative value creation processes to further refine them into our own services. It allows us to completely rethink the hitherto valid axiomatic basis of economic models and confronts us with questionings of fundamental premises concerning our legal norms—questioning that seemed unthinkable to us since the times of the *ius civile*,[1] simply because technically impossible. The networked universal computing machine has stripped traditional business models of their premises virtually overnight, thus depriving business of its foundation. It has changed our demands as well as our demand-satisfaction rituals so fundamentally that long-practiced social processes and traditions have lost their meaning or been replaced by entirely new models of demand-satisfaction at a speed that is nothing short of breathtaking. In a nutshell, transformative technologies, directly or indirectly catalyzed by the world computer in just a few decades, have led to upheavals of established business and operating models and the disappearance of entire industries but also to new social habits, rituals, and needs as well as models of demand-satisfaction that we would never have dared to dream of before. As special and worthy of consideration as each of the transformation phenomena outlined here may be, in our observations, reflections, and discussions today, we use the term *digitization* to refer, grosso modo,[2] to our ability to completely rethink long entrenched premises, beliefs, and social as well as business rituals and processes.

Enterprises are increasingly relying on *digital means*[3] as elementary foundations of their business models. As a result, companies in certain industries (e.g., finance, insurance, entertainment and music, telecommunications) have successively

[1] *Ius civile*. Even though Roman law was initially a body of law that arose from many years of practice without written laws, so-called customary law, very early on ancient Rome created a universally applicable set of rules that has gone down in history as *ius civile*. In the *ius civile*, fundamental legal concepts and goods were defined and regulated, which still influence our legal norms today (e.g., property conception, commercial law, rights to political participation).

[2] *Grosso modo* is an adverbial locution whose etymological roots are in Latin "grossus" (*approximately*) and "modus" (*way*, *method*). In English, *grosso modo* mostly enters in the meaning of *roughly*, *approximately*, or *coarse*.

[3] In terms of *digital means*, the industry clearly delineates two terms that sound very similar but have important differences in their conception: digitalization and digitization. Gartner, for example, defines digitalization in its IT Glossary as: "Digitalization is the use of digital technologies to change a business model and provide new revenue and value-producing opportunities; it is the process of moving to a digital business." Similarly, definitions of digitization have been proposed, with Gartner defining the term in their IT Glossary as follows: "Digitization is the process of changing from analogue to digital form, also known as digital enablement. Said another way,

transformed their originally analog business models into digitalized ones. New technologies (or innovations regarding the use of technologies) are emerging at an increasingly rapid pace, enabling business model transformations on an unprecedented scale and disruptiveness, turning established analog business models upside down, and replacing traditional players in a market with new and innovative entrepreneurs. You are certainly familiar with these famous examples where analog business models have been replaced by digitalized ones that function in a fundamentally different way. For example, in that, digital business and operating models have decoupled from space and time, where analog ones are still tied to them: "Facebook, the world's most popular media owner, creates no content. Alibaba, the most valuable retailer, has no inventory. And Airbnb, the world's largest accommodation provider, owns no real estate" (Goodwin 2015).

Svyatoslav Kotusev observes in *The Practice of Enterprise Architecture* (Kotusev 2018) that with the steady growth, use, and proliferation of digital means, organizations are transforming into socio-technical systems, where a clear demarcation between business and business-enabling digital technology is increasingly blurring or disappearing. What was originally distinguished as business and technology are merging and constituting a common backbone of business models that is neither distinguishable in form nor in essence. From this perspective, we can already view most of today's enterprises as socio-technical systems that ultimately form a carefully tuned conglomerate of organization and people, processes for shaping efficient and effective cooperation, and digital means and technologies united by common enterprise goals and mission.

A brief look at the anatomy of digitized, socio-technical systems reveals their potential. However, this consideration also gives an impression of the inherent complexity of socio-technical systems, with which enterprises must find a creative and novel way of dealing. The digitization of socio-technical systems initially implies that elementary means and tools (*resources*) are increasingly based on data and networked computers that manage and utilize this data. On the one hand, these resources are used as components in services; on the other hand, they are building blocks in a value chain for planning, designing, implementing, deploying, and operating these services. The fact that resources in an enterprise are increasingly digitized and flow into virtually all the essential building blocks of the enterprise value chain digitalizes (directly or indirectly) the entire cooperative system—i.e., the functions and disciplines (*organization*), *processes*, *services*, and *service offerings*. An effect of the digital material of all building blocks that should not be underestimated is that they can be replicated virtually infinitely often and quickly and are flexibly adaptable and highly scalable, with marginal costs close to zero and negligible space-time expansion. The digital material of modern enterprise building blocks travels across digital networks, enabling both duplication and teleportation of digital and semi-digital building blocks in a matter of seconds (Fig. 1.1).

