

2 | Architecture Orientation Framework

In this chapter we present an explanatory model for dealing with architecture. This model provides orientation by positioning the significant elements of architecture in an architecture orientation framework using simple question words. The focal point of this framework is the role of the architect. We also use the framework to convey knowledge and experience throughout the rest of the book. It enables you to think about architecture in a structured way and to orient yourself.

Overview

2.1	Motivation	24
2.2	Overview of the Framework	26
2.3	Architectures and Architecture Disciplines (WHAT)	29
2.4	Architecture Perspectives (WHERE)	30
2.5	Architecture Requirements (WHY)	31
2.6	Architecture Means (WITH WHAT)	32
2.7	Organizations and Individuals (WHO)	34
2.8	Architecture Method (HOW)	35
2.9	Summary	36
	Further Reading	36

Basic concepts of the architecture orientation framework

Figure 2-1 shows the basic concepts covered in this chapter and visualizes how they relate to each other.

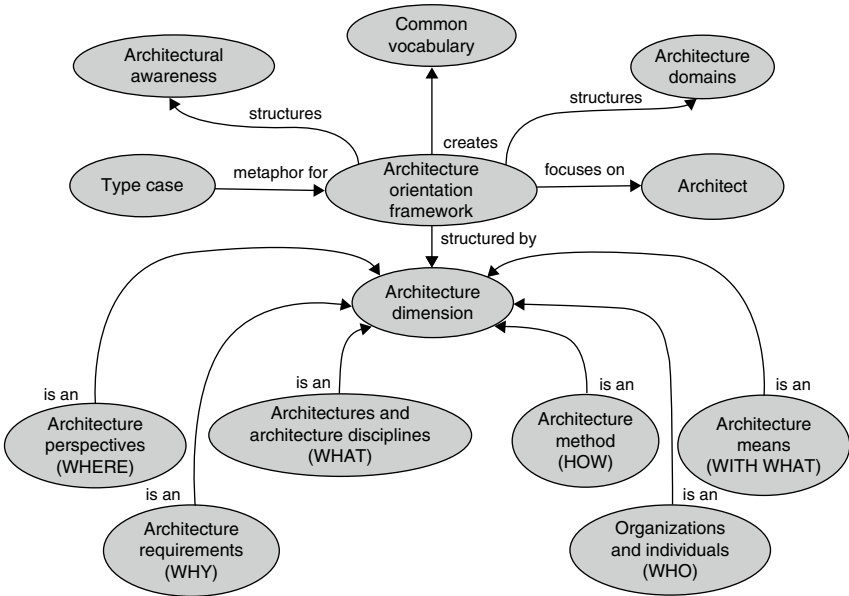


Figure 2-1: Basic concepts of the orientation framework

2.1 Motivation

Varied and dynamic environment

Architects work in a very varied and dynamic environment. New technologies are flooding onto the market, new tools promise increases in efficiency and productivity, lean methodologies promise risk-free project management, and new architecture concepts, such as service orientation and cloud computing, claim to reduce the inherent complexity of IT systems. As an architect, you must be able to understand, classify, and finally assess all of these developments and new features in order to select a suitable solution for your specific problem. You must therefore arrange and classify such new topics accordingly and compare them with your existing knowledge. In addition to mastering this flood of information, your tasks include making architectural decisions, defining guidelines, and managing your team professionally. You must also take on board customer requirements, analyze them, and design viable architectures. The selection of suitable products, and therefore the communication with suppliers, is also an important part of your role.

Develop architectural awareness

To be successful in this environment, you must be aware of these varied aspects—you must develop an *architectural awareness* that enables you to clas-

sify, evaluate, and put all aspects into an overarching and holistic architectural context. Every architect develops such a way of thinking about architecture over the course of his or her career. It reflects your understanding of architecture and enables you to structure your daily work. The quality of this awareness is relevant strategically and in the long term, since an architectural awareness is a basis for life-long learning and thus for being a successful architect. Concrete knowledge is of course important, but it can be learned and has a shorter life than an architectural awareness. Without this understanding, it is difficult to position, apply, and assess knowledge. During the course of your activities, you will also undergo experiences that you have to classify in the same way as your concrete knowledge. This will enable you to make better decisions in the future based on your wealth of experience, and to make these decisions more easily and more consciously.

