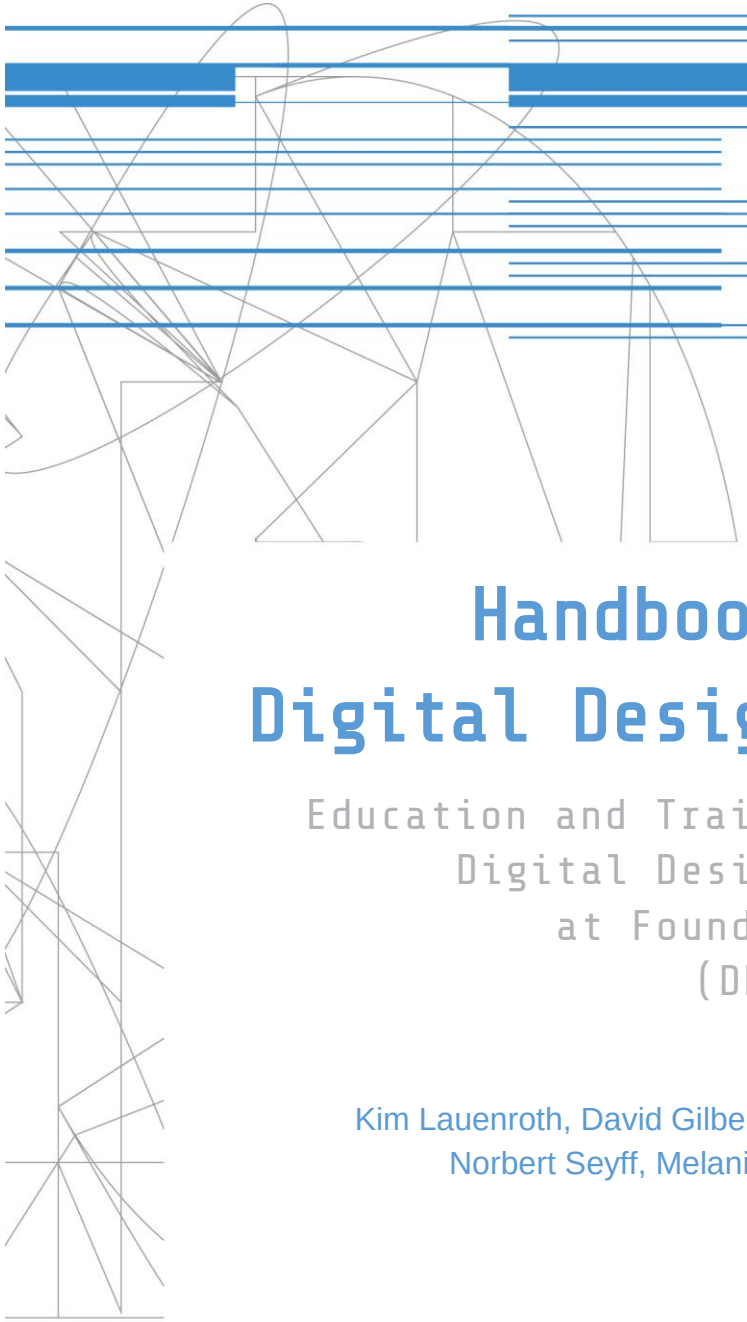


DIGITAL DESIGN PROFESSIONAL

Handbook for the Digital Design Professional

Education and Training Handbook for the Digital Design Professional at Foundation Level (DDP FL)

Kim Lauenroth, David Gilbert, Michael Kemper, Karsten Lehn, Norbert Seyff, Melanie Stade, and Marcus Trapp

Version 1.0.0
June 1st, 2021

Terms of Use

Acknowledgements

The contents of this handbook were reviewed by Michael Burmester and Martin Glinz. Tracey Duffy performed the English review of the handbook.

We thank everybody for their involvement. Our special thanks go to Bernd Aschauer, Nikola Eger, Thomas Geis, Saskia Hehl, Thomas Immich, Jens Kawelke, Rolf Molich, Knuth Polkehn, Lars Sonnabend, Hans-Jörg Steffe, Stefan Tilkov, Marcus Winteroll, and many others for their support in the preparation of this handbook.

The handbook was approved for release on March 8, 2021 by the Council of IREB e.V. upon recommendation by Martin Glinz.

We thank everybody for their involvement.

Scope of this handbook

This handbook provides an introduction to Digital Design based on the syllabus for the Digital Design Professional (DDP) at Foundation Level. It complements the syllabus and addresses three groups of readers:

- Students who want to learn about Digital Design and take the certification exam can use this handbook as a companion book to training courses offered by training providers, as well as for self-study and individual preparation for the certification exam.
- Practitioners who want to learn about Digital Design can use this handbook as a reference guide to set up a digital building process and to find appropriate tools end templates for their daily work.
- Training providers who offer trainings on the DDP Foundation Level can use this handbook as a complement to the syllabus for developing their training materials or as a study text for the participants in their trainings.
- Professionals in industry who want to apply proven concepts and knowledge in their practical work will find a wealth of useful information in this handbook and the provided examples.

This handbook provides a link between the syllabus, which lists and explains the learning objectives, and the relevant literature on Digital Design. The structure of the handbook matches the structure of the syllabus.

The authors, our reviewers, and IREB have invested a significant amount of time and effort into preparing, reviewing and publishing this handbook. We hope that you will enjoy studying this handbook. If you detect any errors or have suggestions for improvement, please contact us at info@digitaldesign.org.

Version History

Version	Date	Comment
1.0.0	2021/06/01	First version of the DDP handbook

Table of Contents

1 An Introduction to the Digital Design Professional 1

1.1 Digital Design as a New Profession 1
1.1.1 The Need for a New Profession 1
1.1.2 Overview of the Digital Design Profession 3

1.2 Understanding Digital as a Material for Building Digital Solutions 3
1.2.1 The Idea of Digital as a Material for Building Digital Solutions 4
1.2.2 Understanding the Context as Part of the Building Process 6
1.2.3 The Difference between Client, Customer, and User 7
1.2.4 The YPRC Case Study as an Example of a Digital Solution 8
1.2.5 A View on Technology in Relation to the Idea of Digital as a Material 11

1.3 An Introduction to the General Building Process for a Digital Solution 11
1.3.1 Cross-Cutting Activity Areas 12
1.3.1.1 Management of the Building Process 12
1.3.1.2 Evaluation of the Digital Solution 13
1.3.2 Core Activity Areas of the Building Process 13
1.3.2.1 Design of a Digital Solution 13
1.3.2.2 Construction of a Digital Solution 15
1.3.2.3 Realization of a Digital Solution 16
1.3.3 Understanding the Building Process as a Parallel Process 16
1.3.3.1 Cooperation between Design and Construction (1) 17
1.3.3.2 Cooperation between Design and Realization (2) 18
1.3.3.3 Cooperation between Construction and Realization (3) 20
1.3.3.4 Cooperation between Design, Construction, and Realization with Management (4) 21
1.3.4 Managing the Building Process at the Solution, System, and Element Levels 22
1.3.4.1 The Power of Understanding the Building Process at the Solution and System Levels 24
1.3.4.2 The Difference between a Concept-Driven and a Realization-Driven Building Process 27

1.4 Competence Profile of a Digital Design Professional 28
1.4.1 The Digital Design Professional as an Education Scheme 28
1.4.2 Ten Principles of Good Digital Design 29
1.4.3 The Digital Design Professional is Not a New Role 31

2 Design Competence 32

2.1 Integration of Digital Design into the Building Process 32
2.1.1 Fundamentals of the Design Process 32
2.1.1.1 The Design Squiggle as a Design Process Model 33
2.1.1.2 The Dual-Mode Model of Design 35
2.1.2 The Three Essential Steps of the Building Process for a Digital Solution 36
2.1.2.1 The Scoping Step 37
2.1.2.2 The Conceptual Step 41
2.1.2.3 The Development and Operations Step 43
2.1.2.4 On the Equal Importance of Scoping/Conceptual and Development Work 46
2.1.3 Quality as a Cross-Cutting Concern of the Building Process 46
2.1.3.1 Holistic Consideration of Quality during the Building Process 46
2.1.3.2 The Difference between the Quality of a Digital Solution and the Quality of a Digital System 48
2.1.3.3 Perceivable and Underlying Quality Attributes 48
2.1.3.4 A Quality Model for Digital Systems 49
2.1.3.5 Describing a Quality Model for Digital Solutions 50
2.1.4 Additional Resources for the Building Process 50

2.1.5 Conclusion: An Idealized Model of the Building Process 51

2.2 Conceptual Work in Digital Design 54

2.2.1 Fundamentals of Conceptual Work 55

2.2.1.1 Concepts Are Ideas in Thought or Communication 55

2.2.1.2 Benefits and Limits of Concepts 55

2.2.2 Pragmatic Document Templates for the Different Abstraction Levels 57

2.2.2.1 Digital Design Brief 57

2.2.2.2 Solution Design Concept 59

2.2.2.3 System Design Concept 60

2.2.2.4 Software Design Concept 61

2.2.2.5 Device Design Concept 63

2.2.3 Documentation Techniques for the Solution Level 64

2.2.3.1 Future Press Release 64

2.2.3.2 Persona for Characterizing Customer and User Groups 65

2.2.3.3 Stakeholder List 67

2.2.3.4 Value Proposition Canvas 67

2.2.3.5 Customer Journey Map 68

2.2.3.6 Business Model Canvas 69

2.2.4 Design Concepts at the System Level and Element Level: General Considerations 70

2.2.4.1 On the Value of Non-Redundancy and Tools for Avoiding Redundancy 71

2.2.4.2 Modularity of Information 72

2.2.4.3 Relationships between Building Blocks 73

2.2.4.4 On Readability versus Structured Documentation 74

2.2.5 Documentation Techniques for the System Level and Element Level 74

2.2.5.1 A Note about the Idea of Perfect Technology 74

2.2.5.2 A General Building Block Template 74

2.2.5.3 Goal Template 75

2.2.5.4 Constraint Template 77

2.2.5.5 Form and Function at the System Level 78

2.2.5.6 Form and Function at the Element Level 84

2.2.5.7 Quality: Quality Requirement Template 94

2.2.5.8 Overview Pictures for Visualization Purposes 95

2.2.6 Putting Everything Back Together for the Big Picture 96

2.2.6.1 Relationships between Building Blocks at Element Level 96

2.2.6.2 Relationships between Building Blocks at System Level 100

2.2.6.3 Relationship between Element Level and System Level 100

2.2.6.4 Relationship between Solution Level and System/Element Level 101

2.2.6.5 Relationship between Design Concepts, Activity Areas, and Steps of the Building Process 102

2.2.7 Conclusion: Learning Conceptual Work 104

2.3 Application of Prototypes in Digital Design 104

2.3.1 Definitions of Prototype 104

2.3.2 Objectives of Prototypes 105

2.3.2.1 Exploring the Problem, User Needs, and Requirements 106

2.3.2.2 Communicating Solution Ideas and Concepts 107

2.3.2.3 Testing and Improving Concepts and Solution Ideas 108

2.3.2.4 Advocating a Solution or Solution Idea 108

2.3.3 Examples of Using Prototypes in Different Disciplines 109

2.3.4 Criteria for Categorizing Prototypes 110

2.3.4.1 Level of Interaction 110

2.3.4.2 Goal of Prototyping 111

2.3.4.3 Level of Fidelity 111

2.3.4.4 Level of Fidelity for each Dimension of a Prototype (Mixed-Fidelity Prototypes) 113

2.3.4.5 Fidelity Profiles of Prototypes Typically Used during the Building Process 116

2.3.4.6 Prototype Categories in Certain Stages of the Building Process 118

2.3.4.7 Degree of Immersion in Relation to the Fidelity Profile of a Prototype ... 120
2.3.5 Tools for Creating Prototypes ... 121
2.3.5.1 Software Design and Development Tools and Technologies for Prototype Creation ... 121
2.3.5.2 Industrial Design Tools for Prototype Creation ... 122
2.3.5.3 Interaction Design Tools for Prototype Creation ... 124
2.3.5.4 Other Tools ... 124
2.3.6 Building and Using Simple Low-Fidelity Prototypes ... 124
2.3.7 Conclusion on Prototyping ... 128

3 Digital as a Material ... 129

3.1 Understanding Technology ... 129

3.2 Perceivable Technology ... 133
3.2.1 End User Devices ... 133
3.2.2 Interaction Technology ... 134
3.2.3 Software User Interface Technology ... 141

3.3 Underlying Technology ... 145
3.3.1 Programming Technology ... 146
3.3.2 Technology for Operating Software ... 148
3.3.3 Digital Communication Technology ... 149

3.4 Technology-Oriented Knowledge Areas ... 151
3.4.1 Software Architecture ... 151
3.4.2 Computational Complexity ... 152
3.4.3 Human-Computer Interaction ... 153

3.5 The Digital Design Perspective on Technology ... 153

4 Cross-Cutting Competences ... 155

4.1 Human Factors ... 155
4.1.1 Fundamentals of Human Sensation and Perception ... 155
4.1.1.1 Visual Attention ... 156
4.1.1.2 Auditory Attention ... 157
4.1.1.3 Steering Attention ... 157
4.1.2 Fundamentals of Human Performance ... 158
4.1.3 Emotions in the User-System Interaction ... 159
4.1.4 The Role of Prototypes ... 161

4.2 Business Models for Digital Solutions ... 161
4.2.1 Business Model Pattern ... 163
4.2.2 Digital Business Models ... 164
4.2.3 Thinking about Future Possibilities for Digital Businesses ... 164

4.3 People Management ... 166
4.3.1 Understanding the Building Process as a Social Process ... 167
4.3.2 Understanding People through Personality Models ... 169
4.3.2.1 The Inner Ring: Abstract versus Concrete (Understanding, Perceiving, and Learning) ... 170
4.3.2.2 The Second Ring: Cooperative versus Pragmatic (Action Focus) ... 173
4.3.2.3 The Third Ring: Directive versus Informative (Communication Style) ... 173
4.3.2.4 The Fourth Ring: Expressive versus Attentive (Energy, Impulsivity) ... 173
4.3.2.5 Conclusion on Working with Personality Models ... 174
4.3.3 The Building Process from a Group Dynamic Perspective ... 174

4.3.3.1 Managing Perception and Learning Potential during the Building Process ... 175
4.3.3.2 Managing the Thinking into the Future ... 176
4.3.3.3 Managing Working Style, Key Competences, and Role Assignment in the Building Process ... 178
4.3.3.4 Managing Leadership during the Building Process ... 181
4.3.3.5 Considering the Working Environment and the Context Understanding ... 183
4.3.4 Conclusion ... 184

5 A Building Process for Beginners ... 185

5.1 Guidelines for the Scoping Step ... 185
5.1.1 Scoping a Wicked Problem ... 186
5.1.1.1 Phase 1: Setting up a design thinking team ... 187
5.1.1.2 Phase 2: Performing the design thinking process ... 187
5.1.1.3 Phase 3: Documenting the results and iterating if necessary ... 188
5.1.2 Scoping a Tame Problem ... 188
5.1.2.1 Phase 1: Interview important stakeholders ... 189
5.1.2.2 Phase 2: Scoping workshop ... 189
5.1.2.3 Phase 3: Document the results and iterate if necessary ... 190
5.1.3 Defining the General Terms for Building a Digital Solution ... 190
5.1.3.1 Guidelines for Defining the Schedule ... 190
5.1.3.2 Guidelines for Defining the Mode of Cooperation ... 191
5.1.3.3 Guidelines for Defining the Budget ... 192
5.1.3.4 Guidelines for Defining Potential Revenue Streams ... 192
5.1.3.5 Guidelines for Defining Available Resources ... 192

5.2 Guidelines for the Conceptual Step ... 193
5.2.1 Phase 1: Explore the solution space for the digital solution from the customer perspective ... 194
5.2.2 Phase 2: Elaborate and evaluate solution candidates from a business perspective ... 197
5.2.3 Phase 3: Approach a promising solution candidate from a feasibility perspective ... 200
5.2.4 Phase 4: Final evaluation of the solution candidate with the client ... 201

5.3 Guidelines for the Development and Operations Step ... 202
5.3.1 Overview of the Process Structure ... 203
5.3.2 Managing Work at the Three Levels ... 204
5.3.2.1 Managing the Work at the Element Level ... 204
5.3.2.2 Managing the Work at the System Level ... 207
5.3.2.3 Managing the Work at the Solution Level ... 210
5.3.2.4 Big Picture on Managing Work at the Three Levels ... 211
5.3.3 Guidelines for Defining Work Items in the Building Process ... 212
5.3.3.1 Overview of Relationships between Work Products and Design Concepts ... 213
5.3.3.2 General Template for Work Items ... 214
5.3.3.3 Solution Work Items ... 215
5.3.3.4 Epics in Relation to the Solution and System Design Concept ... 216
5.3.3.5 User Stories in Relation to Element Design Concepts ... 218
5.3.3.6 Concept Work Items for Working on Design Concepts ... 222
5.3.3.7 Prototype Work Item ... 223
5.3.3.8 Evaluation Work Items ... 224
5.3.3.9 Technical Work Items ... 225
5.3.3.10 Writing Defects ... 227
5.3.4 Phase 1: Initial release planning and backlog preparation ... 228
5.3.4.1 Elaborate Element Design Canvases for Each Element ... 228
5.3.4.2 Creating a Story Map for Defining User Stories ... 231
5.3.4.3 Define Work Items and Estimate, Prioritize, and Manage the System Backlog ... 231
5.3.5 Phase 2: Developing the first release of the digital solution ... 233
5.3.6 Phase 3: Further evolution during operation ... 234
5.3.7 Phase 4: Retirement ... 235

5.4 Lean Startup as an Alternative Approach 235
5.5 Conclusions on the Building Process for Beginners 237
6 Achieving Good Digital Design 238
6.1 Contributions of the Digital Design Professional 238
6.2 The Importance of Practical Experience and Heuristics 240
6.3 The Importance of Teamwork 241
I. References 243
II. List of Figures 249
III. List of Tables 252

1 An Introduction to the Digital Design Professional

This chapter introduces the Digital Design Professional (DDP) as a new education scheme for designing digital solutions. In this handbook, we use the abbreviation DDP. The introduction starts in Section 1.1, which explains the motivation behind the need for a new profession called Digital Design for the future demands and challenges in a digital society. The core idea of Digital Design is presented is Section 1.2: understanding digital as a material that can be used to build digital solutions.

With the understanding of digital as a material, Section 1.3 introduces the building process for a digital solution. Understanding this building process is necessary to understand the scope of the DDP. Section 1.4 concludes the introduction and presents the competence profile of a DDP in relation to the building process for a digital solution.

A comprehensive case study describing the building process for a fictitious digital solution is provided with this handbook and is used as a basis for examples. The solution is called YPRC (Your Personal Running Coach). YPRC provides a holistic training service for newcomers to running and consists of a dedicated smartwatch, a smartphone app, and a service portal. Full details of the case study can be found in the supplemental material for the handbook.

Further details of the case study are presented throughout the handbook. There is no assumption that the case study is known in advance. It is even better not to read the case study in advance since the details of the case study will be elaborated during the course of the handbook.

To support the reader with this handbook, we use the following formatting:

Definition of an important term

Example from the YPRC case study

Furthermore, the heading Consideration for daily work is intended to emphasize that the text under this heading contains practical advice for daily work and goes beyond the content of the certificate.

1.1 Digital Design as a New Profession

1.1.1 The Need for a New Profession

The development of digital technology changes the nature of digital solutions and can be characterized by the following levels[1]:

- *Digitization* is the use of digital technology to solve problems with digital data that had previously been solved with non-digital data.
- *Digitalization* is the use of digital technology to create solutions and business processes that are not feasible with non-digital means.
- *Digital transformation* occurs when digital solutions impact people and society by changing people's habits and lives with digital means.

[1] There are several definitions of the terms digitalization and digital transformation (cf. [Bloo2018]). We prefer the definition that understands these terms as levels that depend on each other.

The beginnings of digital computing were all about digitization. When computing hardware became increasingly more powerful, cheaper, and less voluminous, businesses and engineers discovered that digital technology enabled solutions that had not been feasible before: digitalization solutions started to emerge. Since then, digitalization has enabled several waves of digital transformation. Today, we have a co-existence of digitization, digitalization, and digital transformation.

For example, analog directories such as phone books were initially digitized as electronic directories. These offered the same look-up features as analog directories but were cheaper, faster, and easier to distribute. This enabled the development of new features—such as a search by given criteria or reverse lookup—that were infeasible with analog directories: digitalization occurred. The advent of the World Wide Web and efficient search technology has transformed people's habits from looking up information in dedicated directories to on-demand searching—this is an early example of digital transformation.

When building a solution with digital technology, the first challenge is to figure out what to build. When figuring out what to build, *stakeholder* is an important umbrella term for all types of people or organizations involved in the building process. The term is defined as follows [Glin2020]:

Stakeholder: A person or organization who influences a system's requirements or who is impacted by that system.

One important stakeholder role is the client:

Client: A person or organization who orders a system or a solution to be built.

Users and customers—as further important stakeholder roles—are defined in Section 1.2. The building team member as a stakeholder role is introduced in Section 2.1.

In projects where there are clients who know what they want to order and stakeholders who know what their needs are and what problems a digital solution shall solve, the challenge of what to build is addressed by *requirements engineering*: identifying the right stakeholders, eliciting requirements from stakeholders, and then consolidating, documenting, validating, and managing these requirements. Based on the requirements, systematic and efficient realization of a digital solution is possible. Such approaches to building a digital solution are referred to as *requirements-driven* approaches.

In product development, however, there are often neither clients who order a solution nor readily available stakeholders who know what their needs and problems are. A similar situation occurs when a client demands an innovative solution but nobody has a clear idea about what such a solution should do and what it should look like. In these situations, the challenge of figuring out what to build must be addressed in a different way. The digital solutions to be built are driven by digital technology; the aim of such solutions is digitalized business that pushes toward digital transformation [Kell2016]. This calls for a *design-driven* approach to building a digital solution: *designers* explore what could be done, create visions for digital solutions, and finally shape the form, function, and quality of the digital solution. As this kind of creative design is analogous to what industrial designers do when designing physical products, the design of digital solutions is called *Digital Design*.

1.1.2 Overview of the Digital Design Profession

Because of the innovative nature of digitalization and digital transformation, a requirements-driven development must be complemented by a design-driven development of digital solutions that allows the use of new technical possibilities. Digital Design [Bitk2017] is a profession that represents a focus shift toward this. Digital Design [LBGH2018] is a profession that aims to improve our ability to design and build better digital solutions. Digital Design is defined as follows:

Digital Design: The creative design of digital solutions.

Digital Design understands digital as a shapeable material (see Section 1.2). This understanding goes beyond a pure technical understanding of digital technology, with the aim of a combination of design skills and technical skills similar to an understanding promoted by industrial design and building architecture.

Digital Design means shaping digital solutions by taking a holistic view of the technical possibilities of digital material, of the economic aspects, and of the current or future needs of people.

Digital Design shapes new and optimizes existing digital solutions by:

- Designing the goals, benefits, and means of a digital solution together—this reflects the holistic view of the solution and system (see Section 1.3.3) and the ability to cooperate with all other activity areas
- Designing both the large and small aspects, with large referring to the solution-level and system-level views of a digital solution, and small referring to the design of the elements of a digital solution (see Section 1.2.1)
- Designing perceivable and underlying aspects of a digital solution together—this refers to the fact that designing the perceivable form, function, and quality of a digital solution requires a profound understanding of the underlying form, function, and quality that enable the perceivable aspects (see Section 1.2.1)
- Designing material and immaterial aspects of a digital solution—this refers to the fact that a digital solution often consists not only of software but also of physical parts (see Section 1.2.1)

Digital Design means taking responsibility for the design of a digital solution and leading the building process for a digital solution from the design perspective. This includes shaping and optimizing the design activities of the building process, as well as intensive cooperation with all other activity areas of the building process (Section 1.3).

In the following, we provide important fundamental information for understanding Digital Design as a profession: The understanding of digital as a material (Section 1.2) and the general building process of a digital solution (Section 1.3).

We conclude this section by introducing the DDP, the education scheme that we have developed to become part of the Digital Design Profession (Section 1.4).

1.2 Understanding Digital as a Material for Building Digital Solutions

Before the different terms are defined, the analogy to building architecture presented in Table 1 is intended as an overview and to explain the overall idea of digital as a material.

The word *digital* is often used as a technical adjective to refer to the representation of data in a binary format. As shown in Section 1.1, binary data is used to shape new business models, social networks, and innovative products and services.

This means that the importance of binary data goes far beyond transporting and transforming information: we use binary data to shape life processes in the same sense that building architects shape life processes by designing space. Therefore, digital can also be considered as a noun—something that is reflected by the idea of digital as a material.

Table 1 – Terminology in building architecture and Digital Design

	Profession	
Reference point	**Building architecture**	**Digital Design**
Subject of design[2]	Direct: space Indirect: life processes	Direct: flow of binary data Indirect: life processes
Material	Building material (concrete, steel, wood, windows, doors)	Digital material (software, hardware, algorithms)
Abstract result	Building	Digital system
Specific result in a concrete context with a defined objective	Residential house, office building, clinic, garden house	Fitness tracking app, online shop for books, enterprise software, social network

1.2.1 The Idea of Digital as a Material for Building Digital Solutions

The idea of understanding digital as a material is intended to reflect the importance of digital for our economy and society. It is also intended to make clear that just like other materials, digital can be shaped to create innovative digital solutions. The core terms of this idea are defined below.

At first sight, these definitions will appear theoretical and abstract. We will use the YPRC case study to explain these concepts afterwards.

We define digital as follows:

Digital (noun): The structure, flow, and transformation of binary data.

This understanding is further intended to make clear that just like other materials, digital can be shaped to create digital solutions. This understanding is the basis for the focus shift away from a reactive technical and requirements-oriented development of digital solutions toward a proactive design-oriented development of digital solutions.

A prerequisite for understanding digital material is defining the term *system*.

System: In general: A principle for ordering and structuring.
In engineering: A coherent, delimitable set of elements that—by coordinated action—achieve some purpose.

[2] According to Walter Gropius [Grop1930], architecture means designing life processes (“Bauen bedeutet Gestaltung von Lebensvorgängen”). We believe the same applies to Digital Design.

The following three terms are useful for communicating about systems:

- *Form*: the elements and the relationships between the elements that make up the system's structure
- *Function*: capabilities provided by an element, by a combination of elements, or by the system as a whole
- *Quality:* The degree to which an element, a relationship between elements, or a capability of a system fulfills defined quality characteristics.

Achieving good quality, communicating about quality and evaluating quality require explicitly defined quality characteristics (cf. [ErMa2008]). Quality characteristics can be defined in various ways. In Section 2.1.3, we introduce exemplary approaches for defining quality characteristics.

Binary data needs a medium that carries the structure and enables the flow of data. This medium is digital material and is defined as follows:

Digital material: The technological means that enable the digital, that is, the structure, flow, and transformation of binary data.

In order to understand digital material, it is important to understand the four important properties of digital material:

1. Digital material has no objective.
2. Digital material has an underlying and a perceivable layer.
3. Digital material has technology-neutral aspects.
4. Digital material can be shaped, within limits, without any programming knowledge.

However, digital material only enables the flow and transformation of data. A flow of data does not exist without a system that processes, transports, and stores the flow of data. The meaning and value of this data (i.e., information) are created only if the data flow takes place between users and a system that can produce, transport, and consume the flow of data.

Such a system is called a digital system:

Digital system: A technical system that realizes a digital solution in a given context with digital means, that is, by processing, transporting, and storing binary data.

The user is an important stakeholder role of a digital system and is defined as follows:

User: A person who uses the functionality provided by a system.

In addition to human users, digital systems can be used by animals (e.g., in digital farming).

Elements of a system can also be understood as (sub-)systems. This allows the definition of systems that consist of a multi-level hierarchy of systems.

With the concept of a digital system, a digital solution is defined as follows:

Digital solution: A socio-technical system that solves a real-world problem with digital means.

A socio-technical system is a system that spans software, hardware, people, and organizational aspects. This understanding follows the understanding of systems from general systems theory

and understands people as part of the socio-technical system that defines the digital solution. This means that Digital Design is about shaping technical (digital) systems and about shaping socio-technical systems (the digital solution) with digital material.

The problem that a digital solution solves can also be referred to as objectives or value propositions that are offered to a customer in a certain context (see Section 2.1.2). The customer is an important stakeholder role for the digital solution and is defined as follows:

Customer: A person or organization who receives a system, a product, or a service.

The term *receiving* includes both buying a solution or obtaining it for free. The definition is very broad to cover various situations. Typical situations are:

- The customer can receive the digital system without any further services. For example, the customer buys office software.
- The customer can receive a product that is embedded in the digital solution. For example, the customer buys a games console that allows games to be purchased via the internet.
- The customer can receive a service that the digital solution provides. For example, a customer can use the digital solution to book a hotel room.

Beyond its intended customers, a digital solution may also have indirect customers. This is the case, for example, when customers employ a digital solution to improve non-digital services that they provide to their customers.

The two layers of digital material manifest themselves in two layers of a digital solution:

- Perceivable layer: form, function, and quality that can be perceived by stakeholders
- Underlying layer: form, function, and quality that is hidden from perception by stakeholders and that enables the perceivable layer

1.2.2 Understanding the Context as Part of the Building Process

In general, context is a network of thoughts and meanings needed to understand phenomena or utterances. In Digital Design, context is defined as follows:

Context: The part of the environment of a digital solution or digital system that is relevant for understanding and realizing a digital solution.

Context includes important stakeholders and, in particular, potential customers and users of the digital solution.

Problem and context are inseparable

The problem solved and the context are inseparable. This means that a digital solution that works in one context does not necessarily have to work in another context.

The important difference between a digital solution and a digital system is that the digital system represents the technical means to achieve an end in a defined context (the digital solution). From a theoretical perspective, two things are important.

First, the relationship between means and end can be complicated or complex (cf. [Snow2005]): a complicated relationship is characterized by a deterministic cause-and-effect relationship

between means and ends. A complex relationship has a non-deterministic part that makes it difficult or even impossible to analyze in advance.

People who do not have a proper understanding of digital material often consider digital solutions complex. However, with a proper understanding of digital material and with training in Digital Design, it is possible to separate the complicated from complex means-end relationships and to deal adequately with both.

Second, means and ends are independent of each other: an end can be achieved by different means and a means can be used to achieve different ends. This often leads to the impression that the end (the digital solution) should be defined before thinking about the means (the digital system).

In practice, means and ends influence each other significantly. It is of course important to start with the end. However, understanding the means to an end improves the understanding of the end as well. This is why design is solution-oriented and emphasizes the importance of prototyping to improve the understanding of means and ends together (cf. [Cros2006]).

The joint consideration of means and ends is particularly important when designing digital solutions. Digital material offers new means, which in turn enable innovative ends to be achieved. Digital Design therefore implies in particular designing the digital solution and the digital system in parallel.

1.2.3 The Difference between Client, Customer, and User

It is important to understand the difference between client, customer, and user to keep a clear focus during the building process. The three stakeholder roles create three idealized external perspectives for the building team of a digital solution:

- The *client* orders the building of the digital solution. To understand the client perspective, it is important to understand the objectives of the client for ordering the digital solution.
- The *customer* wants to receive value (i.e., a system, product, or service). To understand the customer perspective, it is important to understand what value should be generated.
- The *user* uses the digital system within the digital solution and therefore takes part in the value creation. To understand the user perspective, it is important to understand how value is created.

In practice, the three stakeholder roles can be independent people or organizations. However, this is a special situation for building a digital solution. In other situations, stakeholder roles are combined:

- The client is a customer and a user: the client orders a digital solution for their own purposes. Example: client orders a company internal ERP (enterprise resource planning) system.
- The client is a user but not a customer: the client orders a digital solution for their own organization to deliver value to customers. Example: client orders a CRM (customer relationship management) system.
- The client is not a customer and not a user, the customer is a user: the client orders a digital solution for external customers who also use the digital system inside the solution. Example: client orders an online shop to sell their products over the internet.

- The client is a customer but not a user: the client orders a digital solution for their own benefit but does not use the digital solution. Example: client orders a website for presenting the client's company to external people (users).

Clear focus on client, customer, and user improves innovation potential

A clear focus is important for two reasons. The first reason is to have a clear picture of what will be designed in terms of the customer and user perspective. In the context of digital solutions, we often encounter the implicit assumption that the user and customer are the same person. For example, a person that orders a hotel room on the website of the hotel is a customer of the hotel and at the same time the user of the website.

This implicit assumption limits the solution space of digital solutions unnecessarily because it assumes that the added value of a digital solution is created only through direct interaction with the solution. A good digital solution can also create value with indirect interaction. Consider the hotel example again: the customer in the hotel could also call reception and speak to a hotel employee, who in turn interacts with the website to book the room as a user. The added value for the customer is the same in both cases but the interaction is completely different. To achieve good Digital Design (see Section 1.4.2), it is important to be able to separate these perspectives to recognize valuable non-digital aspects of a digital solution that can be improved or supported with digital means.

The second reason why having a clear focus is important is to avoid confusing the different perspectives of a particular person on the stakeholder roles. First, in several digital solutions, one person can be the customer and the user at the same time. Such a person may have a certain idea about the value (the customer perspective) and about how the value is created (user perspective). Second, the clients often overestimate their knowledge and understanding of the customer and/or the user perspective. This may lead to false assumptions and often to suboptimal or even weak digital solutions. To achieve good Digital Design, it is therefore necessary to carefully evaluate and clarify the input from the clients.

In everyday life, it is of course easy to use the term user synonymously to the term customer. To improve the readability of the handbook, we also use the term user in this way. However, in situations where the distinction is important, we use both terms and recommend that you pay attention to a precise use of both terms during the design of a solution.

1.2.4 The YPRC Case Study as an Example of a Digital Solution

We will now illustrate the terminology introduced with the YPRC case study.

In our case study, the client is the startup committee that could offer the funding for building the YPRC solution. The startup committee believes that there is a potential market for an innovative digital solution that offers coaching services to runners. This belief brings us to the customer perspective.

From a customer perspective, the objective of YPRC is to provide an all-round solution for beginners in long-distance running that do not have or cannot afford a personal coach but want to have guidance that goes beyond an existing simple fitness app. At the core of the YPRC solution is the idea of offering a coaching experience that is comparable to an experienced personal coach who is running side by side with the customer. Although this is only a rough characterization of the YPRC solution, it creates an initial understanding of what YPRC is about.

Now, we will look at the system level. In the story line of YPRC, the team evaluated different alternatives for shaping the digital system of YPRC. In essence, two alternative system forms were discussed. Alternative A is a digital system where the personal coach is replaced by artificial intelligence (AI) that gives coaching advice to the runner. Alternative B is a digital system where the personal coach is connected to the runner via a digital data link. This data link is used to transfer the health data of the runner to the coach and to realize a voice connection between runner and coach.

At first glance and without further evaluation, both ideas seem reasonable. However, they represent completely different strategies for realizing the objective of potential customers that we have defined at the solution level. Hence, the form of the digital system will look different in each case. With this example, we want to highlight the important terminological difference between a digital system and a digital solution. The digital system is a tool, and the digital solution adds the perspective of a transformation of this tool toward defined *objectives*. Building a digital solution requires consideration of the objectives together with the digital system that shall achieve the objectives. The example further shows that the digital system and the technology behind the system (here, artificial intelligence) can significantly influence the objectives of the digital solution. Without the idea that digital technology can provide a coaching experience comparable to a real coach, the solution idea of YPRC would not be feasible.

If alternative B is realized, this form of YPRC requires a number of elements: the runner who wears the YPRC smartwatch and uses the YPRC smartphone app, as well as a web portal that is connected to the app and that is used by the runner's remote coach. The relationships between these elements can be visualized with a simple figure:

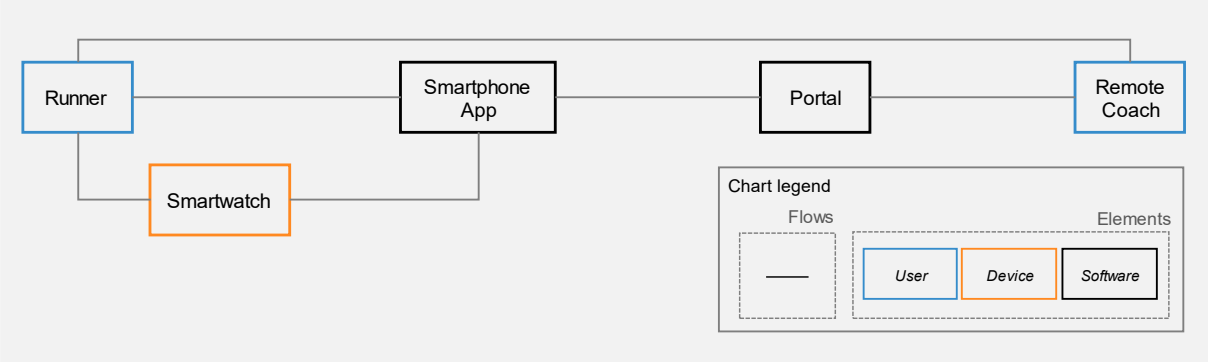


Figure 1 – Simplified form of the digital solution YPRC

This figure does not show much except that there are five elements that have some relationships between each other. Nevertheless, the elements of YPRC provide an important message: a digital system inside a digital solution is not limited to software. Nowadays, a digital solution can consist of dedicated hardware devices that have been built specially as part of a digital solution. Examples of such devices are smartwatches, smart speakers, and devices for home automation. We will now look at the functions of YPRC to go into further details of YPRC.

The function of the YPRC smartwatch is to measure the pulse of the runner and to show the current pulse to the runner. The pulse data is continuously transferred from the smartwatch to the smartphone app. Note that the term continuously refers to a quality of this function. The runner can use the smartphone app to view the pulse data history after their training session. So far, this is a common fitness app.

The distinctive functions of YPRC are enabled by the web portal. The smartphone app transfers the training data of the runner to the portal. The portal uses artificial intelligence technology to analyze the runner's data and to provide training tips to the runner. The runner can subscribe to this artificial intelligence training support by paying a monthly fee. Furthermore, the runner can book a personal remote coach for a training session. During a training session, the remote coach can view the runner's data via the portal in real time (note the quality *real time*) and give the runner immediate (again, a quality) running tips via a voice connection. With these details on the function of YPRC, we can create a more informative figure:

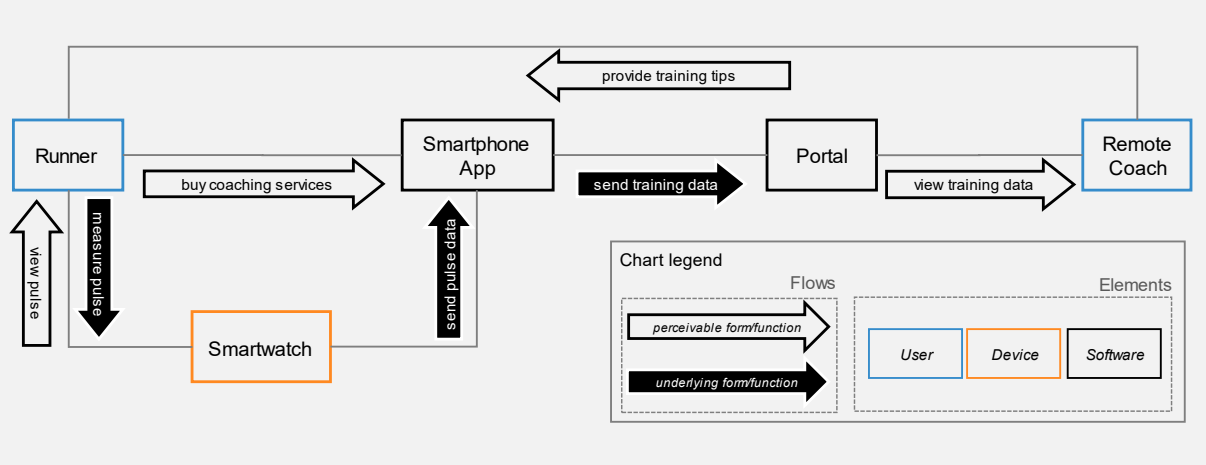


Figure 2 – Simplified form and function of the digital solution YPRC

Although YPRC seems to be a simple digital solution, Figure 2 already shows a complicated system with several relationships, functions, and qualities.

The idea of the perceivable and underlying layer is the second property of digital as a material and can create a more practical view of this digital solution. For example, the data transfer between smartwatch, app, and portal belongs to the underlying form and function of the YPRC. A fast computation and data transfer (underlying quality) are important for the solution because they are necessary to enable the remote coaching service in real time, but from the perspective of the runner (and the coach), they are not perceivable. The runner can only perceive the actual coaching tips and the remote coach can only perceive the real-time visualization of the training data. To highlight that these aspects belong to the underlying form and function, the description is written in black in Figure 2.

The distinction between perceivable and underlying form, function, and quality is not only a means of structuring complicated digital systems more simply: in the first instance, it is a central property of digital material. Relevant perceivable functions of a digital solution, which generate real added value, are made possible only by the underlying form, function, and quality.

To design digital solutions holistically, this means that the perceivable and underlying form, function, and quality must always be considered and designed together. Digital Design means neither just visual design of the solution nor interaction design or exclusively technical design of the system. A one-sided focus on perceivable (visual or interaction) or underlying (technical data flows) aspects will always lead to suboptimal solutions or even unrealizable solutions.

Assume, for example, that YPRC would opt for the artificial intelligence (AI) approach and could develop a really strong AI that can provide useful training tips for a runner (underlying function). Assume further that these tips are then visualized on a poorly designed smartphone app user

interface (perceivable form): it is very likely that users will not accept this service. Vice versa, if a great smartphone user interface provides the weak training tips of a poor artificial intelligence, users will also not accept this service.

1.2.5 A View on Technology in Relation to the Idea of Digital as a Material

To conclude this introduction to the idea of digital as a material, a view on technology is necessary. You may have noticed that the descriptions of the YPRC case study have essentially omitted technical terms and the naming of technologies. Only the terms app, portal, and artificial intelligence (AI) have been used.

This technology-neutral perspective is the third property of the idea of digital as a material: we can talk about a digital solution at a very detailed and concrete level with a minimum of technical knowledge. Although software itself is an important technology, the term software also does not appear in our description.

The absence of software reflects the fourth property of the idea of digital as a material: it is possible to create a digital solution without programming software. We can use a very simple example to illustrate this fourth property.

Assume we want to create a very simple digital solution for ordering pizza. Our pizza restaurant uses a popular messaging service as the digital communication channel (for example, WhatsApp). You can send your order and delivery address to the restaurant as a message. The restaurant confirms your order with a message, including the price to be paid. If you agree with the price, you can pay your order with an existing payment service (for example, PayPal). Once the payment has been confirmed (the restaurant receives an email from the payment provider), the restaurant starts preparing the order. When the order is ready for delivery, the restaurant sends another message to inform you that your pizza is on the way. This digital solution is of course a very primitive one, but it is a working solution for ordering pizza online.

Nevertheless, this simple example should not create the impression that designing digital solutions does not require technical expertise at all—it absolutely does require technical expertise. Only through technical expertise is it possible to make use of the capabilities of technology and to design digital solutions on a technology-neutral level (cf. third property of digital material) that can then actually be realized. Furthermore, the software used must of course be developed and provided by someone in advance.

1.3 An Introduction to the General Building Process for a Digital Solution

With the understanding of digital as a material, we can now look at the building process for a digital solution. In general, a process is defined as follows:

Process: A set of interrelated activities performed in a given order to process information or materials.

In the context of digital solutions, the word *building* can seem unusual at first glance. We deliberately chose this general term because it fits well with the idea of digital as a material. For us, building means crafting a digital solution using digital material.

In the following, we introduce a number of terms to describe the building process. These terms refer to activity areas and possible outcomes of the activities. In Section 1.4, we use these activity areas to define the competence profile of a DDP.

The activity areas must not be confused with roles within a project setup. Roles can be defined from these activity areas but depend on the particular process model or project situation.

Existing disciplines for developing elements of a digital solution (e.g., software engineering, industrial design, usability engineering, product management, software testing, etc.) can be matched to one or more of the activity areas presented. Where necessary, brief references to existing disciplines are made in this handbook. However, further details on the relationship to existing disciplines are not part of this handbook.

This does not mean that the DDP should ignore or neglect these disciplines. In fact, the exact opposite is the case. With this general understanding of the building process, a DDP is able to cooperate with a wide range of existing disciplines. In this section, the building process for a digital solution is described in a schematic way. We distinguish between three core activity areas and two cross-cutting activity areas of a building process:

- Core activity areas:
  - Design
  - Construction
  - Realization
- Cross-cutting activity areas
  - Management
  - Evaluation

This basic understanding of the building process is important for understanding its various aspects and for understanding the integration of Digital Design in the building process. Nevertheless, this understanding is not sufficient for actually defining and performing a concrete building process. A concrete process for beginners is described in Chapter 5.

1.3.1 Cross-Cutting Activity Areas

1.3.1.1 Management of the Building Process

Building a digital solution involves a complexity that requires a dedicated management activity area. It is defined as follows:

Management: Leading the building process in cooperation with all other activities.

We distinguish between three perspectives for the management of the building process:

- *Project management perspective*: coordination of activities, time, and budget
- *People management perspective*: managing stakeholder expectations, managing the cognitive process of stakeholders, getting the right people and skills for the activity at hand
- *Product management perspective*: developing a short-term and long-term strategy for the evolution of the digital solution

The project and people management perspectives are detailed in Section 1.3.4. Product management goes beyond a foundation level and is not an explicit part of this handbook.

1.3.1.2 Evaluation of the Digital Solution

The activity area evaluation is intended to focus on the quality of the work products. The digital solution realized is also considered a work product. We define evaluation as follows:

Evaluation: A systematic process for determining the value, quality, or appropriateness of something.

In Digital Design, evaluation particularly determines whether a digital solution or a work product used to create a digital solution actually has the qualities and properties that it should have according to the design concepts and the stakeholders' needs. This means that in the building process, evaluation is always related to a work product and therefore related to one of the core activity areas. We therefore consider evaluation as an inseparable part of the core activity areas. The respective perspective on evaluation is therefore described together with the core activities (see Section 1.3.2). However, the independent definition of evaluation is intended to emphasize the importance and the generic applicability of evaluation within the building process.

In order to capture and structure the evaluation work, we define a dedicated concept type:

Evaluation concept: A description of the evaluation approach for a work product.

Evaluation concept is again a generic term in order to capture the broad scope of quality assurance work during the building process. In contrast to concepts created by the core activity areas (see Section 1.3.2), the evaluation concept does not describe the digital solution. An evaluation concept describes the approach for evaluating a certain work product of the building process. The reason for this is to make the evaluation approach explicit.

1.3.2 Core Activity Areas of the Building Process

1.3.2.1 Design of a Digital Solution

Design is an activity area that deals with the future and thus which (digital) solutions can change an existing situation into a preferred one. Design is a difficult term because it has several meanings (cf. [ErMa2008]).

Design: 1. A plan or drawing produced to show how something will look, function, or be structured before it is made.

2. The activity of creating a design.

The definition of the activity design makes use of the term design as a result. The result design is defined as a plan or drawing produced to show how something will look, function, or be structured before it is made.

Designing thus means envisioning and properly describing a desired future by means of design concepts. This understanding of design has three aspects:

1. Elaborating and understanding the desired future
2. Defining and shaping a digital solution that shall create this future by means of a design concept
3. Evaluating the quality of the design concept

The first aspect requires empathy, imagination, and creativity. The second aspect requires skills in digital technology and conceptual work. The third aspect requires skills in various quality assurance directions (e.g., technical feasibility, function suitability, usability).

Talking about a desired future is a rather abstract and big idea. Design literature often uses *design problems* or *design goals* as alternative terms. However, we prefer *future* instead of problems or goals because it describes what design is really about: envisioning a future and creating it.

Envisioning a future together with all relevant stakeholders is a real challenge and must by no means be underestimated.

In order to describe the desired future, design creates design concepts to describe the digital solution that shall create the desired future. A design concept is defined as follows:

Design concept: A description of the design of a digital solution, of a digital system, or of an element of a digital solution under the assumption of perfect technology.

Design concept is a generic term. It can be considered as building plans for the digital solution in various abstraction layers including all important elements (e.g., dedicated devices, user interfaces, functionality, etc.). Keep in mind that a digital solution is more than software. This means that the term design concept also includes the description of dedicated devices that are designed especially for the digital solution at hand. As a consequence, the creation of the design concept may require the involvement of various disciplines (e.g., requirements engineering, interaction design, industrial design, and service design).

The assumption of perfect technology [WaMe1986] is an important simplification for the design of digital solutions and means in particular, defect-free technology as well as infinite computing capacity, storage capacity, and infinite communication capacity. It simplifies the development of design concepts in several ways. We will come back to the details of design concepts and the assumption of perfect technology in Section 2.2.5.1, where we give a detailed introduction to conceptual work.

The responsibility of design does not end with the design concept. The definition above also includes two important quality assurance aspects of design.

Envisioning a desired future means that design must ensure that the future envisioned is desirable for all relevant stakeholders. This aspect is about involving all relevant stakeholders to ensure their acceptance of the future defined.

Creation of a design concept of a digital solution that will create this future means that design must ensure that the design concept has the potential to create the future envisioned and that the digital solution defined creates the future envisioned. This aspect has two dimensions.

In the first dimension, the design concept itself is the focus of quality assurance, i.e., the defined form, function, and quality must be validated to determine whether they are suitable and capable of creating the future envisioned (*Are we building the right digital solution?*). This aspect also requires the involvement of stakeholders to validate the design concept. Besides the design concept, the creation of prototypes is an important technique for validating certain aspects of a design concept. In Section 2.3, we provide further details on the application of prototyping in design.

The second aspect is about the proper construction and realization of the digital solution according to the design concept (*Are we building the digital solution right*?). This aspect requires close cooperation between the activity areas construction and realization. Details of this cooperation are discussed in Section 1.3.3 immediately after the introduction of these terms.

1.3.2.2 Construction of a Digital Solution

Construction is an activity area that deals with the technical details of a digital solution to prepare its realization. We define construction as follows:

Construction: The creation of the realization concept of a digital solution that will create the desired future.

This understanding of construction has two aspects: first, the creation of the realization concept of the digital solution; and second, the evaluation that the digital solution described by this concept will create the desired future envisioned by the design activity.

The realization concept is again a generic term, it complements the design concept and is defined as follows:

Realization concept: A description of a digital solution with real technology.

The realization concept must use real technology and has to deal with all technical details that are necessary to realize the digital solution and its elements[3]. This can include the definition of the physical structure (e.g., physical components and materials) and technical structure (e.g., microprocessor and board) of dedicated devices, the definition of the software structure (e.g., software components), the definition of the realization technology (e.g., programming languages, frameworks, technical sensors, etc.), and the definition of the technical infrastructure (e.g., the definition of proper computing centers).

This broad range shows clearly that construction of a digital solution may require the involvement of various disciplines (e.g., software engineering, industrial design, and production engineering) and that realization concepts have various instances (e.g., software architecture concepts, physical building plans, electronic layouts). We will come back to the challenge of involving different disciplines in Section 1.3.4.

Of course, the realization concept and the design concept depend on each other. We discuss the relationship between both concepts in Section 1.3.3.1 when we discuss the cooperation between design and construction.

The quality assurance aspect of construction is about the realization concept. Like the design concept, the quality assurance of the realization concept has two dimensions.

First, the realization concept must describe the necessary technical capabilities for creating the future envisioned. This especially includes the aspect that the defined technologies must achieve certain qualities (e.g., reliability of the digital solution). In Section 2.1, we provide further details on the relationship between quality and technology.

[3] This handbook focuses on design concepts and does not provide further guidance for working on realization concepts.

Second, construction has to further ensure that the realization concepts defined are properly realized. Here, close cooperation between design and realization is necessary (see Section 1.3.3).

1.3.2.3 Realization of a Digital Solution

The activity area realization deals with the factual implementation of the digital solution and is defined as follows:

Realization: The implementation of the digital solution according to the defined design concepts and realization concepts.

Like construction, this understanding of realization has two aspects: first, the implementation of the digital solution according to the design concept and the realization concept; and second, ensuring that the digital solution implemented will create the desired future envisioned by the design and construction activities.

The realization of a digital solution is by no means a trivial endeavor. As with construction, the realization of a digital solution may require the involvement of various disciplines. Depending on the types of elements of a digital solution, different disciplines must be involved:

- Dedicated devices: planning and performing the manufacture and delivery of the devices, including the development of the software parts of the device
- Software elements: setup of a development environment for the actual implementation, setup of a production environment for operating the software elements, and rollout of the software

Evaluation during realization is fundamental to every building process. Realization must ensure that the digital solution is implemented according to the design concept and the realization concept to ensure that the digital solution creates the envisioned future. This quality assurance is performed in a joint effort together with design and construction.

1.3.3 Understanding the Building Process as a Parallel Process

From a naïve perspective, the core activity areas can be considered as subsequent activities: design creates the design concept, which is then transformed by construction into a realization concept, which is then implemented in the realization activity. Such an approach is often called a *waterfall approach* but has been disregarded as impractical since the very beginning of software development (cf. [Royce1970]).

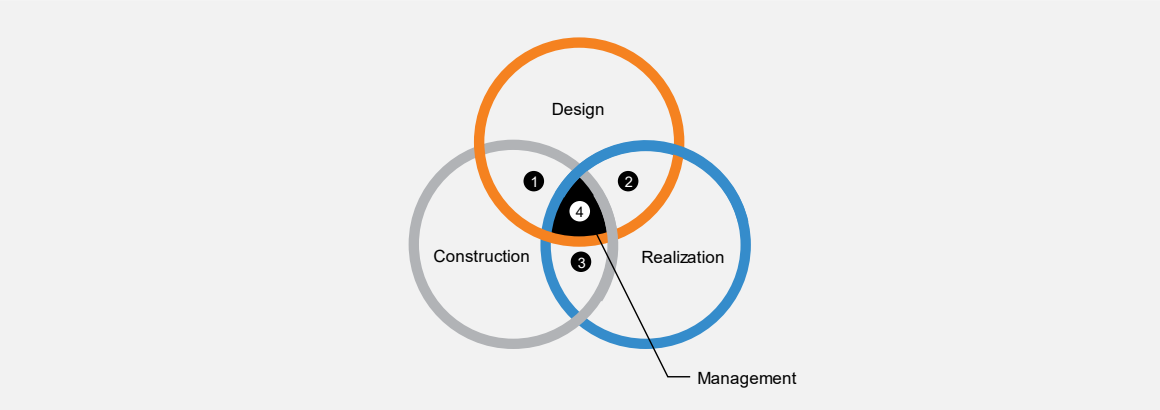


Figure 3 – Intersection between the core activity areas of the building process

A more realistic view of the building process is the understanding of all three core activity areas as ongoing activities that are performed in parallel. Figure 3 shows the activities of the building process as a Venn diagram. The four overlapping areas are important because they define the area where the different activities cooperate during the building process.

In the following, we discuss the cooperation between design, construction, and realization, including evaluation in detail (the headline of each subsection refers to the number of the area considered). Readers with experience in building digital solutions will already be familiar with the content presented. The comprehensive introduction is aimed primarily at beginners. The management of the overall building process is discussed in Section 1.3.4.

1.3.3.1 Cooperation between Design and Construction (1)

The difference between design and construction is very important for understanding the building process for a digital solution. Both activity areas work at a conceptual level and often use the same languages for communication and documentation (for example, diagrams or technical drawings). Therefore, it is quite easy to mess up both activity areas.

Design and construction are about the perspective

The most important difference between design and construction is the perspective. Design looks at the outside of a digital solution with the aim of understanding and elaborating the desired future. Construction looks at the inside of a digital solution and is concerned with defining the technical realization of the desired future.