---

digitization takes an analogue process and changes it to a digital form without any different-in-kind changes to the process itself" (Gartner Glossary 2020).

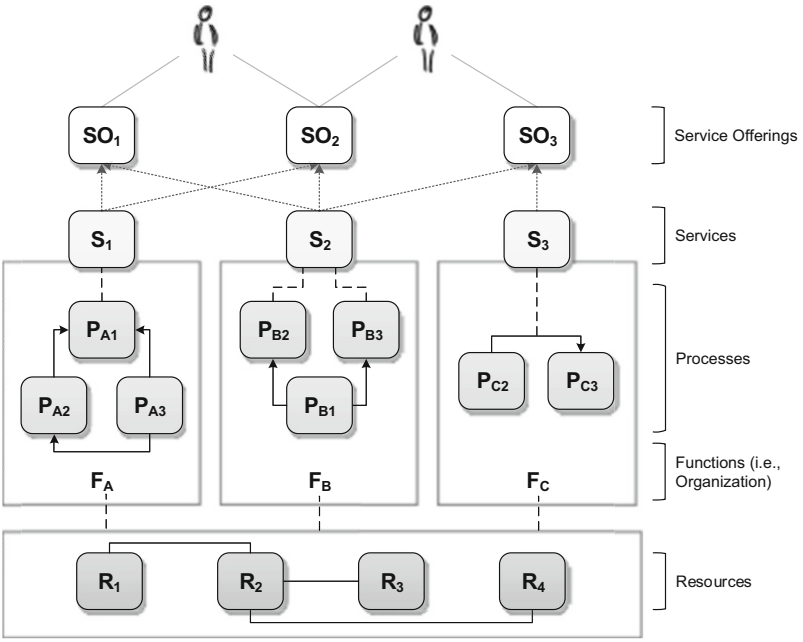

Fig. 1.1 Digitized business in the twenty-first century

Putting all this together, we can say that differentiation between business and IT has (meanwhile) become artificial at least if not obsolete altogether. At the same time, we note that dealing with, if not mastering, complexity has become the central challenge for enterprises in the twenty-first century.

Challenges for Enterprises in the Twenty-First Century

In *Collaborative Enterprise Architecture: Enriching EA with Lean, Agile, and Enterprise 2.0 Practices*, Stefan Bente, Uwe Bombosch, and Shailendra Langade (Bente et al. 2012) state that complexity in enterprises results from a combination of opportunities and challenges that can be directly or indirectly attributed to the phenomenon of advancing digitization. Whether changes in the business context, such as government regulation, mergers, and acquisitions, or whether new business models or changes in the range of enterprise resources used to run the business (e.g., new technologies, retirement, replacement or consolidation of resources, or supplier changes), these are all complexity drivers in modern enterprises. In addition, there are other drivers and challenges, such as ensuring unconditional quality with respect to all of the above resources, for example, their evolutionary fitness, where they need to be easily adapted to changing business contexts (i.e., modifiability, adaptability), or their reliability, where adequate scalability, availability, performance, or security must be ensured. But also an efficient and effective engineering discipline for

digitized resources increases the complexity compared to the analog pendant, because—unlike in traditional engineering disciplines—important material of digital resources is model and thus language. Moreover, the engineering of a product does not end (anymore) with its delivery—rather, the engineering continues while the resources remain in use. While in traditional engineering disciplines the requirement to turn a car into an anesthetic gun by flipping a switch leads to sheer horror on the engineering side, such requirements are at least conceivable in the case of systems whose building blocks are digital in nature and thus data-based. Another complexity driver is the segregation of business as well as globalized, cooperative processes, which leads to a segregation of resources in central enterprise structures as well as on the level of their change vehicles (e.g., projects). In particular, segregation undermines desired synergies and economies of scale. Also, the fact that actual assets (e.g., information) often take a back seat to pseudo-assets (e.g., applications), or are scattered across them, often leads to fundamental misconceptions in the sense of confusion between ends and means. In summary, it can be said that business ecosystems have experienced significant growth in potential as a result of digital materials on the one hand but have also become inherently more complex on the other—both in their breadth (i.e., *many touchpoints* in processes organized on the basis of the division of labor) and in their depth (i.e., *complex touchpoints*).

Potential of Architecture to Address Enterprise Challenges

In the course of this introductory chapter, I will further elaborate on the distinction between enterprise, domain, and solution architecture. In addition, throughout the book, I will distinguish between architecture as a subject, architecture as a discipline, and architecture as an organizational capability (i.e., architecture function). At the same time, however, I will ignore such distinctions where they do not contribute to understanding, or subsume them under the term *architecture*.