Architectural awareness should be structured like a type case into which you can sort new experiences and new things learned, and retrieve them as and when they are required. “Learned” refers to the knowledge aspect of practicing as an architect. Architecture principles, styles, and patterns, but also specific platforms, such as JEE and .NET, fall into this category. “Experienced” covers specific experiences from the real world, for example, whether one of the afore-mentioned platforms works in practice, or how to deal with tensions within a project team. To stay with our metaphor, the type case supports the orderly arrangement of parts, where each section is a container for elements that share certain characteristics—thus a specific classification that they all have in common. This enables you to derive the general characteristics of new information learned and new experiences gained from the understanding of the characteristics of the section into which you have sorted them.

The structure of the architecture type case should take into account your varied fields of activities. It must therefore consider architecture in its entirety, and not, for example, restrict itself to primarily technical aspects. It is therefore important to place you the architect at the center of the consideration. The type case should also enable you to open further sections within a section, to guide your awareness to further structuring paths within a section, and to develop the type case further over time. In addition, despite having to be comprehensive and extensible, it must still be intuitive and understandable so that you can use it efficiently. You will only be able to act successfully in practice if you can explain the layout and structure of your architecture type case, and thus your holistic understanding of architecture, in simple words.

The type case represents a basic model for explaining the architecture domain and spans the framework within which you operate as an architect. Based on the previously defined requirements for comprehensiveness, extensibility, simplicity,

**Structure
architectural
awareness**

**Define architectural
structuring
characteristics**

**Derive the
architecture
orientation
framework**

ity, and understanding, the following sections present an architecture orientation framework that can be viewed as a type case.

2.2 Overview of the Framework

Basis of the architecture orientation framework

The framework presented below has arisen from visualizing the daily life of an architect and considering the requirements formulated in the previous section. A framework should be simple. It is therefore important to restrict yourself to the few most important dimensions, or rather main sections in the sense of the type case metaphor. However, at the same time, these dimensions should be extensive enough to be able to describe the varied nature of architecture. It should also be possible to subdivide the dimensions further in a useful way so that you can extend the framework. The framework must also be easy to understand and based on practice. So what distinguishes an architect in practice? In principle, you provide answers to questions and problems put to you by customers, team members, suppliers, or even questions you pose yourself. Therefore, an architecture orientation framework with a structure based on open question words is a sensible and practical approach.

Question words as main dimensions

The main dimensions of the framework are:

Table 2.2-1: Dimensions of the architecture orientation framework

Question word	Dimension	Explanation
WHAT	Architectures and architecture disciplines	The WHAT dimension contains basic principles and definitions of architecture. It therefore lays the basis for working as an architect. It also classifies architecture according to the various fields of activity in which architects work (e.g., software architecture, data architecture, or security architecture). Fundamental knowledge and experience belong to the WHAT dimension.
WHERE	Architecture perspectives	The WHERE dimension covers the different levels at which architecture takes place and the views with which architecture can be described. The use of different perspectives enables you to concentrate on one problem at a time. You use this dimension to include different ways of looking at things.
WHY	Architecture requirements	The WHY dimension is dedicated to the requirements IT systems must satisfy in general and architectures in particular. From the wealth of requirements you are confronted with, you must be able to identify those that are architecturally significant and design an architecture that meets these requirements. In the WHY dimension, you can arrange the requirements an architecture must satisfy.

Question word	Dimension	Explanation
WITH WHAT	Architecture means	The WITH WHAT dimension structures the different architecture means you can use whilst carrying out your trade. It thus enables you to classify different architecture means.
WHO	Organizations and individuals	The WHO dimension looks at the role of the architect and the influence of individuals and organizations on architecture. It examines the interaction between organizations, individuals, and architecture more closely. Considering this dimension allows you to act successfully. In the WHO dimension, you can include knowledge and experience from your social and organizational environment.
HOW	Architecture method	The HOW dimension structures the architecture method. It details the most important architectural activities that you perform during your work. Here you can store proven methods and access them again as and when necessary.

The framework can be visualized as shown in Figure 2.2-1. This image places you, the architect, at the center, and we use it repeatedly throughout the book to place the topic in question in the context of the framework, thus providing you with better orientation.