To emphasize this difference, we have intentionally defined dedicated result categories—design concept for design and realization concept for construction—and introduced the idea of perfect technology. Ideally, design concepts and realization concepts together define the whole digital solution without any redundancy. Further details on the creation of design concepts, including the assumption of perfect technology, are discussed in Section 2.2.

YPRC example. Table 2 shows exemplary relationships between a design concept and a realization concept based on the YPRC case study.

The examples show that technical feasibility and the degree of freedom are an important shared responsibility between design and construction. Every decision during design or construction may have an impact on the other activity area and on the final digital solution.

Table 2 – Exemplary relationships between a design concept and a realization concept

Design concept	Realization concept
Mock-up of a user interface of the YPRC smartphone app	Technical components and libraries for the user interface: HTML, CSS, and angular java script
Data model that describes the health data of the runner	Tables in an SQL database for storing the health data
Communication between the YPRC smartphone app and the portal	Web service description of the interfaces
Shape of the YPRC smartwatch housing	Technical form of the housing, including manufacturing plans

Even though design and construction are different activity areas, they must work together and complement each other. For each and every form, function, or quality defined by a design activity, a corresponding technical implementation is necessary to realize it.

Technical decisions are a shared responsibility that creates innovation potential

From a practical perspective, it must be possible to realize a form, function, or quality defined during design with the technology available[4]. People making design decisions must have enough technical expertise to assess the feasibility of their decisions or must recognize when they need to consult expertise from construction.

YPRC example. Assume that somebody defines that the artificial intelligence will run on the smartphone app and will provide real-time training tips to the runner through an artificial voice assistant. Such a function requires significant computation power that may not be available on a smartphone. In this situation, a construction expert with knowledge about smartphones is necessary to evaluate the technical feasibility of this function.

Vice versa, people making construction decisions must have enough design expertise to assess the impact of their decision on the design of a digital solution and must recognize when they need to consult experts from design. Assume, for example, that during construction of the YPRC case study, somebody decides that an open-source SQL database is selected for storing the user and health data in the portal. Assume further that this database technology is only able to scale up to a certain amount of data and a certain number of transactions per second. Such a decision may impact the scalability and the number of users that the software can handle. In this situation, this technical limit must be discussed with design to determine whether the limit is acceptable for the final realization.

The cooperation between design and construction is not only about limits and restrictions—design and construction decisions may also create additional possibilities. If, for example, during construction, a technology is chosen that enables additional functions that were not thought about in the design, design should be informed about these possibilities in order to possibly integrate them into the digital solution.

1.3.3.2 Cooperation between Design and Realization (2)

The cooperation between design and realization must be distinguished according to the type of element to be realized. Before we discuss the special properties of software and physical parts in detail, let us first consider the general aspects of the cooperation between design and realization:

- *Clarifying conceptual details for realization*: Design has to provide the conceptual details of the digital solution that are necessary to implement the solution. At the same time, realization is responsible for pointing out inaccuracies and gaps in the design concept of the solution so that design can correct and improve them.
- *Quality assurance of perceivable form, function, and quality*: The perceivable form, function, and quality provide the surface of the digital solution to users and are the responsibility of design. Therefore, design is responsible for ensuring that the perceivable form, function, and quality of the digital solution create the desired future and are realized

[4] If a desirable form, function, or quality is currently not realizable, it may trigger research and development activities that could lead to technologies that make them realizable. However, this is not within the scope of this handbook.

according to the design concept. The concrete approach for fulfilling this responsibility depends on the particular building process.

Design and realization should work together as early as possible

The cooperation between design and realization can start very early during the building process with the creation, for evaluation purposes, of prototypes that cannot be created solely by design. Examples of these prototypes are interactive prototypes that require factual implementation of software (see below). In such a situation, an evaluation concept that describes the goal and measures of the evaluation activity should be created. For example, an HTML user interface prototype for an app is implemented to evaluate the usability of the overall visual design of the app. The evaluation concept should capture the concrete shape of the prototype and the procedure for performing the evaluation.

With respect to digital hardware devices, the design of a physical device ends with a model of the product (also called a pre-production prototype) that will be reproduced in an expensive and challenging mass production process. In contrast, the design of a software element is an ongoing activity during the whole realization process.

The reason for this difference originates from the inherent complexity of implementing the software part of digital solutions. The implementation (also known as programming) of software is seen as an intellectual challenge, since even the simplest programs can push the human mind to its limits (cf. [Glas2006], [Wein1971]). Important questions about the details of a software implementation arise during the actual implementation of the software because the act of programming forces us to think in logical precise structures of the programming language.

Late design decisions are an important aspect in Digital Design

For the cooperation between design and realization, this means that various detailed design decisions (for example, the behavior of a software in exceptional situations) have to be taken during realization in order to implement the software properly. In practice, this detailed work on software is often perceived as unsatisfactory from a design point of view since the development of software is often very lengthy in all details.

Nevertheless, the possibility of late design decisions is the special strength of software. When used correctly, this possibility can significantly increase the speed and efficiency of the entire building process. A good building process deliberately plans for late design decisions. This means that only those design decisions that must be taken upfront before realization are actually taken.

A good example of the benefits of late design decisions is the user interface of a software. User interfaces that are relevant for the main use cases of a software should be designed as early as possible to evaluate the user acceptance of the designs. The design of user interfaces with less importance can be created shortly before the actual realization takes place.

A second benefit of late design decisions in software realization can be achieved in quality assurance. Certain aspects of a software can best be validated in real operation. A good example here is the behavior of software in the case of errors. The usability of the visualization of error messages in exceptional cases highly depends on the real-life situation of the user. The validation and improvement of the design of a software in such situations can best be achieved with the real implementation.

Using the possibility of late design decisions properly is a real challenge and requires careful preparation (for example, selection of the proper technology to enable flexibility in the

implementation) and experience (for example, to identify those decisions in the design of the software that can be delayed).

The design of a physical product requires a different cooperation

Realizing a physical product requires a different cooperation approach. Industrial design and product engineering are dedicated disciplines that deal with mass products. A comprehensive overview of the subject is beyond the scope of this handbook. Nevertheless, we give a brief introduction to highlight important challenges.

All relevant design decisions and quality assurance measures must be taken before the mass production of the physical product starts. The cooperation between the design and realization of physical products therefore relies massively on different types of prototypes. In Section 2.3, we elaborate further details of prototypes. In the following, we give three examples of such prototypes:

- The *functional prototype* does not have the final shape of the product but contains all relevant function elements. It is created to validate the technical function of a product.
- The *appearance prototype* does not provide functionality but represents the final shape of the product. It is used to validate the visual appearance of the product with users.
- The *pre-production prototype* is a fully realized sample of the product for final and comprehensive quality assurance measures.

The last aspect of the cooperation between design and realization is the quality assurance of the solution implemented. Typical examples for this aspect are user/customer acceptance tests and usability tests. Different types of evaluation concepts can be defined for this purpose. They typically include test cases that describe the concrete application of the digital solution, including test data that defines the circumstances under which the digital solution is used. More sophisticated evaluation approaches include automated tests and A/B tests. However, we consider such approaches as advanced level approaches.

1.3.3.3 Cooperation between Construction and Realization (3)

Like the cooperation between design and realization, the cooperation between construction and realization must be distinguished according to the type of element to be realized. Before we discuss the special properties of software and physical parts in detail, let us first consider the general aspects of the cooperation between construction and realization:

- *Clarifying technical details for realization:* Construction has to provide the technical details of the digital solution that are necessary to implement the solution. At the same time, realization is responsible for pointing out inaccuracies and gaps in the realization concept of the solution so that construction can correct and improve them.
- *Quality assurance of underlying form, function, and quality:* The underlying form, function, and quality provide the foundation of the digital solution and are the responsibility of construction. Therefore, construction is responsible for ensuring that the underlying form, function, and quality of the digital solution are realized according to the realization concept. The concrete approach for fulfilling this responsibility depends on the particular building process.

Good examples for the cooperation between construction and realization in terms of quality assurance are automated tests (e.g., unit tests) of individual elements of the digital solution and

static analysis of source code (e.g., SonarQube) to ensure a certain level of quality in the source code. Again, the detailed approach for this should be captured in a dedicated evaluation concept.

The construction of a physical device ends, together with the design process, with a model of the product that will be reproduced in a mass production process. The construction of a software element is an ongoing activity together with design during the whole realization process.

Construction and realization in software development can benefit from delayed decisions

Important technical decisions for a software element have to be taken before the actual realization starts (see Section 1.3.1). Nevertheless, the cooperation between construction and realization can also utilize the idea of delayed decisions. For example, if a certain functionality requires the selection of an existing open-source component and the decision about this component impacts other parts of the solution, this decision can be delayed until shortly before the actual implementation of that component starts. The benefit of this delayed decision is twofold: first, there is a better understanding of the details of the function at hand (e.g., data structures or interfaces to other components), which leads to a better-informed decision on the component; second, in the course of the building process, new components may emerge that were not yet available during the planning of the digital solution. Such a situation is not uncommon, since the software community works continuously on the further development of its technologies.

For a physical device, the cooperation approach is similar to the cooperation between design and realization. Again, all relevant decisions and quality assurance measures for the product must be taken before the mass production starts. Therefore, the cooperation between construction and realization also relies on prototypes (see Section 1.3.2.2). However, construction uses these prototypes to work on the underlying form, function, and quality of the digital solution. For example, an appearance prototype can be used to evaluate whether the technical components fit into the housing of the device.

1.3.3.4 Cooperation between Design, Construction, and Realization with Management (4)

In the following, we briefly discuss the challenges of managing the building process. This introduction is mainly intended to create awareness of the challenges and is not meant as a guideline for planning and management.

The previous three subsections showed various pairs of intersections between design, construction, and realization. These intersections focus mainly on the content details of designing, constructing, and realizing the form, function, and quality of the digital solution, including their evaluation of the form, function, and quality. The management and planning of the building process require cooperation between all three activity areas since the competences of all three areas are necessary to create a coherent perspective on the building process.

Iterative and incremental work is the core cooperation mode

The definition and alignment of the details of the building process are by no means top-down activities from design to construction to realization. Such an approach typically leads to suboptimal results. For the development of software in particular, the agile development movement has shown that an iterative and incremental management approach is much more suitable.

Such an iterative and incremental approach requires close cooperation between experts of all three activity areas since each activity area provides certain input for another activity. The bad news is that there are mutual dependencies between the activity areas and that these complex

dependencies make it impossible to define a general planning approach for the building process for a digital solution. In the following, we illustrate these dependencies with some examples.

- The appearance and behavior of the user interface (design) drives the decision for the selection of technical components for the implementation of the user interface (construction). A certain technology may provide features for implementing the user interface that design has not thought about (realization). Not using this potential will create a digital solution that does not make use of the full potential of technology. A plan for building a digital solution with a significant number of user interfaces should plan for an extensive exchange between design, construction, and realization experts to make use of the full potential of technology.
- The development environment (the environment for implementing and testing an element of a digital solution) can be viewed as a pure realization aspect. Modern development environments, however, provide important features that are also useful for design and construction, such as tools for modelling data structures. Certain development environments especially support rapid prototyping, i.e., detailed design aspects of a digital solution can be defined during the realization and immediately validated and improved together with stakeholders.
- The planning of the evaluation of certain aspects of a digital solution (e.g., the user acceptance of the processes defined) must be aligned with other activities. On the one hand, the part of the solution that shall be evaluated must be realized properly. On the other hand, the proper evaluation setup (e.g., a usability laboratory) must be set up properly, including inviting potential test candidates. The planning of such evaluation activities requires a rather long-term planning that may conflict with other activities.
- The implementation sequence of a digital solution can be driven by various aspects. A typical approach is driven by *user stories* (a description of the solution that creates benefit for a user, see Section 5.3.4.2) or a *minimal viable product* (MVP, see Section 5.4). Such implementation approaches typically cut across the whole structure of a digital solution since a functionality must be realized through the whole solution. A technically driven approach first implements all important basic functionality and then starts to implement the functionality relevant for the user. Such an approach requires more time until a user can provide feedback.
- The definition of time schedules is often driven by realization costs and time to market. Such an approach typically neglects the effort for design and construction, including the evaluation of the design and realization concepts. A better effort estimation and budget allocation can be achieved when design and construction are considered explicitly in the budget. Especially in project situations with a fixed budget, the design and construction competence are of great importance in obtaining the best possible digital solution for the available budget.

1.3.4 Managing the Building Process at the Solution, System, and Element Levels

The building process for a digital solution can be structured according to the hierarchy of abstraction levels presented in Figure 4:

- *Solution level*: understanding and shaping the goals and business model (if applicable) in relation to customers, stakeholders, existing systems, and the digital system as a black box

- *System level*: understanding and defining the overall digital system in relation to the objectives of the digital solution as an organization of the existing system, human beings as users, and elements to be developed
- *Element level*: understanding, defining, and realizing a particular element of the digital system that realizes the digital solution in relation to users, existing systems, and other elements

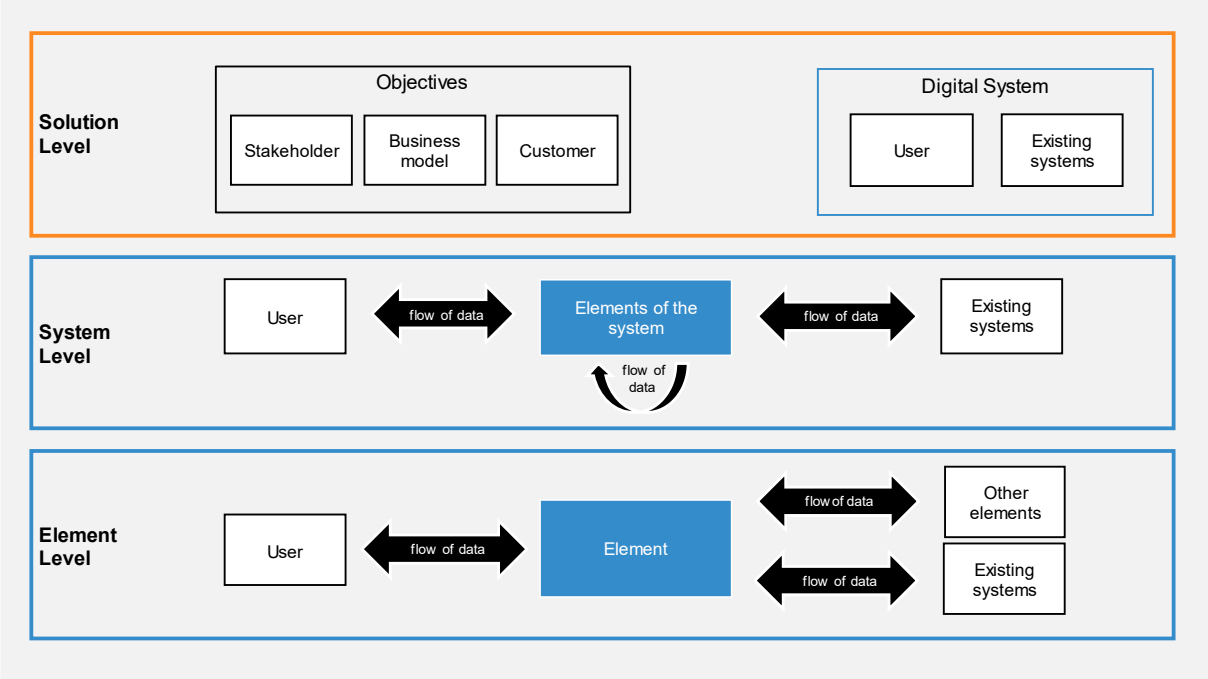


Figure 4 – Overview of the abstraction levels solution, system, and element

This understanding of solution and system represents an important terminological difference to other fields. Some disciplines consider the customer, the user, and existing systems as system context and only the elements as the system. This understanding was suitable for digitization (transforming existing things into the digital world). However, digitalization and digital transformation require the broader understanding outlined above, since novelties are created and the solution and system are shaped in parallel. The YPRC case study is an example of this. During the actual development of YPRC, new ideas for selling additional services emerged that became part of the business model.

The separation between solution, system, and element level is of great importance for understanding and managing the building process for a digital solution. Even very simple digital solutions typically consist of more than one element. As soon as a digital solution does not run on a single device, two elements are involved: the front-end part and the back-end part of the solution. The YPRC case study is also a simple digital solution and already consists of three elements.

The three abstraction levels should not be understood as three process steps that follow each other. They are meant to structure the view of a digital solution. The building process for a digital solution always starts with the solution level. With an initial understanding of the desired future, an iterative process starts that takes place at all three abstraction levels in parallel.

1.3.4.1 The Power of Understanding the Building Process at the Solution and System Levels

The building process for a digital solution always starts with the solution level. At this level, major decisions are made:

- The main objectives of a digital solution, including the customers
- The value proposition and business model behind the digital solution (if applicable)

Next to the solution level, the system level is important for defining the overall form, function, and quality of the digital system to achieve the defined objectives. With a proper initial understanding of the solution and system levels, the following activities can be performed:

- Planning for people necessary to build the digital solution
- Planning of the long-term evolution of the digital solution
- Planning and management of the subsequent building process for the elements
- Ongoing alignment of the building processes for the elements with each other and the system level
- Making core decisions on the technologies that will be used to realize the digital solution
- Evaluation and further development of the main objective of the digital solution
- Evaluation and further development of the overall idea of the digital solution
- Evaluation and further development of the business model (if applicable)

Planning for people necessary to build the digital solution

The main ideas of a digital solution are a starting point for planning the skills that are necessary to build the digital solution. Depending on the types of elements involved, different skills and a specific team diversity are required to create an effective building team setup[5]. Especially if the digital solution consists of one or more dedicated devices, skills in industrial design, production engineering, and embedded system development are necessary to build the dedicated device. Furthermore, the building of a dedicated device requires different management skills compared to a pure software-based digital solution. Also, for the software part, the system level provides initial guidance for the necessary skills, especially if special-purpose software is required.

YPRC example. For the YPRC case, we definitely need industrial design and production engineering skills if we want to create the smartwatch on our own (buying a white label product is also an option). For the software part, we need experts in smartphone app development, and for the artificial intelligence coaching part, we need experts in building artificial intelligence software.

Planning of the long-term evolution of the digital solution

Digital solutions can evolve over time—for example, new functionality is offered, existing functionality is improved. The system-level understanding of a digital solution can be considered as a starting point for planning this evolution.

[5] See Section 4.3 for more details on people management.

YPRC example. The description of the YPRC case study implies a natural order for this evolution: start by selling the smartwatch together with the app and the free portal for storing data. When there are sufficient users of the app, start the development of the artificial intelligence coaching and sell this service. Even later, start the development of the personal remote coaching and sell this service. This simple, long-term plan already provides an initial building plan for the digital solution. Nevertheless, this is not the only possible approach for building YPRC. In Section 2.1, we will come back to this plan and discuss possible alternatives.

Planning and management of the subsequent building process for the elements

A digital solution typically consists of more than one element that can and should be built in parallel. Such a parallel building process requires planning and management. The long-term evolution outlined in the previous paragraph tells us that we will start with the smartphone including the app and the portal for storing data. The building process for all three elements can start in parallel and can be managed from a system-level perspective as well. The main tool for allowing parallel building processes is a precise definition of the functionalities and technical interfaces between the various elements.

YPRC example. For the YPRC case, we have to define the interface between the smartwatch and the app and the interface between the app and the portal for storing data. The definition of these interfaces is a joint task of design and construction. A clear definition of the interfaces between the elements allows a structured building process for each element since each element can rely on a clear set of data and functions that it can expect from other elements.

Ongoing alignment of the building processes for the elements with each other and the system level

Building a digital solution is not a top-down process. Instead, there is extensive exchange of information between the system level and the element level since the building of the individual elements will of course lead to new insights and ideas about the digital solution. The most important points for insights are interfaces between the elements. Any change of the interface must be shared immediately with the element affected so that the building process for these elements can incorporate the change. The availability of new technology is another source of change.

YPRC example. The availability of new artificial intelligence technology in the YPRC case study may allow new coaching services to be offered. These insights must be shared between the building processes for the elements to make use of the new technology.

Making core decisions on the technologies that will be used to realize the digital solution

With an understanding of the solution and system levels, it is possible to explore different technologies for realizing the digital solution.

YPRC example. The different elements of the YPRC case study can be realized by various technologies (for example, Java for the portal). If innovative technologies are being considered, an early understanding of the system level is very useful for discussing the applicability of innovative technology. For example, artificial intelligence (AI) is considered for realizing an AI-based coach. Developing and training an AI that is capable of coaching runners during training is very challenging. The main challenges (for example, training data, sufficient computation power) for such an approach can already be discussed with an initial understanding of the system level.

Evaluation and further development of the main objective of the digital solution

When a digital solution is driven by the market and is not created for a closed organization, the acceptance of the digital solution on the market is very important.

YPRC example. The main objective of YPRC is to provide an affordable "all-round coaching solution for beginners in long-distance running." At first glance, this objective appears to be reasonable since long-distance running is a popular sport. Nevertheless, an evaluation of the defined objective—for example, by means of a market survey or a customer survey—can be beneficial in order to better understand the acceptance of this idea on the market. A typical question can be: Is there a sufficient number of potential customers on the market that will buy such a solution?

In Section 2.1, we provide further details on possible approaches to the evaluation and further development of the main objectives.

Evaluation and further development of the overall idea of the digital solution

The solution level already summarizes the main idea of a digital solution. This main idea can also be used for evaluation purposes.

YPRC example. The main idea of YPRC consists of three elements: sell a smartwatch together with a free smartphone app and a portal for storing health data. Through the app, the owner of the smartwatch can purchase additional coaching services. Even at this abstract level, the description of these ideas can be used to discuss YPRC with potential customers to gain more insights about the acceptance of this solution.

In Section 2.1, we provide further details on possible approaches to the validation and further development of the overall system level of a digital solution.

Evaluation and further development of the business model

With an initial understanding of the solution and system levels, it is possible to evaluate initial business model ideas. In Section 4.2, we provide further details on different types of business models for digital solutions.

YPRC example. The description of the main idea of YPRC in the previous paragraph already outlined certain aspects of the business model of YPRC (sell smartwatch, sell premium services) and the cost drivers of YPRC (cost for development and mass production of a smartwatch, cost for developing and maintaining the smartphone app and the portal, and the operation costs for the portal). This information can be used to elaborate a first draft of the business model of YPRC and to evaluate whether the business model can become a success.

All these activities are possible only if there is a proper understanding of the solution and system levels of the digital solution. In Section 2.2.2, we discuss practical approaches for creating design concepts for the solution and system levels as a starting point for documentation and as a foundation for managing the building process at the solution and system levels.

1.3.4.2 The Difference between a Concept-Driven and a Realization-Driven Building Process

As with the activities, the building process at system level and element level takes place in parallel (see Figure 5). In the previous section, we showed that the process starts with the solution and system levels. When the solution and system levels have been understood properly, the design and construction of each element can start, but in close cooperation with the design and construction of the solution and system levels.

When the realization of the first element of a digital solution starts, an important management shift has to take place. The start of the realization of an element turns the building process from a concept-driven process (i.e., pure concept working) into a realization-driven process (i.e., driven by implementation work). A concept-driven process that focuses on design and construction is cheap and fast. New insights can be incorporated into the concept easily and at little cost. A realization-driven process is, generally speaking, expensive and slow. Realization teams (for example, software development teams, cf. [SeRP2017]) require constant input to remain productive. The incorporation of new insights and correction of significant errors are expensive. Parts that have already been built may have to be altered at additional cost.

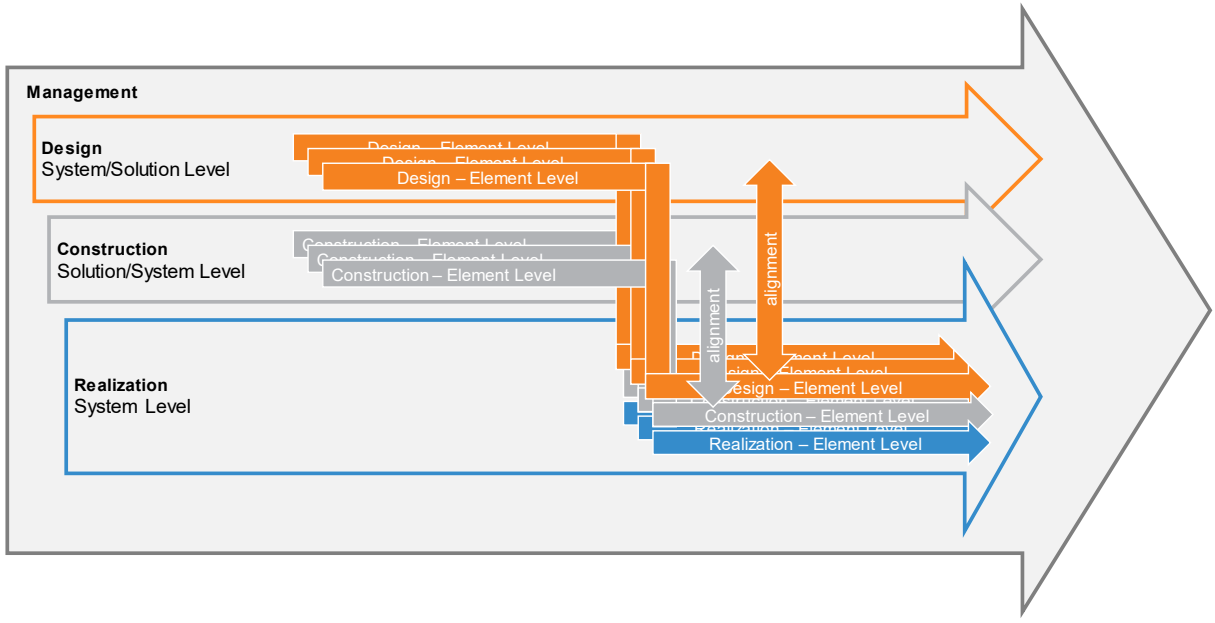


Figure 5 – Activities of the building process work in parallel at system and element level

The difference between concept-driven and realization-driven processes is not an argument for working as long as possible in the concept-driven process to create concepts that are as detailed and as verified as possible. It is meant as a warning that the start of realization is costly, and that the important and costly details must be clarified before realization starts. Catching these important details requires experience and training.

1.4 Competence Profile of a Digital Design Professional

A Digital Design Professional (DDP) is a person who is considered competent in the field of Digital Design. This handbook describes the skill set necessary for a foundation level understanding of Digital Design. An overview of this skill set is given in the following subsection. In addition, it is important to recognize that the DDP is not a new role. We elaborate on this in Section 1.4.2.

1.4.1 The Digital Design Professional as an Education Scheme

The structure of this handbook follows the idea of the pi-shaped competence profile of Digital Design [Bitk2017] inspired by the Greek letter π as a symbol and is structured as in Figure 6.

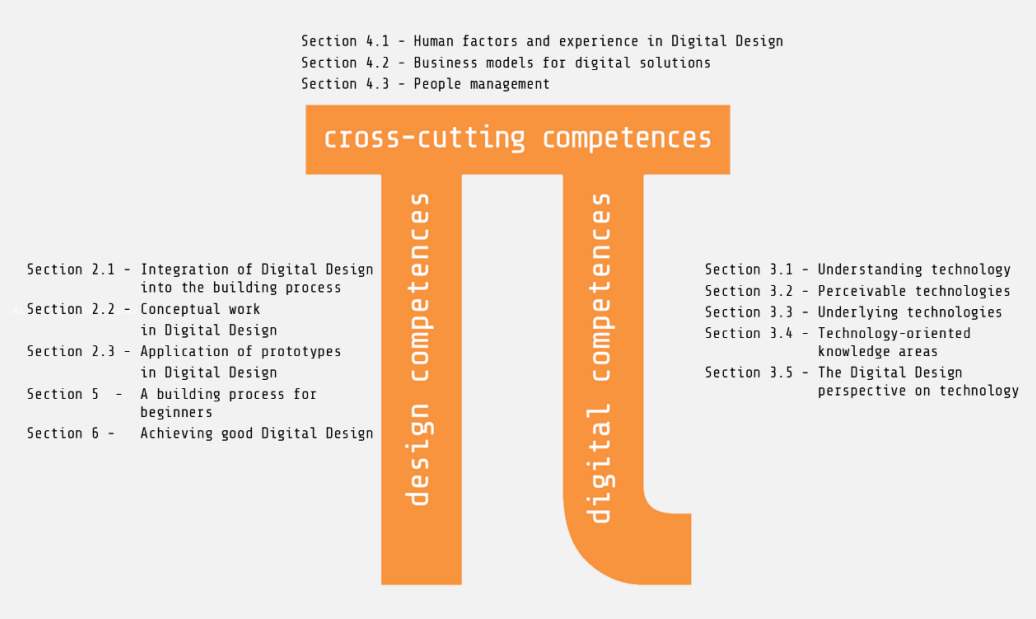


Figure 6 – The π-shaped competence profile of a DDP at foundation level

The left leg of the pi represents the design competence. At foundation level, the following aspects are important:

- Integration of Digital Design into the building process (see Section 2.1)
- Conceptual work in Digital Design (see Section 2.2)
- Application of prototypes in Digital Design (see Section 2.3)

The right leg represents the understanding of digital as a material. At foundation level, the following aspects are important:

- Understanding technology (see Section 3.1)
- Perceivable technologies (see Section 3.2)
- Underlying technologies (see Section 3.3)
- Technology-oriented knowledge areas (see Section 3.4)
- The Digital Design perspective on technology (see Section 3.5)

The top of the pi represents cross-cutting aspects of Digital Design. At foundation level, the following aspects are important:

- Human factors and experience in Digital Design (see Section 4.1)
- Business models for digital solutions (see Section 4.2)
- People management (see Section 4.3)

In order to apply this broad competence profile, Chapter 5 presents a building process that is intended for beginners in Digital Design. This process provides a set of concrete and pragmatic techniques that are suitable for building digital solutions with medium complexity.

The competence profile of a DDP is completed by the ten principles of good Digital Design. The following subsection presents these principles and Chapter 6 discusses how the methods and techniques from this handbook contribute to achieving the ten principles of good Digital Design.

1.4.2 Ten Principles of Good Digital Design

In general, a profession is defined by its methods, techniques, and values. The ten principles of good Digital Design presented in the Digital Design Manifesto [LBGH2018] define the principles that guide the values of Digital Design and the attitude of people who work in the Digital Design profession.

The following subsections introduce the ten principles in detail according to the Digital Design Manifesto [LBGH2018]. Keep in mind that the ten principles are not a checklist; instead, they describe an attitude toward digital as a material and the design of good digital solutions. They always refer to good Digital Design and by this they mean both the design process and the result. This is important because we think that the process and the result are inseparable.

P1 - Good Digital Design is useful and usable

Digital technologies are supposedly some of the most powerful technologies that have ever been invented by humankind. Good Digital Design uses these technologies to create benefits and added values.

At the same time, however, digital technologies are supposedly the most complex technologies that have ever been invented. This complexity must not become a problem for the user. Therefore, good Digital Design is devised such that the resulting digital solutions can be used easily, with joy and a good experience by the user.

P2 - Good Digital Design is elegant and aesthetic

Just like well-designed (analog) products or buildings have their own elegance and aesthetics, good Digital Design solutions are also elegant. This elegance refers to the users' expectations for the perceivable form (e.g., elements, such as aesthetic interfaces or elegantly designed user devices).

However, *invisible elements* are also elegant. Along with the underlying form, function, and quality of a digital solution, algorithms, data structures, and software architectures can develop their own elegance. The invisible elements convey this, for example, through their simplicity, efficient processing, reusability, good maintainability, or intelligent use of the technical possibilities offered by digitalization. They thereby promote their own usefulness for the development of digital products, systems, or services.

P3 - Good Digital Design is evolutionary

No digital solution is perfect from the very beginning—all digital solutions continue to evolve. In addition, all digital solutions are continuously subject to external influences, which inevitably leads to the necessity to change. Good Digital Design is devised such that it has a long lifetime. Changes and further development are made as simple as possible or are not hindered unnecessarily.

P4 - Good Digital Design is exploratory

Believing that you always know in advance which solution will work is naive. This is particularly true for the innovative field of digitalization. Good Digital Design is exploratory—it allows users, designers, and developers various options for achieving their objectives. It draws conclusions from users' behavior to detect the best way forward and to develop this path further.

P5 - Good Digital Design focuses on the person as a whole

User-centered design is an important design principle but is too short-sighted as the user is also a person within society as a whole. The revolutions expected as a result of digitalization therefore mean that the focus must be shifted to the person as a whole and their environment and society.

Today, for example, many fully digitalized workplaces often tie employees to their desks and lead to a lack of movement for many office workers. Good Digital Design can design digital workplaces such that movement elements become an integral part of the work and therefore promote employees' health.

P6 - Good Digital Design anticipates the effects of its results

Digitalization does not take place in a vacuum; it has an effect on all people and on society as a whole. Current developments show that not all effects of digitalization are really desirable and can even lead to undesirable side effects. Some of these effects only become visible in the course of time when a large number of customers have adopted a solution. Sometimes, it turns out that a solution is used in an initially unintended way or by customers it was not designed for. This results from interactions between customers and other social effects within society.

Therefore, good Digital Design anticipates the effects of its results and soundly balances the advantages of a solution with the disadvantages that arise.

P7 - Good Digital Design respects data protection and data security

Which data is stored and how it is processed is a Digital Design issue. Therefore, data protection and data security begin in Digital Design. From the very beginning, good Digital Design considers applicable data protection laws and uses data sparingly, meaning that it uses only data that is indispensable for the intended purpose.

In good Digital Design, sensible and critical data is specially protected according to its importance through careful design and the use of current technologies that take account of risks of data theft.

P8 - Good Digital Design is sustainable and creates sustainability

Digital technology uses a lot of energy every day. Manufacturing user devices such as smartphones, tablets, or smart watches also consumes a lot of resources. Energy and valuable resources can be saved through intelligent design. Conversely, intelligent digital solutions can create sustainability. For example, digital means of communication can reduce the necessity for travel and thus save energy. Intelligent digital control systems already save energy in a lot of areas of industry and private life today.

Good Digital Design must contribute to sustainability and therefore favors solutions that minimize energy and resource consumption in relation to its benefit. However, sustainability in Digital Design goes beyond the environmental dimension and must be understood as a set of dimensions that also includes individual, social, economic, and technical aspects (cf. [BCDE2015]). This also covers maximizing the quality of the digital solution to ensure a long lifetime. In digitalization,

sustainability also includes anticipating disposal. Therefore, before realization, good Digital Design plans what happens with data or end devices when solutions are no longer used or when users pass away.

P9 - Good Digital Design appreciates analog and digital means equally

Analog and digital are not contradictions—they merely describe the poles of a spectrum. Just because something that was previously analog is now digital (e.g., paper books compared to e-books), this does not necessarily mean that the digital solution is better.

Good Digital Design does not have to maximize the digital aspect. Digital means should only replace analog means where this is appropriate and constructive. If an analog element is equal to or even better than the digital element, the analog element should be used. The high potential of hybrid digital solutions can only be used to achieve real innovations when analog and digital means are appreciated and considered in the same way.

P10 - Good Digital Design uses digital means only where this is necessary

Without doubt, digitalization is a leading force for progress. However, this is precisely why it must not become an end in itself, because in this case, it would lose its credibility and power of innovation to improve the lives of all people within society.

Good Digital Design uses digital means intentionally and where this is necessary and creates benefits for the lives of people within our society.

1.4.3 The Digital Design Professional is Not a New Role

It is important to recognize the difference between a role and a profession:

- A role is a position that a person can take in a given situation. A role is defined by its tasks, rights, duties, and responsibilities.
- A profession is an occupation that requires specialized education.

The DDP is not a role—it represents a training program for entering the profession of Digital Design. This foundation level handbook aims to provide a broad overview of Digital Design and at the same time, provide hands-on methods and techniques for practical work. These competing goals require a selection of the content presented and a simplification of the content. In order to not lose sight of the big picture at foundation level, references to further literature are given.

During the building process for a digital solution, a DDP can work in various roles that are related to the activity area design (e.g., product owner, business analyst, requirements engineer, usability engineer). However, due to the broad scope of Digital Design, certain roles will require additional specialized education to achieve good Digital Design (see Chapter 6). References to further literature are given as a starting point for this education.

The core advantage of being a certified DDP is that it gives you a broad understanding of the building process for digital solutions. Furthermore, you then have a broad competence in the design of digital solutions, including the necessary material and cross-cutting competences. This broad knowledge from the DDP education supports in particular experts with specialized training in better integrating their personal strengths into the entire building process for a digital solution.

2 Design Competence

In this chapter, we introduce the world of design competence for Digital Design.

We start with the integration of Digital Design into the building process in Section 2.1. The design of a digital solution cannot be understood without understanding the whole building process. The main reason for this is that the design of a digital solution evolves over different abstraction levels (solution, system, and element) and these abstraction levels do not correspond to a process model. Quite the opposite is true: due to the complexity of a digital solution, important design decisions on the overall solution that shape the digital solution are made very late in the building process. We discuss this in detail when we introduce the fundamentals of design processes together with a pragmatic process model for the building process for a digital solution.

Besides process knowledge, design competence consists in particular of the ability to communicate with the relevant stakeholders about the planned solution (cf. [Cros2006]). This communication especially includes the elaboration of details and the review of the planned solution. Concepts and prototypes are central tools for this communication. In Section 2.2, we therefore introduce conceptual work in Digital Design, with Section 2.3 introducing the application of prototypes in Digital Design.

2.1 Integration of Digital Design into the Building Process

The main characteristic of the process of designing a physical product is the separation of design from manufacture: the creative act of determining and defining a product's form and function takes place in advance of the physical act of making (realizing) the product, which consists purely of repeated, often automated, replication (cf. [Nobl1996]).

Every process model that works with implicit assumptions of mass production processes at the end of the design process is of limited use for building a digital solution. The design of a digital solution is an ongoing process during the whole building process (see Section 1.3.3). Building digital solutions requires process models that provide guidance for integrating design activities into the whole building process.

Nevertheless, studying the fundamentals of design processes is important for developing your own understanding of how design works (cf. [Dors2003]). In Section 2.1.1, we therefore provide an overview of the fundamentals of design processes as an entry point to the world of design and design processes.

In Section 2.1.2, we present three essential steps of the building process for a digital solution and discuss their relationship to Digital Design. In Section 2.1.3, we discuss the consideration of quality as a cross-cutting concern during the building process. Section 2.1.4 presents further important fields of competence for the building process for a digital solution. On the one hand, this overview serves as a starting point for the further acquisition of competence and on the other hand, is intended to show how Digital Design is related to other fields. To conclude the integration of Digital Design into the building process, we present an idealized model of the building process in Section 2.1.5.

2.1.1 Fundamentals of the Design Process

Learning to design is like learning any other complex skill: it requires education and training [Cros2006]. Design education is based on design process models and a description of design

work, but these do not replace training and real practice. Cooking is a good analogy: assume you want to learn to cook good Italian pasta with tomato sauce. You get a recipe and follow the instructions for cooking the sauce. The result can be good or bad because the recipe covers only the abstract steps and ignores the concrete case. Maybe the tomatoes were too watery when you first tried to cook them, and the sauce would have had to cook longer than described in the recipe to make the sauce tasty. Or maybe you had the wrong kind of tomatoes.

You can study the sauce recipe as long as you like; the result will probably not be much better next time. The pasta sauce will become better if you start experimenting with the recipe (e.g., with tomato paste or other tomato varieties) and critically examine the results. This will improve your understanding of the inner relationship between the recipe (concept), ingredients (materials), and the pasta sauce (solution). You can only gain an intuitive understanding of the procedure and improve your skills in cooking pasta sauce through numerous experiments. Over the course of this process, you need the recipe less and less and toward the end, you can cook the sauce without a recipe because you have internalized it.

The important lesson from this analogy is to not confuse cooking with studying the recipe. The recipe is the starting point and not the end. Design process models are, just like the work instructions in a recipe, an essential part but they are a starting point and it is useful to study them. Following a design model will allow you to achieve a basic understanding of how design works and such models are a good starting point for a DDP at foundation level. However, keep in mind that you really develop design skills by practicing and experimenting.

In the following, we present a model that provides an initial understanding of the design process and allows us to discuss important aspects of every design process. To conclude the fundamentals, we present a model that allows us to discuss the nature of design work.

2.1.1.1 The Design Squiggle as a Design Process Model

The first model is the design squiggle by Damien Newman ([Newm2020], see Figure 7). It shows that a design process is typically a chaotic and iterative activity that somehow leads to a clear understanding of a particular design solution. The design squiggle shows three different and strictly separated stages:

- *Research & synthesis*: This stage is about understanding the problem space and gaining insights into users, customers, and the situation at hand. It is typically characterized by a rather unstructured journey with a lot of noise and uncertainty. However, at some point in time, the process leads to an understanding of the problem.
- *Concept/prototype*: Once there is an initial problem understanding, it is possible to create initial concepts and prototypes to explore possible solution ideas. Again, this process is not really linear. Concepts or prototypes may lead to completely new insights into the problem. It may even mean that the original problem has to be discarded completely and that the process has to start all over again.
- *Design*: At some point in time, one solution idea comes forward as the final solution. Now, the process becomes linear, since the one solution has to be elaborated in all details until it becomes the final design. It is important to recognize that here, the term design refers to the final result (design as a noun) and not the activity (design as a verb).

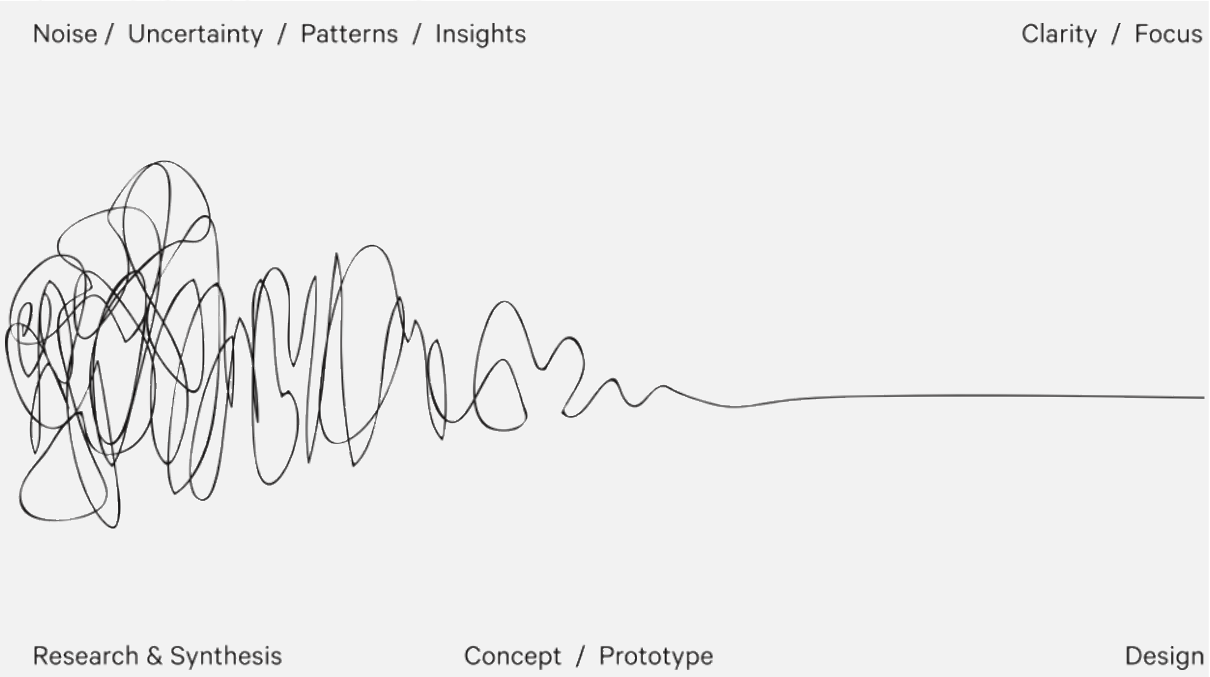


Figure 7 – The design squiggle [Newm2020]

Furthermore, the design squiggle presents three important core aspects of the design process that should become part of the attitude of every DDP.

Attitude 1: Always try to understand the environment before working on solution ideas

A typical beginner's mistake is to understand design only as the third stage of the squiggle. A prerequisite for creating good design solutions is a proper understanding of the overall environment for which the solution will be designed. Special emphasis should be given to the context of use, i.e., to the customer of the solution and the environment in which the solution will be used.

A profound understanding of the environment is important because many detailed design decisions depend on the tiny details of the environment. A typical pitfall for a DDP is the belief that they already have a good understanding of the environment and do not need a detailed research stage for a particular solution. Today, the environment can change very quickly, so knowledge once acquired also quickly becomes outdated. We therefore highly recommend that you do not skip the research stage and that you clearly separate the research from the work on the digital solution.

In Section 2.1.2, we provide further guidance on how to understand the environment.

Attitude 2: Ongoing evaluation of everything

A second beginner's mistake is to believe too quickly in your own understanding of the environment and the solution ideas that have been created. The design squiggle clearly shows that the design process is a rather chaotic one that goes back and forth between the different stages. One reason for this is that the initial understanding of the environment is seldom right and that initial solution ideas are seldom the best solution. Inexperienced designers often learn this lesson when their solution ideas fail with the customer or the market.

Experienced designers have learned to tolerate the uncertainty and have made the evaluation of everything (understanding of the environment and of solution ideas) part of their attitude, always

looking out for opportunities to evaluate their understanding and their design. This does not mean that experienced designers always apply heavy evaluation methods; it only means that experienced designers are aware of the limits of their own understanding, which in practice, also requires discussion of insights from the first-person perspective.

In Section 2.1.2, we provide further guidance on how to approach evaluation within the building process.

Attitude 3: Iteration as a working mode

A third beginner's mistake is to assume a linear design process. The design squiggle highlights that every design process is a rather chaotic and iterative process. Only at the end, when the solution idea is really clear, does the process become rather linear.

Experienced designers have learned to tolerate this way of working and have made iteration (i.e., going back to understanding the environment and working on solution ideas again) the normal way of working. It is only through several iterations that several solution ideas can be created and evaluated to identify those solution ideas that really are promising. This does not mean that the whole process and the whole team performs an iteration—sometimes, an iteration can also be a rather short event that takes place only in the mind of a designer.

In Section 2.1.2, we provide further guidance on how to work iteratively within the building process.

2.1.1.2 The Dual-Mode Model of Design

Being a good designer means developing a design personality. For us, a design personality especially means cultivating the ability to switch between the first-person and third-person perspectives. The third-person perspective means a view of design activities that is as objective as possible, with a focus on models, theories, (customer/user) research, principles, etc. The first-person perspective on design means understanding design from your own experience (from doing design) and learning in interaction with experienced designers.

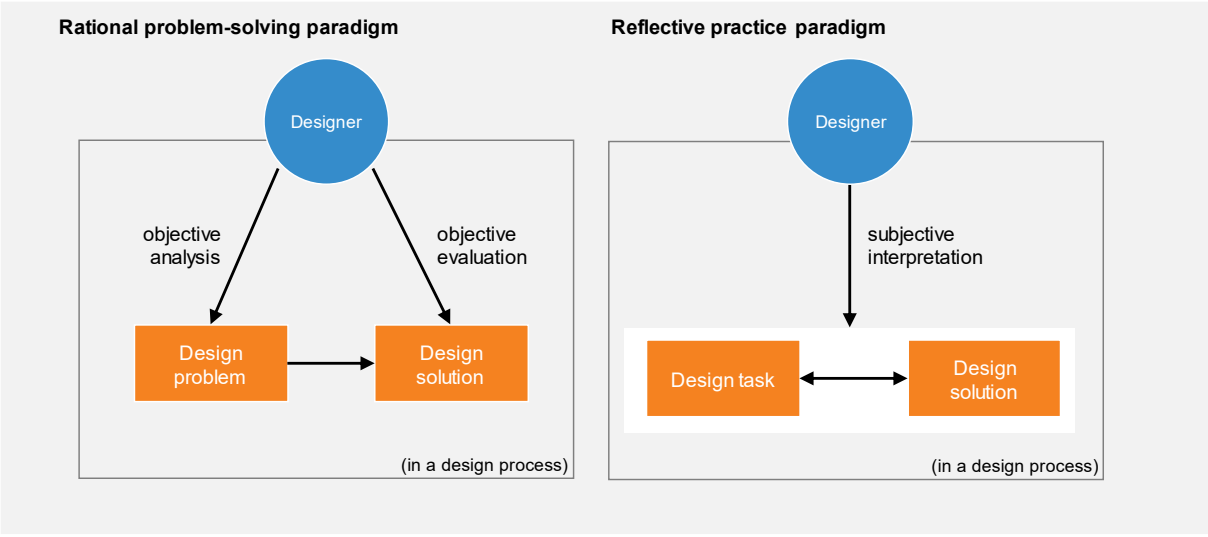


Figure 8 – The dual-mode model of design [Dors1997]

The dual-mode model of design [Dors1997] describes these two perspectives with two modes of design activities (see Figure 8):

- *Rational problem-solving paradigm/the objective mode*: in this mode (left part of Figure 8), the designer works objectively on the design problem (working in the problem space) through rational analysis and observation and wants to solve it. The goal here is to work as objectively and as rationally as possible. The insights from analysis and observation are transformed into a design solution (working in the solution space). The solution itself is again observed, analyzed, and evaluated.
- *Reflective practice paradigm/the subjective mode*: in this mode (right part of Figure 8), the designer works subjectively on the whole design situation, wants to understand it, and aims to define a way to proceed with the design task. The designer works on the design task (the given situation and the timeframe) in relation to the desired design solution. The designer is aware of their subjective perspective and their own background theory as well as the abilities of their implicit knowledge. Also, they can consciously adopt alternative perspectives on the design task. The subjective mode is important when the design task is unclear, ill-defined, or violates ethical or moral values of the designer.

For beginners in design, this model provides three important lessons:

- Design can be approached with an objective or subjective attitude. Both modes are important, and a skilled designer must make use of the two modes and switch between them when necessary.
- Working on the design process (for example, planning the next steps to analyze the problem or to create a prototype) is part of the reflective practice mode. There are many ways to approach a design problem and the message of the reflective practice mode is that the way to proceed in a given situation depends on subjective factors (for example, experience and education).
- If you are stuck with your design problem or with the solution, switch to the reflective practice mode and try to understand the whole picture. Question the process, question the given problem, and question your current understanding.

There is a lot more literature available on the fundamentals of design. In Chapter 6, we recommend several books as further reading.

2.1.2 The Three Essential Steps of the Building Process for a Digital Solution

The building process for a digital solution can be separated into three essential steps: the scoping step, the conceptual step, and the development and operations step. Keep in mind that in each step, the design process follows a process that is similar to the design squiggle (see Figure 7). Furthermore, the design work will switch back and forth between the objective and the subjective modes during the whole process (see Section 4.3). The main difference between the steps is the abstraction level at which the design takes place.

In the following subsections, each step is described in an abstract way. For each step, the general results and activities are described. Figure 9 summarizes the three steps and the general work products of each step (colored boxes) at their particular abstraction level. The color of each box indicates the activity area responsible; boxes with more than one color belong to several activity areas. Keep in mind that the different concept types shown in Figure 9 do not necessarily have to be maintained as independent documents or in independent tools. The distribution of concept types among documents or tools is a question of work organization.

The goal of the following description is to provide a general understanding of the different steps of the building process, including the general results and the relationship between the steps and the results. It is important to recognize that the three steps are not intended as a linear process model that can be used for immediate application. Iterations are important to create a feedback loop between the three abstraction levels (see Section 1.3.4.2). Instead, the steps describe different categories that are different in their nature and therefore, we refer to the three steps as being essential. The three steps can be used to understand the overall structure of a building process for a new digital solution. They can also be used to understand and structure the further development of a digital solution.

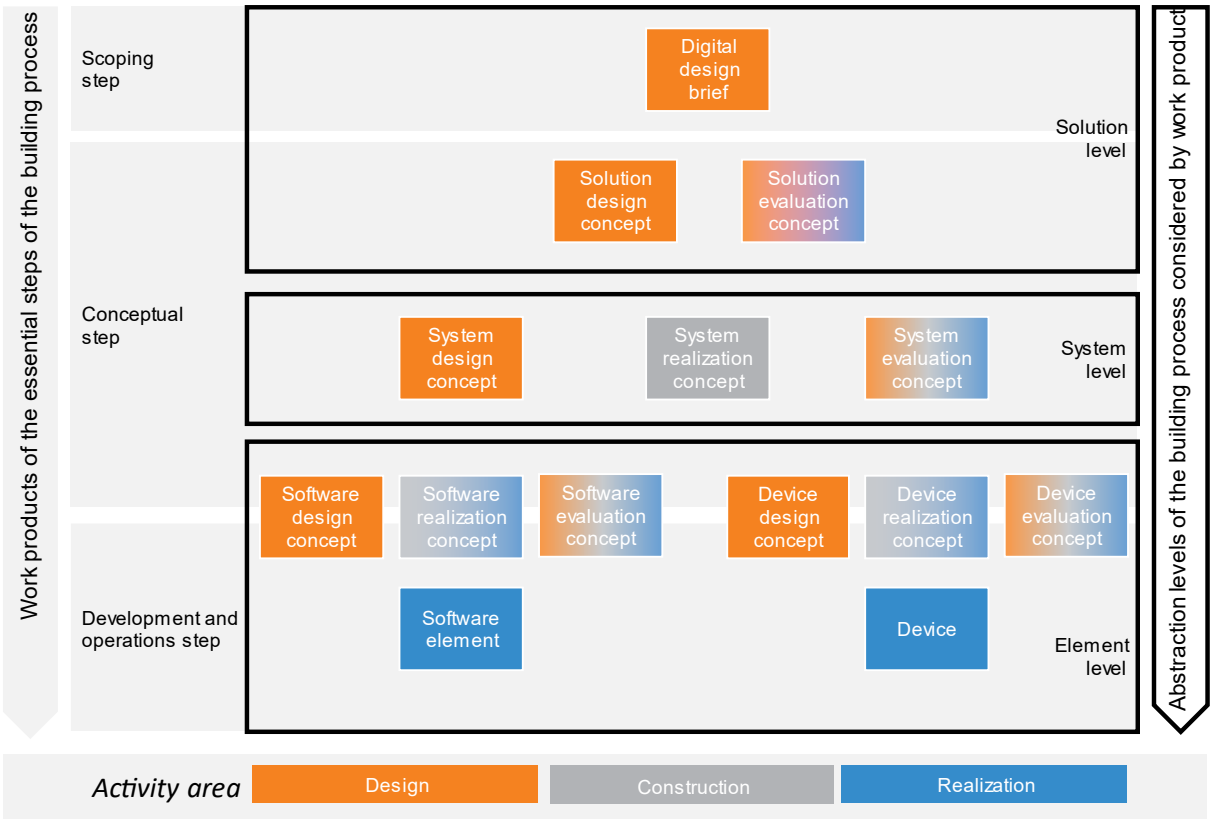


Figure 9 – The three essential steps and work products of the building process

An applicable building process for creating a new digital solution is presented in Chapter 5, including detailed methods, techniques, and process guidelines.

2.1.2.1 The Scoping Step

The goal of the scoping step is to elaborate a common understanding of the case for action (why are we starting a new building process?) and of the intended digital solution (what do we want to achieve?) with the client and relevant stakeholders. From a Digital Design perspective, scoping a digital solution should distinguish between four perspectives of the context (see Figure 10): the environmental, building, technical, and social perspectives.

These four perspectives must be considered together, because they define the area of conflict in which the digital solution will be created. In the following, each perspective is explained, including its relationship to the other perspective.