In summary, we can say that architecture makes significant contributions in the areas of enterprise strategic development (e.g., through contributions to strategy formulation or strategic alignment of the enterprise), strategic planning (e.g., through architecture baseline, target and roadmap development, or transformation planning), enterprise portfolio management (e.g., through contributions to investment planning, prioritization, risk, time, and resource planning), as well as in the areas of solution lifecycle, project, and service development (e.g., through contributions to project and service management, service architecture and design, implementation, and operations). If we shorten this view to the essentials of architecture, then architecture is both a portfolio and planning discipline as well as a transformation and implementation discipline.

For example, the architecture discipline addresses the impact of change initiatives on applications in an enterprise, the realization of organizational capabilities through applications, or the optimal support of enterprise services through applications. Other examples of architecture contributions include planning and implementing

services based on processes; analyzing the impact of decommissioning[4] or changing applications, services, or platforms; and analyzing the propagation of errors across systems. Architecture is also concerned with identifying optimization potentials in the operational assets (i.e., landscape assets) of a company—for example, gaps or redundancies in services with regard to their support of an enterprise business model. Finally, architecture contributes to reducing complexity, identifying and resolving overlapping responsibilities, and ensuring that landscape assets[5] such as services, applications, and platforms comply with regulatory constraints.

The successful orientation, design, and operation of business models today is increasingly synonymous with the establishment and evolution of their digital means and building materials, enabling the adaptation and evolution of modern business models to ever-changing environmental conditions. This new blurring of the distinction between a business model and the digital means on the basis of which it is realized requires a disciplined and systematic approach to planning, developing, delivering, and operating enterprise services. Architecture is a cornerstone discipline here, ensuring both intelligent decision-making and sustainable implementation. It increases both the effectiveness and efficiency of planning, developing, and operating enterprise services. Architecture achieves this by providing the transparency needed to make the right decisions, translate decisions into quality solutions, and to deploy those solutions correctly. Furthermore, the architecture identifies risks and proposes remedial actions; promotes the reuse of intellectual, conceptual, and physical assets to reduce redundancy and exploit synergies in the enterprise; and adequately balances conflicting expectations of time-to-market versus quality requirements. Architecture establishes effective communication between business implementation and business-enabling disciplines, like the information technology discipline. In this respect, it acts as a change agent on a journey that transforms an enterprise from a state of being overwhelmed by complexity to a rationally organized state that enables it to respond efficiently and effectively to the challenges of a digitized world. The contributions of the architecture discipline presented here demonstrate its invaluable importance to the effective operation of modern enterprises.

When Architecture Fails to Fully Leverage and Contribute Its Potential

While the importance of architecture and its potential contributions are apparent on the one hand, it is by no means certain that they will be recognized and brought to full fruition in every organization. Thus, an inadequate understanding or definition of architecture leads to an unclear architecture mandate and to reducing architecture to a purely technical discipline, without recognizing that the architecture discipline could make equivalent contributions to business domains. Furthermore, an insufficiently developed understanding leads to architecture learning and standardization taking place in silos, without architecture being understood as a holistic discipline. In

[4] See the *Decommissioning Reward* pattern in this book's pattern catalog (Chap. 6).

[5] See the *Landscape Asset* pattern in this book's pattern catalog.

summary, in such enterprises architecture is misinterpreted as having no impact on the business, leading to this potential being left untapped. Another common problem in enterprises is an ineffective or completely absent architecture organization. That is, architecture understanding and skills are either non-existent, underdeveloped, or lack traction to drive appropriate change. Other symptoms include the lack of a systematic approach to driving and approving architecture decisions, inconsistent tool base and metrics, and unilaterally balanced incentive systems.

A culture of firefighting replaces serious planning when there is a lack of alignment between business and information technology departments. Holistic, transversal, and fully integrated architecture plans are either not developed or do not have the buy-in of relevant parties. In such organizations, a focus on short-term success at the expense of sustained pursuit of longer-term plans can be observed, as well as an uncontrolled proliferation of redundant enterprise assets and processes. Architecture organizations must succeed in demonstrating that architecture costs and investments are distinctly linked to business outcomes. Here it is particularly important to monetize both—costs and benefits—not just in the short term but in the medium and long term. Both costs and benefits should consider monetized risk, architecture debt[6] and solution complexity, as well as architecture development, operations, and maintenance.