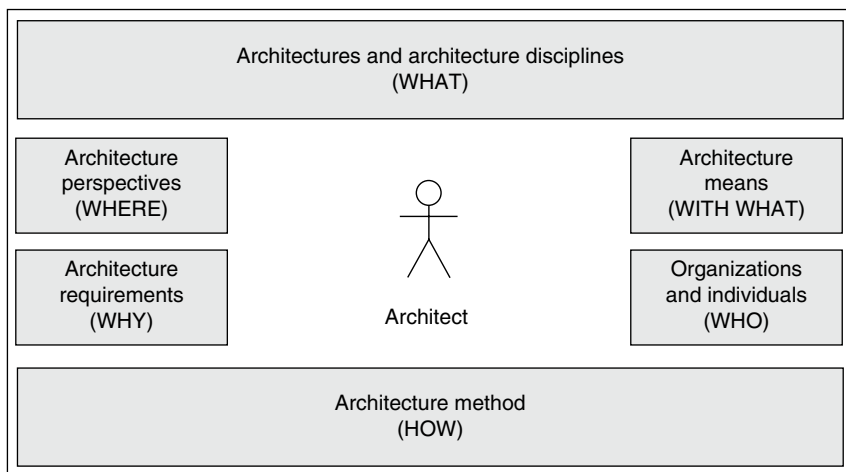


Figure 2.2-1: Overview of the architecture orientation framework

A framework structured according to the question words specified enables you to ask basic questions and thus in practice, enables simple and systematic orientation. Architectural activity can thus take place based on an explanatory model in

Framework in action

which you are aware of the different dimensions at any given time. In the course of a project, for example, during analysis, design, and implementation, you will continually ask yourself which means (WITH WHAT) to use in which way (HOW) to realize a specific requirement (WHY). For example, you document the desire for a distributed architecture during the analysis by holding a requirements analysis workshop (HOW), document it in a requirements document (WITH WHAT), and you guarantee it in the architecture design by using a corresponding architecture pattern (WITH WHAT). In addition, depending on the architecture discipline (WHAT), you will, for example, use different perspectives (WHERE) to consider relevant aspects of the IT system for the current activity. It is not possible to assign all aspects uniquely to one and only one dimension, since the aspects themselves are multi-dimensional. Methods such as the Unified Software Development Process (USDP) are a good example of this. They define, for example, a basic process on one hand, and on the other, document the means with which a system is realized and the perspectives from which it can be considered. In the sense of our architecture orientation framework, you should generally assign such methods to the HOW dimension and the other process-independent elements of the methods to the other dimensions. For orientation purposes, it is important that you establish criteria that enable you to make assignments to the dimensions. The basic question you have to answer is: *“What is the essence of the topic under consideration?”* Once you have answered the question, you can make an assignment.

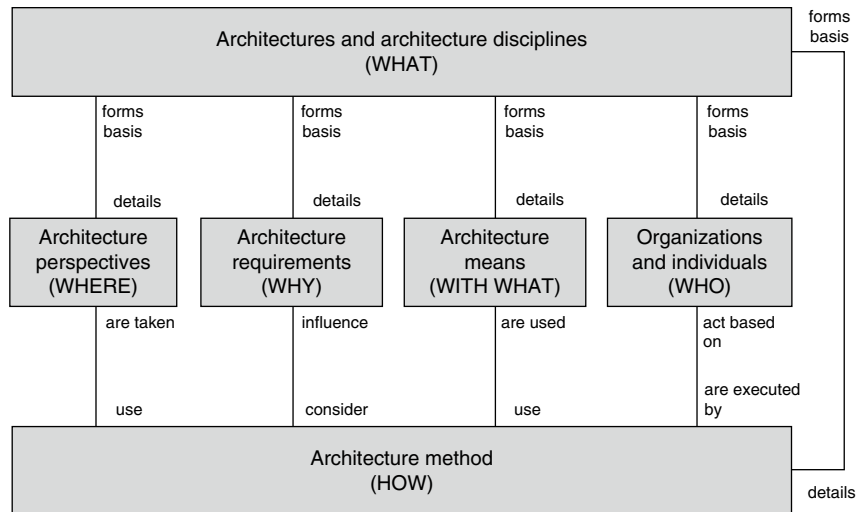


Figure 2.2-2: Relationships between the dimensions

The individual dimensions of the framework are related (see Figure 2.2-2). The WHAT dimension forms the basis for all other dimensions since it contains the basic architecture knowledge and important architecture definitions. All other dimensions detail the basic principles contained in the WHAT dimension. In an architecture method (HOW dimension), the elements of the other dimensions are also put into a methodological and time context. For example, an architecture method describes which architecture perspectives (WHERE dimension) have to be applied and which architecture means (WITH WHAT dimension) have to be selected to fulfill specific architecture requirements (WHY dimension). The architecture method also describes which activities are to be performed. Individuals (WHO dimension) base their actions on the method.