The environmental perspective

The environmental perspective focuses on the environment in which the digital solution realized is embedded. This perspective is often called context (cf. [BeHo1998], [CPRE2020], [CPUX2018]) and it can be separated into different facets (cf. [ISO2018]):

- *Direct usage context*: the environment (e.g., place, time, and surrounding conditions) in which the digital solution realized will be operated and used
- *Physical environment*: environment in which the context of use will be embedded (e.g., a department, a family)
- *Social environment*: environment that is relevant for and impacts the surrounding context (e.g., a company)

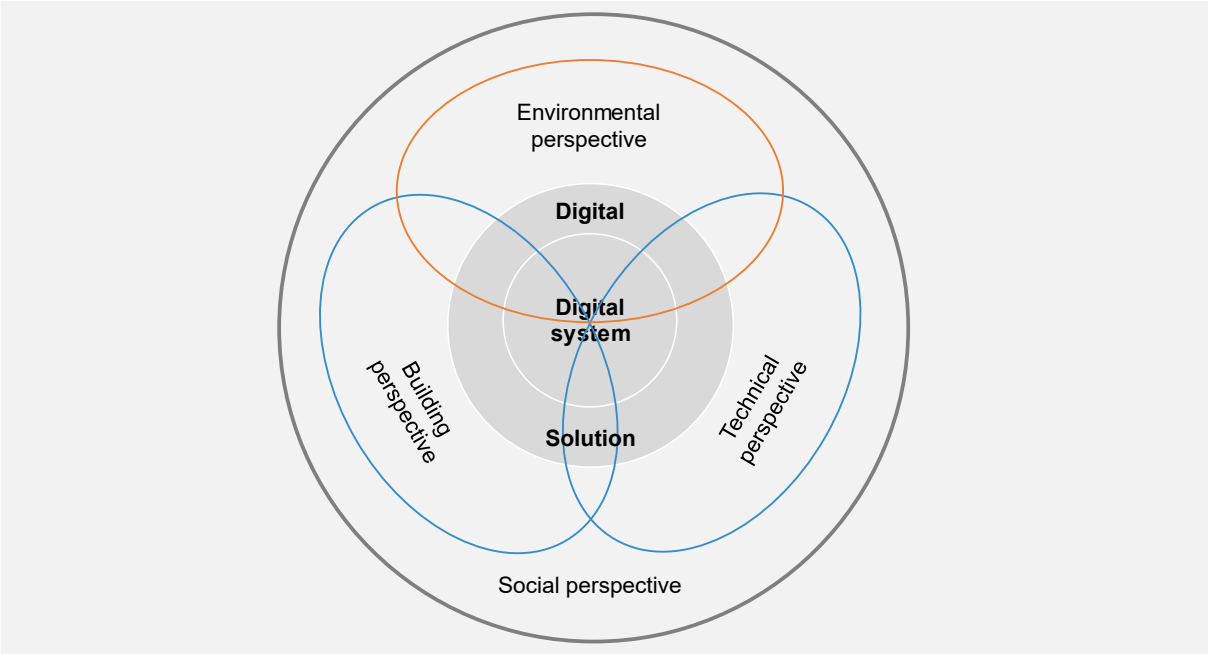


Figure 10 – The context scoping perspectives of Digital Design

Understanding the context of use and the direct environment of a digital solution is particularly important because the environment is an important information source for designing the digital solution. However, an in-depth analysis of the environment during the scoping step is difficult. The idea of an intensive initial analysis of the environment originates from a time when digitization was the primary problem. It assumes that the context already exists and is accessible for analytical methods in order to understand requirements that can be used to shape the digital solution. For digitalization and digital transformation, this understanding remains important but is not possible in the beginning since the environment created by a novel digital solution does not exist yet and is therefore only partially accessible for analytical methods. However, it is precisely here that the designer is called upon to develop a felt understanding for the environment from their subjective perspective in the sense of the reflective design paradigm.

A second challenge for understanding the context of use and the direct environment of a digital solution during the scoping step is that there is often no clear idea of the digital solution. Therefore, the recommendation is to characterize the environment of a digital solution during the scoping step in general by focusing on the direct and wider environment. For example, it is useful to focus on the problem or the case for action that is the motivation behind the need for a new digital solution instead of focusing on the customer during this early stage. In these early discussions

with the client, influencing factors for potential changes of the environment will occur and thereby new ideas of the value proposition are highlighted. This enables the preparation of the value proposition definition in the conceptual step (see below). This is done by collecting initial ideas for successful business models or customer journeys, creating awareness for the need for new business models or value propositions (see Section 4.2), by describing the motivation behind the digital solution (the case for action), and by establishing a common language for describing, designing, and analyzing the evolution of the digital solution over the course of the building process (see Section 4.3).

YPRC example. In the YPRC case study, this early and broad focus was the idea of improving the training experience for runners with digital technology. There were some ideas of what this improvement could look like. The case for action was that there seems to be some digital technology and data (remote communication, training data) that had the potential to improve this training experience.

The building perspective

The building perspective focuses on the part of the context in which the digital solution is built. The building perspective includes the following facets:

- *The building team*: the people (called building team members as stakeholder role) who are actually performing the activities of the building process
- *The building organization*: the organization that is driving and supporting the building process
- *The client organization*: the organization responsible for building the digital solution (e.g., the client organization) or the company of that the building organization is part of

The building perspective is often neglected when planning a new digital solution. The availability of skilled personnel and resources is often taken for granted. Such a utilitarian viewpoint on the building perspective poses a significant risk for innovative ideas, especially if skilled personnel is a scarce resource.

Another factor in this perspective is the client organization. Existing solutions or products in the client organization may compete with the new solution. In particular, when a digital solution aims to replace existing services (e.g., selling goods online as a replacement for warehouses), organizations that provide these existing services tend to start fighting against novel solutions to protect their existing business. Considering this perspective from an early stage is very important for understanding the forces and values that drive the political landscape in an organization. It is important to recognize that the building perspective does not include competitors or related solutions that are provided by other organizations. These aspects are part of the social perspective.

The technical perspective

The technical perspective focuses on the technical context that provides and develops digital technology. The technical context includes the following facets:

- *Available technologies*: the technology itself that is used for building digital solutions.
- *Technology providers*: organizations that supply hardware or software that can be used to build the digital solution. Providers become especially important when they provide a core technology (e.g., a programming framework) for a digital solution. In such a situation, the

new digital solution (and the organization that uses the digital solution) will depend on the provider.

- *People*: technology is provided and developed by people. They must be hired or trained in order to deal with technology.

During the scoping step, a broad consideration of the technical perspective is recommended to gather a broad understanding of available technology. In addition to existing technology, emerging and potential future technologies (e.g., blockchain) should also be considered during the scoping step. Even if a technology is in an early stage and not yet ready for the market, it could mature during the building process for the planned digital solution. Here, it is important to remember that we are in the scoping step and aim at a broad understanding of the technical possibilities.

YPRC example. A good example for this broad view on technology from the YPRC case study is the use of artificial intelligence (AI) on the smartphone. In the case study, the current smartphone hardware was not yet powerful enough to provide real-time AI coaching tips. Nevertheless, for example, the capacity of smartphone hardware will grow and eventually, the hardware will have sufficient power to provide real-time AI coaching tips. Taking this into consideration during the design of the YPRC case study is vitally important in order to be able to include this feature when the technology is ready.

The social perspective

The social perspective represents the overall social world in which the digital solution is embedded. The social perspective can include various factors that may become relevant. Important aspects of the social perspective include:

- Competitors that provide similar digital or analog solutions (e.g., companies that offer running apps)
- Companies with related services (e.g., companies that sell running services or running equipment)
- Organizations within the domain of the digital solution (e.g., running clubs)
- Political organizations/NGOs in the domain of the digital solution (e.g., political parties)
- Important political or social persons in the domain of the digital solution (e.g., the sports minister of a country)

In order to capture the results of the scoping step, we recommend creating a Digital Design brief (abbreviation: design brief):

Digital Design brief: The description of the context, vision, scope, and general terms for building a digital solution.

A detailed template for a design brief is presented in Section 2.2, including relationships to the scoping perspectives presented above.

A design brief can be created quickly if the client is clear about the scope of the planned solution. It can also become a project in its own right if the client does not have a clear picture of the digital solution, or if the organizational situation of business and IT is very complex, which is typically the case in large companies. In such a situation, a skilled and experienced DDP can take the role of a consultant to support the client in developing a powerful vision and a high-quality Digital Design brief. Here, the reflective practice mode of design is important (see Figure 8).

2.1.2.2 The Conceptual Step

In the conceptual step, the stakeholders and the building team elaborate an initial common understanding of the digital solution (including the digital system) that is sufficient to accept the risk of starting the development and operations step. The emphasis here is explicitly on risk since conceptual work serves this purpose in particular. The extent and level of detail of the conceptual work should be defined in terms of risk. For example, if the scope of a digital solution is small and well understood, a very short concept phase may be sufficient. If, on the other hand, many aspects of a solution are still unclear, much more extensive conceptual work is recommended.

Although the digital solution and the digital system are closely related (see Section 1.2), we recommend creating two dedicated design concepts: one for the digital solution and one for the digital system. The main motivation for creating two separate design concepts is that both concepts have different target groups and different purposes (see Section 1.2 for the difference between the solution level and the system level).

The solution perspective in the conceptual step

The solution design concept focuses on the socio-technical perspective and captures the client's and customer's requirements for the digital solution. It is defined as follows:

Solution design concept: The description of the goals, the business model, and the overall idea of a digital solution.

The business model describes what the digital solution shall achieve from the perspective of the client. This can include measurable goals, a cost driver, revenue streams, or assumptions regarding aspects such as key partners or customer segments. The overall idea is a description of the digital solution in the business language of the client and in the language of the customer in terms of value proposition. To capture the approach for evaluating the overall digital solution, we recommend creating a concept for this part of the quality assurance:

Solution evaluation concept: The evaluation concept for a digital solution.

At the solution level, there are several evaluation approaches that can be applied (e.g., prototypes, focus groups, expert reviews, or surveys). In general, it is important to recognize that the evaluation clearly focuses on the aspects that are defined by the solution, i.e., the business model and the value proposition. The solution evaluation concept is created initially in the conceptual step.

Further refinement of the solution evaluation concept takes place during the development and operations step (see below), since detailed aspects of the digital solution will be defined during this step and hence must be evaluated during this step.

YPRC example. The price for the remote coaching service is an example of this refinement. During the realization of the app, the idea of selling single remote coaching sessions for a certain price might arise. Defining this service is part of the solution level and evaluating the acceptance of this service (e.g., by means of a customer survey) is defined as part of the solution evaluation concept.

The system perspective in the conceptual step

The system design concept focuses on the system part of the digital solution, captures the system level, and is defined as follows:

System design concept: The description of the system-relevant objectives and of the overall form, function, and quality of a digital system.

The system design concept represents the system perspective—it details the digital system inside the overall idea of the digital solution at a level of detail that allows technical realization to be planned. As a complementary perspective to the solution design concept, the system design concept focuses on elements, buildings blocks, and their relationships and uses a rather technical language that is typically not directly accessible for non-technical stakeholders. Nevertheless, the solution and system design concepts are closely related, and it is the responsibility of design to keep both concepts consistent with each other.

YPRC example. If the business model of YPRC (solution design concept) defines that running sessions are purchased on a per session basis, the system design concept must somehow provide a possibility to pay for the service. If the business model changes—for example, the runner can obtain coaching sessions as a kind of reward—the system design concept must define the possibility to get these rewards.

Depending on the complexity and technical novelty of a digital solution, the recommendation is to create a realization concept that focuses on detailed realization aspects of the digital system inside the digital solution:

System realization concept: The description of the technically relevant system objectives and of the overall technical form, function, and quality of a digital system.

The system realization concept is the responsibility of construction experts. The concrete form and content depend on various factors and are not in the scope of this handbook. Nevertheless, it is the responsibility of design to evaluate the need to create such a system realization concept together with other stakeholders to minimize the risk of taking conceptual decisions that may result in expensive or even unrealizable solutions. Furthermore, it is a shared responsibility of design and construction to keep the system design concept and the system realization concept consistent with each other.

YPRC example. An example of a system realization concept in the YPRC case study could be the voice communication for the remote running coach. If the project team is unsure whether a remote voice connection over a mobile internet connection is technically feasible, they should consult voice over IP experts and start to elaborate a system realization concept that explores possible realization alternatives for such a functionality.

In order to capture the approach for evaluating the digital system, we recommend creating a dedicated concept for this part of the quality assurance:

System evaluation concept: The evaluation concept for a digital system.

The system evaluation concept combines the design, construction, and realization aspects. Therefore, the system evaluation concept is elaborated with experts from all activity areas. From

a design perspective, the system evaluation concept deals especially with the quality requirements that the whole system must fulfill.

YPRC example. A good example of a system evaluation concept from the YPRC case study is again the remote coaching service. For this service, several elements of the YPRC system must work together: the watch measures health data that is transferred to the portal by the app. From the portal, the data is then transported to the PC of the running coach.

This means that the health data is transferred through three elements before the coach can see it. In order to ensure a visualization of the health data in near real time, several system tests should be performed in various settings.

Such tests can be defined very early in the design process for the system. It is even possible to create a dedicated prototype for this purpose. An example of such a prototype during the conceptual step is described in the YPRC case study.

From a construction and realization perspective, the system evaluation concept deals with system tests and system integration tests. These tests deal with the technical integration of the whole system or with a subset of elements of the system.

Similar to the solution evaluation concept, the system evaluation concept is created initially during the conceptual step and is further refined during the realization of the digital solutions.

The creation of the solution-level concepts and the system-level concepts is an important step since both levels provide the foundation for deciding to start the next step of the building process. Therefore, the recommendation is to spend adequate effort creating and validating both levels by means of various techniques and prototypes.

Depending on the type of solution concerned, a DDP must involve additional experts in the creation process of both concepts. Different fields of competence are explained in this respect in Section 2.1.4.

2.1.2.3 The Development and Operations Step

The actual realization of the digital solution takes place in the development and operations step. We consider the development and the operation of a digital solution together in one step since the typical digital solution evolves continuously (e.g., new functions are added, existing outdated functions are removed). This means that the operation of a digital solution and further development during operation should be considered from an early stage in every building process. Although this topic is not the responsibility of Digital Design, it impacts the design of a digital solution. We will come back to this topic later in this section.

The important difference between this step and the previous conceptual step is that the concepts must be elaborated to a level of detail that allows the digital solution to be realized.

It is part of Digital Design to elaborate the elements of the digital solution. These details are typically captured in a design concept at element level:

Element design concept: The description of the element-relevant objectives and of the form, function, and quality of an element of a digital solution.

In contrast to the system design concept, the element design concept focuses on a particular element of the digital solution and must reach a level of detail that is sufficient for construction and realization. Furthermore, the element design concept is used to discuss and evaluate the

details of a particular element with relevant stakeholders. It is the responsibility of Digital Design to determine the right stakeholders that need to be involved in a particular detail of an element. For example, if the element under consideration is used by the user, feedback should be obtained from the user on all user-relevant aspects (e.g., shape of the user interface, functionality).

Furthermore, it is the responsibility of design to decide on prototyping activities to further evaluate important details of an element. For example, if the suitability of an important function for the user is unclear, or if the realizability of a functionality is questionable, you should plan for a prototypical realization of a function in order to evaluate the function.

There are important dependencies between the element level and the system level, and it is the responsibility of design to keep the system and the element design concepts consistent with each other. The details of these relationships are discussed in Section 2.2.6.

From a process perspective, the necessary level of detail can only be reached with an iterative design, construction, and realization process. This means that a DDP typically works together with additional experts from design (e.g., industrial designers, interaction designers, requirements engineers, UX professionals), construction (e.g., software architects, electrical engineers), and realization (e.g., software engineers, production engineers). A detailed list of related competences is presented in Section 2.1.4.

In addition to the element design concept, construction and realization activities typically create realization concepts that complement the element design concept. We call such concepts element realization concepts.

Element realization concept: The description of the technically relevant element objectives and of the technical form, function, and quality of an element of a digital solution.

In order to capture the evaluation approach, the element evaluation concept is defined as follows:

Element evaluation concept: The evaluation concept for an element of a digital solution.

Like the system evaluation concept, the element evaluation concept is a joint concept of the design, construction, and realization activities. The scope of the element evaluation covers the particular element and the interaction with the elements that are directly related to the element under consideration. From a design perspective, element evaluation includes the following two aspects:

- *Evaluation of the design of a particular element*: for example, the design of the functions offered and the design of the user interface. This evaluation can be performed by means of prototypes (e.g., user interface mock-ups) or by means of a part of the element that has already been implemented (e.g., a usability test).
- *Evaluation of the proper implementation*: for example, testing that the behavior is implemented as defined in the design concept. This evaluation is typically performed by means of test cases based on the design concepts and dedicated test activities that perform the test cases.

From a construction and realization perspective, the element evaluation includes different technical aspects. Important aspects include tests of technical interfaces to other elements of the system. These tests are typically performed by means of integration tests. Another important

technical perspective is the evaluation of the detailed implementation (e.g., by means of unit tests) and the evaluation of technical quality criteria related, for example, to maintainability of the source code (e.g., by means of code reviews or static code analysis).

The concrete procedure for creating the concepts at the element level and for realizing the particular elements depends on the process model selected for building the digital solution. In general, a preparatory phase should be planned before the start of implementation. The first details of the elements are worked out in this phase. Furthermore, preparatory measures for the realization are performed—for example, setting up workshops, work areas, as well as development tooling.

Building an initial version is different to evolving a solution during operation

Once a first version of a digital solution has been realized and is in operation, there is an important shift in the focus of the building process for a digital solution.

The first part of this shift is that the digital solution in operation requires care and maintenance. Users may report bugs that require fixing or may request additional functionality. This maintenance effort is often underestimated and creates a conflict between the maintenance and the further evolution of a digital solution.

The second part of this shift is that every decision on the further development of a digital solution must take into account that there is already an existing solution out there. From a design perspective, this shift creates opportunities and risks. Changes include obtaining feedback from real users or customers in real settings. This feedback can be used to understand and further improve a digital solution. On the other hand, the risk is that users or customers need to adapt to modification of the digital solution. Furthermore, the evolution of a digital solution must take existing technical structures and constraints into account.

For example, modifying a digital solution will require an update of the software parts or even a replacement of existing devices. Such an update or replacement may require substantial effort and planning depending on the type of digital solution concerned. Another challenge is created by updates and modifications of existing technologies that are used to build or operate a digital solution. For example, if the operating system of a smartphone is updated, modification may be necessary in order to keep the digital solution operational.

Solutions in operation present further possibilities for the design

Finally, an existing digital solution in operation allows for additional methods that are relevant for the further design of a digital solution (cf., e.g., [SaLe2016]):

- *Data-driven design decisions*: It is possible to explicitly include functionality to measure the behavior of users or customers to learn more about their needs. For example, it is possible to measure the time that a user spends in an app to evaluate the attractiveness of a particular app.
- *A/B testing*: A/B testing is an experiment-driven approach for evaluating design alternatives. Two alternative functions are realized and offered in a random fashion to different user or customer groups. The digital solution measures certain data about the user behavior that can be used to evaluate and compare the design alternatives.

Both methods rely on capturing detailed user/customer data. Therefore, these techniques have to be applied with care and require detailed consideration of data protection and privacy legislation in order not to violate existing laws.

2.1.2.4 On the Equal Importance of Scoping/Conceptual and Development Work

The term *upfront* is used for scoping or conceptual work and often has a negative connotation in literature (see, for example, [Meye2014]). It is expected that a DDP will have to deal with the discussion surrounding this term. Therefore, a clear position on this issue is necessary. The scoping and conceptual steps in Digital Design are a must because this is the only way to acquire relevant information that is necessary for effective and efficient development.

A concept-driven process that focuses on design and construction is cheap and fast. New insights can be included in the concept easily and at little cost. A realization-driven process is, generally speaking, expensive and slow. Realization teams (for example, software development teams, cf. [SeRP2017]) require constant input to remain productive. The incorporation of new insights and correction of significant errors are expensive. Parts that have already been built may have to be altered at additional cost.

This does not mean that a digital solution must be elaborated in every detail before the actual implementation can start. On the other hand, designing all the details of a digital solution shortly before the implementation takes place is also not a sustainable all-purpose process model.

The difference between concept-driven and realization-driven processes is not an argument for working as long as possible in the concept-driven process to create concepts that are as detailed and as verified as possible. It is meant as a warning that the start of realization is costly, and that the important and costly details must be clarified before realization starts. Catching these important details requires experience and training.

There are many aspects of a digital solution that can be designed and validated prior to implementation, especially by means of prototypes (see Section 2.3). There are also many aspects of a digital solution that can be validated based on the solution implemented. The real challenge is to decide which category an aspect belongs to and making this decision requires a lot of expertise (cf. [Rein1997]).

2.1.3 Quality as a Cross-Cutting Concern of the Building Process

2.1.3.1 Holistic Consideration of Quality during the Building Process

The quality of a digital solution is determined by various aspects. These include the technology chosen to build the digital system, the process used to develop the digital solution, and the understanding of quality within a given context (e.g., project setting, cultural aspects). To the best of our knowledge, holistic quality models that consider all these aspects do not exist, but researchers are communicating the need for more holistic approaches [BrDP2005].

The goal of this section is to create awareness of the importance of quality and the broad influence of various aspects on quality. This awareness is necessary to actively manage and shape the quality of the digital solution during the whole building process and to build a high-quality digital solution.

Key aspects to consider for a holistic consideration of quality during the building process include:

Quality as an attitude

All the aspects mentioned below are necessary to produce good quality. However, we would like to highlight that good quality depends above all on the attitude of all the people involved in building a digital solution. This is because the people involved make the many small, detailed decisions that ultimately produce the overall quality of a solution. If the people involved have a good attitude toward quality, they will make these decisions in terms of good quality. We have formulated essential elements of these attitudes in the ten principles of Good Digital Design (see Chapter 6).

Awareness that process quality influences product quality

Although for most companies, the goal is to deliver *high quality*, they need to be aware that the quality of the building process can have a significant influence on the quality of the digital solution and the digital system. Standardized and continuously improving processes [Demi2000] allow quality to be planned. This also means that the creation of a high-quality digital solution is a joint and holistic task of the building process: all activity areas must work together to deliver a high-quality digital system that realizes the digital solution. Quality control and management and the continuous evaluation of different artifacts built need to be key aspects of the underlying process as described in Section 2.1.

Actively managing quality and understanding different quality aspects

Improving the quality of a digital solution that has already been realized may require significant effort. Therefore, the desired quality must be defined and pursued from the very beginning of the building process. This includes considering different perspectives (i.e., technical, environmental, building process, and social) as discussed in Section 2.1.2.1, and also differentiating between qualities that can be seen and experienced by users and qualities of a solution that are hidden to users (see Section 2.1.3.3). Furthermore, a DDP needs to be aware of and understand the quality attributes for both the digital solution (see Section 2.1.3.5) and the digital system (see Section 2.1.3.4) in order to be able to plan accordingly.

Considering risk and value

Providing an adequate and expected level of quality to customers/users of a digital solution so that they accept and embrace the digital solution is what a DDP should aim for. Quality does not come for free and often, providing a higher level of quality also means that developing the digital solution is more expensive, eventually leading to higher costs for customers/users. Adequate consideration of risk and value is therefore important. This also includes understanding at what level of detail a quality aspect needs to be described and also which form of documentation—including the definition of adequate test metrics—is needed in order to reduce the risk of developing a digital solution that does not satisfy the needs of the customers/users [Glin2008].

Differentiating between the quality of the digital solution and the quality of the digital system

Understanding the difference between the digital solution and the digital system in general is important for a DDP. This is also true when it comes to quality. Considering the difference but also being aware that the qualities of the digital system influence the digital solution allows quality to be planned and, for example, adequate technologies selected to realize a digital system (see Section 3.1). This aspect is described in more detail in the following section.

2.1.3.2 The Difference between the Quality of a Digital Solution and the Quality of a Digital System

The quality of a digital solution has a significant impact on its acceptance and success. Nevertheless, the digital solution and the digital system are not identical. The digital system is the instantiation of the technical aspects (i.e., the hardware and software) of the digital solution and is therefore only part of the digital solution (see Section 1.2). There are qualities of a digital solution that are independent of the qualities of the digital system. However, the qualities of a digital solution can have an impact on and influence the qualities of the digital system. Vice versa, the qualities of a digital system become part of and add to the qualities of a digital solution. In particular, this is true for perceivable qualities that are visible and can be experienced (see Section 2.1.3.3).

For example, an online hotel booking service for tourists is a digital solution. This digital solution could, for example, be realized by a smartphone app that provides the features for booking a hotel online. This app is the corresponding digital system for the hotel booking service. The online booking service itself has qualities of its own, such as the freedom to book a hotel from all over the world, searching for hotels in various countries (see hedonic qualities in Section 2.1.3.5). These qualities can be defined independently without having a particular digital system and its qualities in mind. However, this might have an impact on the requirements of a digital system. The corresponding digital system can have its own qualities. For example, the booking app should be easy to use, demonstrate good performance, and provide good aesthetics. These qualities of the digital system add to the experience and quality of the digital solution.

2.1.3.3 Perceivable and Underlying Quality Attributes

With regard to quality attributes, it is important to differentiate between quality attributes that are perceivable to the users and those that are underlying or hidden to the users. The quality of a digital system includes both perceivable qualities (also called external qualities) and underlying qualities (also called internal qualities):

- Perceivable quality attributes are visible to a user or can be experienced by a user (e.g., usability, reliability). They create value for users, help to fulfill requirements, and foster system acceptance.
- Underlying quality attributes include qualities that are hidden to the user (e.g., maintainability) but enable developers to evolve and maintain the system at low cost.

However, although the differentiation between perceivable and underlying quality attributes allows better planning and testing for quality, it is important to note that internal quality attributes can affect external ones [Mcco2004]. For example, if the code of a digital system cannot be easily understood by developers, this makes it harder for them to fix bugs and maintain the system (underlying quality). This can have negative effects on the reliability of the digital solution, which is perceivable by its users.

Differentiating between perceivable and underlying quality also allows for the creation of an adequate evaluation strategy to cover both perspectives [FrPr2009]. In order to evaluate perceivable quality attributes, users of the system need to be involved, while underlying qualities can be evaluated using automated tests to ensure that, for example, a piece of code (a unit) meets its design and behaves as intended. These unit tests can be run by developers and do not need any involvement from users in order to be executed.

2.1.3.4 A Quality Model for Digital Systems

There are several models for defining the quality of a digital system (e.g., McCall [McRW1977], Boehm [BBKL1978], ISO/IEC 9126 [ISO2001], ISO/IEC 25010 [ISO2011]). This is also true for approaches and models that describe non-functional requirements that often include quality requirements as a key category [Glin2007].

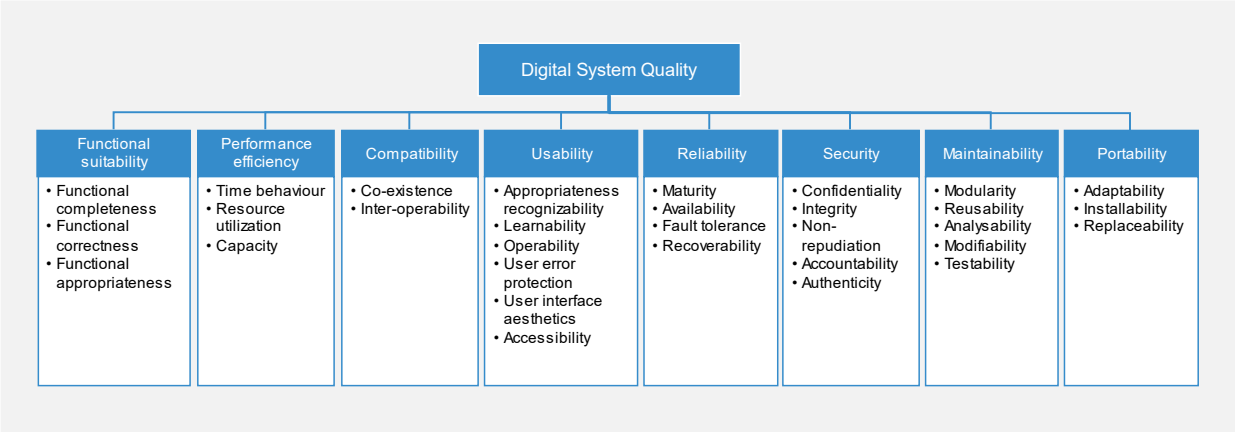


Figure 11 – Adapted ISO/IEC 25010 quality characteristics [ISO2011]

Definition of system quality characteristics according to [ISO2011]:

Functional suitability: This characteristic represents the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions.

Performance, efficiency: This characteristic represents the performance relative to the number of resources used under stated conditions.

Compatibility: Degree to which a product, system, or component can exchange information with other products, systems, or components and/or perform its required functions while sharing the same hardware or software environment.

Usability: Degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use.

Reliability: Degree to which a system, product, or component performs specified functions under specified conditions for a specified period of time.

Security: Degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.

Maintainability: This characteristic represents the degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it, or adapt it to changes in the environment and in requirements.

Portability: Degree of effectiveness and efficiency with which a system, product, or component can be transferred from one hardware, software, or other operational or usage environment to another.

When it comes to quality characteristics, ISO/IEC 25010 is a good starting point for understanding the quality of a digital system. It defines eight main quality characteristics that can be applied to

digital systems. Each of these characteristics includes several sub-characteristics (see Figure 11); in total, there are 31 sub-characteristics [ISO2011]. ISO/IEC 25010 mentions usability as a key quality of a digital system, which mainly refers to the degree to which a system is easy to use. However, it does not explicitly mention user experience as a quality attribute, which is considered in other norms such as ISO/IEC 9241-210 [ISO2019] or the CUE model [ThMa2007].

2.1.3.5 Describing a Quality Model for Digital Solutions

Regarding digital solutions, we advocate a broader and holistic view of quality where user experience (UX) aspects such as satisfaction, enjoyment, and pleasure are considered. This view can be influenced by many factors that are not necessarily part of the digital system, such as the reputation of the brand. The quality model by Hassenzahl et al. [HaTr2006] is a good starting point: it distinguishes between pragmatic and hedonic qualities. Pragmatic qualities focus on effective and efficient task support. Overall, they refer to the usability and utility of a digital solution. This also means that it is the digital system and its perceivable qualities that strongly influence the pragmatic qualities of a digital solution.

Although both qualities focus on users, hedonic qualities go beyond the qualities of digital systems (see Section 2.1.3.4) and provide a new perspective on qualities of a digital solution. The perceived hedonic quality focuses on the subjective perception and the sensations and emotions caused by the use of the digital solution. Understanding the user's positive emotions allows for further investigations. These can include, for example, understanding the extent to which the digital solution will motivate or excite the user. Will it even change the user's life? Such aspects are subjective and hard to tackle but can really affect the user and allow them to bond with the digital solution. Studies show that users can clearly differentiate between pragmatic and hedonic qualities and that users form an overall judgement of the attractiveness of a digital solution based on both qualities. This also means that basically, both quality aspects are equally important. However, depending on the particular digital solution and the needs and requirements of different stakeholders, one or the other aspect could play a more significant role. The questionnaire AttrakDiff [HaTr2006] was developed to evaluate the different product qualities and provide an overall judgement regarding the attractiveness of a digital solution (see Section 4.1).

The quality aspects considered for the digital solution can even go beyond UX and also consider customer experience (CX), which includes all interactions a customer has with a company. This can include, for example, calling the company's support hotline.

Concluding this discussion of quality, we can summarize that usability is an important aspect of user experience, which is an important aspect of customer experience. Note that customer experience and user experience are not identical, especially if the customer is not also the user (see Section 1.2.3).

2.1.4 Additional Resources for the Building Process

The competences provided in this handbook define the foundation level, which means that the skills provided are necessary basics but are not sufficient to cover the whole spectrum of skills necessary to design digital solutions.

A DDP therefore has to be aware of the fact that additional skills are necessary.

Exemplary skills include:

- Requirements engineering for understanding, validating, and managing requirements of complicated solutions
- Business analysis for understanding and evaluating business-driven solutions
- Industrial design for shaping physical devices as part of a digital solution
- Usability engineering for designing and evaluating the interactive part of digital solutions
- Software testing for the systematic quality assurance of software parts of a digital solution

This list is of course not complete and comprehensive. However, it provides an overview of several related fields and has two purposes:

1. To provide references to important additional fields that a DDP at foundation level should be aware of
2. To create humbleness in view of the many different disciplines and to make it clear that a DDP can draw on a wide range of skills to design a digital solution

A person can certainly have numerous competences in these fields. It should be clear, however, that no one will master all fields to the required depth. The direct conclusion that we can draw from this is that every building process requires teamwork and that the right competences must be involved at the right time (see Section 6.3). With regard to the design of a digital solution, it is the responsibility of a DDP to ensure that people with the necessary skills are available to create a successful digital solution. This responsibility begins at foundation level.

2.1.5 Conclusion: An Idealized Model of the Building Process

In this section, we have presented the three steps of the building process together with an introduction to design processes. A typical beginner's question at this point in time is as follows: Now I know the steps and something about a design process, but what does the building process really look like in reality? The answer to this question is usually disappointing for beginners: we do not know in detail. Most companies tailor this general process to their particular needs. The definition and tailoring of building processes is a management skill and goes beyond the scope of Digital Design.

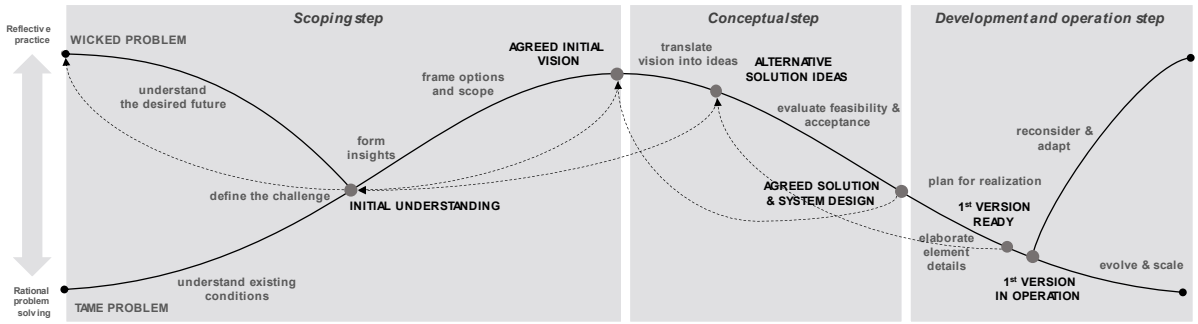


Figure 12 – An idealized model of the building process

However, we are aware that this is not a satisfying answer. We therefore give an overview of an idealized building process (see Figure 12) that summarizes all the content presented in this chapter. We describe the model briefly, starting on the left side with the scoping step.

Working in the Scoping Step

At the beginning of the building process, we are in a situation where we face an abstract and open space of possibilities. In order to simplify the situation, the distinction between tame and wicked problems [RiWe1973] is useful to define two different situations for starting a building process. A wicked problem can be defined as follows:

Wicked problem: A problem that is difficult or impossible to solve because of incomplete, contradictory, and changing requirements.

The tame problem is the opposite of a wicked problem:

Tame problem: A problem that is well defined with clear and stable requirements.

YPRC example. The YPRC case study is an example of a wicked problem: "Develop a new coaching experience for long-distance runners." The wicked aspect of this exemplary problem is especially created by the term "new," since nobody really knows what new means. An example of a tame problem is "measure the health data of a runner during a training session." Health data can be defined in an analytic fashion. Furthermore, measurement technology is also available.

When facing a wicked problem, the reflective practice mode (see Section 2.1.2) is the correct working mode. We have to understand the desired future together with all relevant stakeholders in order to reach an initial understanding of what the digital solution is about. Tools for this understanding are early concepts (e.g., sketches) and prototypes (e.g., paper prototypes or story boards). The scoping dimensions introduced in Section 2.1.2.1 are a proper tool for analyzing the different aspects of the wicked problem. When facing a tame problem, the rational problem-solving mode (see Section 2.1.1.2) is the correct working mode. We have to understand the existing conditions in which the problem exists for two reasons: first, to understand what has to be achieved, and second, to make sure that the problem is not a hidden wicked problem.

No matter what type of problem we face, we make our understanding of the overall problem more specific and try to reach an initial mutual understanding of what we want to achieve. If it is not possible to reach this mutual understanding, the scoping step must start all over again under the assumption that we are facing a wicked problem.

With a mutual understanding, we again work in the reflective practice mode and start to make our understanding of the scope even more specific. This is done by discussing the problem from various perspectives (e.g., by looking at competitors or analyzing digital technologies that might help solve the problem). At the end of this process, a concrete and agreed vision for the digital solution is defined in the form of a design brief. If it is not possible to reach this agreement among all relevant stakeholders, the building process must start another iteration to define a new vision. In Figure 12, the iterations are represented by dashed lines. The visualization of these iterations describes the latest point in time where the need for an iteration can be recognized.

Working in the conceptual step

With an agreed initial vision (and the other details from the Digital Design brief), the conceptual step can start. When starting the conceptual step, we are again in an abstract and open situation since there are several alternative ways of achieving the vision. The reflective practice mode is the correct working mode in this situation since we have to translate the vision into various solution

ideas (see Section 2.1.1.2). For this purpose, we can develop initial concepts and create fundamentally different prototypes to explore these various solution directions. Details on this are included in Section 2.2 and Section 2.3.

The understanding of the different solution ideas becomes more and more concrete during this work. An important result during this work is alternative solution ideas. Developing and evaluating alternative solution ideas is a key approach when performing design work. Alternative ideas allow us to explore the possible solution space in a systematic way and increase the possibility of finding a good solution idea. If the solution ideas developed are not promising enough, the initial vision should be questioned, and the building process should go back to the scoping step in order to develop a new vision for the digital solution.

During the concretization, the working mode shifts from the reflective practice to the rational problem-solving mode. The various solution ideas should be translated into system ideas. The feasibility and acceptance of these ideas must then be evaluated with relevant stakeholders. At the end of this process, agreed initial solution and system designs for the digital solution are created that are sufficiently detailed to accept the risk of starting the development. If it is not possible to reach such a design, the development of alternative ideas must start again.

Working in the development and operations step

Although agreed solution and system designs are concrete from the perspective of the conceptual step, we are again in an abstract and open situation because the various realization details of the system must be defined together with an initial plan for realizing the digital solution. In this situation, the rational problem-solving mode is again the proper way of working since many detailed design decisions have to be made in order to elaborate and evaluate the various details of the digital solution. For digital solutions, this part of the process is a real challenge and requires experts from various domains.

From an idealized perspective, the end of this work is a first version of the digital solution that is ready for operation. Now, a critical decision is necessary: is this solution ready for operation or not? If the answer is no, the process goes back to the conceptual step since the overall solution idea is not appropriate and new solution ideas must be developed. This step back seems to be rather radical but is the only proper answer since the modification of details of the solution idea are of course part of the building process. If it is not possible to bring the selected solution idea into a shape that is ready for operation, the only proper answer is to create new solution ideas.

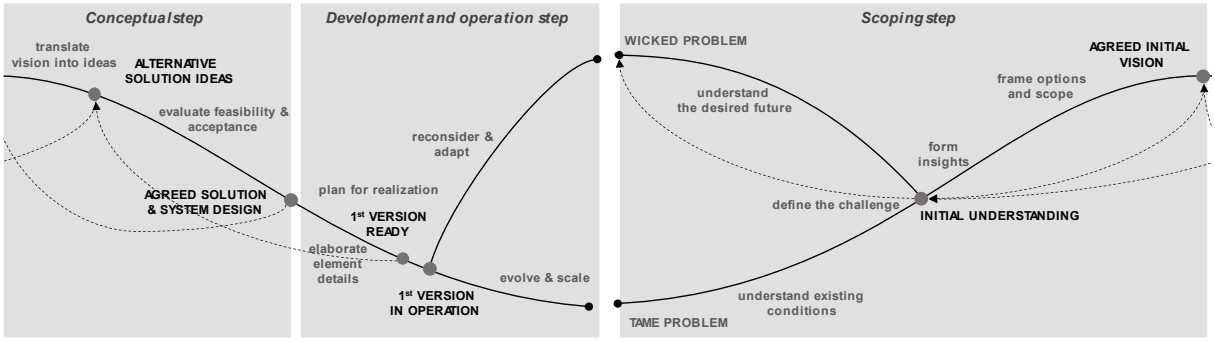


Figure 13 – The end of the building process is a new beginning

As soon as the solution is in operation, we are in a position to obtain feedback from real customers in real life. Our understanding of the achievement of the solution now becomes more and more concrete. In general, two different outcomes are now possible: one outcome is that the solution

is accepted by the customers and generates the desired value and can therefore start to evolve and hopefully scale to become a successful solution. The alternative outcome is that the solution is not really accepted and does not meet the business objectives although all previous evaluation measures indicated that customer acceptance was very likely. We consider such a situation as a kind of wicked problem since we have to reconsider our whole understanding of the solution. Regardless of the outcome, another scoping step will take place and the idealized model of the building process can be applied again (see Figure 13).

2.2 Conceptual Work in Digital Design

Conceptual work is a core aspect of Digital Design work. To cover the wide scope of the building process for a digital solution, a broad range of different concept types is necessary in Digital Design. The different types of concepts were introduced in Section 2.1 as part of the building process.

When talking about conceptual work, it is important to recognize the difference between the content perspective and the creation process perspective:

- The *content perspective* deals with the content of concepts and the relationships between the different parts of one concept and the relationships between the content of different concepts (detailed in this chapter).
- The *creation process perspective* deals with the creation of one or more concepts as part of the building process (detailed in Chapter 5).

Mixing both perspectives leads to an incorrect understanding of concepts, namely the misunderstanding that a number of different concepts imply a *waterfall approach* in which one concept (e.g., the solution design concept) must be completed before the work on a second concept (e.g., the system design concept) can start. This sequential approach is almost never applied in practice.

In this chapter, we focus on the content perspective, present templates for each concept type, and illustrate their relationships from a content perspective. Understanding the content perspective is a prerequisite for efficient and effective conceptual work. The creation process perspective is a second aspect and is even more challenging than the content perspective. In Chapter 5, we illustrate the creation process perspective with an exemplary building process for beginners.

The world of concepts is typically overwhelming for beginners in Digital Design because of the different types of concepts, the different perspectives, and the different abstraction levels (solution, system, and element) addressed by the concept.

In the following, we start with the fundamentals of conceptual work (see Section 2.1.1) in Digital Design and then work our way through the abstraction levels down to the details of the element level. Once we get there, we work our way back up and connect the details into one big picture, presented in the last section of this chapter. This leads to the structure outlined below.

Section 2.2.2 introduces document templates for the different abstraction levels and presents the overall structure of each document template by describing the content of each chapter in the document template. To improve the readability of this overview, we have separated the description of the document template from the description of detailed documentation techniques that can be applied to create the content of a particular chapter of the document template.

The documentation techniques are presented separately for the solution level in Section 2.2.3 (Digital Design brief and solution) and the system/element level in Section 2.2.5 (system/software/device design concept).

The main reason for this separation originates from substantial differences in the structure of design concepts at the solution level and system/element level. We elaborate on these differences in Section 2.2.4. Section 2.2.6 concludes this section by putting all the pieces and details together into one big picture.

2.2.1 Fundamentals of Conceptual Work

2.2.1.1 Concepts Are Ideas in Thought or Communication

In general, concepts are ideas that occur in thoughts or in communication (written or verbal) and are considered as elements of thoughts (cf. [MaLa2015]). In Digital Design, conceptual work means working mentally to create a digital solution, i.e., defining the objectives of the digital solution and the corresponding form, function, and quality of the digital solution. Concepts can occur in a rather linear verbal/written form or in a highly structured technical form.

YPRC example. The following brief description of the YPRC case study can be considered as a verbal conceptual description: YPRC provides a holistic training service for newcomers to running and consists of a dedicated smartwatch, a smartphone app, and a portal. The smartwatch measures the runner's health data that is visualized by the app. The app communicates with a portal in the background. Via the portal, the runner can purchase artificial intelligence (AI) coaching and remote personal coaching.

This short verbal description is an efficient tool for communicating the main ideas of YPRC to various stakeholders. However, such verbal descriptions are of only limited use for defining and communicating all details that are relevant for actually building YPRC—this requires more sophisticated concepts that present the various aspects of a digital solution in a structured way.

2.2.1.2 Benefits and Limits of Concepts

A design concept is created for several purposes:

- *Thinking tool for taming supposedly complex solutions/systems:* A design concept structures thoughts about the digital solution. In the following, we detail this aspect by describing dedicated documentation techniques that, on the one hand, serve as document templates, and at the same time, provide support for structuring thoughts about a digital solution, the corresponding system, and its elements. This structured approach is important for understanding the system and solution as a whole and for identifying those parts of the solution/system that are really complex or *only* complicated (see Section 1.2.1). In Section 2.2.5, we will come back to this point by describing the particular building blocks that make up the whole digital solution.
- *Basis for construction and realization*: A design concept serves as input for the activity areas construction and realization (see Section 1.3). In contrast to requirements-driven work (see Section 1.1, e.g., from requirements or usability engineering), the conceptual work in Digital Design adopts the solution-driven perspective from design (cf. [Cros2006]) and focuses on concrete solution ideas instead of an intensive definition and analysis of requirements that lead to a solution. This does not mean that requirements are neglected in Digital Design. Where necessary, requirements are documented and used, especially

quality requirements, constraints, and goals. Furthermore, the stakeholders' requirements, in particular those from the client and the customer, are crucial input when creating design concepts.

- *Communication tool*: A design concept communicates the digital solution to various stakeholders. To support this communication purpose, different types of design concepts must be created. In Section 2.2.2, we discuss this issue again and explain the various target groups for the different concepts.
- *External memory for complicated solutions/systems*: A design concept serves as an external memory during the whole lifecycle of a digital solution. This is because the amount of information about a typical digital solution that is created during the building process far exceeds the capacity of human memory. To serve as external memory, all design concepts must be revised and optimized continuously. This allows the typically complicated structures of solutions/systems to be handled with an acceptable level of effort.
- *Evaluation tool*: A design concept can be used to evaluate the digital solution described—for example, in terms of business cases, customer or user acceptance, or legal issues.
- *Reference point for evaluation*: A design concept serves as a basis for evaluating that the digital solution realized corresponds to the needs of the stakeholders and for ensuring that the digital solution is built according to the concept defined. Additional evaluation concepts are therefore created at each abstraction level (see Section 2.1).

Creating and working with design concepts is inexpensive but has certain limits:

- *Concepts are never complete*: Even the best and most detailed design concept will be incomplete. The reason for this is simple: life is too complicated to be anticipated and captured completely in a concept. This fact is not a weakness, however, since concepts are a communication tool. Many weaknesses and missing details of design concepts are identified in particular when the development of a digital solution starts. A DDP at foundation level should accept the fact that concepts are incomplete and will be extended and revised during the whole building process. This work is by no means a waste of time. The constant work on design concepts is an important backbone for keeping the building process under control from a design perspective since the design concepts serve as external memory (see above).
- *Concepts always leave room for interpretation*: Just as concepts are never complete, concepts can never be free of interpretation. Interpretation is a core feature of human communication and since concepts are communication tools, they have to be interpreted. A DDP at foundation level should always keep this in mind and should always look out for potential misunderstandings and misinterpretations of concepts.
- *Concepts are not the digital solution:* Working with concepts is one of the main tasks in Digital Design. However, concepts are a means to an end. Concepts are tools that serve the building process (see Section 2.1.2). The goal of the building process is to bring a digital solution to life. A DDP at foundation level should be aware of this and should take conceptual work seriously but always keep in mind that a good digital solution is more important than good concepts. In Chapter 6, we discuss this topic again when we present the ten principles of good Digital Design.

- *Sophisticated concepts can create false confidence*: Good design concepts with high-fidelity prototypes of the digital solutions can be very impressive, especially to inexperienced clients. The downside of this impressiveness can be that clients and stakeholders can get a false confidence in the success and the current state of a digital solution and become too optimistic about the potential success of the digital solution. This point is not an argument for creating sloppy design concepts—it is meant as advice to handle concepts carefully.

The benefits and limits of concepts do not determine the intensity and the level of detail that are appropriate for a particular situation. Using the right intensity and level of detail of conceptual work requires experience.

2.2.2 Pragmatic Document Templates for the Different Abstraction Levels

In this section, we introduce a number of templates for the different types of design concepts introduced in Section 2.1. The definition and selection of these templates is based on the authors' practical experience.

In literature, there are many other approaches to conceptual work, and experienced experts will certainly know and be able to apply other techniques and define other template structures. When the templates described here were defined, the focus was on easy learnability and quick applicability. In this sense, the following templates should be understood as pragmatic. They especially serve as a good starting point for beginners to conceptual work.

2.2.2.1 Digital Design Brief

Table 3 shows a pragmatic template for a Digital Design brief. The Digital Design brief is used to clarify the overall idea and the scope of the building process for the digital solution in the scoping step with the client (see Section 2.1).

This means that the Digital Design brief not only provides first information on the intended digital solution, it also provides information on the process that is intended to build the digital solution. In order to support both purposes of the Digital Design brief, the template addresses the intended digital solution in the first two chapters (context and vision). Chapter 3 defines the potential scope for building the whole digital solution and Chapter 4 defines the general terms for the building process. The table lists potential techniques that support the structured documentation of the content of a particular chapter.

The *Doc. technique* column of the table lists some advice for creating each section of the Digital Design brief. A detailed example of documentation techniques can be found in the YPRC case study. Details on the stakeholder list and the future press release are given in Section 2.2.3.

Table 3 – Document template for a Digital Design brief

Chapter	Content	Doc. technique
1 Context		
1.1 Case for action	What is the reason/rationale/motivation of the client for building a new digital solution?	Textual description
1.2 Potential customers	List of persons (including a short description of each type of person) potentially becoming a customer of the digital solution	Bullet list of customer types with textual explanation
1.3 Potential users	List of persons (including a short description of each type of person) potentially becoming a user of the digital solution	Bullet list of user types with textual explanation
1.4 Potential further stakeholders	A list of further persons who are considered relevant for the building process or the digital solution	Stakeholder list
1.5 Related solutions	Descriptions of solutions that are to be understood as analogous to the planned solution	Textual description for each solution
1.6 Potential competitors	Descriptions of possible existing solutions that can be considered as competitors	Textual description for each competitor
2 Vision	A vision statement of the desired future that shall be created by the digital solution. Alternative ideas for the digital solution can be described, if necessary.	Future press release
3 Scope		
3.1 Solution space	Characterization of the solution space that shall be explored to realize the vision	Textual description
3.2 Potential technologies	Descriptions of technologies that are considered suitable for realizing the vision	Textual description for each technology
3.3 Constraints	Description of constraints that the digital solution must fulfill	Textual description for each constraint
3.4 No-gos	Description of technologies, features, and other aspects that are considered unacceptable	Textual description for each no-go
4 General terms		
4.1 Schedule	A rough timetable for building the digital solution, divided into timetables for conceptual work, potential prototypes, and a timetable for developing a first version of the digital solution	Timetable with timeframes
4.2 Mode of cooperation	A description of the mode of cooperation between the building team and the relevant stakeholders. This includes, for example, regular coordination meetings, but also rights and obligations of the building team and the relevant stakeholders.	Textual description
4.3 Budget	Budget for building the solution, divided into the steps conceptual work, development, and operation budget	Table with different budget items
4.4 Revenue streams	Brief description of potential revenue streams for the digital solution, if applicable	Textual description
4.5 Resources	Resources (personnel, workshops, factories, etc.) required or available for the conceptual and development steps	Table with different resources

2.2.2.2 Solution Design Concept

The solution design concept describes the digital solution from the client's perspective (see Chapter 1 for the difference between solution level and system level). Table 4 shows a pragmatic template for a solution design concept. In Chapter 1, it introduces the digital solution to the case for action and the vision of the digital solution. Chapter 2 focuses on the context of the digital solution and describes the intended customer segments, user groups, and further relevant stakeholders of the digital solution.

In contrast to the Digital Design brief, with regard to customers, users, and stakeholders, the solution design concept clearly focuses on the solution. In the design brief, potential customer types, user types, and stakeholders are described to identify possible solution alternatives. In the solution design concept, a particular solution is defined, including the relevant customer segments, user groups, and stakeholders.

YPRC example. In the YPRC case study, family doctors might be considered as potential users. They could gain access to the runner's health data as part of a medical checkup. This service would lead to potential stakeholders (e.g., people from health data legislation). If this function is not considered during the conceptual step, this user group, including the related stakeholders, do not become part of the solution design concept.

When defining customer segments and user groups in the solution design concept, it is important to keep in mind the difference between the customer and the user in terms of stakeholder roles (see Section 1.2.3). Chapters 3 and 4 focus on the concrete value and customer experience that a digital solution shall deliver and on the business model that will drive the digital solution.

Table 4 – Document template for a solution design concept

Chapter	Content	Doc. technique
1 Motivation	Motivation for the digital solution, including the case for action, an introduction to the digital solution, and the vision statement for the digital solution	Textual description and future press release
2 Context		
2.1 Customer segments	Description of the people who are customers of the digital solution	Persona templates
2.2 User groups	Description of the people who are users of the digital solution	Persona templates
2.3 Further stakeholders	List of further stakeholders who are relevant for the digital solution	Stakeholder list
3 Value proposition	A description of the value proposition of the digital solution	Value proposition canvas
4 Customer experience	A description of the customer experience that the digital solution shall offer to customers	Customer journey map
5 Business model	The business model that will sustain the digital solution	Business model canvas
6 Constraints	Description of constraints that are relevant for the design of the solution	List of constraints

We recommend using canvas techniques such as the value proposition canvas, customer journey maps, and the business model canvas to document the value proposition, the customer experience, and the business model. In the case of a complex business model for digital solutions, we recommend consulting literature from business analysis (cf., e.g., [PaYC2010]) for more details on the documentation of business models.

There are types of digital solutions that do not have a business model of their own. Typical examples are internal systems within a company, such as enterprise resource planning systems or customer relationship management systems. These digital solutions are part of a larger value chain and therefore only part of an overall business model. In such a situation, we recommend documenting the business model that the digital solution is part of. We further recommend trying to capture the value the particular digital solution contributes to the overall business model.

2.2.2.3 System Design Concept

The system design concept describes the digital system that realizes the digital solution (see Section 1.2.1 for more details on the difference between solution and system). Table 5 shows a pragmatic document template for a system design concept. The template reflects the structure of form, function, and quality introduced in Section 1.2.1.

Chapter 1 of the template provides an introduction for readers who are unfamiliar with the digital solution. Chapters 2 and 3 of the template describe the goals and the constraints that are relevant for the design of the system. The explicit documentation of goals and constraints is important to provide rationales for the design of the overall system.

The main focus of the system design concept is on the form of the digital system (Chapter 4). The template lists the user types (as part of the system context), existing objects/systems, and the elements to be realized. With this approach, the system design concept gives an overview of the different elements of the digital system including their relationships.

Table 5 – Document template for a system design concept

Chapter	Content	Doc. technique
1 Introduction	Description of the general system idea as a starting point for readers who are unfamiliar with the digital solution	Textual description
2 Objectives	Description of goals that are relevant for the design of the system	Goal templates
3 Constraints	Description of constraints that are relevant for the design of the system	Constraint templates
4 Form	Overview of the digital system and its context as an introduction	Overview picture
4.1 User types	User types that will use the digital system as part of the context	User type templates
4.2 Existing objects	Objects in the context whose existence is assumed	Object templates
4.3 Existing systems	Systems that serve as elements as part of the context and whose existence is assumed	System templates
4.4 Elements to realize	Elements that have to be realized	Software element/device template
5 Function	Exemplary description of the functions that the system provides	Scenario templates

6 Quality requirements	Description of the qualities of the system	Quality requirement templates

Compared to the chapter on form, the chapter on function of the digital system will be rather short and described in an exemplary way. The main reason for this is to keep the abstraction level of the system design concept high enough to serve as a concept that focuses on the form of the whole system. Therefore, the chapter on form is separated into the different types of elements.