According to Bente (Bente et al. 2012), there are four deficiencies that result from an insufficiently established enterprise architecture capacity[7]: architecture does not achieve an adequate impact (e.g., reducing complexity or making enterprise operations more cost- and resource-efficient), architecture does not fully exploit its contribution potential (e.g., architecture contributions from strategy through to operations), architecture fails to evolve with an ever-changing business ecosystem, and architecture fails to address the enterprise as a whole. In addition, Carsten Sensler and Thomas Grimm (Sensler and Grimm 2015) describe in *Business Enterprise Architecture: Praxishandbuch zur digitalen Transformation in Unternehmen* prerequisites for a successful enterprise architecture, which—if they are missing—lead to architecture not unfolding its full potential. As prerequisites, Sensler and Grimm see recognition of a problem that requires substantial improvement, such as enterprise transformation (i.e., organizational will), sufficient leadership support and resources to address the problem (i.e., feasibility), and a master plan that appropriately describes the desired transformation goal (i.e., vision and plan).

Exploiting the Potential of Architecture

So the million-dollar question that arises from the above deficit considerations is: how can we ensure that architecture fully realizes its potential to effectively make valuable business contributions in an enterprise? The short answer is that effective

[6] Architecture debt (also known as technical or design debt) is a concept that reflects the cost of future rework. In other words, costs incurred by choosing a simple solution now rather than an approach that is more resilient to rework in the future. In other words, costs incurred today to avoid rework costs in the future.

[7] See the *Architecture Capacity* pattern in this book's pattern catalog (Chap. 6).

architecture is achieved by embedding it as an organizational capability in an enterprise operating model. However, what is an enterprise operating model? In simple terms, an operating model defines and standardizes the cooperation within and between enterprise functions with the goal of optimizing them in both efficiency and effectiveness.[8] An operating model typically standardizes enterprise processes, roles, and responsibilities (i.e., disciplines), information that is exchanged across processes, and means (i.e., resources) to facilitate cross-process collaboration, coordination, and information exchange (Fig. 1.2).

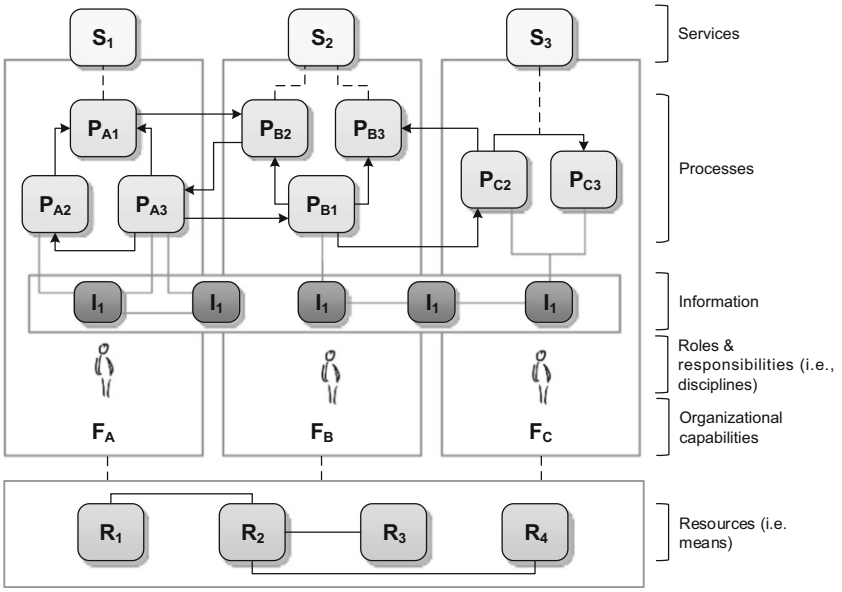

Fig. 1.2 Operation model anatomy

But how is architecture practically embedded as an organizational capability in an enterprise-wide operating model? The short answer here is by establishing an enterprise architecture function and integrating it with other enterprise functions. But how is this achieved in a meaningful way? Enterprise architecture as a topic has a high degree of maturity and is well developed. Many standardized resources exist, such as architecture frameworks, consulting services, or platforms and tools. Thus, standardized enterprise architecture frameworks provide comprehensive guidance. They define core concepts and recommend practices for planning, developing, maintaining, and executing architecture. Also they offer a variety of components

[8] Efficiency versus effectiveness. Something is effective to the extent that it achieves an intended result. In contrast, something is efficient to the extent that it achieves something with minimal use of resources. It is possible to be effective without being efficient and vice versa.

such as generic architecture view models, artifacts, processes, principles, or maturity metrics.[9] The genericity of architecture frameworks requires their adaptation to concrete problems, which means that they are helpful heuristics—but no off-the-shelf solutions. An example of such a framework is the Zachman Framework for Enterprise Architectures,[10] which is named after its founder John Zachman and is one of the first frameworks in the field of enterprise architecture. Another framework is called TOGAF,[11] which is an industry- and vendor-neutral as well as community-based framework maintained by the Open Group.