**Relationships
between the
dimensions**

Using this framework, you can establish a common vocabulary and understanding, which makes communication within the team easier. It therefore not only benefits individual architects in arranging their architectural thoughts, but also makes collaboration with others more efficient, since a common framework can reduce misunderstandings. The framework can therefore be a catalyst for successful collaboration within the team. Of course, the framework merely represents one possible model for thinking about architecture and arranging your thoughts. In our experience, however, the model is very practical and makes daily work easier. The following sections give an overview of the individual dimensions of the architecture orientation framework.

**Common vocabulary
and practical
suitability**

2.3 Architectures and Architecture Disciplines (WHAT)

The WHAT dimension is dedicated to basic architecture knowledge. The elements of this dimension enable you to explain the nature of architecture, define architecture, and relate architecture to other IT domains and non-IT domains, such as building architecture. With a good understanding of the basic terminology outlined here, you have the basis for understanding and working with the other dimensions.

**WHAT dimension
represents the
foundation**

Designing architectures successfully is a challenge given the inherent complexity of IT systems. Today, architectures must recognize the usual requirements such as reliability, availability, and scalability, and in addition, offer a basis for realizing functional requirements. You are thus faced with the challenge of considering various architectural influencing factors, such as functional and qualitative aspects, and balancing them out sufficiently for the specific problem at hand. In addition to a well-founded basic architectural knowledge, you also need deeper knowledge of specialist fields. For example, for the integration of IT systems, you must have a very good understanding of the integration platform to be implemented and possible integration approaches, such as message-based or

Varied aspects

process-based integration. This need for specialist knowledge has led to the establishment of various *architecture disciplines*. During the course of your career, you will generally decide to specialize in one of these disciplines. An architecture is thus often created as a team effort through the collaboration of architects from the individual disciplines.

Architecture disciplines

Therefore, in the WHAT dimension, in addition to looking at the basic principles of software architecture in detail, we briefly present further architecture disciplines. We do this in an overview in order to position the disciplines within the architecture orientation framework and to separate them from one another. We will look at the following disciplines:

- > Software architecture
- > Data architecture
- > Integration architecture
- > Network architecture
- > Security architecture
- > System management architecture
- > Enterprise architecture

We will examine the contents of the WHAT dimension in more detail in Chapter 3.

2.4 Architecture Perspectives (WHERE)

Concentration on perspectives

Architectural thinking and practice are complex. Psychological studies indicate that people can only process 7 ± 2 units of information simultaneously [Miller 1956]. Added together, the quantity of information covered by all aspects of an architecture vastly exceeds this figure. It is therefore extremely difficult to grasp the building blocks of a system, how they are grouped, how they interact, how they are distributed, and their behavior at runtime all at once. To be successful despite the restrictions of human understanding, you have to reduce the complexity by examining only one manageable part of an architecture at any one time.

Architecture levels

Architecture can take place at different levels. It is therefore important to always be clear about the level you are dealing with. This is the only way of applying useful means and disciplines for the architecture level in question. The levels possible range from organizations to systems all the way down to individual building blocks.

Architecture views and models

At each level, you can take different *architecture views* of a system. In their entirety, the views give a complementary image of the architecture to be implemented. *Architecture view models* enable you to look at architectures systematically and in a way that reduces their complexity for this purpose. They group relevant

views from which architectures are to be considered into one model, thus enabling them to be shown in their entirety. The *4 + 1 view* from Kruchten [Kruchten 2000] is an example of an architecture view model. Architecture frameworks, such as the *Zachman Framework* [Zachman 1987] and the *Reference Model for Open Distributed Processing (RM-ODP)* [ISO10746 1998], also contain architecture view models.

Chapter 4 discusses the individual architecture levels and views. It also takes a closer look at the different views of the architecture models mentioned.