YPRC example. Examples from the YPRC case study are given in brackets:

- User types (runner and runner's coach)
- Existing objects (the runner's smartphone and the PC of the runner's coach)
- Existing systems (the map data provider and the payment provider)
- Elements to realize (the runner's watch, the runner's app, and the portal)

The main motivation for choosing this level of detail for the system design concept comes from the idea of the system level as a management tool for the overall building process. It is only with a detailed understanding of the form of the digital system that you can structure the subsequent work appropriately (see Section 1.3.4). The details of the various functions of the system are defined at the element level, where all necessary information for an adequate description is available.

An often-underestimated factor at the system level is the names of the different elements. We recommend that you choose the names of the different elements with care, as these names are used for communication about the different elements during the construction process. The names should be expressive and reflect the function of the element. Ambiguous names or names that are very similar can easily lead to misunderstandings. For example, in the YPRC case study, the runner uses the *runner's app*. The element used by the coach is called *portal* and not *coach's app*. The name *coach's app* is close to *runner's app* and people might talk only about *the app*, which would definitely lead to misunderstandings.

2.2.2.4 Software Design Concept

With the software design concept, we go into details of a particular software element of the digital system. Table 6 shows a pragmatic template for a software design concept. The name of the software element described from the system level must be used as the title of the concept. For example, the software design concept in the YPRC case study uses *Runner's App* as the title.

Table 6 – Document template for a software design concept

Chapter	Content	Doc. technique
1 Introduction	Description of the element idea as a starting point for readers	Textual description
2 Objectives	Description of goals that are relevant for the design of the software element	Goal template
3 Constraints	Description of constraints that are relevant for the design of the software element	Constraint template
4 Form	Graphical overview of the relevant context of the software element	Overview picture

Chapter	Content	Doc. technique
4.1 Hardware interfaces	Perceivable and underlying hardware interfaces that the software element expects in the context	Perceivable and underlying hardware interface template
4.2 User interface	Description of interfaces of the software element for the user	User interface template
4.3 Software interface	Description of interfaces to related software systems (i.e., application programming interfaces, API)	Software interface template
4.4 Entities	Description of data that the software element stores	Entity template
5 Function		
5.1 Use cases	Description of use cases that the software element supports	Use case template
5.2 Functions	Description of underlying functions that the software element provides	Function template
6 Quality requirements	Description of the quality requirements that the software element must fulfill	Quality requirements template

Like the system design concept, the introduction (Chapter 1) of the software design concept serves as an overview of the digital solution and is intended to make a particular concept readable without additional information.

Chapters 2 and 3 of the template describe the goals and the constraints that are relevant for the design of the software element. Like the system design concept, the explicit documentation of goals and constraints is important to provide rationales for the design of the software element. The goals and constraints documented at system level and element level are typically related to each other or refine each other.

Chapter 4 deals with the form of the software element, including its relevant context. It provides an overview of the overall digital system from the perspective of the software element, including a description of the relevant environment from the system design concept (users, existing objects, existing systems). In addition, the chapter describes the perceivable and underlying hardware interfaces that the software element expects in the context.

The description of these hardware interfaces is important for capturing the technical environment in which a software element will operate. Typical hardware interfaces include displays, audio input and output, and communication hardware (e.g., an internet connection). The form of the element itself is described by the user interfaces for interaction with the user, software interfaces for interaction with other software systems, and entities for describing the information stored by the software element.

Chapter 5 deals with the function of the software element. Use cases describe the perceivable function by means of interaction steps between the user and the software element. The underlying function is described by means of dedicated function templates. Each underlying function represents a relevant transformation of data.

In contrast to the scenarios at system level, the use cases and functions at element level must provide a complete description of the function of the software element at a certain abstraction level. For consistency reasons, the use cases and functions at the element level must not

contradict the scenario description at the system level. We provide further details on this in Section 2.2.5.

Chapter 6 concludes the software design concept with quality requirements that the software element must fulfill.

2.2.2.5 Device Design Concept

Besides software elements, a digital system can consist of devices that are designed and manufactured especially for the digital system. A device consists of hardware and software. Table 7 shows a pragmatic template for a device design concept. The name of the device described from the system level must be used as the title of the concept. For example, the device design concept in the YPRC case uses *Runner's Watch* as the title. The structure is similar to the template of the software design concept (see Table 6). We therefore discuss only the differences between both templates.

The main difference between a software element and a device is that the device combines hardware (e.g., CPU, memory, and storage) and software. This means that the device must provide all necessary hardware and software that is necessary to fulfill its purpose.

This difference is reflected in Chapter 3 and Chapter 4 in the template. The description of the context of a device (Chapter 4) provides only an overview picture of the context and all hardware interfaces are part of the form description of the device since they have to be realized to create the device. In addition to the hardware interface, Chapter 4 has a section on physical parts of the device.

Table 7 – Document template for a device design concept

Chapter	Content	Doc. technique
1 Introduction	Description of the general device idea as a starting point for readers that are unfamiliar with the digital solution	Textual description
2 Objectives	Description of goals that are relevant for the design of the device	Goal template
3 Constraints	Description of constraints that are relevant for the design of the device	Constraint template
4 Form	Graphical overview of the relevant context of the device	Overview picture
4.1 Physical parts	Description of the physical parts that make up the device	Part template
4.2 Hardware interfaces	Description of the perceivable and underlying hardware interfaces of the device	Perceivable/underlying hardware interface template
4.3 Software interface	Description of (underlying) interfaces to related software systems	Software interface template
4.4 User interface	Description of the (perceivable) interfaces to users of the software element	User interface template
4.5 Entities	Description of data that the device stores	Entity template
5 Function		

Chapter	Content	Doc. technique
5.1 Use cases	Description of use cases that the device supports	Use case template
5.2 Functions	Description of underlying functions that the device provides	Function template
6 Quality requirements	Description of the quality requirements that the device must fulfill	Quality requirements template

It is important to recognize that the device design template on its own is not sufficient to build a complete device. It must be complemented by dedicated concepts that deal with the design, construction, and realization of the hardware parts of the device.

Additional experts (e.g., from industrial design, product engineering, etc.) and concepts (e.g., technical drawings, circuit diagrams, product plans) are necessary when designing and building a device. The device design concept presented is necessary for the Digital Design perspective and for designing the digital part of the device.

2.2.3 Documentation Techniques for the Solution Level

In the following, we introduce a number of techniques that are useful for working with design concepts at the solution level (see Section 1.3.4). Table 8 gives an overview of the techniques and their applicability in the different design concepts.

Table 8 – Overview of documentation techniques for the solution level

Documentation technique	*Digital Design brief (references to Table 3)*	*Solution design concept (references to Table 4)*
Future press release	Documentation of the vision (Chapter 2)	Documentation of the vision (Chapter 1)
Persona	-	Documentation of customer types (Chapter 2.1) and user groups (Chapter 2.2)
Stakeholder list	Documentation of potential stakeholders (Chapter 1.3)	Documentation of relevant stakeholders (Chapter 2.3)
Value proposition canvas	-	Documentation of the value proposition (Chapter 3)
Customer journey map		Documentation of the customer experience (Chapter 4)
Business model canvas	-	Documentation of the business model (Chapter 5)

2.2.3.1 Future Press Release

The future press release [Ross2019] is a technique for describing the vision for a digital solution. It takes the reader into the future and describes the central success that the planned digital solution will have achieved. To create a future press release, [Ross2019] defines the following rules:

Rule 1: The future press release must be stated at a future point in time at which success has been achieved and realized. It thus describes the central purpose of the digital solution.

Rule 2: Start with the customer. The press release must make clear why the digital solution is important to customers and how their experience has been improved. This rule focuses on the context of the digital solution. A short story from a customer, including a quote, is a good tool for a brief description of this aspect.

Rule 3: Set an audacious and clear goal. The achieved goals described in the press release should be measurable and can also include financial or market share results. A quote from a manager is a good tool for a brief description of this aspect.

Figure 14 – A future press release

Rule 4: Outline the principles that led to success. This rule focuses on the essential details of the solution and can also include important constraints for the solution. A brief feature list is a good tool for a brief description of this aspect.

All in all, the future press release should focus on the future functional gain drivers or pain relievers but also the future emotional experience from the client and user perspective. The future press release should fit on a single page. A photo conveying important aspects of the solution should also be included. Figure 14 shows a screenshot of a future press release for the YPRC case study. The full text of this future press release can be found in the YPRC case study.

2.2.3.2 Persona for Characterizing Customer and User Groups

Personas are a common user/customer research-based way of describing customer and user groups [Coop2004]. They are often used to synthesize ethnographic research and to build empathy:

Persona: A fictitious character representing a group of people with similar needs, values, and habits who are expected to use a system or benefit from it in a similar way.

Personas are a psychological design aid that allows designers to understand the habits and values of the prospective users and also develop empathy for them. They are an important tool in the design process for developing the goals of the customer or user and a crucial instrument for idea generation and validation of design concepts. Good personas help to reveal the experience, the behavior, and the motivation of the customer or user. At the solution level in particular, they help to identify which customers or users are the most important ones to address in the design process.

In a first step, personas can be set up based on hypotheses. However, good personas should then verify these hypotheses and always base them on valid user research.

Personas can help in the design process regarding the following points [CRCN2014]:

1. Building a common understanding of a customer or user
2. Determining the functions that a successful digital solution must fulfil
3. Being an anchor point for design decisions and joint discussion with stakeholders
4. Support in the evaluation of design concepts

Personas are therefore also an important tool for avoiding self-referential design, where designers project their own needs, goals, skills, and mental models into the solution.

In contrast to user profiles, which are often stereotypical profiles, personas should be based on first-hand ethnographic data. In contrast to market segments, which focus strongly on socio-demographic aspects, personas aim at usage behavior and goals.

Essential criteria for good personas are:

- Does the persona come across as a real person and not a caricature or a stereotype?
- Is the narrative power of the persona compelling?
- Does the persona highlight core attributes and high-level goals of the user?
- Is the persona focused enough to enable concept decisions?
- Is the persona usable?
- Is the persona appealing in its formal quality?

In order for personas to be effective, some stumbling blocks need to be avoided [Fla2018]. One such stumbling block is that, all stakeholders should understand what personas are and what they are used for. Then personas should not be created in silos and simply imposed on a project. It is also important that leadership fully supports the personas in terms of content. To make sure that personas are used in the end, it is also important to place them prominently (e.g., displaying them up in the office).

A persona should consist of a picture, basic background information, and information about goals and behavior. In addition, the pains with the current situation or solution can also be specified.

Figure 15 shows a persona template with the following elements:

- *Demographics*: name, photo, and further details of the persona
- *Behavior*: information on the behavior of the persona in relation to technology, communities, hobbies, and the solution
- *Goals*: information on the goals of the persona in relation to life, work, and the solution

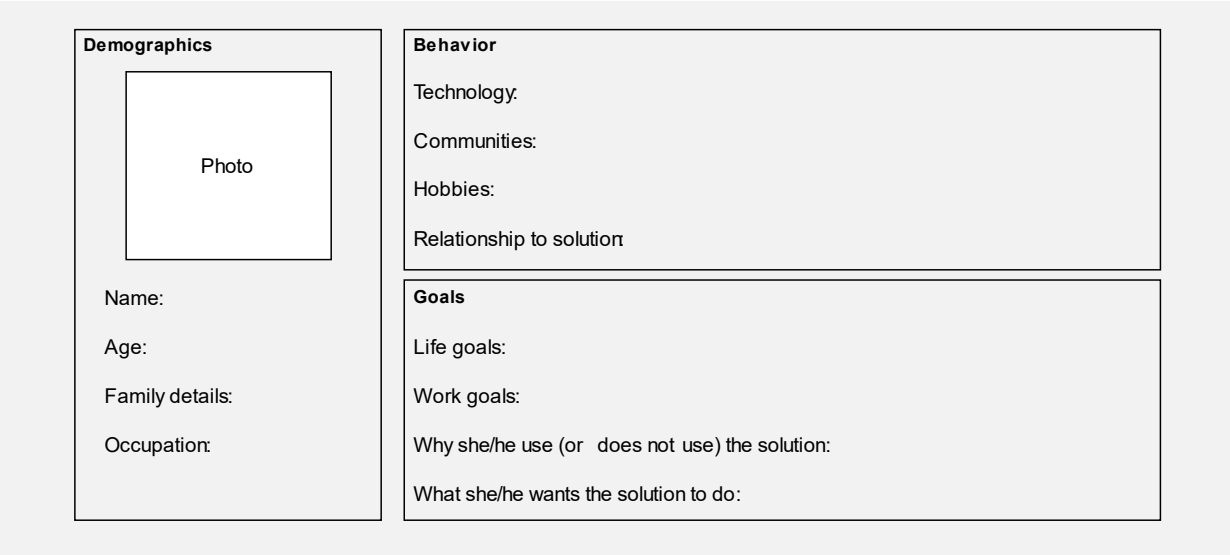


Figure 15 – A persona template

2.2.3.3 Stakeholder List

A stakeholder list is a structured table for documenting stakeholders. It consists of the following columns [CPRE2020]:

- Name of the stakeholder
- Function (role) of the stakeholder
- Personal and contact data, including availability
- Area of responsibility and extent of expertise

In addition, documenting non-public information about stakeholders can be useful: temperament, level of understanding, and commitment (see Section 4.3), as well as goals and interests in relation to the digital solution.

2.2.3.4 Value Proposition Canvas

The value proposition canvas is a template for describing customer profiles in relation to the values offered [OPBS2014]. The persona templates (see Section 2.2.3.2) are used to describe each customer type in general, whereas the value proposition canvas focuses on the particular value of the digital solution for the customer. It allows a more detailed description of the value propositions and the target customer segments. It also allows an evaluation of the fit between the intended value and the expectations that the customers are supposed to have.

A value proposition canvas (see Figure 16) consists of a customer/user profile and a value map. The customer/user profile describes:

- *Customer jobs*: what customers are trying to get done at work or in their lives
- *Gains*: the outcomes and/or benefits customers want to achieve in relation to their jobs
- *Pains*: bad outcomes, risks, and obstacles in relation to customers' jobs

When using persona templates to describe customer groups (see Section 2.2.3.2), this part of the value proposition canvas should be consistent with the persona descriptions.

The value map describes:

- Products and services offered to customers by the digital solution
- Gain creators, which describe how products and services create gains
- Pain relievers, which describe how products and services relieve pains

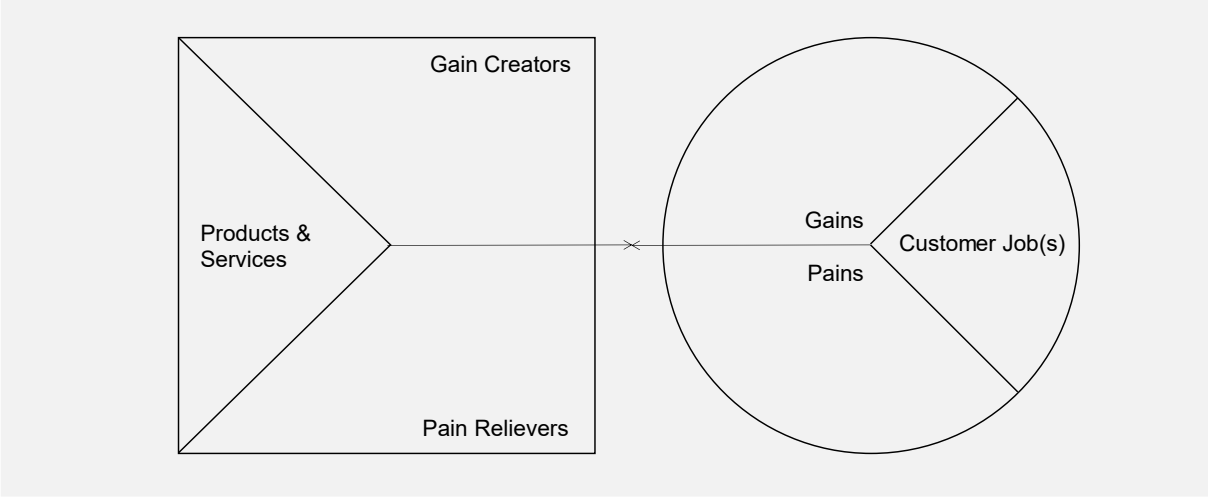


Figure 16 – A value proposition canvas template

The template shown above is used as a physical canvas, i.e., the canvas is printed on a large poster (or drawn on a whiteboard) and is filled with sticky notes to document the content in the different parts of the canvas. For documentation purposes, the canvas can be redrawn with a drawing tool, documented with a photo, or documented with a bulleted list for each part of the canvas.

In a solution design concept, a value proposition canvas is created for each type of customer (if the digital solution has more than one type of customer). An example and instructions for working with the value proposition canvas are given in Section 5.2.

The value proposition canvas can be used before, during, and after developing an in-depth understanding of the customers. If it is used before, it highlights what a building team needs to learn about customers and what to evaluate in terms of value propositions. If the team uses it afterwards, it helps the team analyze and evaluate the fit between the intended value and the customer expectations.

The value proposition canvas can be applied to new and existing value propositions and customer segments alike. In both cases, it helps to structure the understanding and the thinking about the digital solution and helps to make ideas more tangible.

2.2.3.5 Customer Journey Map

A customer journey map is a tool from service design [PoLR2013] that can be used to work on the form, function, and quality of digital solutions and thereby on the customer experience.

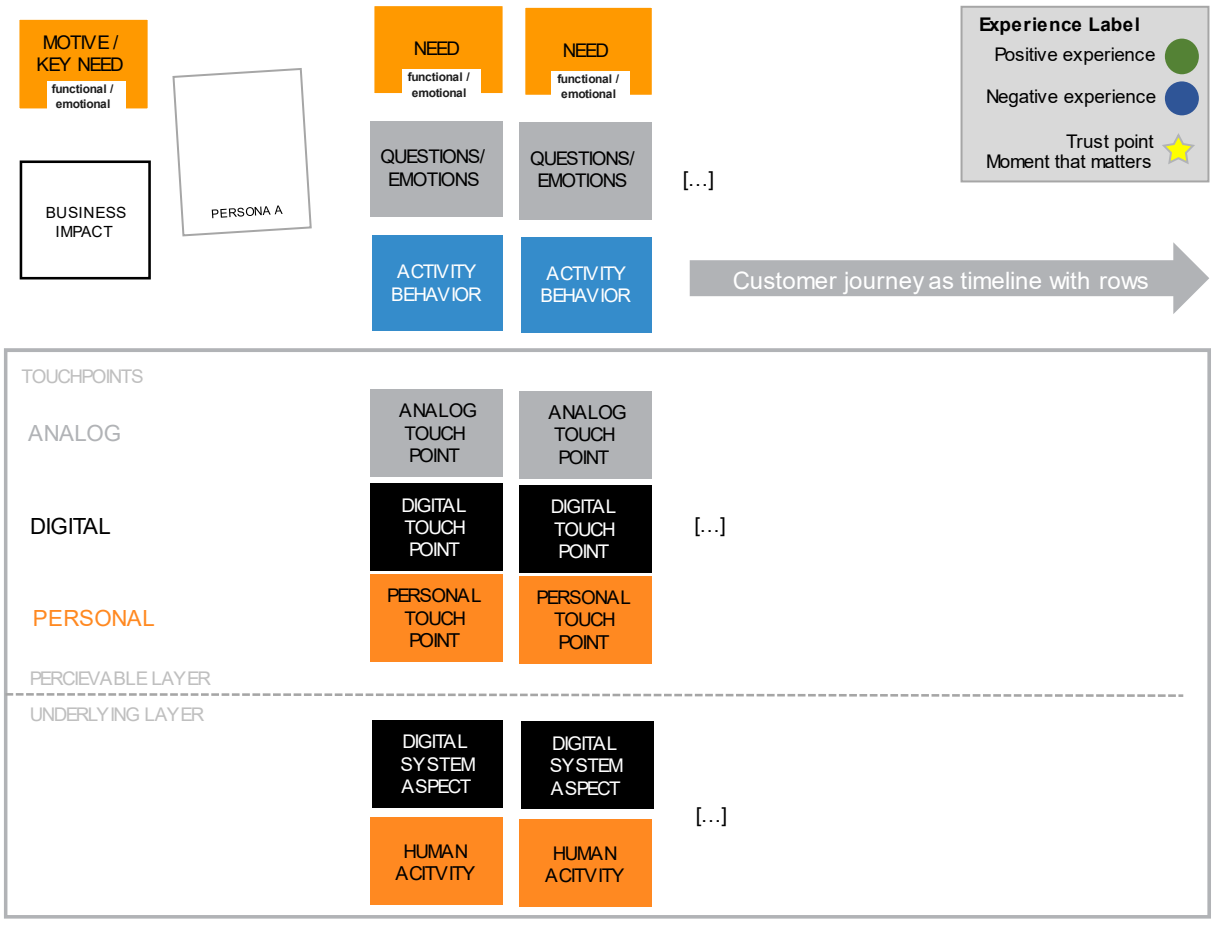


Figure 17 – A customer journey map template

Figure 17 shows a template for a customer journey map canvas with the following elements:

- The motive of the customer is expressed as a key customer need. We recommend distinguishing between functional needs (e.g., *improve personal running performance*) and emotional needs (e.g., *avoid pressure from groups*).
- A picture and the name as a reference to the customer persona whose journey is shown. We recommend creating customer journey maps for all customer personas defined in the solution design concept.
- A short statement on the positive business impact that the journey presented will create (e.g., *increases sales*).
- A table for the timeline of the journey. Each column represents one touchpoint with a detailed visual and/or textual description as one row. One row consists of the following elements:
    - The activity or behavior of the customer, with optional needs and questions/emotions of the customer
    - The perceivable layer of the digital solution in terms of analog, digital, or personal touchpoints
    - The underlying layer of the digital solution in terms of a digital system aspect or a human aspect

Besides the timeline, experience labels are useful for highlighting the quality of the experience for the customer in terms of positive or negative experience and special trust points or moments that are of special importance for the customer.

2.2.3.6 Business Model Canvas

Understanding values and the nature of relationships that create these values is supported by the value proposition canvas. In order to actually realize a sustainable digital solution, the business model perspective is necessary as an additional perspective. A business model canvas [OsPi2010] is a template that describes a business model in a compact form.

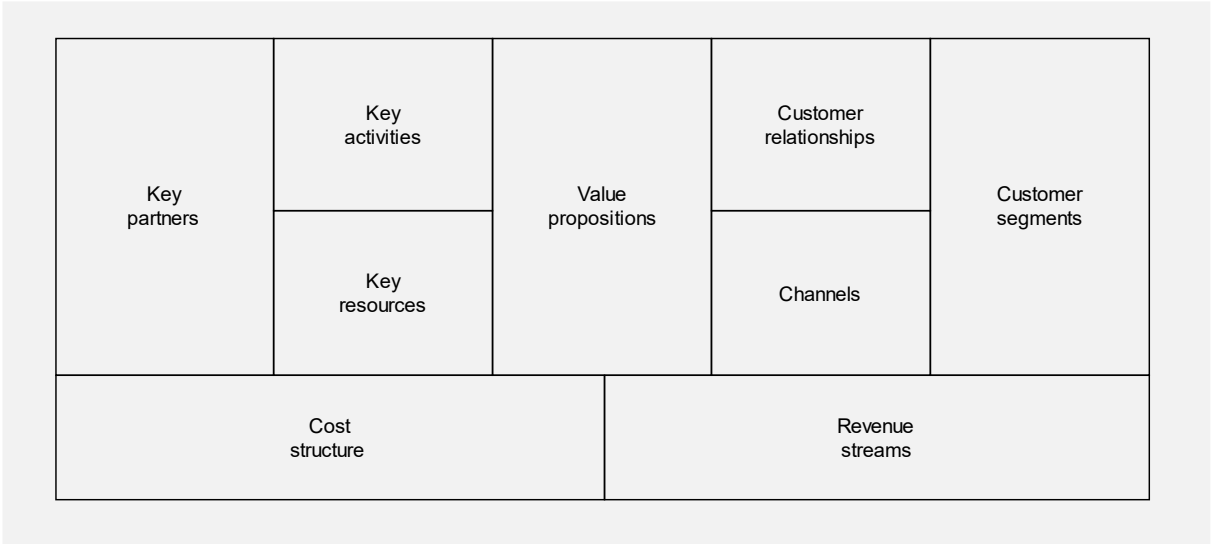


Figure 18 – A business model canvas template

It consists of the following elements (see Figure 18):

- *Key activities:* important activities in executing a company's value proposition
- *Key resources*: resources that are necessary to create value for the customer
- *Partner network*: description of partners for establishing the business model
- *Value propositions*: the collection of products and services a business offers
- *Customer segments*: description of customer segments
- *Channels*: distribution and delivery channels
- *Customer relationships*: intended relationship with customer segments
- *Cost structure*: costs necessary for running the business
- *Revenue streams*: income from the business

The business model canvas combines two central perspectives: the outside perspective (customer) and the inside perspective (building or client organization). When working on the business model canvas, a building team will naturally switch between both perspectives and can thereby incorporate different stakeholder perspectives.

2.2.4 Design Concepts at the System Level and Element Level: General Considerations

So far, we have presented document templates for all levels in Section 2.2.2 and documentation techniques for the solution level in Section 2.2.3. Now, we enter the world of the system level and element level. For beginners, this is a significant step in terms of writing style and working mode. To make this step easier, we start with a general discussion on the difference between solution level and system/element level in terms of writing style and working mode.

The concepts at the solution level (Digital Design brief and the solution design concept) rely on continuous text documentation in a linear fashion and a canvas-oriented documentation approach. To create design concepts at the system level and element level, a different and more structured writing style is better suited for the following reasons:

1) It must reflect the level of detail necessary to build a digital system and its elements, and
2) It must support integration into the other activities of the building process.

The writing style used in the creation of system or element design concepts (in short: SoE concepts) is fundamentally different from the writing style we use in the Digital Design brief, in the solution design concept, in everyday life (e.g., in letters), or at school or university (e.g., for exams or theses). This *everyday* language is typically subject to the premise that communication is executed in a linear form, i.e., a letter or seminar paper is read from beginning to end in order to grasp the meaning of the text or communication. By the way, this of course does not mean that the texts are also written in linear form (i.e., from front to back).

An SoE concept is subject to the same linear structure—this is the nature of language—and is defined at two levels:

- The *headline level* defines the document structure of a concept and gives clear guidance and access to the content of each section of the concept.
- The *building block level* (i.e., a concrete aspect of the system or element, see Section 2.2.5) is used to describe a particular aspect of the system (or the element) that has to be realized[6].

[6] Similar approaches for describing concepts can be found, e.g., in architecture (cf. [AIIS1977]).

However, an SoE concept is not created with the goal of being read and understood in a linear fashion; it must merely have a clear structure. A good model for clear structures is hierarchically structured documents—for example, lexicons, catalogs, or technical books are created according to tree structures. Furthermore, software tools are a must for managing and maintaining such types of concepts.

If high priority is given to linear comprehensibility in SoE concepts, this often leads to SoE concepts that contain a lot of explanatory text added in places but being much better suited to other parts of the document. This inevitably leads to redundancies in the concept, to more text that must be read, and at the same time to an increased risk of inconsistencies and gaps.

In a nutshell, an SoE concept is neither a Shakespeare play nor a non-fiction book for children. An acceptable analogy for a concept is construction plans. It is not primarily about telling a nice story or describing a subject matter in a vivid way; it is primarily about describing a digital solution in all its facets in as compact a form as possible—for example, in order to be able to implement it or to estimate the effort required for implementation.

To be completely clear, a concept that is easy to read and digest can certainly be considered a small work of art. However, the terms *easy to understand* and *nice to read* mean something completely different in the context of design concepts than in the context of novels or non-fiction.

In the following, we provide a more detailed explanation of the writing style for SoE concepts.

2.2.4.1 On the Value of Non-Redundancy and Tools for Avoiding Redundancy

In explanatory texts, repetition of facts is often understood as a didactic means of promoting the learning of certain information or to avoid looking up important information in other parts of a text.

Redundancies increase the effort for reading concepts. They pose a risk to the quality of SoE concepts and the process of creating them, because redundant information is a source of misunderstandings and inconsistencies and reduces the ability to modify and extend the SoE concepts. Furthermore, redundant documentation alone increases the effort required for creation and maintenance since the information must be described redundantly. This increase is also noticeable when copying and pasting since text fragments generally cannot be copied one-to-one from one section of the document to another. We recommend browsing the YPRC case study concepts to get an understanding of this way of working.

Cross-references are the stylistic tool of choice for avoiding redundancies in SoE concepts. The principle for defining cross-references is simple. If a piece of information (e.g., a set of rules, a data set, or a function) is needed more than once in the concept, then this information should be described in isolation and referenced with a cross-reference at the appropriate places. Let us look at an example with a high level of redundancy (each bullet point represents a building block).

- A1: The user record consists of a username (min. 6 characters, max. 15 characters, no special characters) and a password (min. 8 characters, max. 15 char., no dollar sign ($)).
- A2: The system must secure access to the user data by entering a username (min. 6 characters, max. 15 characters, no special characters) and a password (min. 8 characters, max. 15 characters, no dollar sign ($)).
- A3: When a new user registers, the system must ensure that:
    - The username meets the following criteria: min. 6 characters, max. 15 characters, no special characters

    - The password meets the following criteria: min. 8 characters, max. 15 characters, no dollar sign ($)
- A4: If the user changes their password, the system must ensure that the password meets the following criteria: min. 8 characters, max. 15 characters, no dollar sign ($).

Of course, the example is exaggerated but it should illustrate clearly that the same information about the structure of the username and password is needed in different places. In this example, the redundancy is very obvious. In everyday work, however, the information presented can be provided in different parts of a document and potentially, even be formulated by different persons.

A less redundant form of description can be achieved by using cross-references.

- A1: The user data record consists of a username (for criteria, see A5) and a password (for criteria, see A6).
- A2: The system must secure access to the user data by entering a username (see A5) and a password (A6).
- A3: When a new user registers, the system must ensure that the criteria for the username (see A5) and the password are met (see A6).
- A4: If the user changes their password, the system must ensure that the criteria for the password are met (see A6).
- A5: The username must meet the following criteria: min. 6 characters, max. 15 characters, no special characters.
- A6: The password must meet the following criteria: min. 8 characters, max. 15 characters, no dollar sign ($).

The second form of documentation shifts the criteria for username and password to a separate building block and refers to the corresponding building block at all appropriate places. This avoids redundant descriptions of the criteria and also improves the readability of the individual statements. At the same time, the modifiability and extensibility are improved since the criteria for username or password only have to be changed or extended in one place.

The downside of this redundancy reduction is, of course, the lack of immediate availability of information about the criteria as well as less linear readability. To access this information, you would have to jump to the appropriate place in the document while reading it. You can minimize this effort by using appropriate tools. In practice, having al lot of cross-references can be an obstacle, especially in the later implementation of the system, since developers may have scattered the information for their implementation task over an entire document.

The use of integrating structures (e.g., user stories, see Chapter 5), which summarize the relevant information for an implementation task in a suitable form, can help here.

2.2.4.2 Modularity of Information

The sentence structure in natural language makes it possible to express any subject matter in a single sentence. We can join sentences with the simple word *and* and insert something in a sentence using commas, just like that, without the sentence ending. Such constructions are often found in everyday texts and serve, for example, to combine relevant information in one sentence. For SoE concepts, the combination of different facts in one sentence is again a source of redundancy and reduces the modifiability or extensibility of the concept. Here is an example:

- A1: The system must secure access to the user data with username and password and block access if the password is entered incorrectly three times.

In this example, the type of access protection is combined with an access protection behavior (three wrong entries lead to blocking). Even though this combination is unproblematic from a linguistic point of view, the combination of both aspects is not desirable in terms of good concepts. A separation into two sentences does not hurt at this point and produces two referenced statements:

- A1: The system must secure access to the user data with username and password.
- A2: If the password is entered incorrectly three times, the system must block access.

When creating concepts, the recommendation is to formulate only one fact (e.g., a problem., function, or a requirement) per building block.

2.2.4.3 Relationships between Building Blocks

Besides understanding the particular building blocks of a concept, it is important to understand the relationships between these elements. Understanding the relationships is important for creating a complete and consistent concept:

- *Completeness* means that the concept is complete in itself, i.e., any information referenced in a concept is actually present and described in the referenced part of the concept. For example, if a concept describes a function *validate date of birth*, that refers to the attribute *date of birth* of the entity *customer*. If the attribute *date of birth* is not described in a particular data element (called *entity*, see below) *customer*, then this concept is incomplete.
- *Consistency* means that a particular piece of information in a template is used in the same way as in all other templates of the concept and that the pieces of information at different places in the concept do not conflict with each other. For example, if a function *validate date of birth* refers to an entity *customer* as customer and another function *validate address* refers to the entity *customer* as *user*, then this concept is inconsistent.

For a DDP at foundation level, it is useful to distinguish between the instructive and constructive perspectives to understand the relationships between the building blocks of SoE concepts.

The instructive perspective provides explanatory information that motivates the form, function, and quality of a digital solution. Important instructive relationships are:

- *Source relationship:* describes the source of a certain element; for example, a company strategy paper demands the development of a dedicated functionality. In this case, the strategy paper is a document that is the source for a particular function.
- *Constraint relationship:* describes a constraint that has led to a particular element; for example, a standard demand that every business interaction is stored in a dedicated log file.
- *Goal relationship:* describes the objective in terms of what the motivation was for a particular element. For example, a goal is formulated as *Support guidance for new users*. To achieve this goal, the digital solution provides a tutorial mode that guides the user to the main functions of the solution. Each element of this tutorial mode (e.g., dedicated user interface) can then be related to the particular goal.

The constructive perspective describes the relationships between the building blocks that describe the form, function, and quality that make up the digital solution. We will come back to the constructive relationship in Section 2.2.6.

2.2.4.4 On Readability versus Structured Documentation

We want to close this section with some remarks on readability. The approach for creating system and element design concepts presented in this chapter will lead to highly structured concepts. Our experience shows that for stakeholders that are not trained in working with such structures, these concepts are difficult to read. We have experienced that stakeholders who deal with such concepts on a regular basis become used to the structure and even appreciate the highly structured concepts.

However, this does not mean that these concepts should be presented to all stakeholders without further modification. Remember, Digital Design means being responsible for proper communication with all stakeholders. This responsibility may mean that design concepts have to be prepared separately for specific stakeholders (for example, company managers) so that these stakeholders can understand the content of the concepts with reasonable effort to provide feedback.

2.2.5 Documentation Techniques for the System Level and Element Level

In the following, we introduce a number of documentation techniques that are useful for working with design concepts at the system level and element level.

2.2.5.1 A Note about the Idea of Perfect Technology

The assumption of perfect technology compared to real technology [WaMe1986] supports the design of digital systems and their elements. It means in particular defect-free technology as well as infinite computing capacity, storage capacity, and infinite communication capacity and speed. This assumption simplifies working on design concepts because the limits of technology do not need to be considered in the design concept.

However, the assumption of a perfect technology does not extend to users or existing systems. Unexpected behavior of users must be considered (for example, when defining use cases, see Section 2.2.5.6). Similarly, the temporary unavailability of existing systems (for example, a payment provider) must be considered when creating design concepts.

The assumption of perfect technology is not free from considerations of the limits of technology

The assumption of perfect technology is merely a simplification for the creation of design concepts. In the further course of the building process, the assumptions implied by perfect technology have to be removed step by step. This is done through intensive cooperation within the activity area construction (see Section 1.3).

2.2.5.2 A General Building Block Template

Using templates is a proven technique for providing a reference structure to represent elements of concepts or specifications (cf. [CPRE2020]) in a modular way, as introduced in Section 2.2.4. A basic template consists of the following sections[7]:

- Identification number with title
- Relationships to other elements
- Description of the particular element

[7] For experts in information modelling, this basic template can be considered a super class (or super entity) that other classes will be derived from.

Considerations for daily work

A pragmatic approach for defining the identification number is referring to the type of element (see below) with an abbreviation and a number (e.g., F-3 for function no. 3, or UC-12 for use case no. 12). With this approach, the reader of a design concept can tell from the identification number which type is referenced. An important rule is that numbers may not be reused and alphanumeric sorting of numbers is not necessary. It takes some time to get used to this in the beginning, as we are used to this from chapter numbers. In practice, however, it does not pose a problem.

Optional sections of a basic template are:

- Source (e.g., useful for referencing additional information)
- Status (e.g., to describe whether an element is, for example, agreed, under implementation, or done)
- Change log (e.g., to document the evolution of an element)

The use of the optional sections depends on the particular building process.

2.2.5.3 Goal Template

Objectives are typically derived from the solution design concept (see Table 4). Good sources for objectives are the vision statement, the value proposition canvas, and the business model canvas (see Figure 18). Nevertheless, it is not useful to simply repeat all objectives in the system design concept. The description of objectives should be limited to objectives that have a clear relationship to the elements of the digital system.

A goal template can be used to capture each particular objective. Figure 19 shows an exemplary goal template with content from the YPRC case study. The headline provides the ID (G for goal), the number of the element (here, 1), and the title. The title of a goal should be a crisp summary statement of the goal. Further details of the goal will follow in the description.

Advice for creating the description

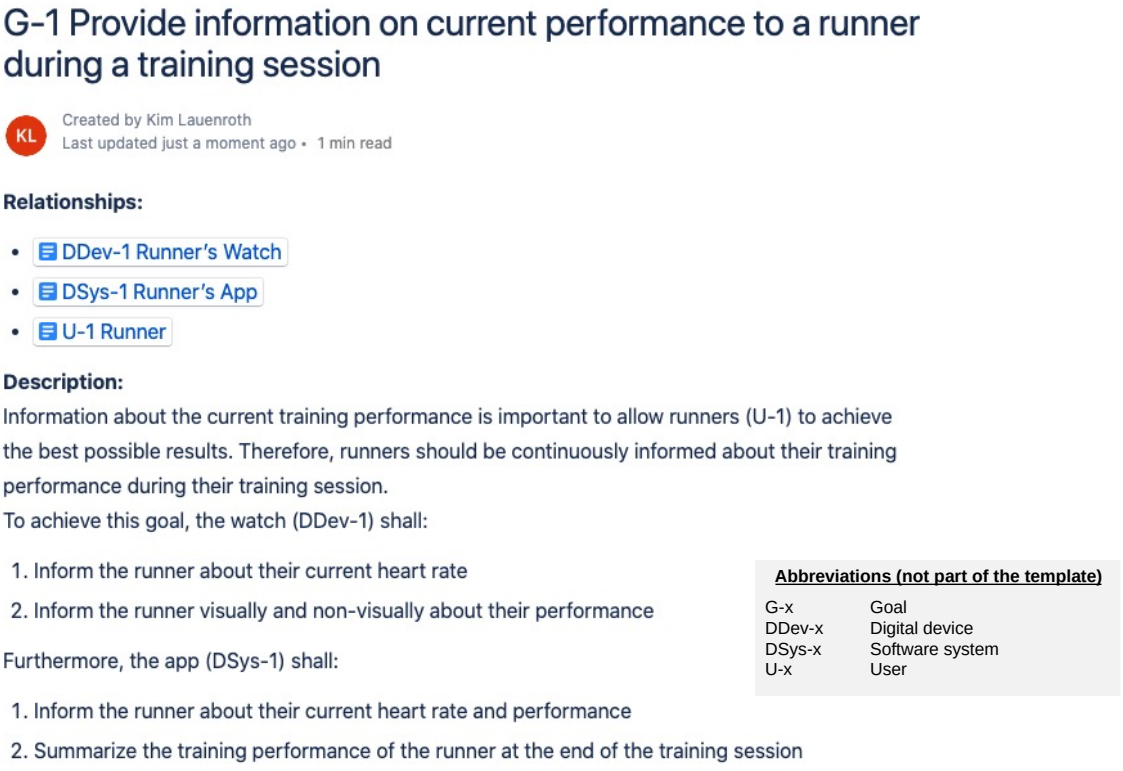
The description consists of a more detailed explanation of the goal, including a rationale about why the goal is important for the digital system. In the example given, the goal is related to the runner; therefore, the goal must describe the importance of the goal for the runner.

It is difficult to formulate a good goal description, especially with respect to functions of a digital system. A good rule of thumb for goal formulation is to avoid mentioning the particular elements of the digital system in the goal formulation. Furthermore, it is useful to distinguish between the following two types of goals:

- *Hard goals* allow for objective measurement.
- *Soft goals* allow for subjective measurement.

If possible, soft goals should be concretized by detailed hard goals or other criteria that allow for the evaluation of the goal fulfillment.

The concretization of a goal can be included in the description as an explanation of how a goal will be achieved by the digital system. Providing these additional details defines a clear border between the goal itself (the first paragraph) and the achievement of the goal (paragraphs two and three).



G-1 Provide information on current performance to a runner during a training session

KL Created by Kim Lauenroth
Last updated just a moment ago • 1 min read

Relationships:

- DDev-1 Runner's Watch
- DSys-1 Runner's App
- U-1 Runner

Description:

Information about the current training performance is important to allow runners (U-1) to achieve the best possible results. Therefore, runners should be continuously informed about their training performance during their training session.
To achieve this goal, the watch (DDev-1) shall:

1. Inform the runner about their current heart rate
2. Inform the runner visually and non-visually about their performance

Furthermore, the app (DSys-1) shall:

1. Inform the runner about their current heart rate and performance
2. Summarize the training performance of the runner at the end of the training session

Abbreviations (not part of the template)

G-x	Goal
DDev-x	Digital device
DSys-x	Software system
U-x	User

Figure 19 – A goal template/example from the YPRC case study

Furthermore, describing the achievement of the goal allows explicit definition of the important elements of the digital system that are necessary for achieving the goal. These relationship descriptions make important design decisions on the digital system explicit to the reader. In this example, it was decided that the watch and the app are responsible for informing the runner of their training performance. The detailed statements can then be considered as goals for the particular elements and documented explicitly in the objectives section of the particular element.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Each element from the system design concept that is mentioned in the text and has a relationship to the goal described is mentioned.

Considerations for daily work

Goals frequently change during the creation of a system design concept. However, they provide a good starting point for understanding and structuring a digital system. We therefore recommend defining the goals of a digital system in detail at the beginning. Such work will typically create a detailed list of goals that has to be revised later in the process. This revision of goals is very important for creating a clear system design concept. The revision especially includes the removal of goals that are no longer considered relevant for the digital system.

Finally, the list of goals for a digital system can be short. Three to five goals at the proper level of detail are often sufficient to capture the core objectives of a digital system. Longer lists may create goals at a level of detail that typically belongs to the element level. For example, the goal in Figure 19 mentions only that the runner shall be informed about the training performance. The goal does not detail how this information takes place or how this information is gathered.

2.2.5.4 Constraint Template

Like the goal documentation, some constraints can be derived from the solution design concept. However, additional constraints may originate from various other sources. Often, constraints will appear when further details of a digital system are elaborated. Good examples are technical constraints or legal constraints.

A separate constraint template can be used to capture each individual constraint of a digital system respectively. Figure 20 shows an exemplary constraint template with content from the YPRC case study. The headline provides the ID (Con for constraint), the number of the element (here, 2), and the title. The title of a constraint should be a brief summary of the constraint. Further details can follow in the description.

Advice for creating the description

The description should provide a detailed explanation of the constraint, including a brief rationale about why the constraint is important for the digital system. Furthermore, the elements that must obey the constraint should be mentioned explicitly.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the system design concept that is mentioned in the text and has a relationship to the constraint described is mentioned.

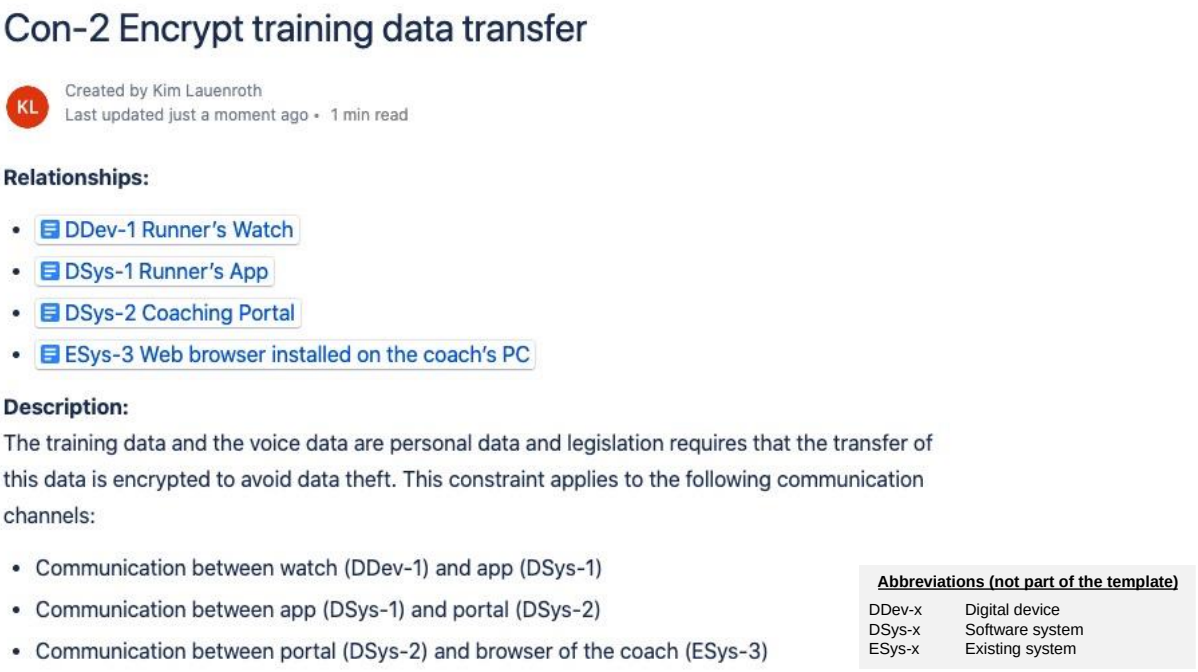


Figure 20 – A constraint template

Considerations for daily work

Defining constraints of a digital system is an ongoing process during the whole building process. It is useful to spend some time collecting constraints at the beginning of the creation of the system design concept. Potential sources of constraints include legislation, regulatory requirements of a domain, or the IT governance of an organization. Nevertheless, you should be aware that the constraint section of the system design concept requires continuous updates. In later stages of

the building process in particular, the constraint section is often neglected because constraints appear to be obvious for the people involved in the building process. A well-trained DDP is aware of this and takes the documentation of constraints seriously.

2.2.5.5 Form and Function at the System Level

At the system level, form means describing all elements that make up the digital system that realizes the digital solution. This includes the user types, the existing elements/objects, and the elements that have to be realized. At the system level, function means describing the functionality provided by the elements of the system as a whole.

2.2.5.5.1 Form: User Type Template

Figure 21 shows an exemplary user template for the runner from the YPRC case study. The headline provides the ID (U for user), the number of the element (here, 1), and the title.

Advice for creating the description

The description is a brief sentence that characterizes the user.

Advice for documenting relationships

In addition to the description, the user template should list references to all other building blocks that are important for the user.These are:

- Goals (instructive relationship) that are relevant for the user type described
- Scenarios (constructive relationship) that the user is part of
- Elements (constructive relationship) that the user interacts with

Considerations for daily work

The information in the element is very brief in itself. However, the description of the user is of great importance for traceability across all design concepts. A separate description is important, for example, especially for systems with different user types, to build a clear reference to the user type considered in use cases or scenarios.

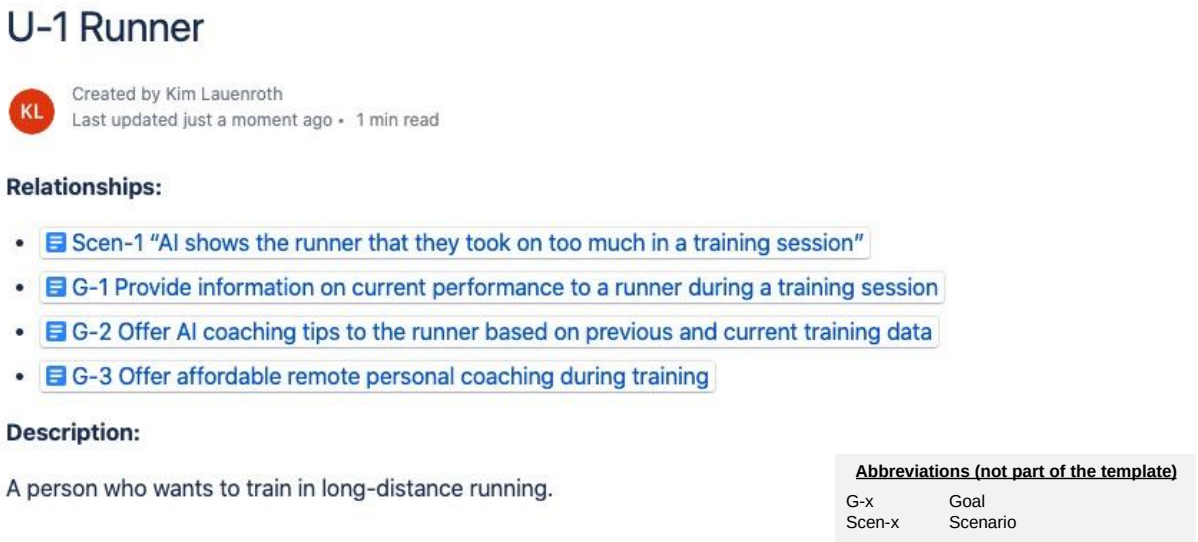


Figure 21 – A user template

2.2.5.5.2 Form: Existing Object Template

Figure 22 shows an exemplary object template with content from the YPRC case study. The headline provides the ID (Obj for object), the number of the element (here, 1), and the title.

Advice for creating the description

The description should consist of a short verbal explanation of the existing object, including an overview of the functionalities that the object provides within the system. References to other elements of the system should be given to clarify the interfaces of the object described to other elements of the system.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Each element from the system design concept that is mentioned in the text and has a relationship to the object described is mentioned.

Considerations for daily work

Similar to the user type, the description of existing objects is not very meaningful in itself. However, explicit documentation is important here for traceability across all concepts. Furthermore, the description of existing objects documents essential assumptions about the digital solution, namely the objects that are assumed in order to be able to use the digital system.

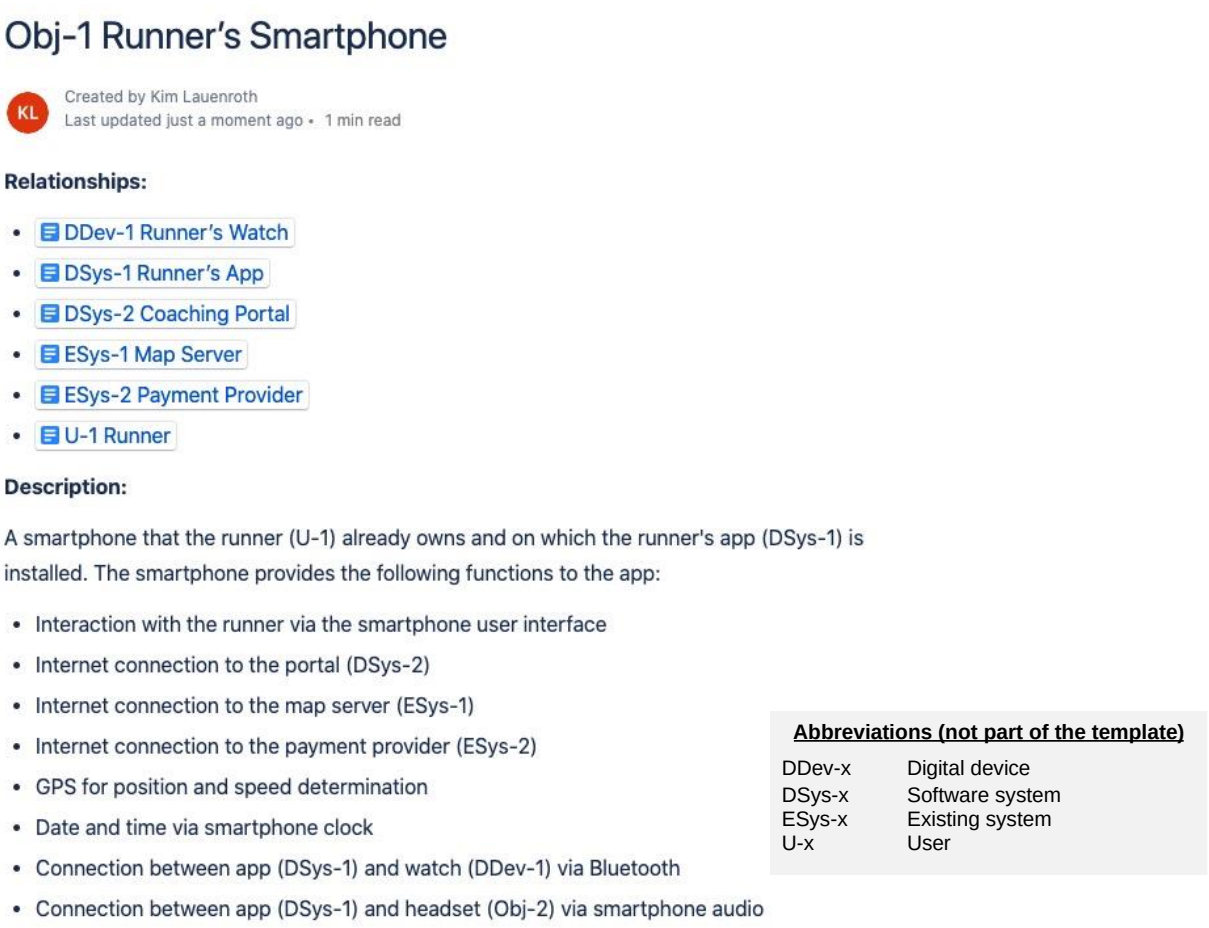


Figure 22 – An existing object template

2.2.5.5.3 Form: Existing System Template

Figure 23 shows an exemplary existing system template with content from the YPRC case study. The headline provides the ID (ESys for existing system), the number of the element (here, 1), and the title. The documentation of existing systems is important for describing the whole digital system and for understanding their contribution to the overall digital solution.

Advice for creating the description

The description in the template should give a brief overview of this contribution (for example, data or services that are provided to other elements).

Advice for documenting relationships

The list of relationships can be derived directly from the description. Each element from the system design concept that is mentioned in the text and has a relationship to the system described is mentioned.

Considerations for daily work

Documentation of existing systems, analogous to existing objects, is important for traceability and for documenting assumptions about existing systems. For example, the YPRC is mandatorily dependent on a map server. Without such an existing system, YPRC would not be able to function.