The question that arises is whether these architecture resources already represent an effective architecture function in a concrete enterprise context or allow an immediate derivation of it. Generic architecture resources such as enterprise architecture frameworks represent architecture as a topic in a structured and generic way (i.e., at the class level) and are thus immensely useful—but not yet guarantees of success. Successful architecture is always enterprise-specific, i.e., a function of the organization's industry, geographic distribution, work culture, or business model. This in turn means that generic architecture resources such as frameworks must always be adapted. Therefore, it is best to think of architecture frameworks and other resources as a loose collection of best practices proposed by experienced architects for practitioners. If you pick and choose from an architecture framework what makes sense to you in your particular context and for your particular purpose, then you use the framework as intended. However, if you understand an architecture framework as something you should read to the letter or follow in an all-or-nothing manner, you are misinterpreting it. For selecting context-appropriate practices, techniques, and approaches from a framework, you must first learn and know the entire framework very well—only then will you have enough context to understand the components of the framework and use them deliberately. However, to realize the full potential of architecture, a framework must first be tailored to the needs of a particular organization and embedded in the organization's particular operating model. Ultimately, a key success factor of architecture is the degree to which collaboration between the architecture and its stakeholders is brought to life. Thus, to practically embed architecture in an enterprise operating model, an enterprise architecture function (i.e., architecture function) must be established and mandated—i.e., the generic architecture potential must be concretely offered and brought into an operating

[9] See the *Architecture Maturity* pattern in this book's pattern catalog (Chap. 6).

[10] The Zachman Framework proposes a basic enterprise architecture schema that provides a structured approach to holistically viewing and defining an enterprise's architecture. The basic scheme distinguishes two dimensions that introduce the intersection of two classifications. While the first classification distinguishes the primitive interrogatives what, how, when, who, where, and why, the second classification is derived from the philosophical concept of reification (i.e., from the process of transforming an abstract idea into its instantiation and vice versa).

[11] The Open Group Architecture Framework (TOGAF). TOGAF proposes an approach to design, plan, implement, and govern enterprise architecture. It describes a generic method for developing architectures. TOGAF suggests a common vocabulary, a generic information model, an adaptable role model, general architecture artifacts, and tooling.

model through the organizational capability of an architecture function. This means that an architecture function is the organizational entity for effectively embedding and operationalizing the architecture discipline in an enterprise.

When an enterprise fails to exploit its architecture potential, it is often because architecture functions do not adequately integrate the architecture discipline into the operating model. The background to this is the misunderstanding that a generic architecture discipline (e.g., in the form of generic frameworks) is already considered a complete or at least sufficient solution to a concrete organizational problem. This misunderstanding is based on a, intentionally or not, misleading suggestion of generic architecture resources, such as frameworks, literature, consulting services, or tools: the suggestion of their immediately operationalizable completeness. The need for as well as the process of instantiating generic frameworks into a well-designed architecture function is only marginally—in any case insufficiently—described in publications and resources.

1.2 Aims of the Book

Motivation

In a nutshell, closing the gap between generic architecture frameworks and an appropriately instantiated architecture function is the central motivation for writing this book. As invaluable as the potential of the architecture discipline is for addressing increasingly pressing challenges of our century, only an architecture function that is precisely integrated into the enterprise operating model will practically realize that potential.

My book therefore positions itself at the intersection of generic architecture resources and practices on the one hand and an enterprise-specific architecture function on the other. In the words of a software architect, one could say that this book provides a model-driven generator[12] of architecture functions. The generator (i.e., my book) presupposes general frameworks, tools, and practices and assists you in generating an architecture function that is optimally embedded in your enterprise operating model. The model underlying my book considers other discipline-specific enterprise organizations, an enterprise value chain, and enterprise services representing the company's market offering as environmental premises. As premises for the application of my book, I consider your desire to concretely establish, or renovate, an architecture function and integrate it into your operating model in such a

[12]Model-driven architecture (MDA) is a forward engineering approach in which executable or semi-executable artifacts are generated from abstract, human-made architectural models (e.g., class diagrams). MDA tools are used to develop, interpret, compare, align, measure, verify or transform models and metamodels. Generator-based architecture approaches very generally decouple solution specifications from their physical generation. They increase domain specificity and the degree of abstraction of the models that architects use to represent a solution. Generators receive architecture models as inputs and create artifacts at the source level or related physical structures as outputs.