2.5 Architecture Requirements (WHY)

For companies, information technology (IT) is a significant means for realizing their business strategies and supporting their operations. You therefore design IT systems and, as a consequence, architectures, not for their own purpose but with a specific business purpose in mind. The primary motivation for architecture is thus not technological elegance but the specific and long-term added value for the business. Of course, you can only achieve this added value if the IT system satisfies the functional requirements placed on it. However, an IT system that satisfies the functional requirements but does not appreciate the non-functional requirements will have no real benefit for the business. An e-commerce shop that satisfies all functional requirements but crashes with a high number of simultaneous users will not support the real business strategy and will probably not deliver any added value.

You must therefore ensure that the requirements placed on an IT system are supported by the underlying architecture of that system. It is essential that you know different types of requirements and their implications for architecture. In principle, there are functional and non-functional requirements. We can differentiate between the following types of requirements:

- > Organizational requirements
- > System requirements
- > Building block requirements
- > Development time requirements
- > Runtime requirements
- > Organizational constraints

The WHY dimension identifies and explains the different types of requirements. You can only design IT systems “fit for purpose” when you are aware of the different *requirements* and take them into account when practicing as an architect. We will discuss these different types of requirements in detail in Chapter 5.

Architecture is not an end in itself

Types of requirements

Relevance of architecture requirements

2.6 Architecture Means (WITH WHAT)

Elements of the WITH WHAT dimension

This dimension is dedicated to the means you use to design and implement your solutions. Using the type case metaphor, this section contains lots of smaller subsections in order to structure the large number of *architecture means* and make orientation easier. During the course of your career, you will continually add new means to these sections and remove obsolete ones. A means becomes obsolete when it is no longer relevant. The spectrum of possible architecture means ranges from fundamental principles to concrete technologies.

Principles

There are elementary means, which, when used and considered, are extremely relevant in establishing successful architectures. These belong to the category of *architecture principles*. One means in this category is the *separation of concerns principle*. Its aim is to clearly separate the responsibilities of building blocks. For example, a building block for visualizing data should not be responsible for saving it in a database. *Architecture principles* are important long term and should accompany every architectural activity. They embody fundamental architecture experiences.

Basic concepts

To ensure that architecture principles are also taken into account in an architecture, you can apply basic *concepts* that support these principles accordingly. You must therefore look at the different concepts and select the appropriate ones depending on the problem at hand. The architecture concepts include basic design and realization paradigms, such as object orientation and component orientation. Means that are based entirely on modeling and generation, such as model-driven software development or model-driven architecture [OMG 2010c], are also elements of this sub-dimension.

Tactics, styles, and patterns

In addition to considering elementary principles and concepts, we recommend having proven architecture solutions in your toolbox so that you can reuse them for similar problems. These solutions, which are based on *architecture principles*, belong to the family of *architecture tactics, styles, and patterns*. An *architecture tactic* helps you to get a first idea about a design problem. You can then develop this idea further. You can also use *styles* and *patterns*, for example, as further means. An *architecture style* documents a proven and successful way of structuring an architecture. Every style has specific characteristics and is a template for the design of the actual architecture. An *architecture style* is also an efficient documentation and communication tool, since the properties of the style used can be understood independently of the actual purpose of the system. There are various options for documenting *architecture styles*. One proven and recommended form is the documentation of the style as an *architecture pattern*. An *architecture pattern* describes *architecture styles* using a general structure. The authors of POSA1 and POSA2 have made a considerable contribution in

this area [Buschmann et al. 1996; Schmidt et al. 2000]. For example, one *architecture style* described in pattern form is the *layers architecture pattern*. It documents the arrangement of system building blocks at different levels. This arrangement achieves a clear separation of responsibilities and avoids a monolithic architecture [Buschmann et al. 1996]. The classic arrangement of presentation logic, business logic, and persistence logic on different layers is a well-known application of this pattern. Architecture styles and patterns are similar means with a different form of description.

A further type of means is *basic architectures*. Basic architectures use the previously identified architecture means in a larger context. Examples of such basic architectures are:

- > Cloud Computing Architectures
- > Dataflow architecture
- > Layered architecture
- > Middleware architecture
- > n-tier architecture
- > Rich client architecture
- > Service Oriented Architectures
- > Thin client architecture

Knowing these basic architectures enables you to expand your architecture knowledge and understanding—and thus arrive at an effective software architecture more quickly.