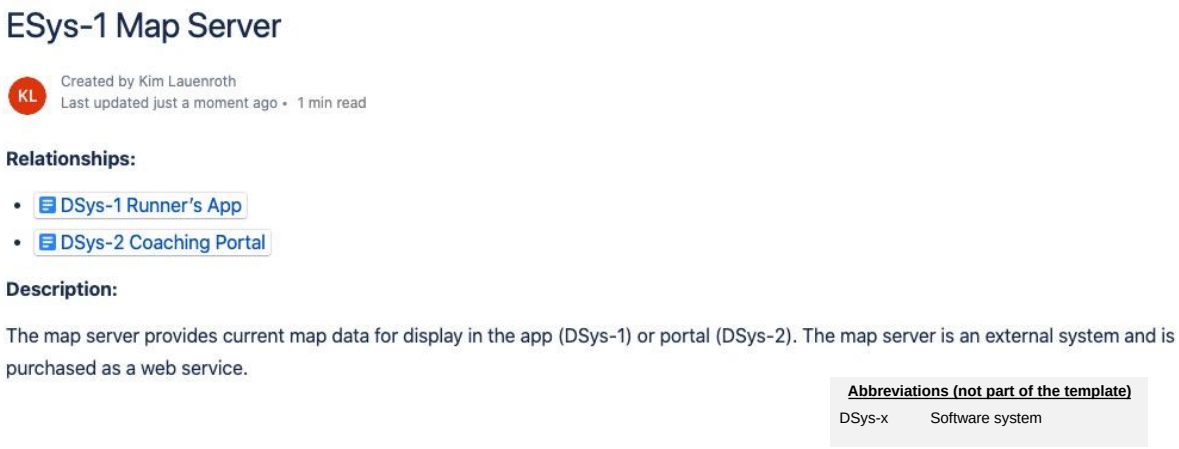


Figure 23 – An existing system template

2.2.5.5.4 Form: Digital Devices and Software Element Template

Even simple digital systems consist of two or more elements that will be realized. The quite simple YPRC case study consists of three elements (watch, app, and portal). The important driver for elements of a digital system is often the underlying form, with elements that provide data and functions to a perceivable element. Capturing these structures of the underlying form is an important task of the system level.

For each element, a dedicated template is created. In order to distinguish between the different types of elements (device and software element), we introduce two different templates: one for devices and one for software elements. The main difference between both templates is the ID, which indicates the type of element.

Figure 24 shows an exemplary device template and Figure 25 shows an exemplary software element template with content of the YPRC case study. The headline provides the ID (DDev for device, and DSys for software element), the number of the element (here, 1 for both examples), and the title. The title should be a short and unique name of the element that is memorable and understandable.

Advice for creating the description

The description should provide a brief overview of the element without providing too many details of the functionality. It should focus on the data that the element will store (remember, data is also part of the form) and on the relationship to other elements of the digital system. Details that are necessary to understand the form of the element can also be mentioned. For example, the runner's watch will have a button that allows interaction with the runner. In the case of a digital system, the description should refer to the existing device that is assumed to operate the software part of the digital system.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the system design concept that is mentioned in the text and has a relationship to the element described is mentioned. At first glance, the description of an element at the system level appears to be quite short and not really useful.

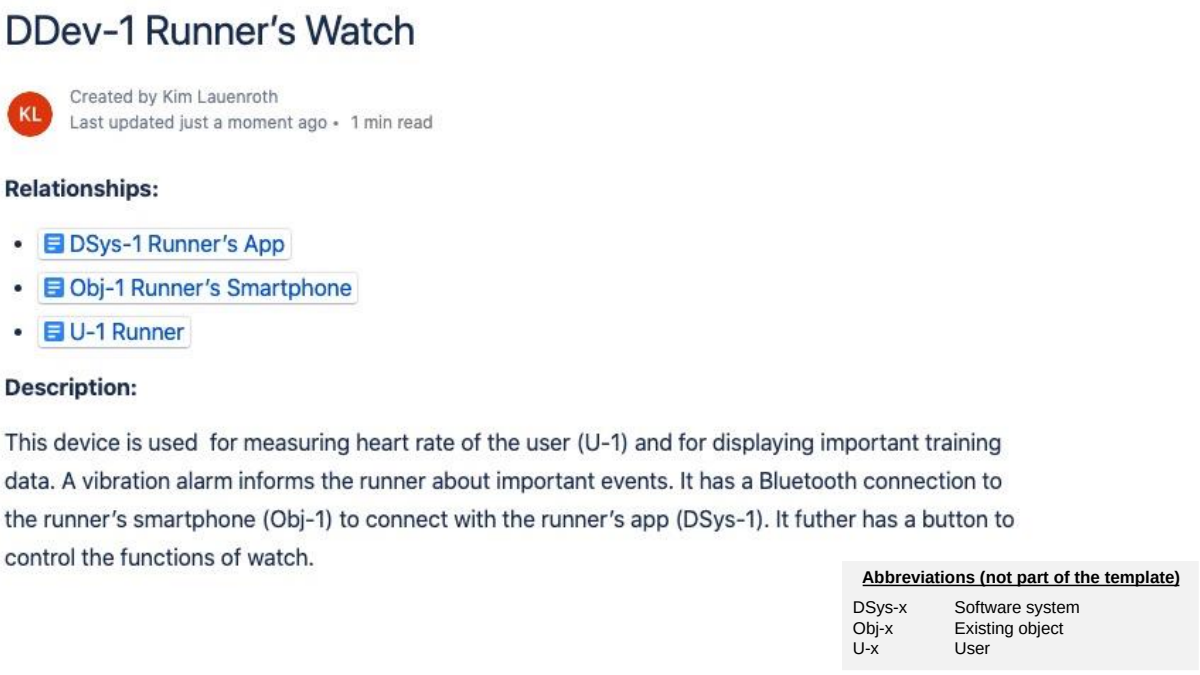
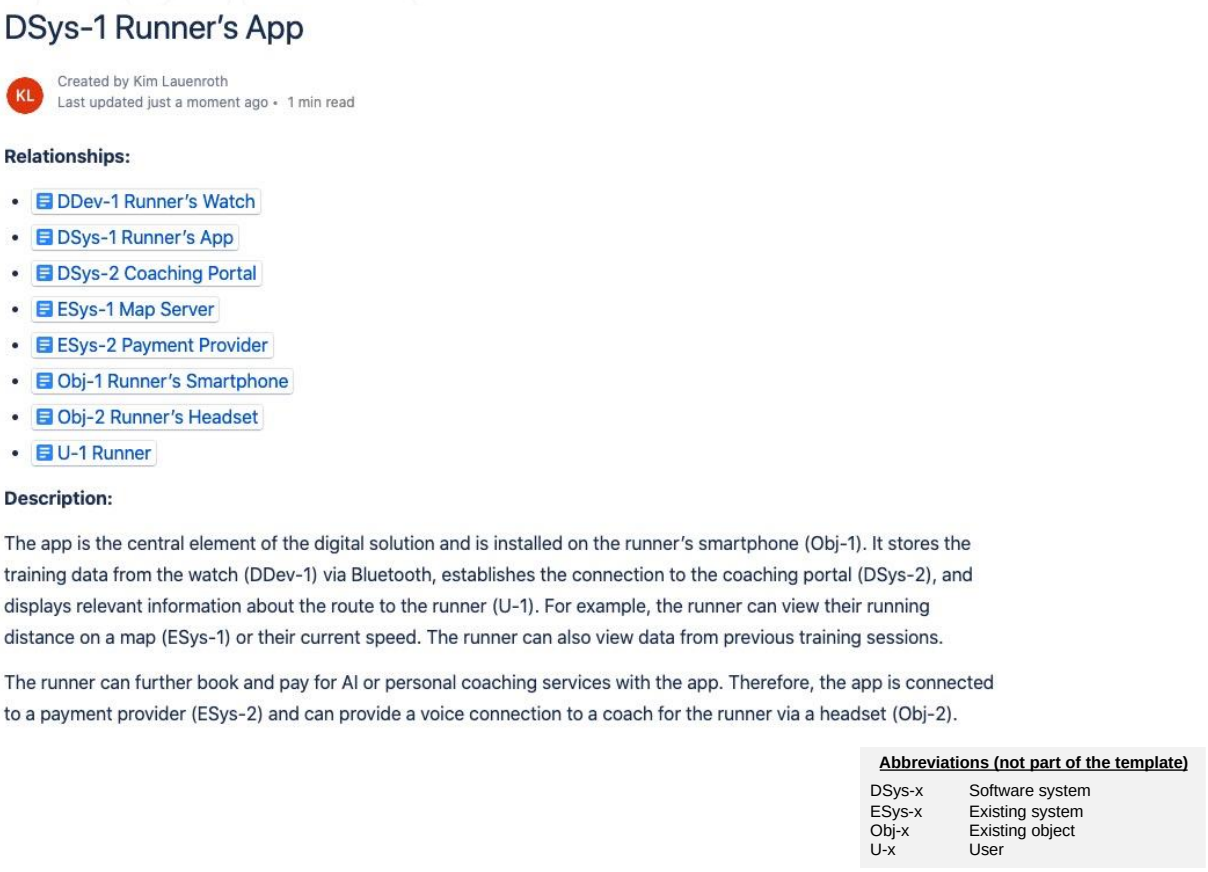


Figure 24 – A device template



DSys-1 Runner's App

KL Created by Kim Lauenroth
Last updated just a moment ago • 1 min read

Relationships:

- DDev-1 Runner's Watch
- DSys-1 Runner's App
- DSys-2 Coaching Portal
- ESys-1 Map Server
- ESys-2 Payment Provider
- Obj-1 Runner's Smartphone
- Obj-2 Runner's Headset
- U-1 Runner

Description:

The app is the central element of the digital solution and is installed on the runner's smartphone (Obj-1). It stores the training data from the watch (DDev-1) via Bluetooth, establishes the connection to the coaching portal (DSys-2), and displays relevant information about the route to the runner (U-1). For example, the runner can view their running distance on a map (ESys-1) or their current speed. The runner can also view data from previous training sessions.

The runner can further book and pay for AI or personal coaching services with the app. Therefore, the app is connected to a payment provider (ESys-2) and can provide a voice connection to a coach for the runner via a headset (Obj-2).

Abbreviations (not part of the template)

DSys-x	Software system
ESys-x	Existing system
Obj-x	Existing object
U-x	User

Figure 25 – A software template

Considerations for daily work

Keep in mind that the system design concept is the documented representation of our understanding of the digital system. If you are not able to provide a compact and clear description of each element of a digital system, it is very likely that you do not have a clear understanding of the particular elements of the digital system. Therefore, this short description is a great measure of your own understanding of the particular elements of the digital system and the effort of writing this short description is well spent to reassure yourself that you have understood the system that you are currently designing.

Furthermore, many stakeholders will work on this level of understanding and the short description of each element will be very useful in later stages of the building process for communicating about the digital system. This means that these descriptions must be maintained during the whole building process.

2.2.5.5.5 Function: Scenario Template

At the system level, the function is created by the particular elements concerned and can be very complicated. To maintain the proper level of detail when you are a beginner in Digital Design, we recommend describing the function of a digital system using scenarios.

As a rule of thumb, for each goal that a digital system shall achieve (see Chapter 3 of the system design concept above), one scenario should be described. For each scenario, a scenario template is created.

Advice for creating the description

Figure 26 shows an exemplary scenario template with content from the YPRC case study. The headline provides the ID (Scen for scenario), the number of the element (here, 1), and the title. The title of a scenario should be a one-sentence summary of the scenario. Further details can follow in the description.

Beginners in scenario writing should stick to five simple writing rules:

1) One goal per scenario: a scenario should explain the achievement of one particular goal in one way. Do not describe alternatives; avoid if-then-else structures.
2) Introduce the story in one sentence at the beginning.
3) Write every sentence with the structure *subject - predicate – object*; subjects are elements of the digital system.
4) Maximum one interaction per sentence: use simple sentences and describe one interaction between two elements of the digital system in one sentence.
5) Include the reference to the element templates to make the interaction more explicit.

These rules lead to very simplistic but clear scenario descriptions as shown in the example. Some additional details (e.g., the reference to the sunny morning in the example) are of course allowed to make the scenarios more readable and to create an understandable story. Together with personas, these kinds of scenarios are a central design tool for reflecting the digital solution and for creating empathy with users.

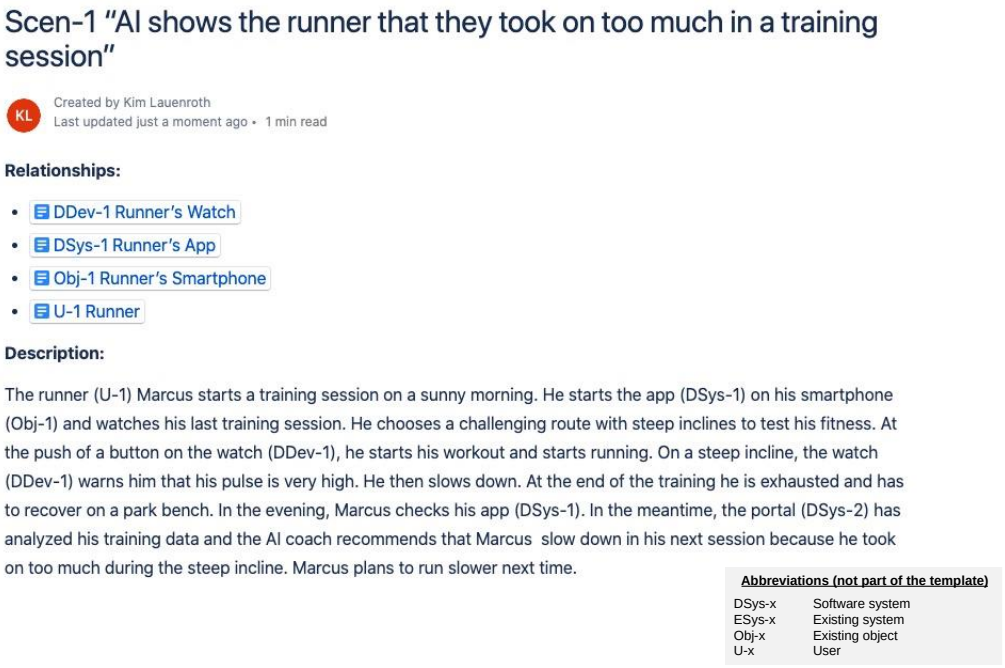


Figure 26 – A scenario template

Referencing the IDs (rule 4) of the element templates as shown in the example will hinder an easy reading of the scenario. Nevertheless, these references are important to show clearly which element of the digital system is involved and how, in which step of the scenario. For example, without the reference U-1, it is not explicit that the runner Marcus is a user of YPRC.

The example shows a scenario with a user involved. Keep in mind that a scenario can also describe a function that consists of pure underlying form. There is no user in such a scenario, only technical interactions between the elements. Such scenarios can be found in particular in

digital solutions that have automation as their goal—for example, the control of a heating system in an intelligent building depending on the available solar energy and other energy sources.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the system design concept that is mentioned in the description and has a relationship to the scenario described is mentioned. In addition, the goal that is achieved in this scenario should be included in the list.

Considerations for daily work

Textual scenarios according to the five rules are of course a very restricted technique for describing the function of a digital system. However, for beginners in Digital Design, these restricted scenarios are a good starting point for practicing the description of functionality at a system level and are especially useful for maintaining the proper level of detail that is suitable for the system design concept.

2.2.5.6 Form and Function at the Element Level

At the element level, form means describing the structure of a particular element in relation to the other elements. We distinguish between the following building blocks to describe the form:

- Underlying and perceivable hardware interface templates
- User interface templates to detail the interaction with users
- Software interface templates to detail the interaction with other software elements
- Entity templates to document the data
- Part templates to describe parts of a device

At the element level, function means describing the functionality that a particular element provides to users or other elements of the digital system. We distinguish between the following building blocks to describe the function:

- Function template to document underlying functions
- Use case template to document perceivable functions

2.2.5.6.1 Form: Hardware Interface Template

Interfaces are an important part of the elements of a digital system. They enable the interaction between different elements and are defined as follows:

Interface: A shared boundary across which information is passed.

With respect to hardware, we define a hardware interface as follows:

Hardware interface: An interface between an element of a system and a device.

This definition is rather abstract in order to cover a wide range of possible levels of interfaces.

We distinguish between *perceivable hardware* interfaces that allow users to interact with a device and *underlying hardware interfaces*, where a system element interacts with a device in a way that is not perceivable for the users of the system. In digital systems, hardware interfaces include, for example, displays, audio input and output, and communication hardware.

Advice for creating the description

At foundation level, it is sufficient to describe perceivable and underlying hardware interfaces in a textual form. The description should provide a brief explanation of the interface, including the other element of the system to which the interface is connected. The description of the interface should also mention the function, use case, or entity that is relevant for the interface.

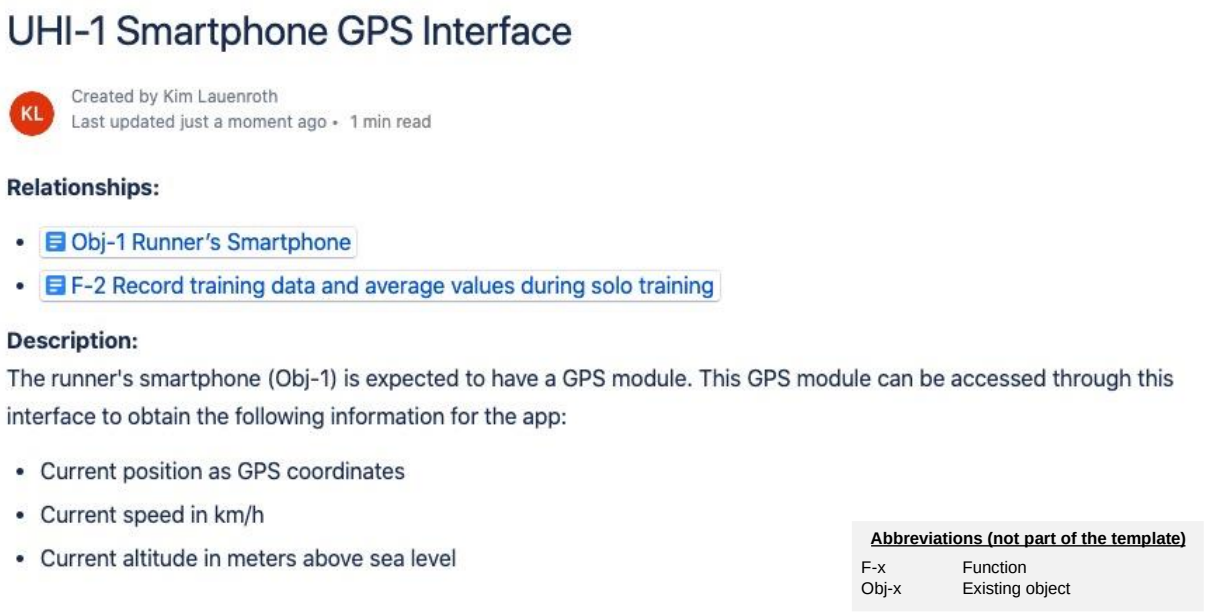


Figure 27 – A hardware interface template

Figure 27 shows an example of an underlying hardware interface template from the YPRC case study. Note the reference to the runner's smartphone as a reference to the object at the system level that provides this interface. Also note the reference to the function F-2 that will use data from this interface (see also below).

Advice for documenting relationships

The list of relationships can be derived directly from the description. Keep in mind that an interface has relationships to other elements at the system level; therefore, you should mention the particular element from the system design concept. It is also possible to mention the corresponding interface from the element design concept of the element referenced.

Considerations for daily work

For beginners, it is useful to distinguish between underlying hardware interfaces that provide connection capabilities and interfaces that provide access to further capabilities of the device that operates the software element (e.g., access to location sensors, see below).

Typical underlying hardware interfaces that provide connection capabilities are:

- Bluetooth or other wireless network connections
- Location sensors to access data from the position network (e.g., GPS or Galileo)
- Internet connection (e.g., LTE or wireless LAN) to connect with existing systems

The internet connection in particular is often taken for granted. Nevertheless, it is very important to document this underlying hardware interface since the internet connection typically has a

significant impact on the capabilities and qualities of an element and therefore the whole digital solution (see Section 2.1.3).

Devices also provide additional capabilities/information to a software element. Typical examples of such underlying hardware interfaces are:

- Clock to get information on time and date
- Battery level information in the case of a mobile device
- Orientation information of a mobile device

The documentation and the knowledge about the hardware interfaces of an element are an important basis for the cooperation with hardware experts.

2.2.5.6.2 Form: User Interface Template

A user interface (see Section 3.2) belongs to the perceivable form and is typically a visualization of data on a screen (e.g., text field) and provides interaction points (e.g., buttons, input fields). The user interface is defined as follows:

User interface: An interface for the exchange of information between a user and a system.

Note that the design of a user interface includes form and function (in particular, the structure and dynamics of information exchange), as well as quality (in particular, usability and user experience).

Advice for creating the description

The template of a user interface therefore consists of a visualization of the user interface and a textual description. Figure 28 shows an exemplary user interface template from the YPRC case study.

In a software design concept, it is very useful to describe the different elements/screens of the user interface with individual templates. This allows each user interface to refer to the necessary data, interfaces, functions, and use cases that are part of a dedicated user interface.

At foundation level, this basic understanding is sufficient to design typical software elements such as apps and business software. More sophisticated technologies—for example augmented reality (see Section 3.2) interfaces—may require other or additional building blocks.

Advice for documenting relationships

The user interface is a core part of every element. This role is also reflected by the various relationships. Note the various relationships mentioned in the text of Figure 28 that refer to related templates.

We recommend documenting the following relationships for user interfaces:

- The use case or use cases in which the interface is used, or the use cases that can be initiated by the user interface
- The functions, hardware interfaces, or software interfaces that provide data for the user interface or that use data from the user interface
- The entities that provide the data that is shown on the user interface

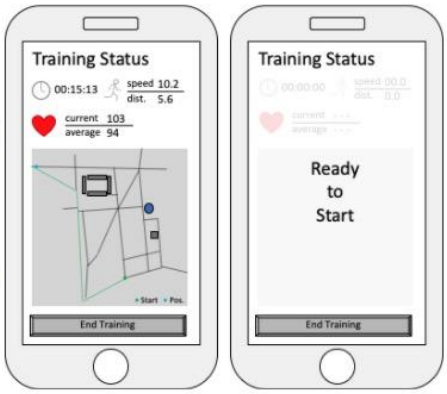
Furthermore, it is useful to document the relationship to a particular goal that is achieved by the user interface.

UI-5 Training Status Screen

Created by Kim Lauenroth
Last updated less than a minute ago • 1 min read

Relationships:

- E-4 Training Status
- E-5 Training Unit
- E-6 Track Data Point
- G-6 Inform the runner about current heart rate and performance with the app
- PHI-4 Smartphone Touch Screen
- SI-4 Map Data Interface (App)
- UC-2 The runner starts a training session with the watch
- UC-6 The runner initiates a solo training session on the app
- UC-11 The runner initiates a coached training session on the app



Description:

The training status screen presents the current training status to the runner during a training session (UC-6, UC-11). While the app is waiting for the runner to start the session with the watch (UC-2), the app shows an overlay message "ready to start" (right screen).
During an active training session, the screen shows the following information:

- Duration of the training (E-5.1)/current heart rate (E-4.2)/average heart rate (E-5.4)
- The current speed (E-4.3) and the running distance (E-5.5)
- A map with the start, the route and the current position (E-6)

The button "End Training" ends the training session (UC-6, UC-11).

Abbreviations (not part of the template)

E-x.y	Entity with optional reference to attribute y
G-x	Goal
PHI-x	Perceivable hardware interface
SI-x	Software interface
U-x	User
UC-x	Use case

Figure 28 – A user interface template

Considerations for daily work

This variety of relationships between a user interface and other building blocks of an element design concept can seem quite complicated for beginners. However, understanding and maintaining these relationships is a core tool for handling even the most complicated digital solution.

Another important aspect of the description of user interfaces is the visual design of the user interface, which requires additional design skills (see Section 2.1.4). In interaction design and visual design, it is common to define a design system to document and organize design materials and to provide guidelines, style guides, and guardrails for design. Atomic design [Fros2020] is an exemplary technique for creating a design system for user interfaces. Nevertheless, we consider design systems as an advanced level topic. Readers who are interested in this topic can start with [Fros2020] as an introduction to design systems.

2.2.5.6.3 Form: Software Interface Template

In addition to user interfaces, software interfaces are an important building block of the underlying form to describe the connection of a software element with other elements or systems:

Software interface: An interface between a software element of a system and an element of the same system or of another system.

This connection can be used to obtain data from an existing system (e.g., map data from a map server) or to include external functions provided by an existing system (e.g., a payment from a payment provider). If a software interface is used to connect with another element that is realized as part of the digital solution, a corresponding interface must be defined in the element design concept of the other element for consistency.

Advice for creating the description

At foundation level, a textual description of software interfaces is sufficient. Figure 29 shows an exemplary software interface template from the YPRC case study. This description mentions the objective of the interface and how the software interface is connected to other parts of the digital system (here, the runner's watch). Check out the device design concept of the watch for the corresponding software interface on the watch that sends the status to the app. In the YPRC case study, we defined the interface *Transfer health data to app* for the runner's watch and *Receive health data from watch* in the runner's app as two corresponding building blocks. At first sight, this may seem like redundant information. In practice, however, this description is necessary since the realization of this connection requires implementation work in both elements. Furthermore, this explicit description supports the definition of a clear interface between both elements.

SI-7 Watch-to-App Status Interface

KL Created by Kim Lauenroth
Last updated just a moment ago • 1 min read

Relationships:

- Con-2 Encrypt training data transfer
- G-6 Inform the runner about current heart rate and performance with the app
- UC-1 The runner connects the watch with the smartphone
- UC-2 The runner starts a training session with the watch
- UC-6 The runner initiates a solo training session on the app
- UHI-7 Watch Bluetooth Interface
- UHI-8 Smartphone Bluetooth Interface

Description:

The watch-to-app status interface is used to send status information from the watch to the app (UC-2). The watch signals the app that the runner has started his training. The watch sends the status "runner active" to the watch to start the data recording in the app (UC-6). When the runner ends his training or decides to make a break, the watch sends the status "runner inactive" to the app.

The connection between watch and app for this interface uses the Bluetooth module of the watch (UHI-7) and the Bluetooth module of the runner's smartphone (UHI-8). Establishing a Bluetooth connection is described in UC-1.

Abbreviations (not part of the template)

Con-x	Constraint
G-x	Goal
UHI-x	Underlying hardware interface
UC-x	Use case

Figure 29 – A software interface template

Advice for documenting relationships

Similar to user interfaces, the software interface can have various relationships to other building blocks of the element design concept. We recommend documenting these relationships carefully as a tool for handling complicated digital solutions.

Considerations for daily work

The careful documentation of software interfaces is usually unfamiliar to beginners in Digital Design since software interfaces are rather technical in nature and can be understood as a construction task. Nevertheless, a DDP must consider software interfaces because Digital Design also includes the design of the underlying form and Digital Design must define what software interfaces an element needs in order to fulfill its desired purpose.

Software interfaces are also an important part of the joint work with realization experts. A profound understanding of existing software interfaces enables a DDP to understand and communicate with software realization experts.

2.2.5.6.4 Form: Entity Template

The last element of the underlying form for the digital part is the entity. It is defined as follows:

Entity: A distinguishable structure of data stored by an element of a digital solution.

The term entity originates from computer science and refers to data structures that are stored and manipulated.

Typical examples of entities in a digital system are:

- User data: data that identifies a user (name, password, address)
- Status data: data that describes the status of the element
- Transaction data: data that describes a certain transaction performed by the digital solution (e.g., purchasing a book)

The explicit documentation of entities is important for the following reasons:

1) To document the available data stored (the table characterizes the entity)
2) To provide a reference point for the description of user interfaces (see relationship to UI-5 in Figure 30)
3) To provide a reference point for the description of functions (see below)

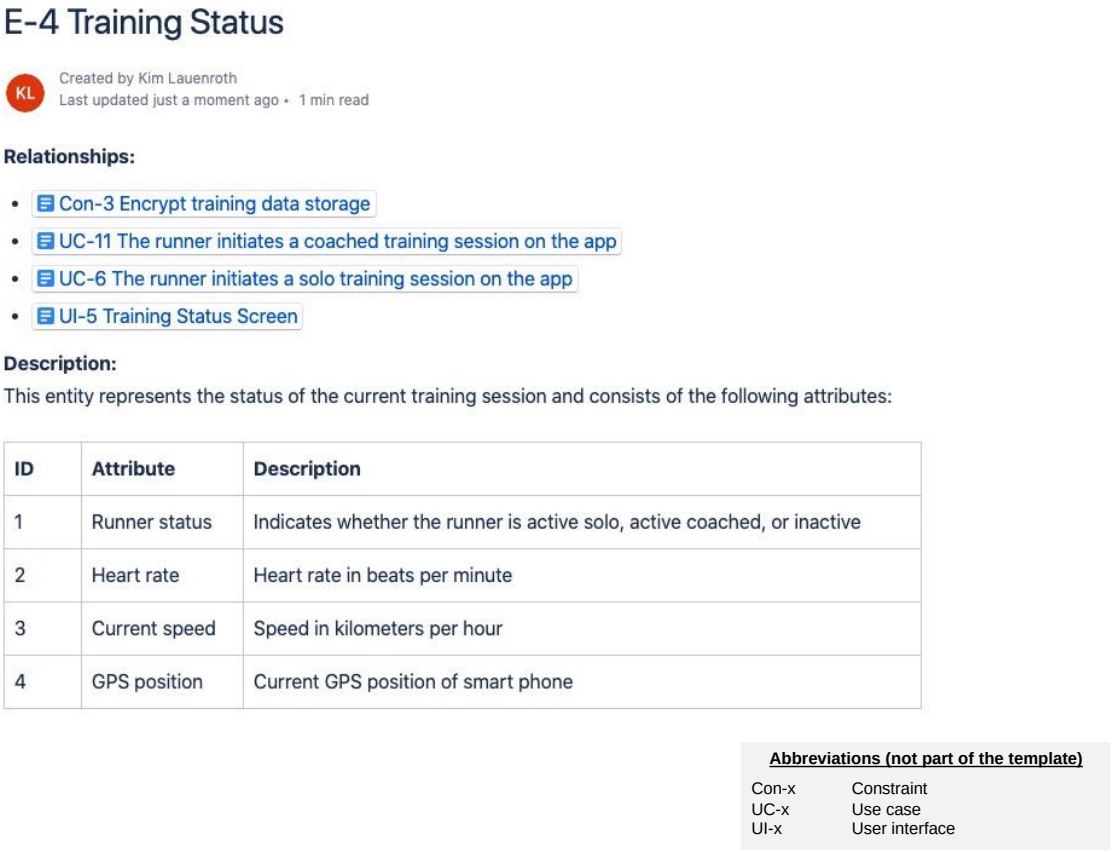
Advice for creating the description

There are several approaches for documenting data structures (e.g., UML class diagrams, entity-relationship diagrams). Experts will recognize the integration of these techniques into the software design concept presented. From a foundation level perspective, it is sufficient to identify and describe the individual entities and the information attributes that define an entity using tables. Figure 30 shows an example of an entity template from the YPRC case study.

Advice for documenting relationships

Relationships of an entity to other building blocks are typically not derived from the description of the entity. Instead, the relationships must be added to the description of the building blocks

connected. For example, if a user interface is described using data from an entity, then that user interface should be included in the list of references.



E-4 Training Status

KL Created by Kim Lauenroth
Last updated just a moment ago • 1 min read

Relationships:

- Con-3 Encrypt training data storage
- UC-11 The runner initiates a coached training session on the app
- UC-6 The runner initiates a solo training session on the app
- UI-5 Training Status Screen

Description:

This entity represents the status of the current training session and consists of the following attributes:

ID	Attribute	Description
1	Runner status	Indicates whether the runner is active solo, active coached, or inactive
2	Heart rate	Heart rate in beats per minute
3	Current speed	Speed in kilometers per hour
4	GPS position	Current GPS position of smart phone

Abbreviations (not part of the template)

Con-x	Constraint
UC-x	Use case
UI-x	User interface

Figure 30 – An entity template

Considerations for daily work

This indirect way of documenting relationships to an entity appears unusable at first sight. From our experience, however, this approach is very pragmatic and allows for efficient documentation of relationships.

2.2.5.6.5 Form: Physical Part Template

The physical part is a building block in the literal sense and is needed for the design of devices:

Physical part: The part that makes up a device.

A physical part can belong to the perceivable or underlying form of a device. It can provide technical functionality (e.g., a CPU or a battery) or can be part of the real form of the device (e.g., the housing of the smartwatch in the YPRC case study). Figure 31 shows an exemplary physical part template.

Advice for creating the description

The description gives a brief overview of what the physical part is. The description of a physical part will often reference other physical parts and hardware interfaces that are provided by the physical part. These interfaces allow the device to interact with its environment and with the user.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the design concept that is mentioned in the description and has a relationship to the physical part described is mentioned.

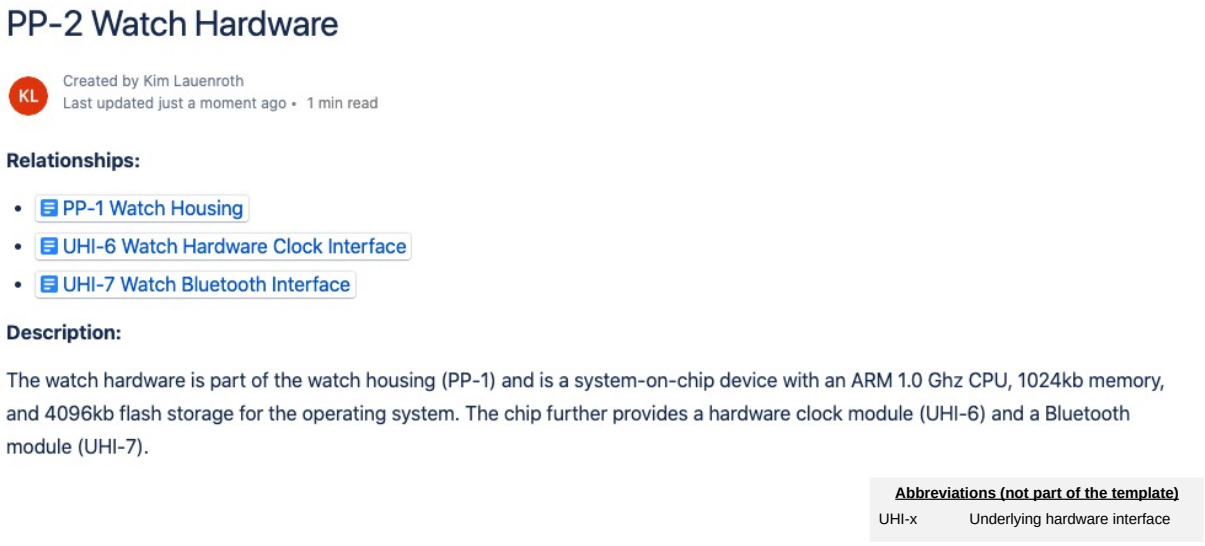


Figure 31 – A physical part template

Considerations for daily work

When describing physical parts, you should keep in mind that the template of the physical part is only a representative of a whole hardware development process (see also Section 2.2.2.5).

2.2.5.6.6 Function: Function Template

At the element level, function can be described with two building blocks: the function template and the use case template.

The function template addresses the underlying function of an element:

Underlying function: The description of the transformation of certain inputs into certain outputs.

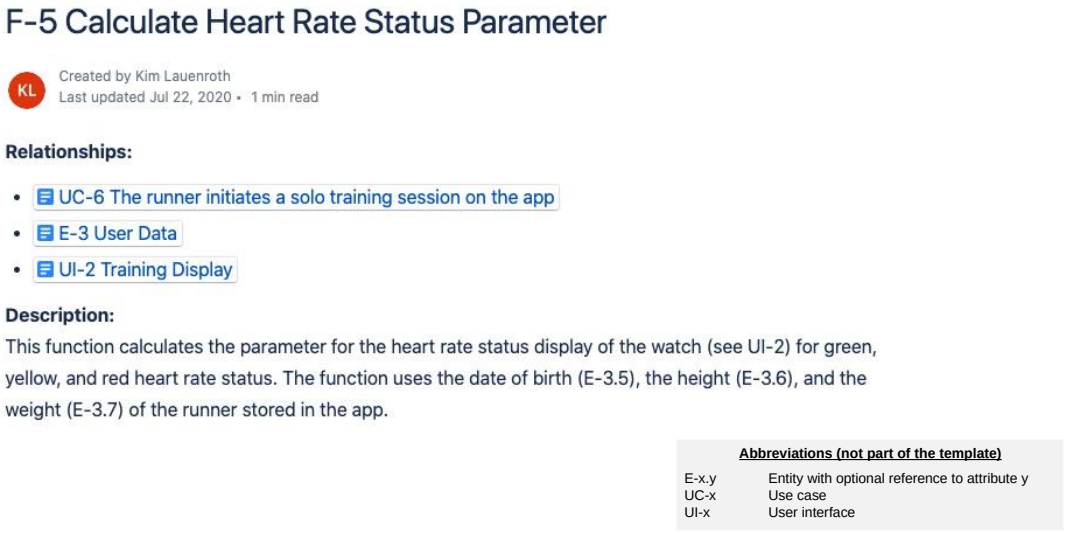
Like the other definitions, the definition of underlying function is rather abstract in order to cover a wide range of possible functions. In daily work, the term "underlying function" is usually used synonymously with the term function. In this handbook, we do the same to improve readability. When the difference is important, we use the full term "underlying function".

Advice for creating the description

There are several approaches for describing underlying functions of a digital system (e.g., UML activity diagrams, UML state machines, algebraic specifications). Figure 32 shows an exemplary function template for describing an underlying function from the YPRC case study, which shows a level of detail that is sufficient for the DDP at foundation level.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the design concept that is mentioned in the description and has a relationship to the underlying function described is mentioned.



F-5 Calculate Heart Rate Status Parameter

KL Created by Kim Lauenroth
Last updated Jul 22, 2020 • 1 min read

Relationships:

- UC-6 The runner initiates a solo training session on the app
- E-3 User Data
- UI-2 Training Display

Description:

This function calculates the parameter for the heart rate status display of the watch (see UI-2) for green, yellow, and red heart rate status. The function uses the date of birth (E-3.5), the height (E-3.6), and the weight (E-3.7) of the runner stored in the app.

Abbreviations (not part of the template)

E-x.y	Entity with optional reference to attribute y
UC-x	Use case
UI-x	User interface

Figure 32 – A function template

Considerations for daily work

Note that the text does not describe the concrete formula for the calculation. This formula is left out because is not necessary for the understanding of the example. It is sufficient to know that such a formula exists. It is added during the realization of the app. This example is intentionally presented to illustrate that a design concept does not have to be complete in an early stage. Certain information (such as the formula) may of course be added later.

2.2.5.6.7 Function: Use Case Template

We have left the use case template as the last template in our explanation because it serves two important purposes at the element level. First, the use case describes the interaction between the user and the element. Second[8], the use case combines all other elements of the element level into one coherent interaction flow.

Use case: A set of possible interactions between a user and an element of a system that provide a benefit for the user(s) involved.

[8] The second aspect is an extended understanding of traditional use cases. Use cases typically focus only on the user perspective. Within the holistic perspective of Digital Design, we prefer a broader understanding of use cases that includes and explicitly refers to other building blocks.

UC-6 The runner initiates a solo training session on the app

Created by Kim Lauenroth
Last updated just a moment ago • 1 min read

Relationships:

- E-3 User Data
- E-4 Training Status
- F-2 Record training data and average values during solo training
- F-5 Calculate heart rate status parameter
- SI-6 App-to-Watch Status Interface
- SI-7 Watch-to-App Status Interface
- SI-9 Transfer Training Data Interface
- U-1 Runner
- UC-2 The runner starts a training session with the watch
- UHI-4 Smartphone Clock Interface
- UI-4 Training App Start Screen
- UI-5 Training Status Screen
- UI-6 Training Unit Summary Screen

Main Scenario:

1. The runner (U-1) is viewing the start screen (UI-4).
2. The watch is connected (E-3.1) and the "Start Training" button is active.
3. The runner (U-1) selects the button "Start Training" (UI-4).
4. The app calculates the heart rate status parameter (F-5) and sends the parameter, the current data and time (UHI-4), and the training status "active" to the watch (SI-6).
5. The app shows the training status screen (UI-5) with the message "ready to start"
6. The app receives the signal "runner active" from the watch (see SI-7) and stores the status "active solo" in E-4.1. (see UC-2 of the runner's watch for details).
7. The app records the training data from the watch (F-2) and shows the training status on the screen (UI-5).
8. The runner (U-1) performs his training session. After a while, the app receives the status "runner inactive" from the watch (SI-7) and stores this status in E-4.1.
9. The app stops the recording of the training data (F-2).
10. The runner (U-1) decides to end the training and selects the "End Training" Button (UI-5).
11. The app sends the training status "inactive" to the watch (SI-6).
12. The app shows the training summary (UI-6).
13. The app transfers the training data to the portal (S-9).

Alternative Scenarios:

10a) The runner (U-1) decides to continue the training and the app receives a "runner active" signal from the watch. The status is stored in E-4.1.
11a) Continue with step 7) from main scenario.

Abbreviations (not part of the template)

E-x.y	Entity with optional reference to attribute y
F-x	Function
SI-x	Software interface
U-x	User
UC-x	Use case
UHI-x	Underlying hardware interface
UI-x	User interface

Figure 33 – A use case template

Advice for creating the description

Figure 33 shows an excerpt of a use case from the YPRC case study. The description consists of a main and one alternative scenario. This step-by-step textual description is common for use cases. Experts will be aware of further documentation techniques in the context of use cases (e.g., UML activity diagram, UML sequence diagrams, or the description of exceptional scenarios). These techniques are considered to be advanced level.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Every element from the design concept that is mentioned in the description and has a relationship to the use case described is mentioned. Use cases typically have several relationships because they integrate several aspects of a digital solution. In Section 2.2.6.1, we discuss the importance of use cases and their relationships in more detail.

Considerations for daily work

At foundation level, it is important to understand the step-by-step description and the importance of references to other elements of the software design concept. Note that almost every step of the example provides references to other elements (other use cases, entities, functions, user interfaces etc.).

With this way of referencing other elements, the use case templates become the most powerful template in terms of expressiveness and traceability. The use case templates put all the other elements into a situational context and describe how and when a certain part of the concept is important.

In other terms, the use case template, together with the function template, is that part of the software design concept where the DDP literally programs the digital system in a textual way.

Although the description might appear to be boring and easy, it is rather difficult to create good use cases, even at this simple level. The case study provides several examples that support learning to write use cases in the style presented.

2.2.5.7 Quality: Quality Requirement Template

Quality is an important aspect at the system level as well as at the element level. Describing quality is as demanding as describing goals. For each quality, we recommend writing a dedicated quality requirement.

For each quality requirement, a dedicated template is created. Figure 34 shows an exemplary quality requirement template with content from the YPRC case study. The headline provides the ID (QR for quality requirement), the number of the element (here, 2), and the title. The title of a quality requirement should be a one-sentence summary. Further details can follow in the description.

Advice for creating the description

The description of quality requirements is an important subject in requirements engineering (cf. [CPRE2020]). For beginners in Digital Design, we recommend describing the quality of the digital system with textual quality requirements. Beginners in writing quality requirements should follow the following rules:

1) Reference the element of the digital system that shall possess the quality described explicitly.
2) Avoid quality requirements that refer to the overall digital system.
3) If possible, describe the quality in a quantifiable way.

Advice for documenting relationships

The list of relationships can be derived directly from the description. Each element of the design concept that is mentioned in the description and has a relationship to the quality requirement described is mentioned.

Considerations for daily work

In Section 2.1.3, we presented a detailed discussion on the importance of quality for designing digital solutions. The quality requirement template is an important tool for beginners in Digital Design for documenting the desired qualities that an element or the whole system shall have.

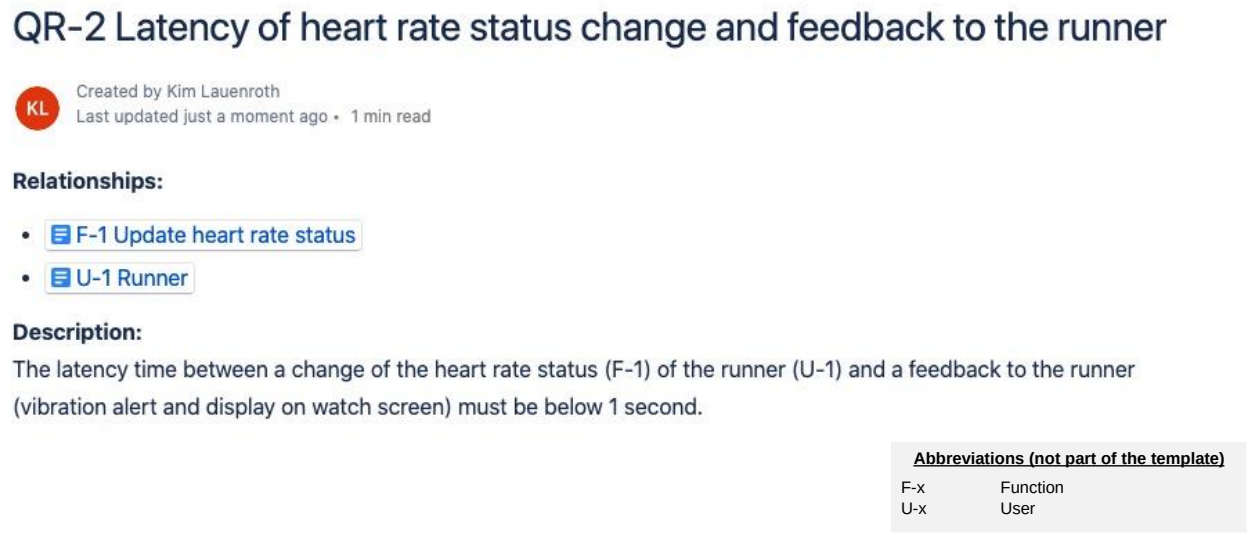


Figure 34 – A quality requirement template

2.2.5.8 Overview Pictures for Visualization Purposes

For beginners in Digital Design, we recommend creating an overview picture that introduces the overall form of the system or an element. Figure 35 shows the overview picture of the YPRC case study. It uses a very simple notation consisting of three elements:

- User symbol to represent the user
- Boxes to represent other elements of the form; the different colors refer to the existing elements and the elements under design
- Arrows to represent data flow/interaction between the elements

Although this notation is quite simple, it is sufficient to provide a good overview of the YPRC digital system.

In the early phases of the conceptual step, the overview picture can be used as the main artifact of work on the design of the digital system. The interaction arrows are sufficient to communicate the ideas of the function. Furthermore, different alternatives of the digital system can be elaborated by means of different overview pictures.

In later phases of the conceptual step, the overview picture becomes rather stable and can serve as a map that shows the overall form of the digital system. In order to fulfill this overview task, the overview picture requires constant maintenance during the whole lifecycle of the digital solution. Design is responsible for keeping the overview picture up to date.

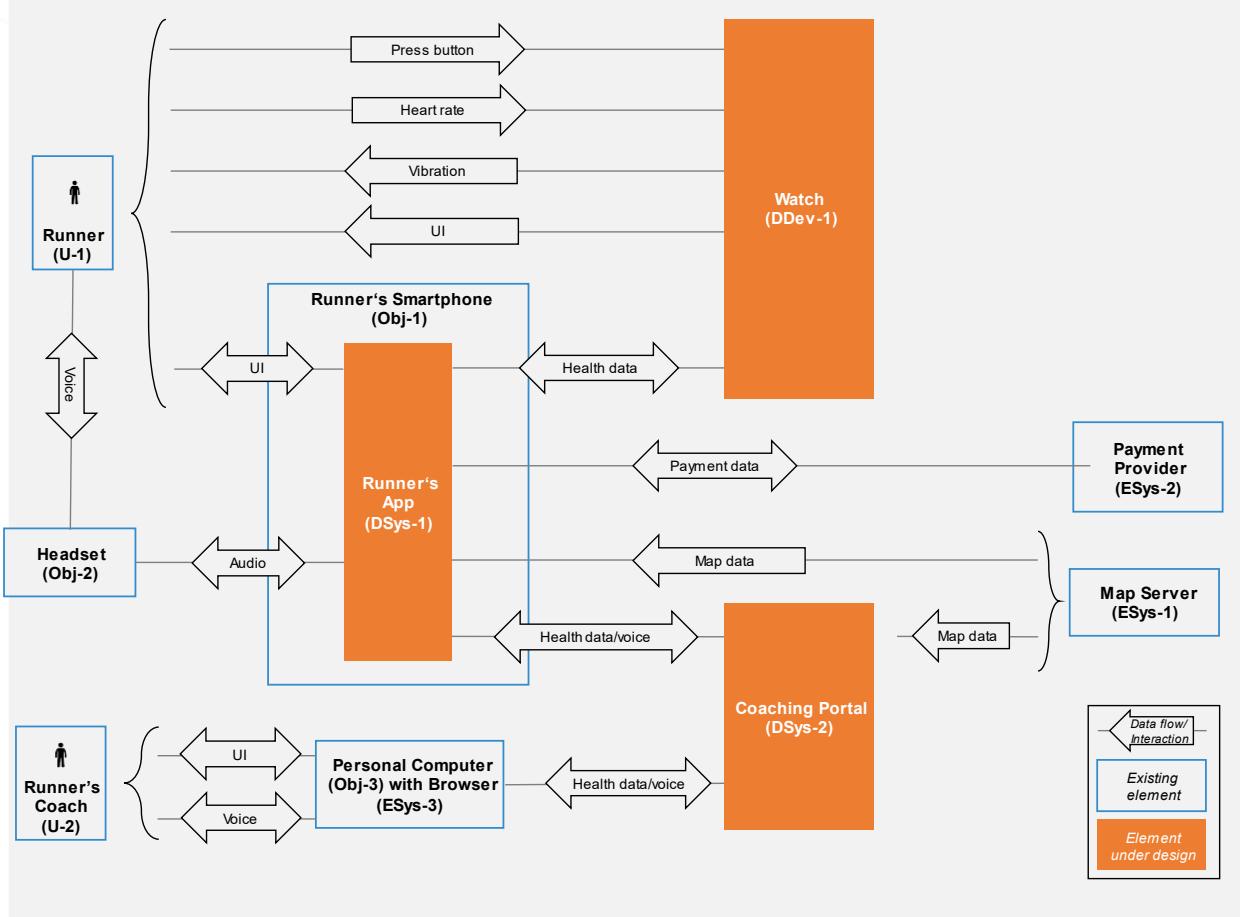


Figure 35 – Overview of the YPRC digital system with its elements

Experts will be aware of other techniques for creating an overview picture (e.g., UML use case diagrams). These techniques are of course also applicable. However, we consider these techniques as advanced level.

2.2.6 Putting Everything Back Together for the Big Picture

Up to now, we have presented document templates and documentation techniques for the solution level as well as the system and element levels. With this presentation, we have covered a lot of details. We assume that reading through this section for the first time might leave some beginners in Digital Design somewhat puzzled about all these details and their relationships.

With this last section, we want to put all the pieces together again in one big picture. We start with the element level and then work upward to the system level and solution level. Finally, we again discuss the stages and the activity areas of the building process from Chapter 1.

2.2.6.1 Relationships between Building Blocks at Element Level

In Section 2.2.5, we introduced different building block templates for working on the design of a particular element of a digital solution. Figure 36 shows these building blocks and their relationships to each other, including building blocks of the system level. The relationships have already been discussed together with the description of each building block. However, understanding the relationships between the building blocks is key to applying them in practice. Therefore, we summarize them again in this section.

For beginners in Digital Design at the element level, we suggest understanding the templates presented in Section 2.2.5 as building blocks in a literal sense. They can be used just like virtual Lego bricks to build up a particular element. Like real bricks, only certain types of bricks fit together. The relationships define which bricks can be combined. We describe them in the following.

Every element needs interfaces to interact with its environment

We start with the device that operates the software. This device can be an existing object (e.g., a smartphone or a computer) or a device that is built especially for the digital solution (e.g., the smartwatch from the YPRC case study). Each device must provide interfaces that allow interaction between the environment and the device. For this purpose, we have defined templates for perceivable or underlying hardware interfaces. When starting work on a new element design concept, we recommend defining the interfaces first. Typical candidates that most devices offer are displays, physical buttons, network connections, and input devices, such as physical buttons, a keyboard, or a touch screen (see Chapter 3 for more details).

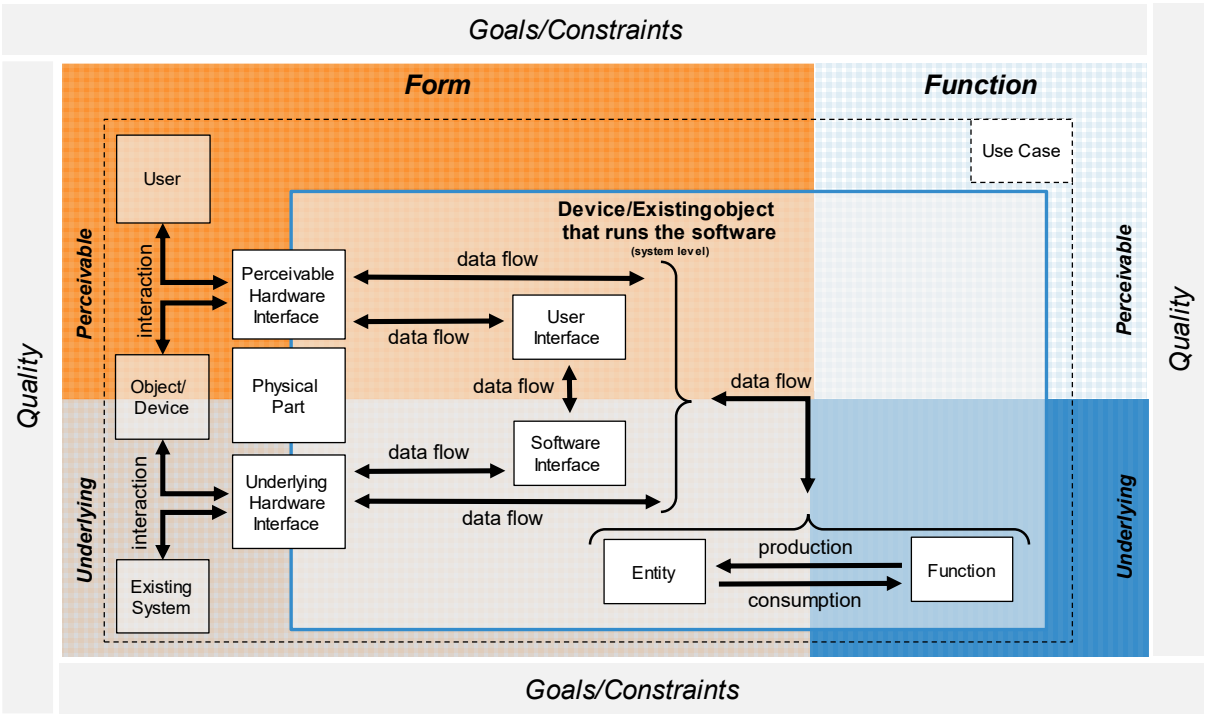


Figure 36 – Building blocks of a software/device design concept

Interfaces create data flows

If we look at interfaces from an element point of view, they can be understood as providers and consumers of data flows. The touchscreen of a smartphone is a good example. The screen describes the visual appearance of the user interface and consumes a flow of data that is visualized. A user interface also produces a data flow that represents the inputs from the user (e.g., touching a certain button or entering some data). Although this sounds quite technical from a design point of view, it is important to define this data flow at a proper abstraction level. In this situation, the assumption of perfect technology is useful. From a design perspective, we do not have to worry about how the screen works in detail to produce or consume the data flow. We only need to define the properties that the screen must have from a design perspective (e.g., the resolution, the color depth, etc.).

User interfaces as a special case of interfaces

Since the user interface is a special aspect of any digital solution, we have defined a dedicated building block for this purpose. With this building block, we describe the concrete visual form of the screen. Moreover, for the visual presentation of the user interface to the user, a dedicated hardware interface is necessary.

User interfaces also create a data flow

A user interface consists of some kind of static form (e.g., text, images) and of some kind of form that allows for interaction (e.g., input fields, text fields that show certain data). Here again, we have a data flow between further building blocks, namely entities and functions. Entities store data that can be visualized on a user interface and functions can produce data that can be visualized. This is a data flow from entities and functions to the user interface. Data flow is also possible in the other direction. A user can input certain data that is stored in an entity or is provided to a function for further calculations.