Architectures of complex systems have to solve several different architecture problems, or rather, balance them out appropriately. Therefore, several *architecture patterns* are used in combination. *Architecture patterns* are also architecture means that do not relate to any specific problem area. That is, they do not address, for example, specific characteristics of a call center architecture. Relying solely on *architecture patterns* to design a solution for such an architecture is therefore not sufficient. It is much more important to include complete architecture solutions in your toolbox as references. These *reference architectures* describe solutions that have been designed for a specific problem domain using different *architecture styles* or *architecture patterns*. They therefore reflect the highest degree of reusability of architectural knowledge and experience.

One very important factor in the success and acceptance of an architecture is that everyone involved (customer, project lead, software developer, etc.) understands and supports it. It is therefore essential that you communicate your ideas and approaches and model the architecture according to these ideas and approaches. To achieve this, you have to use the correct means to express the architecture.

Basic architectures

Reference architectures

Modeling means

These means can vary depending on the target group. For example, for a bid presentation, it may be sufficient to visualize the significant building blocks of an architecture using graphical elements. However, you will need more expressive means for the architecture design in order to exclude misunderstandings and take account of all significant architecture aspects, such as the structure and dynamics of an architecture. The means used in this context are used to model the architecture and belong to the family of *architecture modeling means*. One example of a widespread, standardized modeling means is the Unified Modeling Language (UML) from the Object Management Group [OMG 2007a].

Technologies

In addition to the architecture structures themselves, the selection of technologies that carry and support the architecture design in the actual implementation is a further important influencing factor for a successful architecture. Therefore, one of the trays of your toolbox should be dedicated to these *basic technologies*. Add new technologies to your toolbox frequently and remove technologies that have become obsolete. Databases, transaction monitors, and middleware can be included in this sub-dimension, for example. Furthermore, to implement an architecture successfully, it is very important that you know possible target platforms and consider their strengths and weaknesses when actually realizing the architecture on a concrete platform. Target platforms, such as Sun's Java Enterprise Edition or Microsoft's .NET, belong to the category of *component platforms* and are important design means for implementing architecture requirements such as scalability, availability, and reliability by providing elementary basic functionality.

Chapter 6 covers these topics

The realization and conscious use of these means make architectural activity easier and contribute considerably to the success of an architecture. Chapter 6 looks more closely at the WITH WHAT dimension, examining the individual architecture means in more detail. In this book, we restrict ourselves primarily to IT-related means. It is, however, also possible to place other means in this dimension, such as presentation and discussion techniques. These are useful for you in your communication with stakeholders.

2.7 Organizations and Individuals (WHO)

Interaction and communication as an architecture activity

Architectures are created by people. As an architect, you interact and communicate with many different groups of people in order to design an architecture. For example, you work closely with the customer and end users of the system to be developed in order to extract the architecturally significant requirements from the requirements placed on the system. You are also the first contact person for project leads, supporting them in the creation of project plans and effort estimations. You also lead project teams from a technical point of view, and act as the communicator and motivator of the architecture to be implemented.

To manage these tasks successfully, you need more than well-founded skills in technical and methodological topics. You must also have good social skills. Even the most elegant technical architecture idea cannot be realized if you cannot convince your team and customer of the idea. Unfortunately, not enough importance is placed on social skills in the role of an architect today, even though as far back as 1968 Melvin Conway raised the theory that an architecture is defined to a considerable extent by organizational influences [Conway 1968]. It is very important to be aware of these organizational influences and the required social skills. This is one of the key differentiators that make a technical specialist an architect.

Relevance of social skills

The *WHO dimension* addresses these social skills and thus outlines the role of an architect in organizations and teams. On one hand it covers general topics such as group dynamics processes, factors for well functioning teams, and the interdependencies of organizations and teams. On the other hand, it presents topics that have arisen from concrete project experiences. These include, for example, *organizational patterns*. Organizational patterns describe successful options for cooperation between roles in projects [Coplien and Harrison 2004]. Chapter 7 looks at these topics in detail.

Elements of the WHO dimension

2.8 Architecture Method (HOW)

As an architect, your objective is to design an architecture that can be used as a foundation for realizing a system. To achieve this objective, you have access to various architecture means, can vary their level of abstraction, and can communicate with different partners such as project leads, developers, and analysts. However, considering these options does not guarantee that you will meet your objective. Even if you realize a system successfully once, it does not mean that you will experience the same success next time. You can only be successful in the long term if you are able to repeat your architectural activities systematically. Therefore, it is very important to be aware of proven methodological approaches and be able to apply them repeatedly. The *HOW dimension* is dedicated to these methodological approaches, or rather the question “what do I have to do to design and implement an architecture?” We will therefore present a general architecture method.

Systematic and repeatable activity

The architecture method contains the following activities that must be executed when you design an architecture:

- > Creating the system vision
- > Understanding the requirements
- > Designing the architecture
- > Implementing the architecture
- > Communicating the architecture

Architecture activities

You can perform the activities several times within an iterative development process.

Activity-specific relevance of other dimensions

Depending on which activity you are currently performing, elements of the other *dimensions* have an effect on the architecture to different degrees. For example, you should use different means and perspectives depending on the activity at hand. In *understanding the requirements*, for example, it is particularly important to select the architecturally significant requirements from the requirements placed on the system.

We will look at this topic in more detail in Chapter 8.

2.9 Summary

Summary: Architecture orientation framework

- > The architecture orientation framework structures architecture using the dimensions WHAT, WHERE, WHY, WITH WHAT, WHO, and HOW.
- > The WHAT dimension (architectures and architecture disciplines) contains basic principles and definitions of architecture. It therefore lays the basis for working as an architect.
- > The WHERE dimension (architecture perspectives) covers the different levels at which architecture takes place and the views that make architecture tangible.
- > The WHY dimension (architecture requirements) is dedicated to the requirements placed on IT systems in general and architectures in particular.
- > The WITH WHAT dimension (architecture means) structures the different architecture means you can use whilst practicing as an architect.
- > The WHO dimension (organizations and individuals) looks at the role of the architect and the influence of individuals and organizations on architecture.
- > The HOW dimension (architecture method) structures the architectural process. It details the most important architectural activities that you perform during your work.

Further Reading

Further reading: Architecture styles and patterns

[Buschmann et al. 1996]
Buschmann, Frank; Meunier, Regine; Rohnert, Hans; Sommerlad, Peter; Stal, Michael, *Pattern-Oriented Software Architecture Vol. 1, A System of Patterns*, John Wiley & Sons, New York, 1996

[Schmidt et al. 2000]

Schmidt, Douglas C.; Rohnert, Hans; Stal, Michael; Buschmann, Frank,
*Pattern-Oriented Software Architecture Vol. 2, Patterns for Concurrent and Net-
worked Objects*, John Wiley & Sons, New York, 2000

[Kruchten 2000]

Kruchten, Philippe, *The Rational Unified Process - An Introduction Second Edi-
tion*, Addison-Wesley, Boston, 2000

**Further reading:
Architecture views
and framework**

[ISO10746 1998]

International Organization for Standardization, *Information technology – Open
Distributed Processing – Reference model: Overview*, [http://www.iso.org/iso/en/
CatalogueDetailPage.CatalogueDetail?CSNUMBER=20696&ICS1=35&ICS2=8
0&ICS3=](http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=20696&ICS1=35&ICS2=80&ICS3=), 1998

[Zachman 1987]

Zachman, John, A., *A Framework for Information Systems Architecture*, IBM
Publication, 1987

[OMG 2007a]

Object Management Group, *UML 2.1.2 Superstructure Specification*, [http://
www.omg.org/spec/UML/2.1.2/Infrastructure/](http://www.omg.org/spec/UML/2.1.2/Infrastructure/), 2007

Further reading: UML

[OMG 2010c]

Object Management Group, *Model Driven Architecture*, [http://www.omg.org/
mda/](http://www.omg.org/mda/), 2010

[Miller 1956]

Miller, G., *The Magical Number Seven, Plus Or Minus Two: Some Limits on Our
Capacity for Processing Information*, *The Psychological Review*, 63(2), 81-97,
1956

**Further reading:
Miscellaneous**

Software Architecture

A Comprehensive Framework and Guide for
Practitioners

Vogel, O.; Arnold, I.; Chughtai, A.; Kehrer, T.

2011, XVII, 478 p., Hardcover

ISBN: 978-3-642-19735-2