Entities and functions as the backbone of each element of a digital system

We have now worked our way from the hardware interface through the user interface down to the functions and entities. Both building blocks represent the backbone for describing what an element can actually do. Entities store the data that functions produce and consume. Therefore, the description of functions and entities goes hand in hand when designing an element of a digital solution.

Software interfaces are the counterpart of user interfaces for the underlying form

So far, we have discussed the user interface. Since interaction and connectivity are core features of digitalization, software interfaces as part of the underlying form are of equal importance. With software interfaces, we describe technical connections between the actual element and other elements of the digital solution (e.g., interaction with a payment provider).

Like user interfaces, software interfaces require a hardware interface that allows the software interface to connect to other elements of the digital solution. In the simplest case, this hardware interface can be some kind of internet connection. Documenting the internet connection might seem superfluous and a kind of no-brainer. However, since it is a core feature, we recommend that you do not omit it.

The data flow of software interfaces relates to entities and functions in the same way as user interfaces.

Quality requirements define qualitative details

As already introduced in Section 2.1, quality is a cross-cutting aspect of every digital solution. Therefore, each building block of an element can be detailed with additional quality requirements.

Use cases for putting building blocks into a frame

So far, we have described building blocks that describe static aspects. The use case is a different type of building block. It defines the dynamic frame that puts the other building blocks of an element into action by means of main scenarios and alternative scenarios.

Figure 37 illustrates this framing with an example from the YPRC runner's app: the creation of a new user account.

The left side of this figure shows the main scenario of use case UC-8. The elements referenced are shown as icons. The use case starts with presenting the welcome screen. The reference points to UI-8, which is the user interface building block that shows the form of the welcome screen. Creating a new account requires the user to push the *Register now* button. This information is given in UI-8 and does not necessarily have to be mentioned in the use case. In the next step, the app shows the registration screen UI-7.

The layout prototype of UI-7 is shown on the right-hand side of the figure as an example of a user interface. Here, we can see further relationships that we have described above: UI-7 has a relationship to E-3, the entity that describes the user data stored in the app. The figure shows an excerpt of the table with three attributes: date of birth, height, and weight. These three attributes are also present on the UI screen since the user has to enter them. In addition, UI-7 has a reference to the hardware interface PHI-4, which represents the smartphone touchscreen.

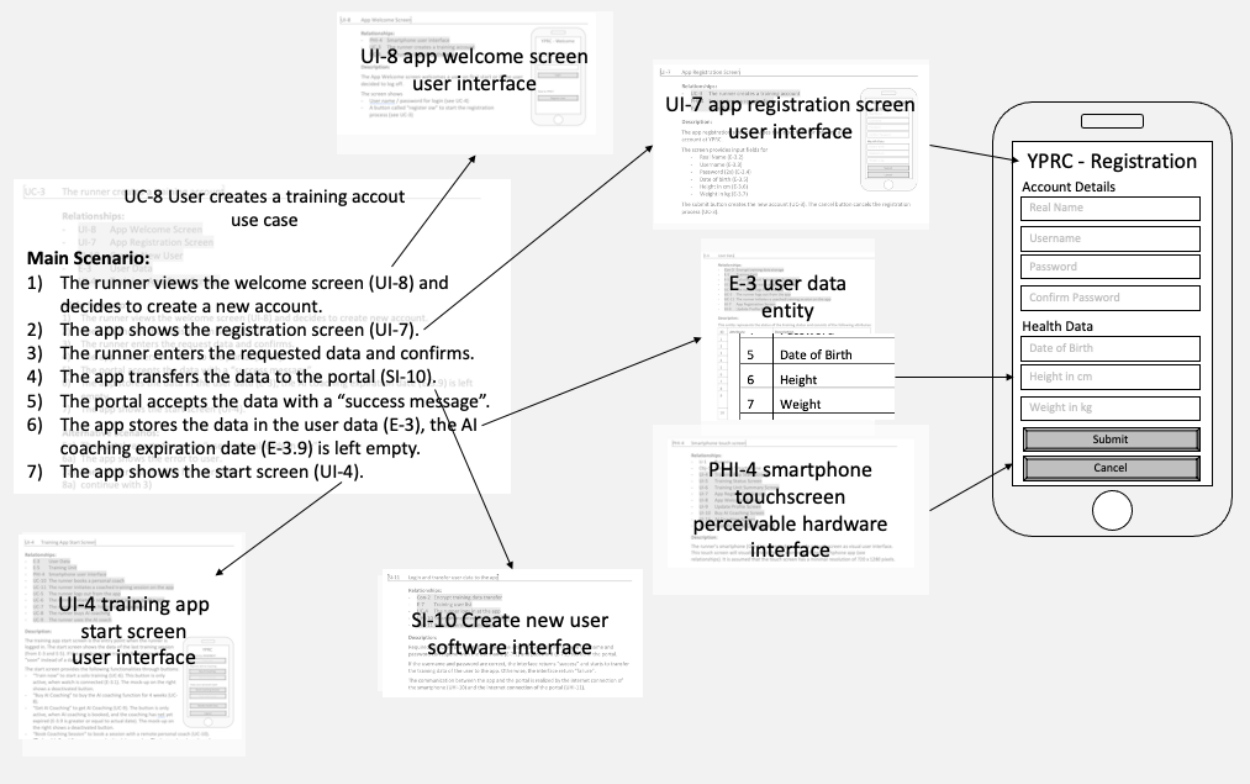


Figure 37 – Example of a use case framing other elements

When the user enters the data and confirms the input (step 3 of UC-8), the app uses a software interface (SI-10) to transfer the data to the portal to create a new user account. If this action is successful, the user data is stored in the app (a reference to E-3) with one exception: the coaching expiration date (the meaning of this can be found in the details of the case study). Once the data has been stored, the use case defines that the app shows the start screen (UI-4). This is the end of the main scenario.

Although the description of the use case itself is very brief and lean, the use case provides a rich description of the registration process together with the elements referenced and thereby serves as a frame that puts several elements into a coherent description of the element.

2.2.6.2 Relationships between Building Blocks at System Level

The larger perspective in terms of relationships is then created at the system level. Figure 38 provides an overview of the building blocks of a system design concept. To describe the form of the digital system, we start with the building block *user*.Here, we again see elements that create the form of the digital system (user, object, existing system, device, and software element). These elements interact with each other. In order to describe the function of the overall digital system, we have seen textual scenarios that describe an exemplary function of the system.

The description of scenarios serves a similar purpose to the use cases at the element level. Nevertheless, the scenario is only an exemplary flow. Therefore, the scenarios are only a partial description of the function of the whole system including the user and its situation. This partial description is the purpose of the scenarios at system level and is not a drawback since the important details are covered by the use cases at the element level.

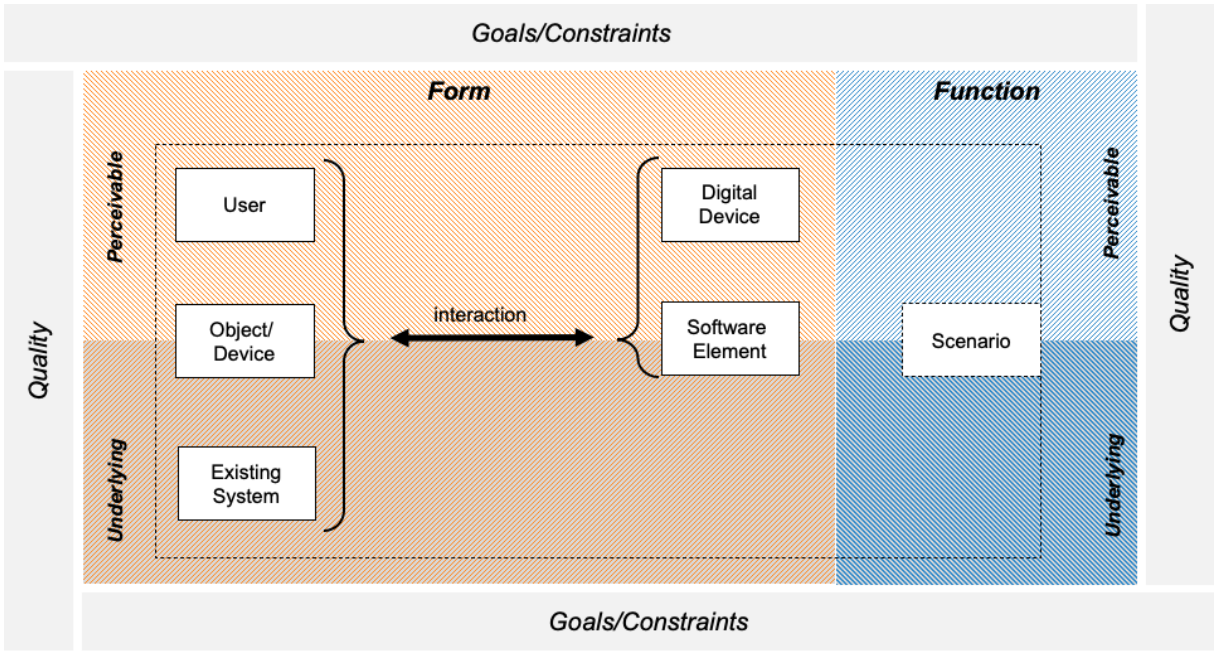


Figure 38 – Building blocks of a system design concept

2.2.6.3 Relationship between Element Level and System Level

We have already mentioned the relationship between element level and system level from the perspective of the element in Section 2.2.6.1. When looking at the relationships between both levels in general, there are two important lessons.

First, a number of building blocks at the element level can be derived from the system level. The relationships between two elements at the system level will result in two interfaces at the element level. Furthermore, the function described by a scenario at the system level will lead to one or more use cases and functions for the different elements at the element level. Second, consistency between the element level and the system level requires the details at the element level to not contradict the description of the overall system. If an element defines an interface to another element, this relationship must also be present at the system level, otherwise we have a contradiction. In the same way, goals and constraints at element level and system level must not contradict each other.

Table 9 gives an overview of the important dependencies between the system level and the element level.

Table 9 – Relationships between system level and element level

Level	Form	Function	Quality
System	Elements of the system, including relationships, describe the static structure.	Scenarios describe the function in an exemplary way.	Quality requirements that the system must fulfill
Element	Every relationship from the system level must be captured by interfaces. All relationships at the element level must be defined at the system level.	The exemplary function from the system level must be covered by the use cases. The use cases must not contradict the scenarios at the system level. Additional behaviors beyond the scenarios at the system level are necessary.	Quality requirements for an element can be derived from the system level. Quality requirements defined for an element must not contradict the system quality requirements. Additional quality requirements for an element are possible.

2.2.6.4 Relationship between Solution Level and System/Element Level

Looking at the YPRC case study, the difference between the design concepts at the solution level and the system/element level is substantial. The solution level is about value propositions, visions, and business models, whereas the system/element level comprises functions, interfaces, entities, and quality requirements.

Table 10 – Examples of relationships between solution level and system/element level

	Example customer/user	**Example value**	**Example revenue**	**Example cost**
Solution level	Customer groups of the digital solution	Value proposition for customer	Revenue stream from digital payment	Cost structure must include operation cost for a data center.
System level	User types consistent with customer groups	Scenarios illustrate the value proposition as examples.	Payment provider must be an element of the system. A scenario should illustrate the revenue stream.	At least one element of the digital system is operated by a data center.
Element level	Technical interfaces and user interfaces in line with the needs of the user types	Complete realization of value proposition by means of use cases and functions	At least one element must provide an interface to the payment provider. At least one use case must detail the payment.	The element in the data center communicates with other elements through a network connection.

Closing this gap is the task of Digital Design. There is no clear rule set or algorithm that allows a system design concept to be derived from a solution design concept. It is the creativity of the people involved (see Section 2.1) that imagine a digital solution and the underlying digital system. There are several ways of realizing a value proposition and at the same time, a certain technology may provide various possibilities for creating value. A good DDP will have the strength to bridge this gap with creative ideas for solutions and systems. The only guidance we can give here is that the concepts at both levels must not contradict each other.

Table 10 shows some exemplary relationships between the solution level and the system/element level to illustrate the complexity.

2.2.6.5 Relationship between Design Concepts, Activity Areas, and Steps of the Building Process

In Chapter 1, we introduced the design concept and the realization concept as general work products of the activity areas design and construction. In Section 2.1, we presented three essential steps of the building process and introduced more dedicated types of design concepts for each step of the building process.

Table 11 – Concept types in Digital Design

Type	Focus/purpose	Context
Digital Design brief	The focus of the Digital Design brief is the project that is intended to create the digital solution. Its purpose is to clarify the project details for all project participants and other relevant stakeholders.	The context of the Digital Design brief is the organization that is driving the project and the potential markets or domains that will be addressed by the digital solution.
Solution design concept	The focus of the solution design concept is the overall digital solution. Its purpose is to define and communicate the overall idea of the digital solution (including the business model) to relevant stakeholders.	The context of the solution design concept is the market/domain that has been chosen for the particular digital solution.
System design concept	The focus of the system design concept is the system that realizes the digital solution. Its purpose is to define and structure the digital system in terms of elements that allow efficient realization.	The context of the system design concept is those elements (users and other systems) of the market/domain that directly interact with the digital system.
Software design concept	The focus of a software design concept is a particular software element (e.g., an app or a web portal). Its purpose is to provide all details that are necessary to realize the software described.	The context of a software design concept is the environment in which the software will be used (including the user) and the device that is used to operate the software element.
Device design concept	The focus of a device design concept is a particular device (e.g., the smartwatch). Its purpose is to provide all details that are necessary to realize the device described.	The context of a device design concept is the environment in which the device will be used (including the user).

The power of the concepts introduced is that they allow you to structure the large amount of information that is necessary to design a digital solution down to the details of the elements. For this purpose, each type of concept in Digital Design has a special focus and describes a dedicated

aspect of the digital solution. In order for the concept to be understood properly, you have to provide sufficient details on the context in which a concept has been created.

Table 11 gives an overview of all concept types, including their focus and their context.

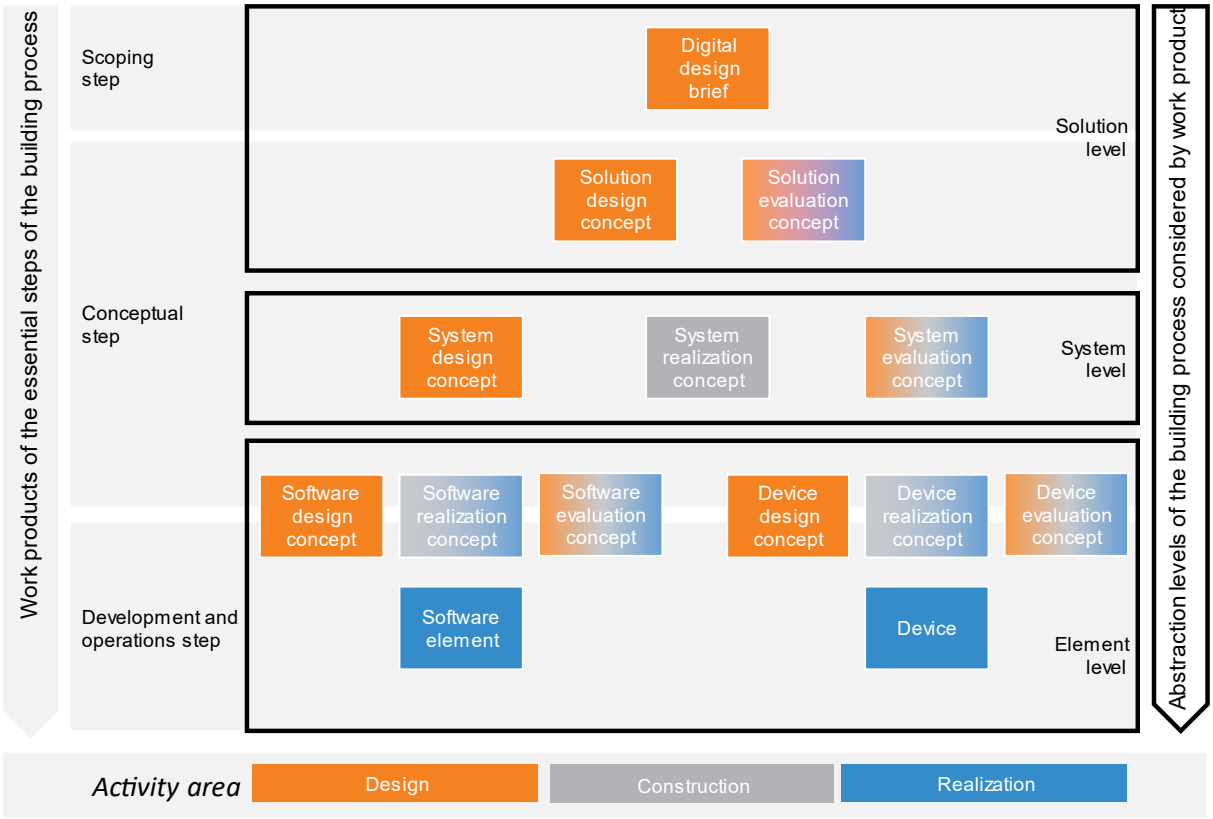


Figure 39 – Overview of design concept types in the different stages of the building process

Figure 39 presents an integrated visualization of steps, work products, and levels. The figure shows each design concept in its main step of the building process and also the relationships between the levels and the design concept. For example, the Digital Design brief belongs to the solution level and is mainly created in the scoping step. The solution design concept also belongs to the solution level, whereas the system design and realization concepts belong to the system level. Both concepts are mainly created in the conceptual step.

It is important to remember that the definition of a system and its elements is a matter of decision (see Chapter 1). For complicated systems with many elements (for example, the software landscape of a large company easily consists of more than 15 different elements), it is advisable to consider defining dedicated subsystems that are considered as joint elements. However, we consider this multi-level system structure to be an advanced topic.

This assignment to steps and levels does not mean that the concepts are done after the particular step is finished. It only means that a first version of the particular design (and realization concept) is ready after the step. The various relationships described above show that it is beneficial to work continuously and iteratively on all design concepts and to modify them if necessary.

The lines between the different design (and technical) concepts indicate the main consistency dependencies. For example, the solution design concept must be consistent with the system design concept. Other transitive relationships are possible. The element design concepts are placed between the conceptual step and the development/operations step. The main reason for

this is that some process models already create partial element design concepts prior to the start of development work.

Figure 39 also shows the different design, realization, and evaluation concept types. Creating and maintaining these concepts requires additional competences in the activity areas construction and realization. The different colors of these boxes indicate the contributions of the activity areas.

2.2.7 Conclusion: Learning Conceptual Work

In this section, we have provided a detailed overview of conceptual work in Digital Design. We have introduced a large number of different concept types and techniques. The goal of this section was to introduce the variety of conceptual work and the main relationships between the different design concepts and their building blocks. A summary of these relationships was given in the previous Section 2.2.6.

We do not expect that reading this section is sufficient to become an expert in conceptual work. The intention of the section was to provide a starting point for working with design concepts. The supplemental materials of this handbook, especially the YPRC case study, provide additional material for learning conceptual work.

Furthermore, we hope that readers who already have experience in the field related to Digital Design will find the broad overview useful in understanding the relationships of their preferred concepts with the view of Digital Design on conceptual work.

2.3 Application of Prototypes in Digital Design

This section looks at how prototypes are applied in Digital Design. It starts with the fundamentals of prototypes in terms of a definition (Section 2.3.1), objectives (Section 2.3.2), and exemplary applications of prototypes in different disciplines (Section 2.3.3).

Section 2.3.4 provides a broad overview of the different types of prototypes and Section 2.3.5 introduces different tools for creating prototypes. Section 2.3.6 introduces paper prototypes based on an example as a ready-to-use and simple technique for beginners. Section 2.3.7 closes this section with a conclusion on the application of prototypes in Digital Design.

2.3.1 Definitions of Prototype

Building a digital solution may cover a wide range of disciplines and professions (see Section 1.3) and Digital Design involves concepts, methods, and techniques from different disciplines—for example, industrial design, interaction design, human-computer interaction, software architecture, and requirements engineering. To ensure unambiguous communication between experts from these different areas, it is important for the DDP at foundation level to understand that there is no one, unique and generally accepted definition of what a prototype is. Experts and stakeholders might all have a different understanding of the term prototype because they have a different background from their respective discipline.

For example, industrial designers use the word *prototype* for a pre-version of a mass production build. On the other hand, they create many early versions of the product in the form of sketches, (computer) models, or simple mock-ups to improve designs—experts from other fields would call these prototypes. Interaction designers use sketches, interactive mock-ups, user flows, wireframe prototypes, or paper prototypes to represent preliminary versions of a (graphical) user interface. These preliminary versions can be explored with and by stakeholders. Software designers or

software architects use early implementations of a certain part of the digital solution (a functional prototype) to study the feasibility of this particular software function.

In order to reduce communication problems with the other disciplines involved in building a digital solution, the following definition provides a broad understanding of the term prototype for use in the context of Digital Design and building digital solutions.

Prototype: A preliminary, partial instance of a design solution.

Prototypes can be used as

1) A manifestation of an idea for a future digital solution in a format that communicates the idea to others or that can be tested with customers or users

2) A model for later stages or for the final, complete version of the digital solution

3) A means of obtaining early feedback on a concept by providing a working model of the expected digital solution before actually building it

Based on [McEl2017] and [IEEE2017]

The use of a prototype or prototypes is also referred to as *prototyping*. As stated in [Dick2019], the term *prototype*—the object—is sometimes used when *prototyping*—the process of using a prototype—is meant.

The wide view on prototypes as defined above may tempt us to see a Digital Design concept itself as a prototype. This might be valid in the abstract sense. However, the definition above links this manifestation to a dedicated objective, such as communication with team members or testing with customers or users. Such testing in particular is difficult to do with a specification—for example, with only the Digital Design concept. As another example, it is hard to test the idea of the future digital solution with users when the idea is presented as a list of user stories in a spreadsheet. Such purposes require a very concrete version of the early digital solution, system, or element. Such a concrete prototype object can be either tangible or intangible—for example, it can be an interactive mock-up that enables the customer or user to interact with parts of the planned digital solution. Examples of intangible prototypes are simple sketches or storyboards that show the (early) ideas much more concretely than an abstract text of a specification can. To emphasize the context and the planned customer or user experience, short narratives in the form of storyboards or fictional (commercial) videos may provide clarity on the digital solution envisioned. See Section 2.3.2 for a discussion on prototyping rationales and objectives and Section 2.3.4.5 for typical names of prototypes used in Digital Design.

2.3.2 Objectives of Prototypes

The use of prototypes is the key technique that enables the DDP to achieve several (partly overlapping) objectives (cf. [McEl2017]):

- Explore the problem, user needs, and requirements
- Communicate solution ideas and concepts
- Test and improve concepts and solution ideas
- Advocate a solution or a solution idea

In all of these cases, the creation of prototypes supports the iteration of problems, requirements, concepts, solution ideas, and solutions. The DDP learns from building a prototype and can use it

to gather feedback from stakeholders and to achieve a subsequent improvement based on this feedback. Such iteration cycles are essential parts of every building process (see Chapter 1 and Section 2.1).

Creating a prototype takes a certain amount of effort. However, this effort is well spent, as the feedback gathered helps you to base decisions on more information and thus make better decisions. When you explore solution ideas in many fundamentally different directions using prototypes, many of these ideas will fail but will also generate new ideas for better solutions. This means that the DDP must be ready to create a prototype for a single purpose only and discard it afterwards. Therefore, the DDP should choose the scope of the prototype and effort for creating it carefully.

Using this power of iterative prototyping can make it easier to find innovative and excellent concepts and solutions and it significantly increases the probability that a solution idea becomes a real innovative product or service.

The objectives listed above are discussed in detail in the following sections. Some of the content is based on [McEl2017].

2.3.2.1 Exploring the Problem, User Needs, and Requirements

The main aim of problem exploration is understanding the user's point of view in relation to the digital solution being created. Prototypes are a good *thinking tool* for exploring the root causes of the user's problems or the core user's needs. They also help you to explore the scope of a potential digital solution based on available or new technology. It is often when exploring this through early prototypes that a more comprehensive and different problem appears than initially assumed.

YPRC example. In the early phase of developing the YPRC digital solution, the initial assumption might be that the remote coach wants to see the current pulse rate of the runner so that they can give the runner training recommendations. Sketching early drafts of the display screens of the user interface allows the planned future digital solution to be discussed with professional running coaches. The result of such a discussion might be that only the changes in the pulse during a certain period of time and not the current value are important. Moreover, the coach must know the resting pulse rate of the runner and the maximum values from previous training sessions. This insight changes the user interface for the coach and in turn, major underlying parts of the digital solution.

Another example based on the YPRC case study might be the initial assumption that the runner must be able to see their position on the route on a map. User studies with runners based on paper prototypes (see Section 2.3.4.5 and Section 2.3.6) might show that they generally know their running route and that they only need the map view on the smartphone for new routes. Therefore, they mostly carry their smartphone in a pocket or mounted on their upper arm. It may also be that runners would like to be informed about leaving a pre-defined pulse rate range through another information channel, such as a vibration or acoustic alert.

The examples above show how prototypes help to illuminate alternative problems in the problem domain. The resolution of each problem ultimately leads to a better digital solution. As this might challenge and potentially change the fundamental concept of a digital solution, it is very important to find such insights at the beginning of the building process (see Section 2.1).

Considerations for daily work

After understanding the problem, the DDP is better equipped to create a real innovation and based on this, to create early design concepts. This understanding is the starting point for the development of solution variants for the problem. Prototypes are very valuable for iterative exploration of solutions. Once such prototypical solutions—which might represent fundamentally different solution approaches—have been created, testing with users should take place in order to compare these alternatives (see Section 2.3.2.3). This comparison and selection of the best alternative should take place once an initial concept is available but nevertheless at an early stage in the building process.

Prototypes can support understanding business strategies in the context of competitive digital solutions, the direction of product portfolios, and the goal of the target users. Prototypes can transform a business direction into a tangible or at least visible object, allowing presentations to and discussion with business stakeholders to become very concrete (see Section 2.3.2.4). Discussions based on such concrete examples are very effective and lead to improvement ideas for the next iteration cycle. This form of prototyping is applicable to all stages of the building process.

Later in the building process, prototypes help to shape the user experience and interactive elements of the user interface. Presentations and testing of such solution manifestations support the understanding of user flows and the discovery of new design aspects.

Understanding the user in general and the user interaction with the digital solution is helpful in all stages of the building process. However, the most crucial part is the understanding of the user needs, which is also included in the first part of the human-centered design process (see [McEl2017], Section 2.1 and [ISO2019]). Moreover, this process not only covers user needs but can also address the needs of other stakeholders, such as clients and customers of the digital solution. Once central problems are clear and a good solution has been developed for these core needs, the digital solution will probably be a success on the market.

New technological developments might lead to digital solutions that no stakeholder has previously envisioned. This can be addressed by the DDP using the dual-mode model of design (see Section 2.1.1.2). Building prototypes of solution ideas that emerge from new technological developments and presenting them to users and other stakeholders helps you to discover and inspire new demands and innovative digital solutions.

2.3.2.2 Communicating Solution Ideas and Concepts

Prototypes manifest thoughts into a physical or digital medium. They transform fuzzy or generalized thoughts into a concrete (example) object. Such an object can be a tangible physical object, like a paper prototype, or an intangible object, like a story map or storyboard (see Section 2.3.4.5). These objects are useful as a basis for communication between and among all stakeholder groups, such as customers, teammates, developers, test experts, and project managers. With a prototype as a concrete discussion base, meetings with these stakeholders become more effective and efficient, especially when the attendees can give concrete feedback on the prototype. By applying this feedback loop iteratively, improvement ideas become more and more clear, and the understanding of all persons involved increases with each iteration.

In general, choosing an appropriate category of prototype (see Section 2.3.4) depends on the communication partners and the goal of the specific meeting. For communication with expert

colleagues, low-fidelity prototypes are often sufficient, as these colleagues share similar mental models. More high-fidelity prototypes are appropriate when feedback on visual designs or complex interactions is required. This takes place mostly in later stages of the building process. Iterations with expert colleagues are useful in all process stages but are especially valuable in later stages, to avoid the DDP focusing too much on their own solutions and own thinking.

Considerations for daily work

When communicating with stakeholders who are responsible for the project, such as project and product managers or customers, the choice of prototype depends on the different meeting objective in the different process phases. It also depends on the current level of understanding of the meeting participants and their ability to imagine the digital solution envisioned. Visual or tangible elements in the form of a prototype can help to develop a better understanding. Moreover, short narratives in the form of storyboards or fictious (commercial) videos (see Section 2.3.4.5) may provide clarity on the vision behind the planned digital solution. As a general guide, in early building process phases, low-fidelity prototypes are useful for early alignment and approval of user flows, use cases, and features. Later in the process, the fidelity level can increase in parallel with the level of maturity of the digital solution. There is a risk that the fidelity level may be too high, such that decision makers or customers might think the digital solution development is already complete. To bring about a decision on specific details later in the process, an intermediate fidelity level (mid-fidelity) for the prototypes or mixed-fidelity prototypes is the right choice.

Software developers or manufacturers who shall produce (a part of) the digital solution need detailed specifications or style guides to scope their work. High-fidelity prototypes can supplement these documents. Presenting prototypes in order to convince future investors or collaborators helps to increase the confidence in the digital solution idea or the Digital Design concept.

2.3.2.3 Testing and Improving Concepts and Solution Ideas

A core use of prototypes is testing the improvement of (parts of the) Digital Design concept and digital solution. They are very useful for feasibility studies. Iterative user-based tests with these prototypes help to ensure a solution is working and help to gather improvement ideas. Even the validation or invalidation of small assumptions—at least with a small number of users—is valuable to bring the building process forward. Technical evaluation of the feasibility or quality assurance is also possible. Such prototypes are most likely developed for single use only (throwaway prototypes), which in most cases is worth the effort if the resulting digital solution becomes successful.

2.3.2.4 Advocating a Solution or Solution Idea

Prototypes are ideal for advocating certain design routes or solutions. Together with insights from previous tests, product managers, business stakeholders, and project managers can be convinced of the right direction for the Digital Design concept or the digital solution. In large project teams, advocating is necessary to provide motivation for certain development solutions that deviate from the initial ideas of the team. Having the right data from iterative (user) testing helps to convince the team.

Incorporating decision makers or developers into the iterative process of giving feedback on alternative prototypical design routes or solutions is valuable for the digital solution as well to convince these stakeholders of these directions or solutions. Moreover, stakeholders should have

the opportunity to observe testing sessions with the digital solution to see for themselves whether solutions work or not. This usually leads to a better understanding for certain design routes.

Considerations for daily work

Based on experience from software development projects, there is, however, a tendency in some projects to start implementing software too early, which often fixes a concrete (part of the) digital solution at early stages of the project. Observations from such projects also show the tendency to create click prototypes that are too sophisticated—for example, with a fidelity level that is too high too early in the process. This usually limits the exploration of (fundamentally) different concept or solution directions because designers and developers hesitate to discard work they have put substantial effort into. In addition, prototypes with a fidelity level that is too high focus the viewer or user of this prototype too much on small details that are usually not in focus at the beginning of the project. This limits the possibility of discussing or evaluating the overall product concept because the discussion partners are truly distracted by details.

2.3.3 Examples of Using Prototypes in Different Disciplines

Prototypes play an important role in various disciplines. The following list presents examples from the digital industry and other disciplines outside the digital domain.

- Building architects, for example, work with floor plans, airflow models for ventilation, heating, and cooling, daylight simulations to optimize the incidence of light through windows, material studies, aesthetic models, and building simulations, where users can walk through the planned building (see [McEl2017]).
- Industrial designers have a long tradition in extensive use of prototypes. They use sketches (e.g., created by real or digital pencils), foam models, or models from additive manufacturing (e.g., 3D printing); they conduct material studies, use aesthetic models, make scaled mock-ups, and create final forms as prototypes before communicating the design result to manufacturing (see [McEl2017]).
- Filmmakers use storyboards and previews to visualize the flow of a movie before actually filming the scenes.
- When developing electronic devices, designers usually create industrial designs and electronic designs in parallel. The placement of electronic components influences the industrial design. Prototypes consisting of selected and assembled electronic components are useful for studying the implications for and optimizing the industrial design (see [McEl2017]).
- Interaction designers who develop user interfaces of software applications use prototypes such as sketches, wireframes, coded prototypes, and visual designs to improve a solution iteratively (see [McEl2017]).
- Software architects and software developers use coded pieces of software as functional prototypes to explore feasibility, verify requirements, or study certain software quality aspects of alternative software solutions.

The examples above show that creating and using prototypes is a common and successful technique in various disciplines. The examples are meant as inspiration and motivation for the DDP to create and use various types and categories of prototypes—which do not necessarily have to be a prototype in a digital format—to build digital solutions.

2.3.4 Criteria for Categorizing Prototypes

There are several approaches for categorizing prototypes. Studying such classifications helps the DDP to understand the broad scope of prototypes and helps with the selection of an appropriate prototype in a given situation and at a specific process phase.

The left-hand column of Table 12 contains an overview of criteria for classifying prototypes that are discussed in more detail in the following sections. The right-hand column of this table shows the categories based on the criteria in the left-hand column. The table is discussed in detail below. Section 2.3.4.6 concludes by presenting a mapping between categories and building process steps.

Table 12 – Criteria for categorizing prototypes

Criterion for categorization	Categories
Level of interaction	*horizontal, vertical*
Goal of prototyping	*exploratory, experimental, evolutionary*
Level of fidelity	*low, mid, high*
Level of fidelity for each dimension of a prototype (fidelity profile)	
Sensory refinement	*low, mid, high*
Breadth of functionality	*low, mid, high*
Depth of functionality	*low, mid, high*
Richness of interactivity	*low, mid, high*
Richness of data model	*low, mid, high*

2.3.4.1 Level of Interaction

A very simple but often-used categorization of prototypes according to their level of interaction distinguishes between horizontal and vertical prototypes. Horizontal prototypes consist of a comprehensive user interface with little or no functionality. Vertical prototypes exhibit a subset of the target functionality in depth but do not cover many functions or the complete user interface.

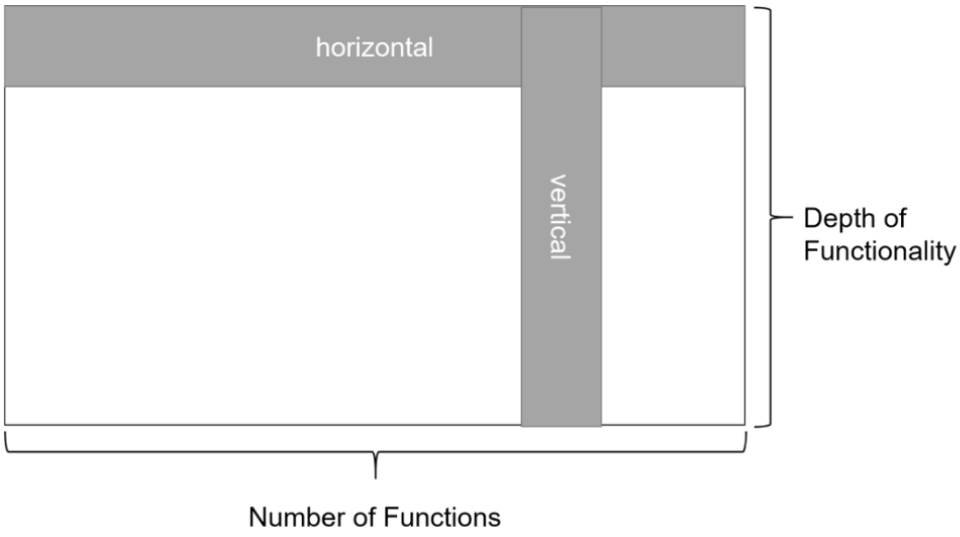


Figure 40 – Level of interaction

Figure 40 visualizes the correlation of these prototype categories with their number of functions and the depth of functionality. This often-used categorization is covered by the categorization based on the five dimensions of prototypes (fidelity profile) presented in Section 2.3.4.4.

2.3.4.2 Goal of Prototyping

Floyd [Floy1984] distinguishes between prototyping for exploration, experimentation, and evolution depending on the goal of prototyping within the given project situation.

Exploratory prototypes support the communication between users and developers of the target system in early phases of the building process. They are useful for collecting ideas, requirements elicitation, and validation. Moreover, visualizing requirements through such prototypes makes the functions of the target system transparent. Exploratory prototyping is useful in early phases of the development process and the focus lies on the exploration of (fundamentally) different alternative concepts and solutions. Such prototypes are often used only for these purposes (throwaway prototypes) and are horizontal prototypes (see [Floy1984]).

Experimental prototypes support the evaluation of the feasibility or usefulness of a certain solution before considerable effort is put into realizing the digital solution. Verifying ideas (for realization) provides experience for a later implementation. Experimental prototyping is useful in early phases of the building process after an initial specification, such as the design brief or an initial concept (see Section 2.2.3), has been written. Such prototypes improve the specification of the target system by either complementing the specification, refining (a part of) the specification, or by serving as an intermediate step between concept creation and realization. Such prototypes are also mostly throwaway prototypes and vertical prototypes (see [Floy1984]).

However, if new ideas, requirements, or features arise later in the building process, it might be worth creating exploratory or experimental prototypes before starting the actual construction or realization of these items.

The aim of evolutionary prototypes is continuous development and further development of prototypes together with the target digital solution. At some point at the end of the realization process, the prototypes are integrated into the final digital solution. If only one prototype is used, it becomes the final digital solution. Using these prototypes for testing purposes as well allows continuous validation of the digital solution under development. By not discarding the prototype or the prototypes but instead integrating them into the final system, they become an inherent part of the final digital solution. Floyd [Floy1984] already stated that an evolutionary prototyping approach is rather a *development in versions*. This approach breaks down a linear waterfall-like development process into successive iterative cycles of (re-)design, (re-)implementation, and (re-)evaluation (see [Floy1984]). Such iterations are the fundamental basis of today's modern agile development paradigms.

2.3.4.3 Level of Fidelity

One categorization of prototypes that is often employed uses the level of fidelity as a criterion. The level of fidelity of a prototype describes, on a continuous scale from low to high, how close the prototype is to the final digital solution. This applies to the visual representation as well as to the behavior of the prototype.

This simple categorization helps with the understanding of the general concept of characterizing a prototype. However, in many practical cases, it is more useful to apply the fidelity level to different dimensions of a prototype individually. A categorization based on five dimensions of

prototypes (fidelity profile) is presented in Section 2.3.4.4. This often-used categorization of prototypes based on the overall fidelity level is covered by the categorization based on the five dimensions. The remainder of this section focuses on a coarse view of the fidelity level of a prototype.

Low-fidelity prototypes are very far away from the final digital solution. Usually, a prototype of this category is in another medium, i.e., it uses different material to the final digital solution. This prototype usually has a limited number of functions and normally has no design visualizations. One example is paper prototypes, which—in their simplest form—are essentially drawings on pieces of paper showing, for example, some target smartphone screens. Low-fidelity prototypes are the easiest and most inexpensive prototype category, take the least time effort, and require only a low skill level to create. Low-fidelity prototypes are most useful for testing core concepts and exploring various alternative ideas. Moreover, they help to identify potential problems of the final digital solution at an early stage. Because low-fidelity prototypes are easy to create, this can be achieved quickly and most economically (see [McEl2017]).

Low-fidelity prototypes are good for testing and evaluating core assumptions or hypotheses about the digital solution. User interactions, information architectures, and mental models of the users can be investigated. When using this prototype category, the focus usually lies on the general use and flow of the digital solution. Examples of low-fidelity prototypes are sketches, paper prototypes, storyboards, wireframe prototypes, mood boards, and simple assemblies of electronic components (component prototype) (see [McEl2017]).

High-fidelity prototypes are at the other end of the fidelity scale and are very close to the final digital solution. Usually, a prototype of this category is in the final medium, i.e., the same material is used as for the digital solution. This type of prototype has high-quality visual designs and includes real content, such as product images in a prototype for a shop app or music files in a prototype for a music app. Most of the functionality and most of the interaction paths are available. As the level of detail is very high, the time and cost involved in creating a high-fidelity prototype are high. A high skill level in specialized disciplines is usually required to create such prototypes (see [McEl2017]), for example, skills in interaction design or software development.

High-fidelity prototypes are good for testing small details and for conducting a final overall evaluation of the digital solution shortly before the first release to the market. Icons, animations, detailed user flows, and the sizes of final buttons and keys can be investigated. This prototype category is also suitable for testing the legibility of type fonts and performing long-term studies on the wearability of (parts of) the digital solution. High-fidelity prototypes are useful for testing the durability or long-term stability of the physical part of the target digital solution. Examples of high-fidelity prototypes are an early pre-mass production version of a smartwatch (pre-production prototype), a fully coded smartphone application, or a completely visually designed digital experience (see [McEl2017]).

Sometimes, the term *mid-fidelity* is used to categorize prototypes. As the name suggests, mid-fidelity prototypes lie in many respects between low-fidelity and high-fidelity prototypes. Mid-fidelity prototypes look more similar to the final digital solution than low-fidelity prototypes but are still simpler than high-fidelity prototypes. Visual designs, more interaction paths, and more functionality are used. Such prototypes can already be in the final medium, i.e., the same material as for the final digital solution. Naturally, this means more effort than for the low-fidelity prototype but the target audience, such as users or customers, gets more context for evaluation. Costs and time effort can be optimized compared to the maximum utility of the prototype. Examples of mid-

fidelity prototypes are click prototypes potentially on target devices, style tiles, simple coded prototypes, and simple electronic prototypes (see [McEl2017]).

This categorization provides a coarse categorization of the overall fidelity level of prototypes. Although the levels *low*, *mid,* and *high* will suffice for most purposes, it is possible to refine this scale, for example, using a continuous scale based on numbers, such as between one and ten, to categorize the fidelity level with more granularity.

2.3.4.4 Level of Fidelity for each Dimension of a Prototype (Mixed-Fidelity Prototypes)

Section 2.3.4.3 addresses the (overall) fidelity level of the complete prototype. This categorization helps us to understand the general theoretical concept of low-fidelity and high-fidelity and to classify prototypes at a coarse level. However, in practice, it is often necessary to consider fidelity levels for dedicated dimensions of the prototype separately. This is especially the case when working in larger projects and in iterations.

To this end, we suggest considering the fidelity for the following five dimensions of a prototype: (1) sensory refinement, (2) breadth of functionality, (3) depth of functionality, (4) richness of interactivity, and (5) richness of data model. This is an extension of the model by McCurdy et al. [MCPKV2006], who consider *visual refinement* as the first dimension. Our model extends this idea with further sensory modalities, such as the auditory, haptic, or balance sense (vestibular sense) as well.

A prototype might have different levels of fidelity (on a scale between low to high) for each of these dimensions that define the fidelity profile of a certain prototype. As already mentioned in Section 2.3.4.3, the fidelity levels low, mid, and high serve most purposes. However, a continuous scale—for example, between one and ten—is possible as well to achieve a categorization with more granularity.

Sensory refinement

The fidelity level of the dimension sensory refinement refers to the quality of the sensory presentation of the prototype, such as the visual presentation, the auditory refinement, or the quality of the haptic output. A low-fidelity sensory presentation might be a hand-drawn sketch of the user interface elements such as simple box-shaped wireframes. On the high-fidelity end are pixel-accurate visuals or at least high-quality visualizations of the user interface that are close to the final digital solution.

YPRC example. One of the early prototypes created in the YPRC case study is the prototype for the coach perspective. The goal of this prototype is to evaluate whether it is possible to coach the runner with a voice connection. Table 13 shows the fidelity profile of this prototype.

Table 13 – Fidelity profile of the YPRC prototype for the coach's perspective with regard to the five dimensions of a prototype

Sensory refinement	Breadth of functionality	Depth of functionality	Richness of interactivity	Richness of data model
low	low	high	mid	low

Figure 41 presents different fidelity levels of the dimension sensory refinement by means of the visual elaboration of one display screen from the YPRC case study. Image a) in this figure shows a hand-sketched screen of a paper prototype as an example of a low-quality visual. Images b) and c) contain screens of a wireframe prototype of low and mid visual quality. Image d) is a high-quality drawing that can contribute to a high-fidelity sensory refinement.

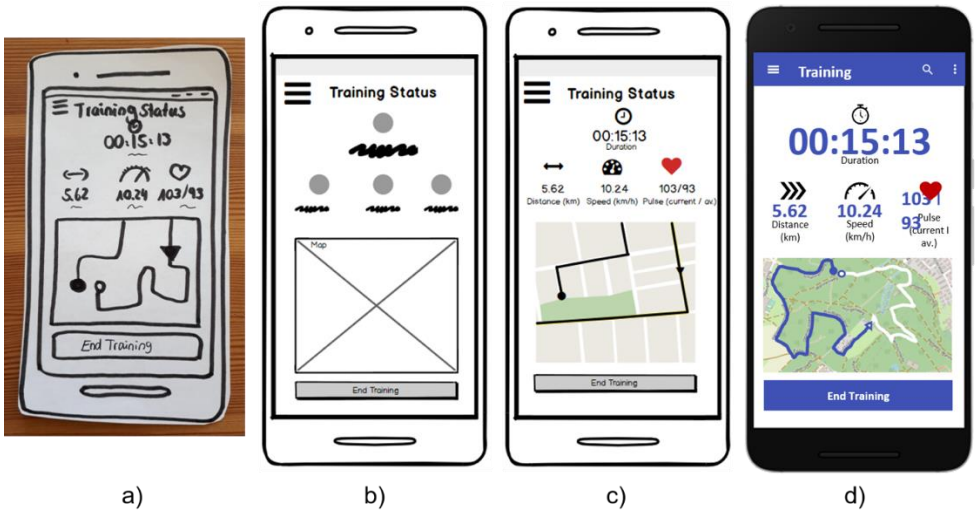


Figure 41 – One example display screen from prototypes of the YPRC case study showing different visual quality levels contributing to the fidelity level of the prototype dimension sensory refinement. Map within the rightmost image: © OpenStreetMap-Mitwirkende, license: www.openstreetmap.org/copyright.

Breadth of functionality

The fidelity level of the dimension breadth of functionality addresses the number of functions that are available in the prototype.

A prototype for the YPRC case study with a high fidelity of breadth of functionality has all planned functions available, such as connecting to the watch, the headphone, and the portal, monitoring the training status, making calls to the assistant, uploading the training data.

Depth of functionality

The fidelity level of the dimension depth of functionality defines the level of detail of one feature available in the prototype.

In the YPRC case study, a low-fidelity depth of functionality might be a simulated call function to the training assistant that simply shows one screen image indicating an established call. In contrast, a high-fidelity depth of functionality might offer all the functions for contacting the assistant, such as establishing a call to a person whose voice has been pre-recorded, uploading training data, receiving training recommendations. This example mainly addresses the perceivable functions of the digital solution and might work with dummy data only. Another form of a high-fidelity depth of functionality for the YPRC case study might be a fully functional recommendation system that employs a fully working artificial intelligence (AI) that suggests data based on the actual training data submitted by the user. This would (also) address the underlying parts of the digital solution. Another example from the YPRC case study is the prototyping project where an existing fitness tracker device with an open-source software interface provides training data such as the pulse rate. As a functional prototype, a smartphone app is programmed to gather this data and another PC app is programmed to receive this data and display it as text on the

console of the PC development environment. Although the level of sensory refinement is low, it covers the complete data transmission chain from the fitness tracker to the PC. It therefore addresses the complete underlying function in order to prove the feasibility of this dedicated function for the final solution. For this example, the depth of functionality can be characterized as high.

Horizontal prototypes have a high-fidelity breadth of functionality and a low-fidelity depth of functionality, whereas vertical prototypes have a low-fidelity breadth of functionality and a high-fidelity depth of (at least one) functionality. Therefore, the categorization of prototypes between horizontal and vertical prototypes (level of interactivity) presented in Section 2.3.4.1 is covered by the categorization according to the fidelity of the prototype dimensions presented in this section.

Richness of interactivity

The fidelity level of the dimension richness of interactivity addresses the simulated precision of the interactive elements—for example, how accurately the transition from one screen to another is implemented and how close the response times to user input are to the final digital solution.

In the YPRC case study, a low-fidelity prototype in this dimension would abruptly switch from one screen to another. Extremely low is a transition from one screen to another when a hand-sketched screen of a paper prototype is manually changed to another hand-sketched screen by a human moderator within a user testing session using this paper prototype. A high-fidelity level would be a smoothly animated transition from one screen to another in a coded prototype.

Richness of data model

The fidelity level of the dimension richness of data model expresses how representative the data employed in the prototype is for the actual target domain.

For example, potential scrolling problems in a list of training data of the YPRC case study can only be studied if the data set used is large enough. Simple data models with one or two example training data sets—as usually used in low-fidelity or mid-fidelity prototypes—might allow testing of several aspects of the list, such as the legibility of the characters or the placement of the pictures of the running route, but will not reveal problems with scrolling in a list of 100 entries on a touch screen. Therefore, a high-fidelity level of this dimension would employ a large number of training data sets with a high variation of the individual data entries.

Considerations for daily work

The choice of the level of fidelity for each dimension allows the prototype to focus on certain aspects to be user tested or analyzed for feasibility. This is extremely useful in practice in order to consciously adapt prototypes to actual prototyping objectives. If the fidelity level differs between these dimensions, McCurdy et al. [MCPKV2006] call such a prototype a *mixed-fidelity prototype*.

The dimensions sensory refinement, breadth of functionality, and depth of functionality are within the scope for the foundation level for the DDP, whereas the last two dimensions—richness of interactivity and richness of data model—go beyond this level.

There is an overlap between the criterion level of interaction (see Section 2.3.4.1), the level of fidelity for each dimension of a prototype, and the overall level of fidelity (see Section 2.3.4.3). As already mentioned above, the categorization by the level of interaction in horizontal and vertical prototypes (see Section 2.3.4.1) is covered by the categorization using the dimensions of a prototype (depth of functionality etc.). Also, the classification using the five dimensions of a

prototype includes the categorization by the (overall) level of fidelity. Thus, using fidelity profiles based on the five dimensions of a prototype is more precise and embraces the other two categorization schemes.

All approaches for categorizing prototypes presented in this handbook are covered by the schemes of explorative, experimental, and evolutionary prototyping (see Section 2.3.4.2) and the consideration of fidelity profiles based on the five dimensions of a prototype presented in this section.

2.3.4.5 Fidelity Profiles of Prototypes Typically Used during the Building Process

As explained in Section 2.3.1, different disciplines have a different understanding of what a prototype is. Therefore, prototype names might be used differently in different disciplines as well or have different meanings even within one discipline. Therefore, important prototype names suggested for use by the DPP are characterized in Table 14 by their fidelity profile according to the understanding and experience of the authors of this handbook. The left-hand column of Table 14 shows prototype names typically used in Digital Design.

The following list is a short description of each prototype named in Table 14:

- An appearance prototype represents the final shape of the product but does not provide functionality. It is typically used to validate the visual appearance of the product with customers or users.
- An experience prototype represents the final shape of the product and provides almost real interactivity.
- A sketch is a simple hand-made drawing of (an aspect of) the digital solution envisioned.
- A paper prototype usually consists of hand-sketched screens of the target display on paper and a description of the interaction between theses screens—for example, what happens if the user presses a certain button. Section 2.3.6 provides details of this powerful tool for the DDP.
- A cardboard prototype can be considered as an extension of a paper prototype into the third dimension. It involves cardboard or similar material representing the shape of a physical product.
- A wireframe prototype, usually called wireframes, consists of sketches of display screens in a digital format showing the layout of the respective screens, i.e., where text boxes, buttons, images, or input fields are located on the display. Usually, no real images or real text are shown.
- A click prototype (or clickable prototype) or interactive mock-up presents display screens (hand-sketched or digitally created) that can be clicked. After such a click, a different screen is shown. Usually, the interactivity is limited, and a special creation tool is needed (cf. Section 2.3.5.3).
- A mock-up is a simple build of the physical shape of a (new) device used for a digital solution usually created in a different medium, such as modelling clay, foam, wood, or material used for additive manufacturing (e.g., 3D printing, see Section 2.3.5.2). A mock-up that does not have the target dimensions is referred to as a scaled mock-up.
- A functional prototype contains all relevant elements for realizing a certain function but does not have the final shape of the product. It is typically created to validate the particular underlying function of a digital solution.

- A coded prototype is a prototypical software implementation of the software part of the digital solution (cf. Section 2.3.5.1).
- A storyboard is a hand-sketched or digitally created sequence of images to visualize the interactive use of a digital solution.

Table 14 – Fidelity profile range of typical prototypes in Digital Design

Typical prototype names	Sensory refinement	Breadth of func-tionality	Depth of func-tionality	Richness of inter-activity	Richness of data model
Appearance prototype	high	low-high	low	low	low
Experience prototype	high	low-high	mid-high	mid-high	mid
Sketch	low	low	low	low	low
Paper prototype	low	low-high	low-high	low-medium	low
Cardboard prototype	low	low-high	low-high	low-medium	low
Wireframe prototype	low-mid	low-high	low-high	low-medium	low
Click prototype or interactive mock-up	low-high	low-high	low-high	low-medium	low
Mock-up	low-high	low-mid	low-mid	low	low
Functional prototype	low	low	high	low	medium-high
Coded prototype	low-high	low-high	low-high	low-high	low-high
Storyboard	low	low	low-medium	n/a[9]	n/a
Story map	low	low	low-medium	n/a	n/a
(Fictitious) video, video prototype	medium-high	low-medium	low-medium	n/a	n/a
Future press release	low	low	low	n/a	n/a
Interface prototype	medium-high	high	low-medium	medium-high	low-high
High-fidelity mock-up	high	high	high	low-high	low-high
Pre-production prototype	high	high	high	high	high
Solution prototype	medium-high	medium-high	medium-high	medium-high	medium-high

- A story map is a two-dimensional visualization of a sequence of (textual) user stories. It describes the narrative flow of a digital solution or the overall process provided by the digital solution. Besides being a form of prototype, it is also a management tool (see Chapter 5).

[9] The abbreviation *n/a* means *not applicable* and refers to the fact that this category is not considered by this prototype name.

- A fictitious video or video prototype visualizes the (future) interactive use of a digital solution in the form of a video. The purpose can be (1) to show the interactive use, demonstrating certain functionalities or usage scenarios to stakeholders, or (2) to convince stakeholders. In the latter case, the prototype is rather a promotional video or a commercial video prototype.
- An interface prototype is a different name for a horizontal prototype (see Section 2.3.4.1).
- A future press release is a potential article in a future issue of a newspaper or news portal. It tells an emotional future story about the digital solution envisioned within its application context.
- A high-fidelity mock-up is a different name for a high-fidelity prototype (see Section 2.3.4.3).
- A pre-production prototype is a fully realized sample of the product. It is typically used for final and comprehensive quality assurance measures.
- A solution prototype is a different name for a mid-fidelity or high-fidelity prototype (see Section 2.3.4.3), depending on how close it is to the final digital solution.

2.3.4.6 Prototype Categories in Certain Stages of the Building Process

Explorative prototyping (see Section 2.3.4.2) is useful in all steps of the building process whenever alternative ideas, concepts, or solutions are considered. Such prototypes are usually not re-used and are often horizontal prototypes.

Experimental prototyping (see Section 2.3.4.2) makes most sense during the scoping and conceptual steps. Such prototypes supplement or refine such concepts and can represent an intermediate prototyping step between design and construction and realization. Therefore, this kind of prototyping is especially useful in the consolidation phases of the design and construction activities and during the realization phase of the building process. This type is usually a vertical throwaway prototype.

Evolutionary prototypes (see Section 2.3.4.2) are elaborated and refined until they become the target system. Thus, they are present during all steps of the building process when such prototyping approach is used.

The overall fidelity of prototypes (see Section 2.3.4.3) increases with the progress in the creation of the digital solution. At the beginning of the building process, having a lot of low-fidelity prototypes is useful. They become more detailed (more high-fidelity) and fewer in number as the project moves forward. Prototypes in-between low-fidelity and high-fidelity (sometimes called mid-fidelity) are useful for business decisions. However, there is a danger of choosing a fidelity level that is too high for a certain prototype. In this case, stakeholders might perceive a more mature digital solution than is really available. Therefore, the DDP has to choose the right fidelity level carefully. High-fidelity prototypes—more often deployed in later building process stages—are useful for testing details to see the complete experience or to serve as an interface to software developers or manufacturers (see [McEl2017]).

Figure 42 illustrates a typical use of prototype categories during the lifetime of a project for building a digital solution. The rising slopes for explorative and experimental prototyping depict an increasing use of these prototyping approaches before its usage reaches a peak. However, the actual use is very much dependent on the target digital solution and the project. Similar patterns of prototype use might occur individually during the design, construction, and realization activities of the building process.

In addition to these overall prototype usage profiles, dedicated parts of typical projects require an exploration of a certain solution, even at a late stage in the project. For example, there might be a need to create alternative appearance prototypes that demonstrate different visual design themes to prepare a decision for the best one. Such activities depict the bumps and peaks in the upper triangular shape of Figure 42.

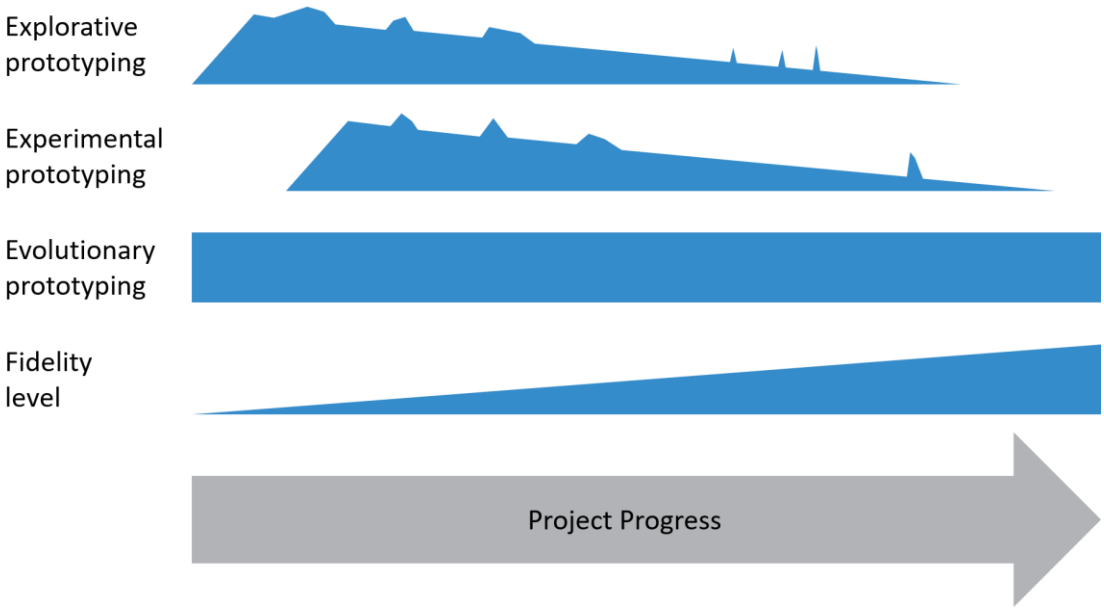


Figure 42 – Typical prototype use (activities) in the course of a Digital Design project

Moreover, a certain function of the digital solution may need a prototypical implementation in order to minimize the risk of a complete failure leading to what is called a *showstopper*. As an example, consider an artificial intelligence (AI) algorithm for a speech output assistant that is performance-critical on the small device a certain digital solution is planned for. In order to ensure that this algorithm is fast enough and to identify optimization needs, an experimental functional prototype can be created. This is an activity preferably done as early in the project as possible in order to continue the work with lower risk.

In another example for the end of the realization activity, an experimental functional prototype of the payment function for a digital solution is created in order to test whether the billing process is working in detail together with other systems the digital solution needs to interface with. These kinds of prototype activities depict the bumps and peaks in the second triangular shape of Figure 42.

In general, mixed-fidelity prototypes (see 2.3.4.4) are very useful for focusing on individual aspects of the conceptual design, the technical design, or the solution realized, such as the three examples described in the previous paragraphs. Therefore, they help most in the middle of the building process. It is impossible to give a general guideline for defining the *correct* fidelity level of the different dimensions of a prototype, i.e., the ideal fidelity profile at a certain project phase or abstraction level. This is because the appropriate fidelity profile is dependent on the specific objective within the specific project situation the prototype is created for—for example, it depends on evaluation goals of a usability test. Nevertheless, the examples show the high flexibility and usefulness of this categorization approach using fidelity level profiles for the different dimensions of a prototype.

2.3.4.7 Degree of Immersion in Relation to the Fidelity Profile of a Prototype

Technical systems for creating a virtual reality (VR) provide a certain degree of *immersion*, which is the technical precondition for enabling the psychological state of *presence* for the user. Presence is the feeling of the user of *being there* in the virtual environment while forgetting the fact of physically still being in a (different) real environment.

We consider the concept of immersion and presence from virtual reality to be useful for applying prototypes in Digital Design. Depending on the planned objectives to be achieved by a prototype, a certain degree of immersion can be targeted. If, for example, a designer colleague shall give quick feedback to a certain user interaction detail for a digital solution, a lower degree of immersion is sufficient than, for example, for a prototype to be used for a customer presentation or an extensive usability investigation.

Following [Jera2016] citing [SlWi1997], “Immersion is the objective degree to which a VR system and application projects stimuli onto the sensory receptors of users in a way that is extensive, matching, surrounding, vivid, interactive, and plot conforming.“

The following description of the immersion elements from this definition are adapted for use in Digital Design. The main adaptations were made for vividness, interactability, and plot.

- Extensiveness indicates the number of sensory modalities presented to the user—for example, if visual output, auditory events, and haptic stimulation by physical forces are present.
- Matching describes the degree to which different modalities fit together and convey consistent feedback to the user—for example, how well the vibration of the smartphone fits to the press of the virtual button in terms of timing.
- Surroundedness describes how all-embracing the cues are—for example, a large display, spatialized audio output, or a large area where gestures are recognized to contribute to this element.
- Vividness indicates the ability to produce a sensorially rich mediated environment, characterized by, for example, the resolution of the display, the brightness and color range, and frame rate or bit rate (cf. [Steu1992]). Interactability is the capability to influence the prototype and the extent of the responses of the prototype based on the user’s actions.
- Plot describes how well a usage scenario is implemented into the digital solution. It goes beyond a single interaction flow and is therefore not fully covered by the interactability. As an example from the YPRC case study, the runner must attach a device for measuring heart rate and set up the smartwatch and smartphone before the run can start. After a series of runs—for example, multiple runs within one week—the runner can evaluate the running statistics on the smartphone or home PC. Including the function of a human or artificial intelligence coach would make the context scenario and thus the plot more complex.

Table 15 presents the relationship between the immersion elements and the five dimensions of a prototype (see Section 2.3.4.5). This mapping can be used to identify important dimensions of a prototype in order to create the desired level of immersion. A strong link (marked by “X”) between a prototype dimension and a certain immersion element indicates how the fidelity level of this prototype dimension influences the respective immersion element and thus the overall immersion degree. The choice of a prototype with a particular fidelity profile can focus on certain immersion elements.

Table 15 – Relationship between prototype dimensions and immersion elements. "X" indicates a strong and "~" a weak relationship.

	Dimensions of a prototype (fidelity profile)				
Immersion elements	**Sensory refinement**	**Breadth of func-tionality**	**Depth of func-tionality**	**Richness of inter-activity**	**Richness of data model**
Extensiveness	X				
Matching	X	~	X		
Surroundedness	X	~			
Vividness	X				
Interactability		X	X	X	X
Plot				X	X

2.3.5 Tools for Creating Prototypes

In order to keep a large solution space open for the exploration of (fundamentally) different solution approaches in the early stage of Digital Design projects, simple but powerful prototyping technologies, such as paper prototyping (see Section 2.3.6), are often the right choice.

The selection of the most appropriate category of a prototype (see Section 2.3.4) and prototype creation tool in a given project situation requires a certain level of experience. The goals of the concrete evaluation determine the most useful prototype category. A beginner in this field should start with simple technologies, such as sketches, paper prototypes, or storyboards. To avoid wasting effort, technologies that are more complex should be used only after careful consideration. The following sections present an overview of the broad scope of prototype creation tools useful for creating digital solutions.

2.3.5.1 Software Design and Development Tools and Technologies for Prototype Creation

Standard software development environments can be used to create coded prototypes, especially if prototypes with in-depth functionality (high-fidelity of depth of functionality) are needed. To create a cost-effective prototype, many simplifications compared to the target development environment are possible.

Web technology as a powerful tool for creating prototypes

Web technologies, such as HTML5, CSS, and JavaScript are useful for quickly creating prototypes with more functionality than click prototypes (see Section 2.3.5.3) can offer. Besides the tool support for the creation of websites and web applications, there are tools available that allow the automatic deployment of web applications to different smartphone operating systems. This deployment to Android and iOS, for example, relies on a single code base using web technologies. Such *hybrid applications*, as these are sometimes called, might even be an option for the final digital solution (see also Section 3.2.3).

When a coded prototype is required, graphical user interface builders (GUI builders) that use graphic libraries of the target operating system are helpful tools. This usually speeds up the development compared to the implementation of a completely customized user interface (library).

Coding prototypes for complex or in-depth functionality prototypes

For more complex or in-depth functionality of the prototype, coding parts of the software application is an option and is sometimes necessary. To speed this creation process up, a simpler programming language or development environment than for the target digital solution is useful. Prototypical programs stay simple by omitting error-handling routines or special cases of the program control flow. In software development jargon, this kind of simplification is sometimes referred to as a *hack*.

If a digital solution runs on standard hardware, such as a smartphone, it usually suffices to use the prototype tools described above to run the prototype on one (standard) screen size and orientation. However, if a new digital solution experience employs a new physical form factor or a new type of hardware, the final shape and capabilities of the device running the software (e.g., screen size, computation capacity) should be considered to create a realistic prototype.

Considerations for daily work

For the DDP at foundation level, the recommendation is to bear the prototyping possibilities presented in this section in mind. If there is an actual need for a coded prototype, a skilled expert for such prototypes must be employed. It is very important that this expert is experienced and skilled enough to simplify the prototype as much as possible and does not create another alternative implementation for the final digital solution. The effort for such prototypes must be limited to allow this work to be discarded when the prototype has served its purpose (throwaway prototype).

The prototyping tools presented in this section are useful for creating prototypes for the perceivable and underlying layers of the software part of the digital solution. Section 1.2 covers the distinction between the perceivable and underlying form, function, and quality in detail.

2.3.5.2 Industrial Design Tools for Prototype Creation

Industrial designers have a long tradition in prototyping and use prototyping tools to design the interaction of the user with a physical product and to study the feasibility of such a product. The following three types are distinguished (see [IDSA2020]):

- Drawing tools for sketches and illustrations
- Rendering software for three-dimensional renderings
- Additive manufacturing tools for tangible mock-ups

Drawing tools and rendering software

There are various types of sketches and illustrations, such as idea sketches, study sketches, referential sketches, memory sketches, coded sketches, information sketches, sketch renderings, perspective sketches, scenarios, storyboards, diagrams, perspective diagrams, arrangement diagrams, detail drawings, and technical illustrations.

Three-dimensional renderings are layout renderings and presentation renderings. Computer-aided design (CAD) tools are used to create sophisticated three-dimensional illustrations.

Tangible mock-ups and physical prototypes

Building tangible mock-ups and prototypes, often called modelling, has a long tradition in industrial design for exploring ideas and optimizing products in the physical space. Studying certain aspects can involve appearance models, assembly models, production models, service models, experimental prototypes, alpha or beta prototypes, system prototypes, final hardware prototypes, off-tool components, appearance prototypes, and pre-production prototypes.

In order to give an impression of the capabilities of this kind of prototyping, which goes beyond the foundation level, Figure 43 shows example prototypes from this field. On the left side hand-made sketches (1) with markers and on the right side photoshop sketches (2) are shown. The middle of the figure shows prototypes created by additive manufacturing (3) based on a CAD model created in Solidworks (CAD application). For additive manufacturing, the fused filament fabrication process was used. This process is also known under the trademarked term *fused deposition modeling*. These prototypes are used to validate ergonomic properties of the pen. The black pre-production prototype (4) was created by the final manufacturer using an injection molding process. The transparent pen (5) is the final product in the Neuland FineOne® Empty variant. Three-dimensional renderings are shown on the right and bottom parts of the figure. These are used for communication purposes.

Figure 43 – Exemplary industrial design prototypes (copyright by Neuland/GENERATIONDESIGN)

Industrial designers sometimes use the word *model* to refer to a prototype, whereas software engineering has a different understanding of this term (cf. [IEEE2017]). Section 2.3.1 provides a definition of the term prototype within the context of Digital Design and a discussion of how it relates to the term model.

Considerations for daily work

For the DDP at foundation level, the recommendation is to train sketching and illustration skills. A basic understanding is sufficient for the prototyping possibilities of using three-dimensional renderings and tangible mock-ups. If there is an actual need for a more sophisticated prototype with such tools, a skilled expert must be involved.

The prototyping tools presented in this section are useful for creating prototypes for the perceivable layer of the physical part of the digital solution. Section 1.2 explains the distinction between the perceivable and underlying form, function, and quality.

2.3.5.3 Interaction Design Tools for Prototype Creation

In digital solutions, interfaces to users play an important role. Interaction design is the discipline that deals with the creation of these user interfaces (cf. [Coop2004]). Therefore, all interaction design tools are useful for building a user interface prototype. Such tools are drawing tools for sketching, storyboarding, and wireframing, paper and pencil for paper prototyping, and rendering software for high-quality renderings.

Considerations for daily work

Sketches, wireframe prototypes, or high-quality renderings of user interface screens can be used for integration into a clickable prototype (also called click prototype or click dummy). Many existing software tools for the creation of such prototypes allow use of the physical target device (form factor), such as smartphones, PCs, or tablets. This makes the prototype more realistic by making clear to the user how small the actual device screen is, for example. There are many click prototype tools available that are suitable for different types of prototypes. Existing product names for such tools include Adobe XD, Axure, Balsamiq, Figma, Flinto, InVision, Mockups, or Sketch. However, for the DDP at foundation level, the recommendation is to stick to simple tools, preferably based on paper and pencil, or to involve a skilled expert who is experienced with these tools.

The prototyping tools presented in this section are useful for creating prototypes for the perceivable layer of the software part of the digital solution. Section 1.2 explains the distinction between the perceivable and underlying form, function, and quality.

2.3.5.4 Other Tools

Furthermore, prototyping tools and technologies from production engineering and electrical engineering may also be used—for example, creating a device prototype using additive manufacturing (e.g., 3D printing) or an early version of a customized printed circuit board (PCB). These prototyping tools are useful for creating prototypes for the underlying layer of the physical and hardware part of the digital solution. Section 1.2 explains the distinction between the perceivable and underlying form, function, and quality.

2.3.6 Building and Using Simple Low-Fidelity Prototypes

Paper prototypes and cardboard prototypes (see Section 2.3.4.5) are easy to build and are practical for creating low-fidelity prototypes. A paper prototype usually consists of (1) hand-sketched screens of the target display screen of the digital solution, and (2) a description of the sequence of these screens when the user interacts with the digital solution (interaction flow). This description can be a storyboard or some other form of specification of the logical flow of the screens—for example, what happens when the user touches a button or when a particular event happens.

YPRC example. Figure 44 shows three paper prototype screens of the smartphone app of the YPRC case study. The left image visualizes the display on the smartphone when the smartwatch is connecting to the smartphone. The other two screens show the content during the running training, such as the time elapsed, run distance, the average running pace, the current and average pulse rate, and the current location of the runner on the map of the planned running route.

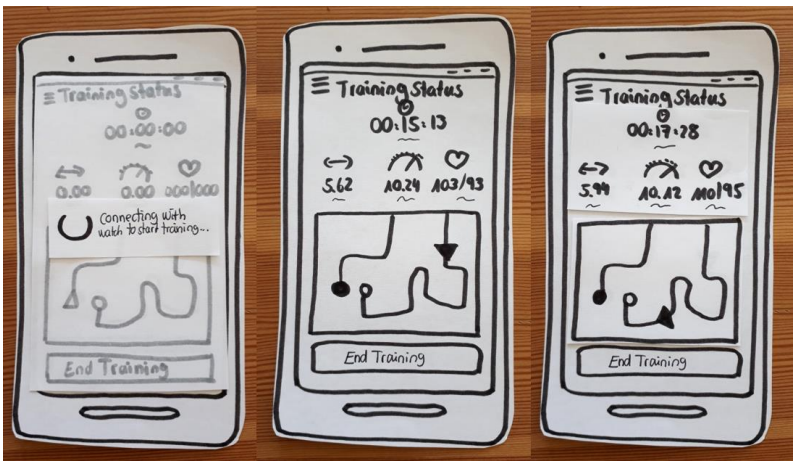


Figure 44 – Example screens of a paper prototype for the YPRC case study

When these display screens are drawn on small pieces of paper or on sticky notes, they fit into a model of the target device (hardware display). Several of these drawn screens can be used to present the screen flow. A large piece of paper or cardboard can serve as such a model.

YPRC example. Figure 45 illustrates such a model with sticky notes from the YPRC case study. The drawn screens on the right side of the figure can be pasted onto the smartphone screen on the left side of the figure during a discussion with stakeholders or a usability test with target users (see below).

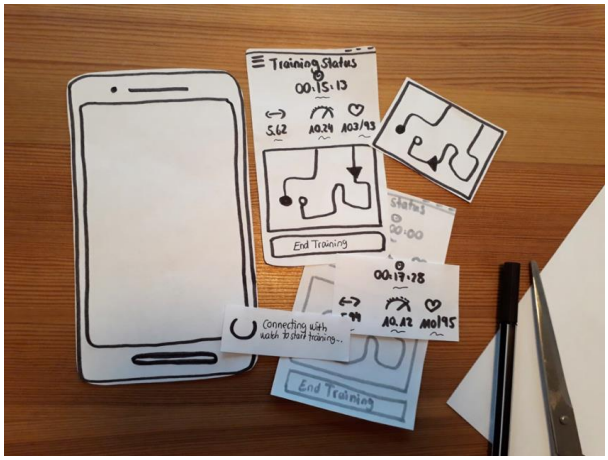


Figure 45 – Example paper prototype for the YPRC case study with several sketched screens

A prototype can be created for almost any device, such as a PC, tablet, smartphone, or smartwatch display. Extending this approach, prototypes made of cardboard or other material are possible, representing an early version of a physical device.

YPRC example. Figure 46 shows a prototype of the runner's watch of the YPRC case study with different alternative display content. The top images show the warning of a pulse rate that is too high and the bottom images show the current and average pulse rates at the same time. As the images on the right show, such a simple prototype can be used to study, with a runner, how suitable such displays are and how well the physical dimensions fit to the purpose of the device, for example, when worn on the wrist.

Figure 46 – Example cardboard prototype of the runner's watch of the YPRC case study

A storyboard[10] visualizes how the user interacts with the digital solution. It visualizes the interaction flow, which defines, for example, what happens when the user touches a button or when a particular event happens, such as exceeding the upper pulse rate limit.

[10] A storyboard is very powerful when it is combined with an information scenario (cf. [RoCa2003]).

YPRC example. Figure 47 shows an example of a storyboard from the YPRC case study. Starting from the upper left image, going down to the lower left and up to the upper right image, the storyboard shows how a runner prepares the smartphone and the runner's watch for a running training session. It then shows the runner running and finally, a situation where the heart rate reaches 180 beats per minute und thus exceeds the personal upper limit of the runner. As indicated, this storyboard shows only part of a larger story that visualizes the interaction of a runner with the YPRC digital solution.

Figure 47 – Example storyboard of the YPRC case study

When testing a particular interaction flow with a paper prototype, a set of sketched screens contains the basic layout of display components and (a representation of) the basic content—for example, buttons and menu items are available. Usually, different flow alternatives for completing a task of the user are prepared.

Usability test using paper prototypes

As an example, the use of a paper prototype for a usability test is described in the following paragraphs. In a usability test, test users use a paper prototype to report on the usability of the design represented by the prototype. In order to prepare such testing, the context of use and the specification of the user requirements—which the prototype is based on—are needed. Tools for defining this include personas, (context) scenarios, or use cases. Then, concrete tasks for the use of the prototype are selected—for example, from specifications available such as a Digital Design brief or the Digital Design concept.

Once the usability test has been prepared, target users try to complete the selected tasks using the prototype. A moderator guides the user and explains the system. A second person simulates the system by manipulating the paper prototype. For example, if the user touches a button on paper, a new screen appears or a menu opens up by, for example, removing a sticky note that covers part of the screen. The user might scroll through a menu, which results in a shift of the paper strip with the sketched menu. Usually, users are very capable of imagining such interactions with the future digital solution, even when just interaction with a paper prototype takes place. This allows valuable user feedback to be gathered. One or more observers record user comments and

observe the behavior of the user. Both user comments and observations can indicate successful designs and (usability) problems of the design. Depending on the case or if the number of resources is limited, one or two persons can conduct the usability test. In this case, one person performs all moderation, simulation, and observation tasks alone, or two persons divide these tasks among each other.

Once the usability test has been completed, the analysis of the data gathered usually leads to improvement ideas for the design. These enhancement suggestions are selected to improve the design or realization of the digital solution. Depending on the fulfilment of the requirements, a new (paper) prototype can be created and used for another study with users. The process described in this section is one realization of the user-centered design process as specified in [ISO2019].

Considerations for daily work

Paper prototyping is a simple, flexible but powerful method that uses paper prototypes in iterative loops to gather valuable feedback quickly from users and other stakeholders. Interaction designers apply paper prototypes frequently to improve user interface designs. In many cases, they prefer this kind of prototype over click prototypes because of its flexibility. Paper prototypes work well for low-fidelity prototypes. However, if interaction flows become complex or high-fidelity prototypes are required, other prototyping tools should be used (see Section 2.3.4 and 2.3.4.7). See [Snyd2003] for an in-depth coverage of this powerful method of paper prototyping.

2.3.7 Conclusion on Prototyping

As this section on applications of prototyping shows, building prototypes to create new designs and subsequently using these prototypes for evaluation is a powerful instrument for the DDP. Prototypes are a preliminary, partial instance of a design solution. They thus represent a manifestation of an idea, a concept, a system, or a solution.

Using prototypes early in the building process makes it possible to explore in fundamentally different directions and to find out the advantages and disadvantages of different alternatives at an early stage. If the prototypes are simple and have low fidelity, it is easy to discard those alternatives that do not work well for the digital solution envisioned. This usually does not lead to much cost but mostly to an additional understanding and a considerable step forward in the building process. However, the DDP must be ready to create prototypes for this purpose only and subsequently discard them. Later in the building process, prototypes help to find out in depth whether a certain critical part of the digital solution is working or not, or what the actual remaining problems are.

The use of prototypes has a broad application spectrum across the building process steps (scoping, conceptual, development and operations) and the different abstraction levels (solution level, system level, element level). The application of the appropriate prototype, especially according to its fidelity profile, depends on the risk of certain parts of the digital solution, the budget available (that can be planned for), and the experience of the personnel involved, especially of the DDP. However, the DDP should use the powerful method of prototyping, advocate for a budget for this activity, and build up experience of using the broad spectrum and the benefits of this technique. Finally, the DDP should not underestimate the strength of paper prototyping, which can be used to resolve uncertainties quickly and reduce risks in building innovative digital solutions.

3 Digital as a Material

The technological possibilities (the digital material) for realizing a digital solution in the area of hardware and software have grown enormously in recent years and will continue to grow in the future (see [Kell2016]). At foundation level, the DDP must know this broad range of possibilities and be ready to keep up with them and their continuous technical development. This chapter is intended as an introduction to the broad range of digital material available.

We start with an introduction to the Digital Design way of understanding technology (Section 3.1). The perceivable and underlying layers of a digital solution strongly influence each other (see Section 1.2). Consequently, their interdependencies must be considered in order to provide a good digital solution. In Section 3.2 and Section 3.3, we give an overview of perceivable and underlying technology following the two layers of digital material introduced in Section 1.2. Section 3.4 presents three technology-oriented knowledge areas that we consider important for a DDP at foundation level. Section 3.5 concludes this section with a Digital Design perspective on technology.

3.1 Understanding Technology

The DDP understands technology as shapeable material for building a digital solution (see Section 1.2.1). Understanding technology as a material essentially means that only a profound knowledge of technology enables the DDP to create excellent digital solutions. The DDP understands the possibilities and limits of, for example, processing and data transportation technologies and memory and storage units. By combining these different building blocks, the DDP creates a digital solution that delivers added value to a user or customer.

The form, function, and quality model as the basis for understanding technology

In order to structure the access to this knowledge and communicate the knowledge of technology to other parties, the DDP uses the form, function, and quality model of digital material. As outlined in Section 1.2.1, this model distinguishes between (1) the perceivable layer, which addresses the form, function, and quality that can be perceived by the stakeholders, and (2) the underlying layer, which is hidden from the perception by the stakeholders but enables the perceivable layer. Figure 48 illustrates this differentiation between the two layers with respect to form and function and provides examples of this view on technology.

Perceivable technology

The left side of the figure shows examples of hardware technologies. The shape of an end user device, such as a smartphone or a tablet computer, can be directly perceived by a user. If special material was used to build the device, such as magnesium or glass, the user can feel this directly. The user can see and feel all the buttons on the device. These are examples of the perceivable form. The way users use the buttons of the device to adjust the volume or to mute audio is (part of) what is known as the interaction flow and this can also be perceived directly. If there is a dedicated mute button, the user can switch to mute, feel the pressure point and movement, hear the sound switching off, and probably see on the button that the device is muted (maybe through a colored indicator on the button). This volume adjustment and audio mute example is part of the perceivable *remote voice coaching* function of the YPRC case study.

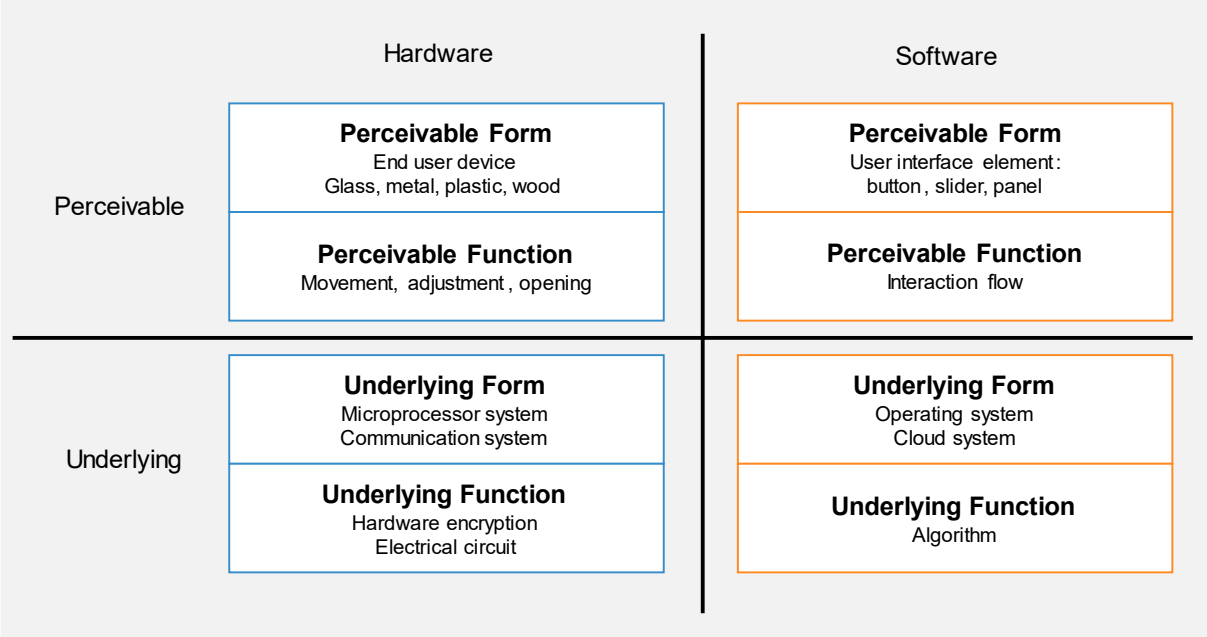


Figure 48 - Examples of technology categorized into (1) perceivable form and function for hardware and software (upper part) and (2) underlying form and function for hardware and software (lower part)

In a digital solution, the hardware function is only part of the solution. To continue the example, the adjustment of the volume—after pressing the (hardware) volume button—therefore needs software that displays the volume change. A user interface element that appears on the smartphone screen showing the status or change of the volume is an example of the perceivable form of the software part (see the upper right part of Figure 48).

The perceivable software part of the interaction flow determines, for example, how many button presses are needed to change the volume from the lowest to the highest possible setting. A more complex example of a perceivable function of the software is the interaction needed to unlock a smartphone by entering the PIN code. Section 3.2 provides more details on perceivable technology.

Underlying technology

Besides these perceivable elements shown in the upper part of Figure 48, a digital solution usually uses more technology that is not directly perceivable, i.e., underlying technology (lower part of Figure 2), to realize its services. In the volume adjustment example, an underlying form of software is that part of the operating system that must be used by the volume adjustment application to actually change the volume. An operating system usually provides an application programming interface (API) as a set of functions or methods that can be accessed by a programming language. How these functions have to be used and how they work is the underlying function of the software.

In order to completely mute the volume, the application might have to successively use the API to reduce the volume one step and use another function to check whether the volume is already zero. This is repeated until the volume is actually zero. In an alternative implementation, the API might provide one dedicated function that mutes the volume in one step or slowly fades out the signal, which makes the iterative use of the API functions described obsolete. There might also be an API function available that enables fading in an audio signal. As already mentioned, this

eliminates the need for successive volume adjustment and volume level control steps from the perspective of the user interface application.

The underlying form of the hardware of a smartphone is, for example, all electronic components that are not directly visible or recognizable with any other human sense (not perceivable). As an example of the underlying form of hardware, a typical smartphone has a (small) loudspeaker and an amplifier in order to play sound and to make (hands-free) phone calls. A communication microprocessor (signal processor) is needed to make phone calls. An application microprocessor is responsible for running (downloaded) applications. The underlying hardware elements of the smartphone can communicate with each other and enable complex use cases, such as the streaming of music using the application processor downloaded through the mobile network function of the radio frequency unit.

The underlying function of the hardware describes how the hardware components (underlying form) work together. In the example above, the amplifier must drive the loudspeaker based on the signals from the communications microprocessor to enable the function of hands-free phone calls. If other applications should also be allowed to play sound, the amplifier must be connected to the application microprocessor as well.

When looking at this simple sketch of the hardware architecture, the following question arises: what happens if the music application plays a song and a phone call comes in? It is obvious that the call must be signaled even when music is played. This can be realized by a mixer (an additional underlying form of hardware) that enables the mixing of signals from both the communication microprocessor and the application microprocessor. Section 3.3 provides more details on underlying technology.

Today, the forms and functions described in the examples above are part of the standard smartphone architecture and are available in all modern devices. However, the example illustrates how (considerations on) underlying form and function influence the possibilities at the perceivable level. If the mixer were not available, a smooth blending of the sound—controlled by the phone call or music application—would not be possible.

This influence of the underlying level on the perceivable level is an important factor in general in order to create digital solutions that balance user needs and the use of new technology. With the advent of the application of artificial intelligence (AI) algorithms in particular, the understanding of the underlying form and function and their influence on the perceivable form and function has become vitally important.

Benefits of understanding technology

A profound understanding of technology and technological developments offers the following important benefits for the DDP:

- *Avoidance of unrealizable solutions*: A profound understanding of the technologies available prevents the definition of unrealizable forms, functions, and quality of a digital solution—for example, the realization of a 4k video chat function is currently not feasible.
- *Inspiration for novel solutions*: Innovative technologies offer capabilities that may enable novel aspects of a digital solution or a completely new digital solution. Examples are new applications based on the application of artificial intelligence (AI).
- *Substantial communication with software experts*: Often, specialists are needed to design and develop parts of a digital solution. Knowledge of the respective technology domains

enables communication with these experts. As an example, knowledge about the existence, the general use, and the functionality of user interface programming libraries helps in such communication.

- *Substantial communication with experts for physical products*: If the digital solution incorporates an in-house physical product, knowledge of the underlying technology enables the DDP to discuss these aspects with the respective experts. This is especially important if the digital solution involves special software and hardware parts. For instance, if the digital solution running on a mobile device requires very high performance from the microprocessor(s), a discussion about the maximum current rating of the available battery becomes important.
- *Substantial communication with vendors or partners*: A profound understanding of the technologies involved allows the DDP to cooperate with partners or vendors who realize parts of a digital solution. This is applicable for vendors of hardware as well as software parts. For example, knowledge about the availability and general function of video compression codecs or white label hardware is important to be able to subcontract such product parts.

As digital technologies are subject to constant development, the DDP must continuously monitor current technological developments and understand new technologies to keep up to date.

There is a difference between understanding technology and developing with technology

It is important to recognize the difference between technological knowledge and the ability to design and develop using this technology. It is only by knowing about the possibilities and limits of the available technology that you can build the best digital solution possible. The DDP does not necessarily have to be an expert in all technological fields. For instance, the DDP must understand all the possibilities and limits of user interactions using the current technologies. However, experts in interaction design deal with this kind of design in depth in order to create an interactive system (see also Section 2.1.4).

The choice of certain technologies has an impact on the quality of a digital system and hence the overall digital solution. Technology choices can manifest themselves in different quality characteristics affecting perceivable and underlying qualities (see Section 2.1.3).

The selection of adequate technologies, therefore, has to be in line with the quality criteria defined (i.e., the quality requirements) for a digital system and the overall digital solution. If technologies are chosen without knowing or considering predefined quality criteria, this can lead to digital systems and solutions that do not meet the expected perceivable quality criteria of their users and hence these solutions might not be accepted by them. Furthermore, inadequate technology choices also can have an effect on underlying quality criteria and prevent or at least make it cumbersome for developers to maintain the system.

All these considerations are true for hardware and software and for the form and function of perceivable and underlying technologies. As discussed in Section 2.1.3.2, a DDP needs to manage quality actively and to do this, needs to have a good overview of existing technologies and understand their effects on key quality attributes.

YPRC example. The YPRC case study provides good examples for discussions. Let us assume for a moment that the central processing unit (CPU) of the runner's smartwatch is selected without closer consideration of quality requirements and without understanding the potential effects of this decision. As an underlying hardware component that defines the form of the digital system, the CPU can have a major impact on the perceivable and underlying qualities of the digital system and overall, affect the quality of the digital solution.

For example, picking a cheap and powerless CPU might lead to performance problems and therefore affect the perceivable quality of the digital system (see Section 2.1.3.4). Furthermore, this could also have an impact on the digital solution because performance problems, for example, might interfere with the aim of the users of enjoying using their running coach (hedonic quality, see Section 2.1.3.5). Selecting an inadequate CPU might also affect other technology choices, such as the selection of the operating system (underlying form, software). In turn, that can have an effect on the development environment needed to develop the software components of the digital system and might also affect the type of user interface provided (perceivable form, software) and the way users can interact with the system (perceivable function, software).

Just like performance issues already mentioned, these effects could negatively impact the perceived quality of the digital system—now in terms of usability and furthermore, could again have a negative overall impact on the digital solution in terms of the joy of using the running coach. In addition to negatively affecting perceivable qualities, the selection of an inadequate CPU might potentially force developers to use an inadequate development environment, which can have negative effects on underlying qualities such as maintainability.

Ideally, a DDP would be aware of these potential negative effects and by understanding them, would support the selection of an adequate CPU for the smartwatch.

3.2 Perceivable Technology

Perceivable technologies are used to realize those parts of a digital solution that a user can perceive directly. A DDP at foundation level should be aware of end user devices, interaction technologies, and technologies for the software implementation of user interfaces.

3.2.1 End User Devices

Standardized end user devices, such as notebooks, tablet computers, or smartphones, are often used to realize a digital solution. These devices can be classified as perceivable technology. On the other hand, they provide technical capabilities that can be assigned to underlying technology (e.g., wireless communication technologies such as WLAN or Bluetooth). However, the DDP has no direct influence on the internal structure of the devices, therefore the perceivable form is the most appropriate category in this context. If standardized end user devices are used as part of a digital solution, the assumed technical capabilities—for example, screen size and density, communication technologies, performance of the processors, memory size—must be clearly defined in order to provide the necessary resources for the digital solution.

The following distinction provides coarse classes of end user devices:

- *Stationary devices at fixed locations*, such as personal computers, smart speakers (e.g., Amazon Echo or Google Home), or smart scales. Because of their fixed location, they can use an outlet as the electrical power supply and the occupied space is—in principle—not limited. Therefore, this category can have the highest performance among these classes in terms of computing power. This means that very fast central processing units (CPUs) and graphics processing units (GPUs), large working memory, high mass storage capacities, and large displays are possible.
- *Portable devices* can be used at different locations. This class ranges from multi-purpose devices, such as notebooks, tablet computers, or smartphones, to—the growing number of—single-purpose devices, such as card readers, fingerprint or retina scanners, consumer goods ordering devices (e.g., Amazon Dash Buttons or Amazon Dash Smart Shelf), or customer satisfaction stations. Because of their portability, these devices usually have to run on a (rechargeable) battery and can be of only limited size to fit into the intended space, such as a pocket, bag, or close to the cash register or the washing machine (for quickly ordering washing powder). A power outlet or wired network connection can be used only in some usage scenarios. Therefore, the available computing power and memory availability are medium compared to the other two classes within this list. The possible display size ranges from medium for the above-mentioned multipurpose devices to small or non-existent for the single-purpose devices.
- *Wearable devices* are worn on the body or even implanted into the body. Examples are activity trackers, blood glucose meters, smartwatches, or other small devices that can be carried. These devices may use only a very limited space and must run on (rechargeable) batteries only. Compared to the other two classes within this list, these devices have the lowest computing power and lowest available memory profile. Also, the display size is very small. Some devices have no display at all and have to use other interface technologies—such as voice technology or wireless technologies—for direct or indirect information exchange with the user.

3.2.2 Interaction Technology

Modern interaction technology consists of a combination of complex hardware and software systems that belong to the perceivable form and function (see Section 1.2). For example, a working smartphone touch screen, which enables touch interaction, requires a complicated combination of hardware and software components in order to generate an appropriate response by the device to the user's touch on the screen.

From the perspective of the DDP, the dynamic aspects of this user experience are most relevant, i.e., the interaction flow enabled by the combination of hardware and software of the interaction technology. Therefore, the DDP focuses mainly on the perceivable functions of the respective interaction technology. See Sections 1.2.1 and 3.1 for an explanation of the perceivable and underlying layers.

One interface type that is still widely used today is the graphical user interface (GUI). When using a GUI, the user navigates—usually within a window—through menus on a hardware screen and selects (menu) items or icons using a mouse pointer (or another navigation device) or keyboard navigation. This kind of interaction through mouse and keyboard is indirect as the item to be selected on the screen cannot be directly clicked on the screen with the finger, for example. One

advantage of a GUI is that it shows all available commands of the interface through menus, icons, and other graphical elements. The user finds commands by exploring these items and does not have to remember any specific command. This supports the psychological function of recognition, meaning that the user only has to recognize the function of a command or option instead of remembering it. Column three of Table 16 summarizes this characterization of a GUI.

Table 16 – User interface paradigms[11]

	Command Line Interface (CLI)	***Graphical User Interface (GUI)***	***Natural User Interface (NUI)***
Interaction	abstract	indirect	direct
Commands	directed	explorative	contextual
Psychological function	recall	recognition	intuition

Before the existence of GUIs, command line interfaces (CLI) were a very common way of interacting with computers. In a CLI, a computer displays a command prompt on the screen waiting for the user to input a command that directs the computer to execute an instruction. The user must type this command into the CLI manually. The user generally has to remember possible commands and their options. Usually, there is only a limited possibility to explore these commands via additional help commands or options, if these are available at all. Therefore, this kind of interaction is characterized as abstract (see column two of Table 16). A CLI can still be activated in many modern operating systems, such as Android, iOS, Linux, or Windows. Today, these interfaces play an important role in batch processing, for IT administration, or for use by computer programming experts in general.

For modern smartphones or tablets, which use high-quality touchscreens, GUIs have been expanded during the past years to enable a more direct manipulation of objects. The user interface of such devices are examples of a natural user interfaces (NUIs) This direct manipulation works through the inspection of objects on the screen and the selection of commands and options using the user's finger or special pens that act as a pointing device. This enables a more direct user interaction than in pure GUIs. Commands and options are available from the context and can be used more intuitively than, for example, in GUIs. This description of an interface is just one example of a natural user interface (NUI). An even more natural interaction method is using voice input and output, which modern smartphones support as well. The aim of NUIs is to enable interaction with computers in the same way as the interaction with the physical world (see [ShRP2019]) or other human beings. These aspects of NUIs are summarized in column four of Table 16.

The user interface types and their characteristics summarized in Table 16 played an important role in the past and are still important today in interaction with computers. Hinman ([Hinm2012]) refers to them as user interface paradigms.

In addition to these user interface paradigms, we can distinguish between different interface types. Many of these types map to the aforementioned user interface paradigms. The interface types that have a certain significance today are described below (cf. [ShRP2019]).

[11] Adapted from [Hinm2012], which is based on a presentation by Dennis Wixon.

Audio interface

The user receives audio signals through a loudspeaker or headphone, sometimes in addition to visual feedback. Such signals play an important role in many everyday interfaces. In home appliances, such as the refrigerator, the dishwasher, or the microwave oven, they signal a certain status of the device. A parking assistant in a modern car is an example of a more complex audio interface that transports information about the distance to other objects in front of or behind the car through the frequency at which a tone is played, for example. For in-car applications in particular, it is important to find the right complexity of the audio interface in order to avoid information overload for the user.

Voice interface

A more complex audio interface is a voice interface or voice user interface (VUI), which enables voice output and voice input. Voice output usually works via a text-to-speech (TTS) system, which allows written (digital) text to be transformed into a voice output. One application of such a system is a screen reader, which is useful for the visually impaired. Modern systems rely on complete sentences spoken by professional speakers. If the database is large enough and the system can combine the correct sentence fragments, such voice output sounds very natural. This output is usually more natural than the artificial sound of older systems that base the output on a concatenation of individual phonemes—pre-recorded by a speaker or artificially created—that make up the spoken language.

The input part of a voice user interface is usually a speech-to-text system that employs speech recognition to identify the contents of the spoken words. Such input systems have already been in use for a long time in call centers to route the customer to the assistant responsible. Modern uses are the voice assistants (e.g., Google Assistant, Apple Siri, or Microsoft Cortana) in smartphones or smart speakers, in which the speech input and output is connected to an artificial intelligence unit. Such applications for single users—for example, for smartphone usage—already work fairly well. However, conversational quality has not been reached yet, i.e., the recognition rate of a smart speaker drops in a family setting, where multiple people communicate with the device. Also, the recognition of the speech of very young children (3-4 years old) often fails as their use of language differs significantly from adult speech (cf. [ShRP2019]).

Touch interface

Touch interfaces use the user's touch as input. Touchscreens have already been in use for a long time, mainly in ticket machines or automatic teller machines (ATM). They became widespread though their use in smartphones. Early touchscreens used a single touch input as the only available input gesture, where the user used their finger to initiate a single event. The ability to recognize multiple finger touches at the same time (multitouch) or the pressure of the touch enables the recognition of more complex gestures. Such gestures include swiping, flicking, pinching, pushing, or tapping. One technology often used in modern touchscreens works based on the measured change of an electrical field on the screen when the user's fingers move closer to the screen (capacitive touch). The main drawback of most touchscreens is the missing tactile feedback.

Gesture-based interface

Touchscreens can only recognize finger gestures that touch the device. If gestures of a complete hand or arm are to be recognized, additional technology, such as depth sensors or cameras, must

be employed (e.g., Nintendo Wii, Microsoft Kinect, or Leap Motion Controller). Simple gesture-based interaction may be realized by using the built-in sensor in modern smartphones (e.g., gyroscopes). In the past, a lot of research was conducted on gestures and their use in technical systems. As a major finding, there is a need to follow a certain syntax like that in a spoken or written sentence in order to understand a gesture. For example, to increase the volume of the TV, the left hand must point to the TV, then in turn, the right hand is raised in order to communicate the gesture to increase the volume. Interesting applications for gesture-based systems are applications for surgeons when they are in an operating room and cannot touch any unsterile objects, such as a mouse or touchscreen. For this type of scenario, an application was developed to allow navigation through a series of images of computer tomography (CT) or magnetic resonance imaging (MRT) recordings based on gestures. One problem that has still not been resolved in gesture-based systems is the robust recognition of the start and end of a gesture and the distinction of a meaningful gesture from a gesture that merely supports spoken words within a conversation. (cf. [ShRP2019]).

Haptic interface

Haptic interfaces provide tactile feedback by means of vibration or another form of counteracting force, for example. A braille display, developed for visually impaired users, can create a detailed relief on a surface, which enables the display of braille characters that can be felt by the tips of the fingers. The braille characters are usually generated with round-tipped pins raised through holes in a flat surface. Vibration actuators can be integrated into smartwatches or wearables. If they are integrated into clothing, a hug by another person can be simulated, for example. In another application, support for learning to play an instrument was demonstrated by this technology. Ultrahaptics is a technology that uses ultrasound to create virtual elements in mid-air. It can be used to create a virtual button to be pressed on demand, which the user can feel. If haptics actuators are embedded into an exoskeleton, they can be used to help (disabled) persons move, walk, or exercise (cf. [ShRP2019]).

Tangible interface

For a tangible interface, sensors are integrated into physical objects that sense, for example, the position, velocity, or acceleration of the respective object. Technologies for this purpose are motion or acceleration sensors or radio-frequency identification (RFID) units. The manipulation of such a physical object causes a digital effect in the application. Well-known examples are the motion sensors in games controllers.

A tangible interface does not necessarily have a single locus of control of interaction, which a mouse-controlled GUI does have. Multiple objects can be manipulated at the same time, for example, by more than one hand and potentially multiple users. In principle, a tangible interface does not enforce any strict sequence of commands or any modal interaction. This freedom helps to enhance the cognitive process of the user to understand and control the user interface. One example of a complex tangible interface is a tabletop with physical objects and digital information projected onto the table. If the objects are manipulated (e.g., moved) the digital information is changed. Such interfaces are used in urban planning, where scaled building models are moved on the table and the digitally generated effects, such as wind and shadows, can be observed through a digital simulation. Variants of interactive tables are tables where the tabletop is a large touch display (e.g., Microsoft PixleSense, formerly Microsoft Surface). It is important for such displays to be able to recognize multiple touches at the same time at different positions. Other

applications for tangible interfaces are toolkits for coding, electronics, and STEM (science, technology, engineering, and mathematics) subjects. Examples are small (inexpensive) one-chip computers that enable discovery learning, exploration, and collaboration when creating small applications with such devices. There are also special toolkits available made for the visually impaired that employ tangible interfaces (cf. [ShRP2019]).

Brain-computer interface

In a brain-computer interface, users control the computer with their thoughts. The neurons in the human brain work by transmitting small electrical signals between each other that change when the person thinks, moves, or feels. These signals are recorded by special headsets, hairnets, or hats. To use a brain-computer interface, users are trained to concentrate on a task and thus control the computer. A lot of research has been conducted in this area, but mass market products are not widely available yet. Applications comprise the control of games or electronic devices in real time. In a pioneering medical research application, such interfaces help paralyzed patients to be more autonomous (cf. [ShRP2019]).

Mixed reality

Milgram and Kishino [MiKi1994] categorized visual displays into the *virtuality continuum* (see Figure 49).

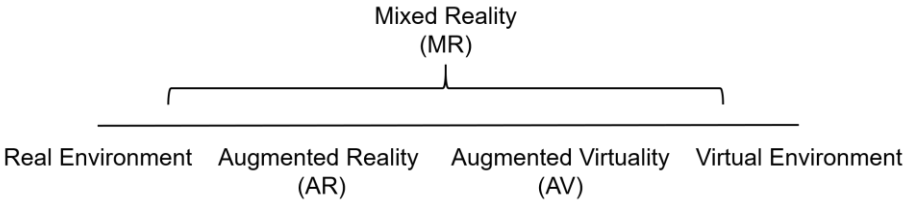


Figure 49 – The virtuality continuum according to Milgram and Kishino [MiKi1994]

At one end of this continuum there is the real environment, i.e., reality, and at the other end a completely virtual environment. Between these ends a combination of virtual elements and reality, which is called *mixed reality (MR)* is presented to users. One MR example is the concept of augmented reality (AR), in which artificial elements are added to the real world or images of the real world virtually. In an augmented virtuality (AV), representations of the real world are added to a virtual environment.

There are currently many interface technologies available that can be categorized as (1) virtual reality (VR), in which virtual environments are created, and (2) augmented reality (AR). More details about VR and AR are outlined in the following.

Virtual reality (VR)

Virtual reality technologies aim to create a virtual environment that a user can participate in. In many cases, the visual sense is stimulated by a stereoscopic display that presents two separate images to each eye and enables a three-dimensional visual impression. Such well-known displays are called VR headsets, with examples being the HTC Vive or Oculus Quest. However, other displays, such as LCD panels or projections, can be used together with special glasses to serve the same purpose. Besides the visual sense, VR technologies can also stimulate the acoustic sense by means of three-dimensional audio content. Unlike standard stereo presentations, the acoustic sounds can be placed at arbitrary positions within the three-dimensional space within a given physical room for multiple listeners. It is also possible to stimulate other human senses, for

example, by using haptic interfaces (see above) or olfactory interfaces. To create a good virtual experience, it is important to enable an interaction with the virtual environment, i.e., the user's head must be tracked such that any head movement or body position change adjusts the scene presented accordingly. The user must be able to manipulate (some of) the virtually presented objects of the scene, for example, by means of standard input devices or tangible or gesture-based interfaces. In this context, the term *immersion* describes the extent to which human senses are stimulated by virtual reality technology. If the degree of immersion is high enough and the user really wants to get engaged in the virtual environment, the user can reach *presence*, i.e., the psychological state that they are really *there* in the virtual environment. Usually, reaching presence is the objective when creating a virtual environment.

The general assumption is that virtual environments benefit from a higher level of realism. However, it is known that realism does not always generate a better presence. Even simple polygon-style or cartoon-like virtual environments can generate quite a strong immersion and presence. Examples are the arcade games from the 1980s or Disney cartoons.

There are many application areas for virtual reality technology. It is well suited for learning and training applications, such as aircraft operation or car or train driving simulations. A museum could show environments in past times in a much more immersive way using VR. Architects can build complete buildings virtually and walk through them together with their clients before they are built. Also, complex technical systems, such as a spacecraft, a car, or a process plant can be prototyped virtually. This allows the functions of the product to be simulated and optimized before the product is created.

Surgeons can be trained on virtual patients before the real operation takes place. There are therapy applications available that help to cure phobias—for example, fear of spiders—by means of confrontation therapy. VR applications can also help to reduce stress when talking in public or reduce posttraumatic stress disorders. When the user enters the situation of someone else through virtual reality, empathy could be created for the person actually in this situation, such as for a refugee who has fled to a foreign country and society.

There are many computer games that are adapted for use with VR technologies; some of them are multi-user games. Complex simulations can be found in entertainment parks—for example, a simulation of a flight to mars. Some airlines use VR for in-flight entertainment purposes or to advertise particular flight destinations of the airline.

One essential problem of virtual reality is cybersickness, especially motion sickness, which occurs for users. The reason for this effect is partly imperfect technology that mainly induces latencies in complex environments. For example, the head is moved and it takes time before the user sees the reaction to this movement in the virtual scene. However, there are also fundamental issues that create perceptual disparities—for example, if the virtual scene shows a strong accelerated movement of the user using an optical flow, such as flying stars, but in reality, the user is standing still. In this situation, the visual sense signals movement while the vestibular sense indicates no movement, which is a perceptual disparity that in turn has the potential to create motion sickness. As long as the user cannot move the same way in reality as simulated in VR, this situation can only be solved by a careful design of the virtual environment.

There are still research questions open: for example, whether it is necessary and important to have a representation of self, i.e., virtual parts of the user's body, in the environment or not. There is research ongoing concerning the most effective motion navigation through the VR that reduces

the potential for motion sickness. Also, the best paradigm and technology for interactions and movements—for example, using head and body movement, the use of a keypad, pointing devices or joystick buttons—is not fully understood yet. Moreover, there is still research ongoing regarding how to best collaborate and communicate with others in virtual environments.

From the games industry, professional 3D tools that help to create immersive virtual environments are available and are being continuously further developed. Examples include the Unity Engine, Unreal Engine, or CryEngine (cf. [ShRP2019] and [Jera2016]).

Augmented reality (AR)

Augmented reality (AR) is an example of mixed reality (see above). In AR, the reality is augmented by three-dimensional artificial elements and the user can interact with reality as well as with the virtual elements in real time. There are basically three types of augmentation possible. The most common form is (1) video see-through AR. In this case, a camera records images of the real world and the augmentation is computationally superimposed onto these images. The result is presented to the user via a display. This can be realized by using a smartphone or a special video see-through headset, which is essentially a VR headset equipped with a camera. If a headset is used, the display can be stereoscopic. (2) Optical see-though AR presents the reality to the user through a semi-transparent display or mirror. The artificial elements are added through the display or projection onto the mirror. An example of an optical see-through headset is the Microsoft HoloLens. In (3) spatial augmented reality, artificial information is projected onto objects of the real world via video. This can be used, for example, to guide a climber thorough a path by means of a projection onto a climbing wall (see [ScHo2016]).

The AR application that is perhaps the most popular is the AR game Pokémon Go from 2016, where players had to look for virtual characters that they could find by moving their smartphone to a certain location. A large application field of AR is medical engineering. Medical images, such as X-ray, computer tomography (CT), or magnetic resonance (MRT) images, can be projected onto the body of patients to plan an operation or even to guide surgeons during an operation. Several therapy and training applications are available within this field. Superimposed diagrams can help technicians to quickly find errors in and repair complex equipment or installations in buildings. AR could help controllers and operators in quick decision-making—for example, air traffic controllers have screens with overlaid information about aircraft movement. Head-up displays (HUD), which are optical see-though displays in military and civil aircrafts, can display important information about weather conditions, for example. In modern cars, HUDs are integrated into the windscreen, where, for example, navigation information is displayed. Such navigation information displayed on a smartphone together with real images of the environment can aid people walking in a city or town and help them find (touristic) points of interest.

A popular entertainment application is the filters on the social media application Snapchat. Users can change or deform their own image or add items to the original image, such as big ears or necklaces. This technology of an AR mirror can be used in virtual try-on applications for sunglasses, jewelry, or make-up, for example. Obviously, the drawback is that the virtual elements can only be seen and not physically experienced.

Challenges for AR include where to overlay which information and how to limit the information to an extent that it is still useful for the user and does not distract or overload their cognitive capacity. Technological challenges include exact capture of the real environment and a perfect integration (registration) of the virtual elements into the reality. The geometric adaptation (geometric

registration) of the virtual objects to the reality—which employs tracking technologies—works quite well already but the adaptation to the lighting conditions (photometric registration) is still challenging for most AR applications. A lower quality of the superimposed elements might be acceptable for entertainment purposes, but the quality must be high for military and medical applications. In particular, if the user or the scene is moving, the situation might become difficult and potentially require a lot of computing power to keep track in real time (see [ShRP2019] and [ScHo2016]).

Ambient interface

By using the interface types described above, a user interacts explicitly and intentionally with a system. Today, there are more and more situations in which a user interacts only implicitly or unintentionally with a digital system. These interactions may appear to show the digital system acting automatically without user input but this is not the case; it reacts in a predefined way to the user's actions. When a user enters their apartment, for example, the home automation system may switch on several lights and other devices and may even perform more sophisticated actions. To enable such features, the digital systems need to recognize the current situation by using (sophisticated) sensors. Therefore, the DDP should understand basic sensing technologies to be able to incorporate them into the design of a digital solution. Examples of such technologies are audio sensors (e.g., microphones), video and light sensors (e.g., cameras, infrared detectors), position sensors (e.g., GPS, GNSS), accelerometers, gyroscopes, and pressure or temperature sensors.

Combination of types

All user interface paradigms and types described above can be combined to tailor the appropriate interface to the target application. For example, augmented or virtual reality applications often include voice, motion, or gesture interaction.

Some of these interface types are already used in standardized end user devices. The selection of an interface type for a custom-made end user device is an important decision for the design of a digital solution and can only be revised at high cost in the further course of the building process. If different interaction forms and interface types are considered, the suitability of these alternatives for a digital solution should be investigated by the use of prototypes (see Section 2.3) in order to reduce the risk of a wrong decision at an early stage.

3.2.3 Software User Interface Technology

User interface (UI) technology falls into the category of software for the perceivable form and function. It determines, for example, both the visual structure (form) of the user interface and the dynamic behavior of the user interface (function). Examples are as follows:

- Windowing, scrolling, zooming
- Speech synthesis
- Speech and gesture recognition
- Software-enabled metaphors, such as pick, drag, and drop
- Software-enabled virtual devices, such as virtual keyboards, virtual sliders, or virtual analog instruments

User interface technology for other sensory modalities also falls into this category. Currently, technologies for audio and haptics input and output are important in addition to visual interfaces.

To run applications on the device categories named in Section 3.2.1, there are operating systems such as Android, iOS, or Windows. Such a framework that is mainly composed of a device and an operating system is called a computing platform or platform.

Technologies for developing user interfaces

For the platforms used most frequently, software development environments are available to support the creation of graphical, audio, or haptic user interfaces. They offer libraries that allow easy access to the features of the device via an application programming interface (API). Such features include showing content on the display, detecting touch events, playing and recording audio via the loudspeaker and the microphone, activating the vibration function, or using the voice assistant.

For most devices, the graphical display output is the most complicated and largest part of the interface. Modern development environments offer graphical editors that make it easy to compose layouts for the display content graphically via mouse interaction. Predefined visual elements, such as buttons, text input fields, and list or image views, are available to quickly create an output screen layout without any additional programming effort. This helps to create a similar look and feel within an application and across different applications on a device. Such editors usually allow a flexible layout. This flexible or automatic layout moves and scales screen elements (at runtime) accordingly such that the application has a nice look and feel regardless of the device on which it is running. Platforms usually allow devices of different display sizes and densities, which makes such automatic adaptation necessary to reduce the development effort.

If a screen layout is created in such an editor, the visual composition is automatically translated into a programmable description that can be manipulated by a programmer and incorporated into the functionality of the application, i.e., a programmable interface to the source code of the application is available. The Extensible Markup Language (XML) is popular for such a description but other markup languages or native program code, such as C#, C++ or Java, can be used as well.

This direct translation of the visual description into program code allows for a division of staff between a user interface or interaction designer and a programmer during development projects. In addition, it helps to reduce the boundary for either a user interface designer or a programmer to take care of or even work in the other discipline.

Modern development environments for application development for mass market devices such as smartphones, notebooks, smart watches are very well documented to reduce the barrier to starting development for a certain platform. This enables an easy start for developers who have little or no experience with the respective platform, or even for non-professional developers.

Application types

For the most frequently used platforms Android, iOS, MacOS, and Windows, we can distinguish between four types of application: (1) native, (2) web, (3) hybrid, and (4) cross-platform applications.

Native applications are applications developed for a single target platform using the dedicated development environment created for this platform. For example, the Google Android Studio is used for Android applications, Apple Xcode is used for iOS applications, and Microsoft Visual Studio is used for Windows applications, employing the respective programming language or languages. The executables of such applications work only on the respective platform and are

very well integrated into the execution environment. The application can be very well integrated into the operating system's user interface and there is excellent access to the resources of the device, such as the microphone or the GPS sensor. The performance can be optimized for the platform and even for certain devices. The development environments for modern platforms provide various means to support the adaptation of the user interface to the individual features (e.g., the display size and density) of the device within the respective platform. The result of this mechanism is similar to the responsive web design for web applications (see below).

Web applications are basically a website tailored for output on a dedicated device. If the application runs in the browser of the device, there are mechanisms available for detecting the properties of the device, such as the operating system, display size and density, and features of the device. With this information, the web application can adapt to the device and, for example, scale the display elements accordingly. For example, the same web application can be used on a notebook with a 17" display and on a smartphone with a 5" display. If the web application supports this mechanism, it is a called a responsive web application. The design for such responsive applications is referred to as responsive (web) design.

The technologies most frequently used to realize web applications are the Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript (JS). HTML is a language—similar to a programming language—that includes semantic information into the content. It can define how such content should be formatted on the display of an output device. For example, it uses special tags to define where a paragraph starts and ends, which part is a heading, and which text parts shall be emphasized (as a candidate to be set in bold font). CSS is a language that can be used to describe how documents are presented. It can define, for example, the colors, and fonts. CSS helps to separate the presentation of the content from the actual content. JavaScript[12] is a programming language that is used to define interactive elements on websites. Besides these three fundamental and widespread technologies, there are many other web technologies with additional features. For instance, there are many JavaScript frameworks available (e.g., as plug-ins) that serve different special purposes for web applications, such as diagram/graph-modelling frameworks (e.g., GoJS), questionnaire frameworks (e.g., formr), or form frameworks (e.g., formanizr).

As web applications run within a browser that has been implemented and optimized for a target platform to efficiently process HTML, CSS, and JavaScript, they are very flexible and run on several devices and operating systems. On the other hand, they mostly require a continuous internet connection to a web server. It is possible to save some data locally on the device but compared to native applications, this memory (web space) is quite limited. As these applications and their data are hosted on a server, it is easy to keep these applications up to date by modifying the source code on the server only. The updated application is downloaded automatically the next time the web application is started. Compared to native applications, the access to certain device features is limited. Web applications are basically a link to a website and are therefore not available in app stores.

Hybrid applications are a combination of native and web applications. One realization of a hybrid application is to use web technologies to program the application and let this code run in a browser control. This browser control is a user interface element of the target platform and can be integrated into a native application (application base). There are tools available for the creation of

[12] The programming language JavaScript is different from the popular programming language Java.

such applications that automatically create native executables for (multiple) selected platforms. This means that several native applications can be created from one single code base for several different platforms. Hybrid applications can be as flexible as web applications, have more local storage, and provide more access to device features than web applications. On the other hand, they are less optimized for a certain platform and therefore their performance might be lower compared to a native application. Today, many applications can be downloaded from application stores that use such hybrid technologies.

The idea of cross-platform applications is to use a single code base to implement the application. The programming languages C#, C++, and JavaScript are often used for such application development schemes. A special tool (e.g., Xamarin) automatically creates native applications based on this source code. Unlike hybrid applications, this source code does not run in a browser. Cross-platform applications usually use native user interface elements of the target platforms and operating systems. This technique allows approximately 75% of the source code to be shared across different platforms. Cross-platform applications share all advantages of hybrid applications and potentially offer a better user interface integration into the target platform and a better performance than hybrid applications. On the other hand, the code base (today) is not completely the same across platforms. It might require platform-dependent, operating system-dependent, or even device-dependent source code parts.

Table 17 summarizes the characteristics of the different application development approaches explained above.

Table 17 – Characteristics of different application development approaches

	Native	Web	Hybrid	Cross-platform
Feature access	high	low	medium	medium
Performance	high	low	low	high
User interface integration	high	low	low	high
Development simplicity	low	high	high	medium
Application store delivery	yes	no	yes	yes
Maintainability	low	high	medium	medium

Games engines can be considered as a special form of cross-platform application creation tool. They offer modelling tools for creating three-dimensional environments with a minimum of programming effort, usually by means of a graphical editor. If needed, *scripts* are a powerful possibility for adding rich interactivity or special functionality to the applications. A single code base can be automatically converted into native applications that run on multiple different platforms. Complex and successful games have been developed in the past with this technology. Games engines are designed and developed for the creation of video games but they are well suited for other applications outside of the games industry as well—for example, for the development of augmented or virtual reality applications (see Section 3.2.2).

Considerations for daily work

The selection of a software user interface technology is a critical decision. It defines which possibilities and application features are available, how the application must be maintained in the future, and which dependencies to third-party tools exist. Replacing such user interface technology late in the building process or in an existing digital solution can be very expensive, since this technology is typically integrated into a digital solution very deeply. Therefore, the user interface technology should be selected with care and the DDP should be involved in the selection process in order to obtain the best possible technology for the design of the solution.

If the creation of prototypes of the user interface of a digital solution (idea) goes beyond simple horizontal prototypes (see Section 2.3), the user interface technology (potentially) selected should be incorporated in such a way that the capabilities of the technology are considered in the design of the user interface. This prevents the development of designs that can only be realized with a disproportionately large effort.

3.3 Underlying Technology

Understanding the underlying technology is as important for the design of a digital solution as understanding the perceivable technology. The main reason for this is that underlying technology is used to build those parts that enable the perceivable form, function, and quality of a digital solution (see Chapter 1 and Section 3.1).

Underlying technology is beyond the parts of a digital solution that a user has direct contact with. Nevertheless, underlying technology may have a significant influence on the solution or system design and thus influence the user experience significantly but indirectly. A DDP needs to have at least a basic understanding of underlying technology to identify parts of the design that may be hard to realize. Furthermore, a decision about a specific architecture style or a decision about a framework that will be used may have already been made by other stakeholders such as system architects or even another DDP from a previous project that the current project is based on. The DDP has to consider these decisions and the consequences in the solution or system design. Knowing underlying technology enables the DDP to talk to construction and realization experts such as software architects or developers and understand their arguments.

To illustrate the importance of underlying technology, let us take another look at our YPRC example.

YPRC example. An important feature of the YPRC digital solution is the remote coaching function. Although the direct function is perceivable (the runner's coach gives coaching advice to the runner), several important aspects of this function are realized by underlying technology:

- The voice communication requires a data connection between the runner's app and the coaching portal (underlying form).
- The runner's training data (speed, position, health data) must be captured (underlying function), stored (underlying form), transferred (underlying function), and prepared (underlying form) in such a way that the coach can use it for coaching purposes (perceivable function).
- The purchase of a coaching session requires payment. The payment itself is realized by a payment provider. The connection with a payment provider system is also part of the underlying form.

Even this apparently simple example shows the importance of understanding the underlying technology for designing digital solutions. A DDP at foundation level should be aware of the following aspects of underlying technology: programming technology, technology for operating software, and supporting hardware. We discuss each of these aspects in the following subsections.

3.3.1 Programming Technology

We have already introduced programming technology as material for implementing the perceivable form and function of a digital solution (see Section 3.2). However, programming technology also determines a significant part of the underlying form, function, and quality of a digital solution.

Beginners often intuitively associate programming technology with programming languages. This often leads to the misunderstanding that writing program code is the central activity in the development of software. This is mainly due to the fact that entering program code in development environments is the main visible activity performed by software developers. However, the central activity in the development of software is rather the creation of a suitable structure in the chosen programming technology, which actually realizes the desired form, function, and quality. This competence belongs to the activity areas construction and realization and is not part of the competence profile of a DDP at foundation level.

For a DDP at foundation level, the understanding of perceivable and underlying form, function, and quality already introduced is sufficient as a foundation for communicating the desired digital solution to experts from construction and realization.

However, programming technology goes beyond programming languages. It offers a rich set of features that a DDP at foundation level should be aware of. We consider the following aspects as important: data storage technology, software frameworks, and API technology.

Data storage technology

Storing and processing data is at the core of digital solutions and is the duty of data storage technology. Digital technology has developed various approaches and tools for this purpose. Wikipedia gives a great overview of the history of databases [Wik2020a] and the different technologies available [Wik2020b].

Considerations for daily work

Each data storage technology has certain advantages and disadvantages in terms of the volume of data that can be processed, the speed of operations, and the flexibility with respect to modifications of the data structures. When discussing data storage technology with technology experts, the DDP should have a very clear picture of the data that the digital solution shall store and process. The DDP can obtain this understanding with the definition of entities and functions from the element design concepts (see Section 2.2). A DDP should be aware that storing data is expensive (e.g., due to realization effort or cloud storage feed). Therefore, when defining data, a DDP should always consider whether the data is really needed and needs to be stored.

Software frameworks technology

Software frameworks have been invented to provide reusable components. Using software frameworks improves the quality of digital solutions since good frameworks have already undergone a thorough quality assurance process. They also speed up the development process since the functions provided by the frameworks are ready to use and do not need to be implemented manually (again).

Considerations for daily work

The selection of the proper software framework is a task for construction and realization experts. Nevertheless, a DDP at foundation level should be aware of the fact that software frameworks can provide important underlying functions that can be important for realizing a digital solution. In the following, we provide a number of exemplary frameworks. The goal of the following list is to show some interesting frameworks for beginners in Digital Design and to illustrate the broad scope of frameworks available:

- *Apache Hadoop* (https://hadoop.apache.org) is a framework for processing large (big) data sets.
- *Camunda* (http://www.camunda.org) is a framework for creating and automating workflows and processes.
- *ElasticSearch* (https://www.elastic.co/) is a framework for searching and analyzing large data sets.
- *Spring* (https://spring.io) is an application framework that provides several functionalities for developing applications (e.g., communication with databases).
- *TensorFlow* (https://www.tensorflow.org) is a framework for machine learning and part of artificial intelligence technology.
- *Unity* (https://unity.com) is a framework for developing two-dimensional and three-dimensional games and applications, including virtual and augmented reality (see Section 3.2).

The main message of this list for the reader is: study frameworks regularly to know what is possible. A DDP should be aware that despite the advantages of using frameworks to save the realization of standard features, there are also some disadvantages of reuse as the reusable features may not be a perfect fit to the current digital solution under development.

Considerations for daily work

By having at least a basic understanding of existing frameworks, the DDP can decide, together with construction and realization experts, whether to use existing frameworks or instead, to spend the time and effort to build a framework exactly as imagined. Furthermore, by using a third-party framework, you tie yourself to a company that you cannot control. The company may change their pricing or licensing model over time to conditions you may no longer be able to accept. You also rely on this company to maintain their framework well and to stay competitive over time. Changes in a third-party framework may force you to change your system accordingly to ensure that it works properly or even runs at all. This could be a time-consuming activity and may be very expensive over time.

API technology

API technology is an approach whose goal is to offer functionality that can be integrated into your own digital solution via a technical interface. There are many different ways of integration but today, when we talk about using or offering an API, we usually refer to an API that is used via the World Wide Web. A technical name for this is *web service*. The YPRC case study makes use of two web services: the map server and the payment provider.

Similar to frameworks, there are a large number of different web services available. The following is a list of some exemplary web services:

- *GraphHopper* provides web services for route planning and route optimization.
- *ApplePay, GooglePay, and PayPal* provide web services for payment transactions.
- *Melissa* is a framework for validating addresses all over the world.

Considerations for daily work

The main benefit of API technology is that the functionality provided is ready to use for a digital solution. However, the business model of the digital solution must take the cost for using web services into account. As with frameworks, a DDP should be aware of the same disadvantages that come with using external APIs. Moreover, you are much more likely to adapt frameworks to better fit the indented digital solution than adapt APIs, as they are fully under the control of an external party.

3.3.2 Technology for Operating Software

Computing technology provides an important part of the infrastructure for building a digital solution. It consists of computer hardware, with processors, memory, and storage as the typical building blocks, and operating systems. Computer hardware is mostly built in large volumes as a standardized commodity.

Specialized hardware (e.g., for data encryption) is used when certain quality requirements (in particular, speed and security) cannot be achieved with standard hardware.

Operating systems are required to manage the computer hardware, provide basic software services such as organizing hardware storage with a file system, and also to provide an environment for running application software.

Hardware and its operating systems can be provided in several ways:

- Part of a standard device (e.g., a smartphone or white label components)
- Part of a custom-made device (e.g., a do-it-yourself smart home controller)
- Local server (e.g., a desktop computer)
- Remote server (e.g., in a data center)
- Service on-demand over the internet (cloud computing)

From a Digital Design perspective, it is important to recognize the broad scope of technology available for operating a digital solution and the different levels of communication technology that may be necessary for the digital solution (see Section 3.3.3).

In order to illustrate this, let us discuss some examples. We assume a medium-sized company producing pasta sauce with a single factory site. The company wants a digital management solution for their production process.

It is possible to create the required infrastructure as an on-premise infrastructure with local servers in the basement of the factory and a closed network between the clients and the server. Such a solution is closed to the outside world and does not require a connection to the internet. There is a risk of loss of data if the servers are destroyed (e.g., through a fire). Furthermore, the hardware must be maintained by an administrator.

The same company can pay for the required infrastructure and use a data center or a cloud service to operate the servers (infrastructure as a service (IaaS)). This operation mode requires an internet connection and relies on a working internet connection in order to be functional. When using IaaS, the company can either decide to realize the management solution exactly as for an on-premise infrastructure or they can build their management solution on a development platform provided by the cloud provider (platform as a service (PaaS)). In doing so, the company uses frameworks and APIs to realize and use common features. As another option, the company could decide to use an existing management solution as a service that is offered by another company (software as a service (SaaS)), regardless of whether this solution is realized on-premise or in the cloud.

Considerations for daily work

A DDP should be aware of the basic advantages and disadvantages of using on-premise infrastructure, IaaS, PaaS, or SaaS for the design, construction, and realization of the digital solution as well as for its business model. For example, if the internet connection is lost temporarily, the factory may be unable to operate. It could be possible to design the system in such a way that it can operate with limited functionality without the server connection. An alternative approach could be to use a backup internet connection (e.g., via mobile internet) to keep the solution up and running. A local installation may be robust against internet failure compared to the remote servers. However, the cost for operating the servers locally and the risk of data failure may be higher than the costs for potential internet failure. It may even be the case that developing a kind of *offline* mode for the remote solution is more expensive than the actual costs for internet failures.

3.3.3 Digital Communication Technology

A core feature of the digital age is connectivity on all levels between users and devices. Most digital business models rely on the ability to offer services all over the world (see Section 4.2). The backbone of this capability is the underlying technology that enables this digital communication.

Communication technology consists of communication hardware, such as cables, antennas, radios, receivers, etc., which is operated by a stack of protocol layers that are realized with computer hardware, and communication software. Together, they provide communication services at various levels, for example:

- Basic services such as Ethernet, WLAN, Bluetooth, and mobile, cellular telephony including 5G, radio-frequency identification (RFID), near field communication (NFC), and infrared (e.g., for face recognition)
- Network services such as the internet or the network that connects phones when a number is dialed
- Application services such as WWW or email

A DDP needs to know the basics of form, function, and quality of underlying computing and communication technology, for example:

- Form: which technologies are available on which devices?
- Function: which services can be provided by these technologies?
- Quality: what is the quality of these services in terms of speed, storage capacity, communication bandwidth, availability, reliability, etc.?

From a Digital Design perspective, communication technology is a ready-to-use technology. However, in terms of form, function, and quality, selecting the proper communication technology has a major impact on a digital solution.

Communication technology in relation to form and function

In terms of form, communication technology enables the form of a digital solution that consists of more than one element. As mentioned above, almost every digital solution consists of more than one element. Different communication technologies allow for different forms. Short-range technologies such as Bluetooth or WLAN allow local networks and also allow elements within a short range to connect without additional costs for the user. A good example is the Bluetooth connection between the runner's watch and the runner's app.

When it comes to long-range communication over the internet, a digital solution will create additional costs. The user of the solution must have an internet connection. Furthermore, the provider of the digital solution must also have an internet connection. Although internet connection is a commodity today, the additional costs should be considered in the business model, especially if the digital solution may create high-volume data transfer. A good example is a video streaming solution. The costs for the data connection for the client and the provider can be substantial since video streaming is a data-intensive function. In terms of function, the communication technology is normally invisible for the user since it transports data between elements. However, when it comes to a connection failure or to a weak connection, a well-designed digital solution can adapt itself to this situation. The concrete method of dealing with communication issues depends on the type of solution. In general, it is possible to define functions in such a way that they can cope with lost connections and restart their work when the connection is available again. A critical issue in terms of communication in the YPRC case study is the remote coaching feature, which relies on a constant internet connection.

Communication technology in relation to quality

In terms of quality, the communication technology has two important factors: bandwidth and reaction time. The bandwidth defines the volume of data that can be transported between two elements in a given time. The bandwidth becomes especially important when larger volumes of data have to be transferred from one element to the other. A low bandwidth will reduce the speed of a function significantly in such a situation. Again, a well-designed digital solution takes this into account. The reaction time of the communication technology becomes important when functions are distributed over different elements. Here, the YPRC case study is also a good example. The remote coaching requires that the runner's health data be transferred in near real time to the runner's coach in order to allow the coach to get a real-time view on the runner's health. If this data transfer were to take a minute, the overall coaching experience for the coach and the runner would be weak, as one minute is a long time when running.

3.4 Technology-Oriented Knowledge Areas

We are aware that the DDP cannot be an expert in all areas of system and software development. However, in the following section, we present three knowledge areas which we consider to be highly important for a DDP at foundation level:

3.4.1 Software Architecture

Software architecture deals with the definition of the fundamental organization (i.e., underlying form and function) of a software system and is an important aspect of the construction and realization of a digital solution.

The architecture of a software system can be understood as a metaphor, analogous to the architecture of a building [PeWo1992]. Like the architecture of a building, the definition of a software architecture is about making *fundamental* decisions regarding structural choices. There are many architectural styles and patterns that can be used to build systems. It is not the goal to cover them all here, but rather to give an overview of selected ones and make the reader understand what aspects of software architecture can also be relevant for a DDP.

These architectural styles include *monolithic* systems, which represent a single-tiered software application that combines user interface and data access code into a single layer. However, nowadays, applications with multiple layers are more common. Such *multi-layered architectures*, for example, can consist of the presentation layer (i.e., the graphical user interface), the application or also called logical layer, providing the actual functionality of the application, and the data layer (e.g., the underlying database). This separation also allows a focus on a particular level of abstraction in the building process. An *event-driven architecture* is the dominant style when it comes to graphical user interfaces. Here, we typically find event emitters and event consumers. Event consumers are notified when events of interest occur and they need to react. For example, in a GUI, the push of a button (e.g., send order) can trigger an event and cause event consumers (e.g., the warehouse system) to react (by, for example, shipping the order and updating the inventory list). Last but not least, there are *service-oriented architectures* (SOA), such as microservices, which are based on loosely coupled services to arrange an application [JPMLT2018].

Selecting a proper software architecture is of great importance for achieving perceivable and underlying quality (see Section 2.1.3). This means that to some extent, the software architecture defines perceivable quality attributes such as usability, security, reliability, and availability, and underlying qualities such as maintainability and extendibility.

Once a decision on a suitable software architecture has been made and the system implemented, making changes to the underlying architecture is costly. However, defining an underlying architecture allows a detailed analysis of the software system’s behavior before the system is actually built [PeWo1992]. This analysis also allows us to understand whether the future software system will fulfill the desired perceivable and underlying qualities and thus the requirements of the different stakeholders. The possibility to understand the consequences of an architectural choice at an early stage can help to reduce risks and save costs [OKKP2015].

This understanding is especially important for fostering efficient collaboration between design and construction during the building process.

3.4.2 Computational Complexity

Computational complexity theory deals with the question of the quantity of computational resources required to solve a given problem [Wegn2005]. Designing a digital solution includes being aware of its complexity. A DDP can seek advice from experts if a digital solution deals with complex problems but ideally, the DDP will already have a basic overview of relevant topics regarding complexity theory. In the following, we provide a high-level overview of that topic. The explanations are strongly based on and aligned with [Wegn2005].

The key focus of computational complexity theory is solving computational problems. These computational problems can be solved by an actual computer but there are also different models of computation such as Turing machines or cellular automata that have equal computing power.

In order to be solved, a computational problem needs to allow for the mechanical application of mathematical steps. This finite sequence of computer-readable instructions is called an algorithm. Algorithms are an integral part of digital systems. They are abstract descriptions for solving well-defined functional problems. All algorithms have inherent time and space boundaries that need to be considered when deciding which algorithm to use to solve a specific problem. This also highlights another important aspect of computational complexity theory: the classification of computational problems. This is done by investigating the amount of time or space (i.e., memory) required to solve a computational problem.

The *big O notation* is commonly used to classify algorithmic complexities and describes the execution time required or the space used by an algorithm. The big O notation defines an upper boundary for an algorithm, which basically describes the worst computational scenario. O(1) refers to an algorithm that, regardless of the input size, will always execute in the same time (or space). This is the ideal case. O(log n) means that by doubling the input, the time and space needed increases only by log(2). An example of such an algorithm is the binary search algorithm. An algorithm with the complexity of O(n) refers to a linear growth, meaning that the time and space complexity grows in proportion to the input. For example, if you double the input, the algorithm needs twice as much time for the calculation. An example of O(n) would be finding an item in an unsorted list. $O(n^2)$ refers to an algorithm with quadratic time complexity, meaning that it scales poorly. For example, if you increase the input size of such an algorithm by the factor 10, the time needed for the calculation increases by the factor 100. Even worse is $O(2^N)$, which refers to an algorithm with an exponential growth rate, starting off with rather small growth numbers and then rising dramatically.

Different solutions exist (e.g., sorting algorithms) for several problems. Understanding the worst-case scenarios regarding an algorithm's time and space complexity avoids the accidental or unintentional use of an inefficient algorithm. It also supports developers in selecting algorithms that are in line with the stakeholder needs and satisfy requested perceivable quality attributes such as performance. However, there are also problems that cannot be solved appropriately with available computing resources or it would be too costly and time consuming to solve them. For such problems, there is sometimes an option to work with approximations and to approximate optimal solutions to such problems, which requires less effort and costs [WiSh2011].

3.4.3 Human-Computer Interaction

Human-computer interaction (HCI) is a knowledge area that is “concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them” [Hewe1992]. It includes knowledge from multiple disciplines such as psychology, computer science, and design. This means that HCI is about connecting humans and machines by considering human needs (see Section 4.1) and providing adequate interaction technologies (see Section 3.2.2). It includes understanding and improving existing interfaces, for example, by observing how humans interact with computers using a particular interface. Furthermore, it is the goal of HCI research to come up with new technologies and novel innovative means of interaction (e.g., augmented reality).

Overall, human-computer interfaces have undergone significant changes in the last two decades. This includes the emergence of mobile and wearable devices and ubiquitous technologies. Furthermore, social media has changed the way people interact with computers. However, graphical user interfaces (GUIs) still remain the dominant means of interaction [Hewe1992], although there are many more types of interaction technologies available, as discussed in Section 3.2.2.

From the perspective of Digital Design, HCI and the user interface mainly influence the perceivable form and function of a digital solution that deals with the immediate interaction between the digital solution and the user. For the DDP, it is important to have an overview of existing interface types and to understand the specific characteristics of these interfaces from the user perspective in order to select the most suitable interface for a digital solution for a specific user and context and to also optimize the digital solution for that selected interface. In other words, this means improving the usability of the digital solution while also considering the characteristics of the interface [Grud1992]. This also includes considering personal needs of particular user groups and making digital solutions accessible and personalizing them. Digital solutions can be improved by considering different principles of design [LWLB2017].

Means of evaluating a digital solution from a user’s point of view include performing usability tests, which are a common way of testing a digital solution on users and understanding how real users actually use the system [Niel1994]. Usually, usability tests are conducted with a small sample of participants and are moderated. To run usability tests on a large scale and at low cost, different methods such as remote unmoderated usability tests exist. Furthermore, user feedback approaches, which often include built-in feedback mechanisms, allow users to communicate their needs directly to developers [Orio2018].

3.5 The Digital Design Perspective on Technology

To conclude the presentation of digital material, let us go back to the Digital Design perspective. From the perspective of Digital Design, the following factors are important:

1. *Compatibility with the intended context*: Technology always requires a certain technical context. Some technologies are available only on certain operating systems. Such a decision can limit the potential market of a digital solution.
2. *Legal constraints*: Some technology (e.g., open source and web services) may have legal impacts on the digital solutions. For example, using web services that are hosted in another country almost always creates issues related to data protection and privacy legislation because user data is partially transferred to another country.

3. *Capability to implement the intended functions*: The technology selected must be capable of implementing the functions intended. This may sound like a no-brainer, but certain detailed functions (e.g., in the context of artificial intelligence) may require highly specialized frameworks.
4. *Inspiration for additional functions*: When selecting a certain technology, a DDP should always study the documentation of a technology or discuss the capabilities of a technology with experts. This may lead to new insights and new ideas regarding the additional functions of the digital solution.
5. *Reuse vs. implementation risk*: Using technology that already exists, especially frameworks and web services, offers the benefit of reuse (see above). The price is that the technology must be used as designed by the provider. Manual implementation can be beneficial in terms of flexibility but comes at the price of additional costs.
6. *Availability of skilled personnel*: The best technology is useless if no skilled personnel is available that is capable of applying the technology. Selecting a certain technology must therefore go hand in hand with searching for competent personnel. Alternatively, the planning of the building process must consider the time and costs for training the available personnel in the new technology and also a slower development pace because of the newly trained personnel.
7. *License costs*: Technology vendors often charge licensing fees for use. These license costs must be considered in the business model.

Factors 1–5 address the technical feasibility of a digital solution. From the Digital Design perspective, they are important for selecting a good technology for the functions intended and can also be a source for innovation. Factors 6–7 address the business model and the overall building process for a digital solution. The costs for manual implementation, for skilled (or unskilled) personnel, and for licensing the technology must be considered in the business model and the overall plan.

The list of factors ultimately shows that technical decisions are closely intertwined with design decisions related to a digital solution. From a Digital Design perspective, it is very important to understand these dependencies. This understanding is not so much important in terms of making technical decisions for a digital solution; rather, it is important for communicating with technical experts and for recognizing a good point in time when additional technical expertise is important for making design decisions.

4 Cross-Cutting Competences

Besides having design competence (see Chapter 2) and competence in digital material (see Chapter 3), Digital Design involves understanding several cross-cutting competences as well. In this chapter, we introduce three cross-cutting competences that we consider important for a DDP at foundation level:

- Human factors (Section 4.1)
- Digital business models (Section 4.2)
- People management (Section 4.3)

4.1 Human Factors

Human users are a core aspect of almost every digital solution. Research in psychology and related domains shows that human behavior and experience are complex. Moreover, the capabilities of human beings have certain limits, for example, when it comes to sensation, perception, and attention or selection and execution of actions. However, the users' attention, their actions, and their emotional responses play a key role in the interaction between them and a digital solution. A DDP should be aware of these facts in order to consult experts and to use prototypes to evaluate and improve digital solutions with regard to human factors.

4.1.1 Fundamentals of Human Sensation and Perception

Human senses connect humans to the external world. This allows humans to interpret what is happening around them and to react accordingly. Sensory receptors receive information about the environment, including the output of digital solutions, in the form of stimuli. For example, rods and cones in the human eye receive light waves. The receptors respond to these stimuli by converting them into nerve impulses that the sensory nerves carry to the brain. This chain of receiving and forwarding information is called sensation. In the brain, the sensed stimuli are interpreted, for example, as an image of a tree. This step is called perception. Perception follows sensation, but sensation and perception are interrelated processes: without sensation, perception would not be possible, as the brain would not receive any information to be interpreted. And without perception, the sensation would be only a collection of meaningless information related to physical stimuli.

Considerations for daily work

The DDP should be aware that the presence of a stimulus provided by the digital solution, such as an icon on a smartwatch display or an auditory alert from the smartphone, does not necessarily mean that the users actually sense and really perceive this stimulus.

One reason for this is that human senses are limited. Affecting all humans, the human senses have physiological limits (*Sensory capacity* in Figure 50). For example, without tool support, humans cannot see infrared light. Moreover, inter-individual differences exist, such as inborn color blindness or inborn hearing loss. In addition, short- and long-term intra-individual differences can occur—normal ageing decreases the functioning of the senses, for example, often gradually. Although the DDP cannot influence the functionality of the senses, the DDP should be aware of and consider the sensory limitations of the (different) users (cf. accessibility). For example, the DDP can use a color blindness simulator (e.g., https://colororacle.org/) to simulate how color-blind users see the output of the digital solution and can thus improve the digital solution if necessary.

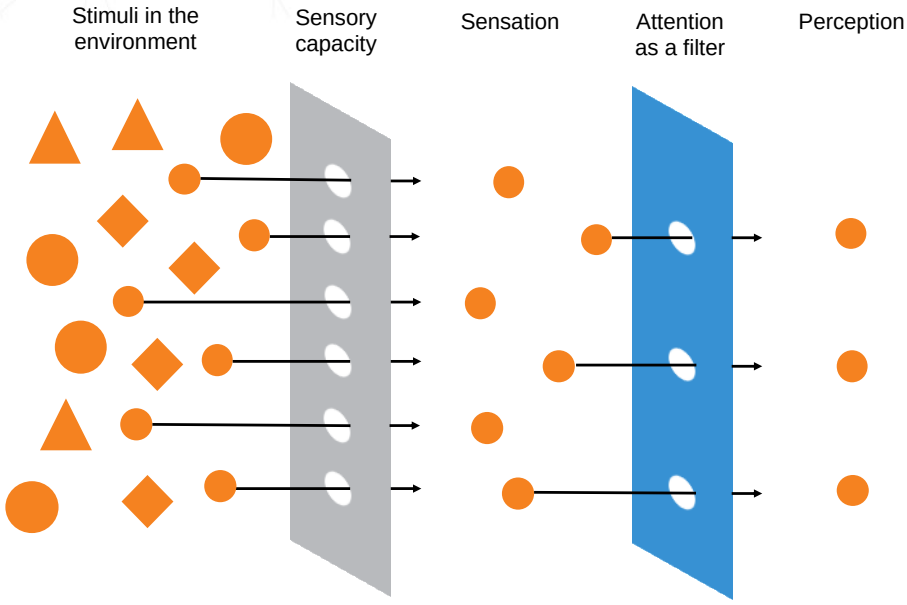


Figure 50 – Simplified model of sensation, attention, and perception

Another reason is that only a small(er) amount of sensed information is actually perceived. Attention steers how many and which stimuli reach the human perception. Thus, attention is also called *filter* [WHBP2016] (*Attention as a filter* in Figure 50), as discussed in the next section.

4.1.1.1 Visual Attention

Most of the information received by the human brain comes from the eyes. To see, humans have to move their eyes and focus because humans can only see if the light entering the eye is focused on the retina. This is called *visual selective attention*. According to [WHBP2016], several factors influence where the eyes look and thus, which details of a scene are noticed by humans. These influencing factors include the extent to which a stimulus stands out from the background (i.e., salience) or the probability of where relevant stimuli might occur. Although visual selective attention is needed for seeing, it can have negative consequences. For example, in the case of *change blindness*, changes in the environment are not noticed [WHBP2016]. This is illustrated in several entertaining video clips: while a passerby is involved in a conversation with an interviewer, a big piece of furniture is carried across the street. At that moment and unseen by the passerby, the interviewer is replaced by another person. Most of the interviewees do not notice that another interviewer is continuing the conversation.

Considerations for daily work

Examples of change blindness in a digital solution include missed system feedback, such as the low battery symbol blinking in the menu bar while the user is watching an embedded product video on a website, or the error message that appears next to the box left unchecked at the top of the booking form while the user tries repeatedly to click the submit button at the end of the form. The changes in the interface (e.g., notification or error message) are not seen, for example, there are too many changes at the same time (i.e., video example) or the changed element (i.e., the error message next to the checkbox) is too far away from the current focus of the user (i.e., the submit button at the end of the form). According to [Budi2018], there are several design aspects that help to prevent change blindness. For example, the website with the video could be dimmed to attract attention to the changed battery level symbol, or a second error message could appear next to the submit button referring to the unchecked box at the top of the form.

4.1.1.2 Auditory Attention

The auditory modality is often neglected when discussing attention. However, in contrast to visual information, humans can receive sound stimuli from any direction, almost at all times, and most auditory information is non-permanent, meaning that humans can take this input only for a (very) short moment [WHBP2016]. In order to manage that situation, humans divide their attention between and receive stimuli from different auditory streams by means of an unconscious fast switching between these streams.

Considerations for daily work

Humans can consciously focus their auditory attention on one source of auditory information or one specific auditory event [WHBP2016]. For example, given two healthy ears, humans can filter out other conversations and focus on one speaker; this is called the *cocktail party effect* [WHBP2016]. Although humans think that they are listening only to that speaker, their subconscious is still listening to the remaining auditory streams, for example, the *background noise in the restaurant kitchen stream* and the *conversation at the next table stream*. However, humans do not listen to the streams in parallel (i.e., at the same time). A very fast and repeated switching between the streams takes place and humans do not even notice this switching. If the brain *detects* something meaningful in one of these streams (e.g., a sudden loud noise in the kitchen, a person's own name being called at the next table), this auditory stream is focused regardless of whether the humans want this or not. However, if the brain labels an auditory stimulus in the auditory streams as *not meaningful*, humans will miss this stimulus: they will simply not *hear* it. This shows that before humans perceive auditory information, it has been subconsciously pre-processed and filtered by the brain.

4.1.1.3 Steering Attention

Users attention is controlled from the top down or from the bottom up [WHBP2016]. In the case of top-down control, the current active goals and tasks of the humans steer their attention (e.g., the aim to end the current running training), while bottom-up control refers to physical characteristics of a stimulus that steer the attention, such as the color and contrast of the *End training* button in the YPRC running app example.

Considerations for daily work

The DDP can try to shape the characteristics of the digital solution in a way that the digital solution might guide the attention of the users in the desired direction; or in other words, that the users will most likely detect the stimulus as intended by the DDP (bottom-up control of attention). As an example of drawing visual attention, the DDP could increase the salience of the stimulus to be detected (e.g., the color of a visual warning message) to make change blindness less likely. The salience can be calculated, for example, with a saliency mapping tool (e.g., http://www.saliencytoolbox.net/). As an example of drawing auditory attention, the DDP could choose a very loud stimulus or a stimulus that has a unique melody so that the stimulus to be detected, such as an auditory notification message, differs from the environment and *pops out*. [Wein2011] provides a small but nice overview of how to get attention with sounds.

However, even with wise design, there are still top-down attention control processes that are almost impossible for the DDP to influence. For example, the DDP can hardly steer where and when the users expect or anticipate a stimulus to occur.

4.1.2 Fundamentals of Human Performance

When humans have perceived information, they decide on an action and execute this action. In doing so, humans can be very fast and accurate, but they can also make errors. Errors are divided into mistakes, slips, and lapses [WHBP2016] (see Table 18). These three error types are described below with examples from the YPRC case study.

Mistakes

In mistakes, the intended action for dealing with a situation is wrong, caused by a wrong diagnosis of the situation (error of interpretation), and/or caused by a wrong selection of an action (error of planning). Thus, a mistake represents the commission of an incorrect action: the humans are simply doing the wrong thing.

YPRC example. The runner Maria might assume that the app automatically stops counting distance and duration after she has finished her predefined running route. She assumes that the "End training" button of the YPRC runner's app is used to stop the counting if the predefined running route is canceled by the runner (error of interpretation). Due to this misdiagnosis of the app functionality, as well as the misunderstanding of the meaning of the button, the runner decides to not take any action. After she has finished her regular run in the forest, she simply closes the app (error of action) and does not notice that the app is still counting distance and duration while she is driving home.

Slips

In the case of slips, the human's interpretation of the situation is correct and they decide on the correct action but ultimately, they perform an incorrect action. This error type therefore represents the commission of an incorrect action, an action that is different from the intended one (doing the wrong thing although the right thing was intended).

YPRC example. For example, the runner Maria knows that she needs to stop the tracking by pressing the "End training" button and intends to do so after finishing her run. However, instead of clicking "End training," she presses the home button of her smartphone: her "standard action" for ending an app prevails. Luckily, she recognizes immediately that she has executed the wrong action. She opens the app again and clicks the "End training" button, as intended a few seconds before.

Lapses

Lapses represent the omission of an action (*forgetfulness*), so no action is performed at all although something should have been done.

YPRC example. The runner Maria knows and aims to click "End training" when she finishes her run. However, she suddenly receives a funny message from her friend. After writing a short reply, she puts her smartphone back into her jacket and starts her stretching exercises. She simply forgets to click the "End training" button.

Lapses often occur in procedures with a series of steps and might be the final action in the sequence.

Table 18 – The three error types

Error type	Situation interpretation	Plan of action	Action execution
Mistake	Correct Incorrect	Incorrect	Correct
Slip	Correct	Correct	Incorrect
Lapse	Correct	Correct	Not (completely) executed

The digital solution should prevent the user from making errors. However, when designing the digital solution, the DDP is acting in an area of conflict: on the one hand, “many of the errors people commit in operating systems are the result of bad system design or bad organizational structure rather than irresponsible action by the person committing the error” [WHBP2016]. On the other hand, “it’s impossible to build a system that is impervious to human error” [Wein2011]. Moreover, users use different error strategies [Wein2011], such as systematically exploring procedures to correct the error or trying out different actions randomly.

However, humans can also detect and correct their errors with the help of the digital solution, such as by means of system feedback. Immediate and prominent system feedback from the digital solution, as well as easy-to-understand instructions and procedures for correcting the error are essential. Thus, the DDP should also design system feedback as well as error and notification messages thoughtfully. The DDP should consider the different error strategies as well as the different error types discussed above. For example, when writing instructions for error corrections, the DDP should follow some general guidelines (see [WHBP2016]), such as keeping the audience in mind, organizing the points mentioned in a logical way, and avoiding passive constructions. Moreover, the DDP should consider that system feedback, just like every output of the Digital Design solution, needs to be sensed and perceived by the users. [John2010] presents and discusses positive and negative examples of visual error messages and summarizes several aspects for ensuring that visual error messages are seen and understood.

There is no blueprint solution for handling human errors in performance: many factors influence the extent to which users make errors, detect errors, and can correct errors. The trickiness continues: the detection and correction of errors by users are activities that are also prone to error.

4.1.3 Emotions in the User-System Interaction

Humans do not just sense and perceive stimuli and then decide on (simple) actions, such as detecting and clicking buttons provided by the digital solution; the digital solution can unleash subjective reactions in the users, such as joy or anger. In turn, these reactions can affect whether and how the users will interact with the digital solution, now or in the future.

Experts and practitioners from different disciplines use different but also overlapping terms to describe this reaction, such as *perceptions and response* [ISO2018], *emotions, beliefs, preferences, perceptions, comfort, behaviors, and accomplishments* [ISO2018], *evaluative feeling* [Hass2008], *positive or negative emotions* [ScKr2010], *affect*, *experience*, *pleasure,* and many more [LRHV2009].

It is also debatable whether these reactions occur before, during, or after the use of the digital solution [ISO2018], during use only [Hass2008], or during and after use [ScKr2010]. However, there is agreement that the interaction with a digital solution—regardless of whether it is the

anticipated, actual, or past interaction—triggers behavioral and emotional reactions in humans and that a positive reaction is something desirable [LRHV2009] (see Chapter 6).

There are various models and frameworks that describe factors that cause and affect such reaction. One such model is the *Component model of User Experience (CUE)* [ThMa2007], see Figure 51. When a user interacts with a Digital Design solution, this interaction is affected by the attributes of the digital solution such as interface design and functionality (*system properties*), attributes of the user such as skills and knowledge (*user characteristics*), and attributes of the particular task and current context of the user (*task & context*). All these attributes shape the characteristics of the interaction, which are perceived by the user as qualities of the digital solution. These qualities are divided into two types of system characteristics: instrumental qualities such as effectiveness, and non-instrumental qualities such as aesthetics. The user's emotional response to a digital solution is shaped by the perception of these two types of system characteristics and emotions mediate between both types of perceptions. Finally, the user's emotional response and the perception of the two system characteristic types influence the consequences of use, such as overall judgment and intention to use.

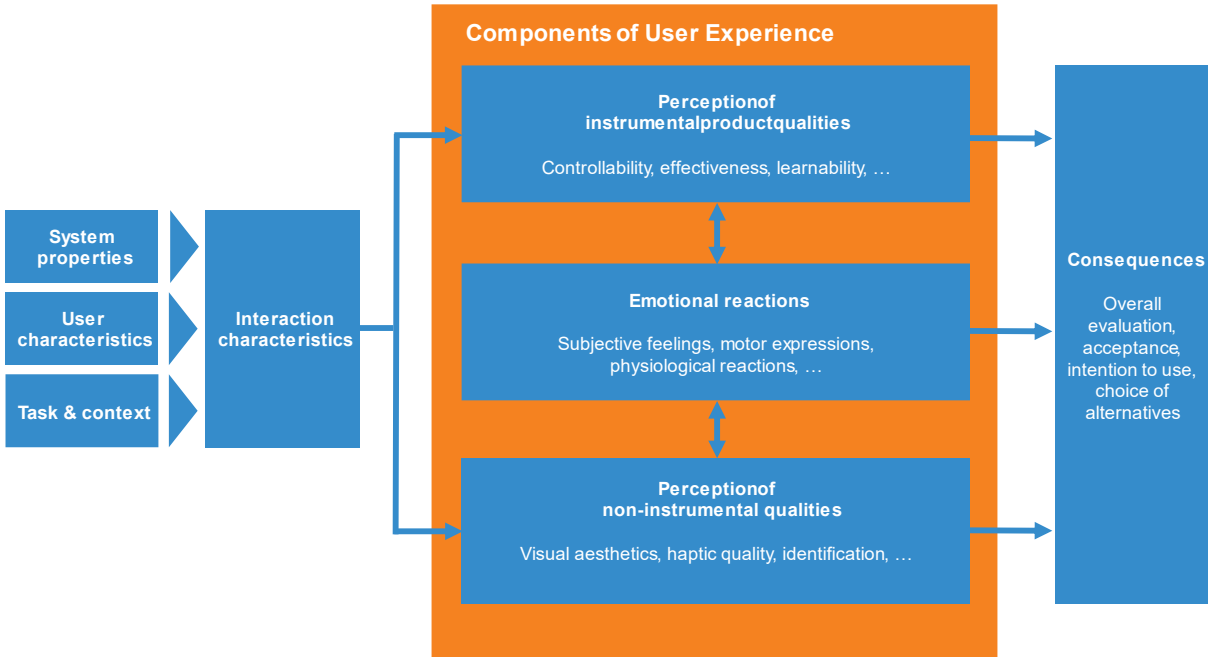


Figure 51 – The Component model of User Experience (CUE) (adapted from [ThMa2007][Ming2020])

The effect of the characteristics of a digital solution on the experience and behavior of its users can be measured. Based on their CUE Model, the research group has developed a questionnaire that allows the DDP to evaluate central aspects of the CUE Model regarding the digital solution under investigation. The modular questionnaire *meCUE* is available in German and English, and hands-on tips for the analysis and interpretation of the results are provided [Ming2020].

Note that other models and questionnaires exist that investigate how a digital solution shapes the experience and behavior of the digital solution user—for example, the model by Marc Hassenzahl and team and the related AttrakDiff questionnaire [HaTr2006] [User2020], or the typology of 20 mood states by Xue and colleagues [XDF2020].

4.1.4 The Role of Prototypes

To avoid a realized digital solution having flaws related to human factors or generating negative reactions, prototypes should be built at an early stage (see Section 2.3). They help the DDP to iteratively understand and influence the behavior (e.g., visual attention, performance, intention to use) and the experience (e.g., positive emotions) of the user in an intended way. See Section 1.3 for a detailed discussion on the evaluation concept.

An evaluation of the prototype can help to identify if and why issues related to human factors occur in the prototype. Evaluations are divided into expert-based approaches (also known as inspections) and user-based approaches (also known as empirical approaches). They can be conducted alternatively or in addition to each other. As an example of an expert-based approach, an expert on human factors works through the prototype looking for issues. In doing so, the expert can use guidelines such as the *interaction principles* [ISO2020], representative task scenarios, and heuristics; these all guide experts in their analysis of the prototype. User-based evaluations cover a broad range of methods that involve users. For example, in a user test with a first prototype, the DDP observes the user's interaction steps with the prototype, records their thinking aloud, and conducts a post-test interview.

Derived from the evaluation results of the first prototype, the DDP might conclude that it is crucial to get the user's attention at certain steps in the future digital solution. Considering this, the DDP creates variants of visual, auditive, and combined alarms. Then, the DDP examines and continuously evaluates which alarm variant attracts the most attention, and then iteratively improves this variant. Before the digital solution is realized, a final evaluation verifies that the alarm designed gets the user's attention at the right time.

Any prototype category can be tested but the prototype category (cf. Section 2.3)—among many other aspects—might affect the extent to which issues in the prototype that are related to human factors can be identified at all. For example, a mobile app prototype with medium visual refinement would be less appropriate for testing whether visual attention failures might occur or not.

Many evaluation strategies and methods exist and most of them can be adapted for any prototype and testing aim. Several books provide good overviews on methods and decision criteria, such as [Barn2011] [SSRW2013] [TuAl2013]. Roughly said, there is a method for every purpose and budget. However, deciding on the right testing methodology as well as planning, conducting, and analyzing a test is challenging. Depending on the DDP's background and expertise level, it is highly recommended that the DDP consults and involves experts. Nevertheless, it is the responsibility of design to foster and plan prototyping as well as iterative prototype testing. In doing so, the DDP should decide wisely when to conduct the testing activities—for example, as far as a sufficient maturity of the solution is achieved or before moving from one stage to the next stage in the building process (see Section 1.3).

4.2 Business Models for Digital Solutions

This section gives the DDP a brief insight into the world of business models. It describes traditional and new models and their dynamics, innovation techniques, and how to position the model within a highly competitive environment.

The bandwidth and speed at which innovative business models are changing the business landscape today is unprecedented. In their leading role, it is very important for the DDP to understand the impacts of this extraordinary evolution and to systematically meet the challenges

associated with it. A quote from Peter Gorb summarizes this challenge (quote taken from [StSc2011]):

"And what designers need to learn, and this is the most important thing, is the language of the business world. Only by learning that language can you effectively voice the arguments for design."

A business model describes how an organization plans to create value [OPBS2014]:

Business model: The rationale of how an organization creates, delivers, and captures value in economic, social, cultural, or other contexts.

The process of business model construction and modification is also called business model innovation and forms part of the business strategy [GeSE2017].

In both theory and practice, the term business model is used for a broad range of informal and formal descriptions to represent core aspects of a business, including purpose, business capabilities and processes, target customers, offerings, strategies, infrastructure, organizational structures, sourcing, trading practices, and operational processes and policies including culture.

Whenever a digital solution plays a major role in the value chain of an organization, there are important relationships to consider between Digital Design and an organization's business model, including its value proposition to internal or external customers:

- The value proposition of the business model (see Section 2.2) must be realized by the digital system as part of a digital solution.
- The capabilities and limits of digital material define the capabilities and limits for delivering the value.
- The customers of the business can be users or stakeholders of the digital solution.
- The users of the digital system can also be part of the value proposition of the business.
- Costs for building and operating a digital system inside a digital solution are often a significant part of the cost structure of a business.
- Creating the revenue stream is often part of the digital system (e.g., collecting data on payment, interaction with a payment provider).

Building a digital solution is therefore not only about the form, function, and quality of the digital system that realizes the digital solution (see Chapter 1), it is also about defining a business model together with the digital system (see Section 2.1) and the related value creation for companies, customers, and society.

Digital transformation (see Chapter 1) is even about replacing existing business models. With the iPod and the iTunes Store, Apple has created an innovative new business model that has made the company the dominant force in music downloads. Skype has given us cheap global phone rates and free calls between Skype users with an innovative business model based on peer-to-peer technology. Today, Skype is the largest international telephone company in the world.

Considerations for daily work

Designing a digital solution means designing a value proposition that is directly connected to an effective business model. A DDP's benefit is that using the methods and techniques from this handbook extends the boundaries of thought to generate new options and, ultimately, to create value for users and business by means of digital material.

Whenever possible, the DDP should integrate the perspective of Digital Design into management models and management thinking and create cases for integrative design of digital solutions and systems. By designing a business model early in the building process, the DDP has the opportunity to systematically invent, design, and implement value propositions to question old, outdated propositions and to test and convert them, together with the stakeholders, into new and innovative business models.

4.2.1 Business Model Pattern

A good introduction to the world of business models is the categorization by Choudary. He distinguishes between two general patterns of business models [Chou2013]:

- *Pipes (linear business models)*: Companies create goods and services, push them out, and sell them to customers. Value is produced upstream and consumed downstream. There is a linear flow, much like water flowing through a pipe.
- *Platforms (networked business models)*: Platforms do not just create and push stuff out; they allow customers to create and consume value. Moazed defines a platform as a business model that creates value by facilitating exchanges between two or more interdependent groups, usually consumers and producers of a given value [MoJo2016]. It is the predominant business model that drives the digital transformation (see Chapter 1).

Besides the two general models, the DDP at foundation level should be aware of further detailed types of business model patterns [OPBS2014]:

- *Unbundling business model*: Business models that combine the three areas of customer relations, product innovation, and the provision and maintenance of infrastructures to different extents (examples are telecommunication providers such as Deutsche Telekom, Swisscom, and AT&T).
- *Long-tail business model*: Instead of offering a limited number of products, the long-tail business model means offering a wide range of different products by making use of superior logistics. This enables a company to make profits from otherwise unprofitable niche products (example: Apple iTunes).
- *Multi-sided platform business model*: A platform enables the interaction of two or more independent groups. The value for an individual group comes from the presence of another group (example: Google.com, the groups are advertisers and search engine users). The more users that use Google's search engine, the more data Google has to improve search results. The greater the market share of the Google search engine, the more advertisers place their ads through Google, which in turn strengthens Google's negotiating position on pricing, and there are several powerful, high-performance control circuits in this business model.
- *Freemium business model*: A standard service is offered free of charge; extended functionality requires a paid subscription (example: linked-in online community).
- *Tied products business model*: A cost-effective or free first-party product or service motivates the use of future paid replacement products or services (example: Gillette razor & blades, HP color inkjet printer). Also known as bait-and-hook or razorblade business models.
- *Open business model*: A collaborative business model that uses external experts to add and secure value (example: a collaborative business model that uses external experts to add and secure value, such as GlaxoSmithKline).

For a foundation level perspective on business models, the list presented is more than sufficient. Readers with more interest in business models can find further business model patterns in [Gass2013].

4.2.2 Digital Business Models

A digital business aims to harness digital material for digital solutions to enable new business models that give an organization a competitive advantage. In contrast, e-business primarily aims to digitize (see Section 1.1) an existing business model, typically with the goals of saving costs or attracting more customers, but without fundamentally changing the business model itself.

In turn, digital business is an enabler for digital transformation, i.e., when digital solutions change the behavior and lives of people and impact society. Social media such as Instagram or entertainment streaming services such as Netflix are examples of where digital solutions drive the digital transformation.

Digital business models are therefore a special case of the business model patterns introduced above (see Section 4.2). In the following, we discuss three examples of digital business models to show the wide range of possibilities:

- Uber relies on the widespread use of smartphones and uses a business model that requires relatively little capital. A traditional taxi company needs vehicles and has the expense of recruiting and managing their employees. Although it would be conceivable to have a business model where customers can order their taxi online and possibly also view trips already completed, the experience of the taxi ride itself would be the same. Uber, on the other hand, asserts itself as a digital company because it is essentially a platform that connects passengers and drivers via the internet while creating a better customer experience.
- Netflix is a great example of how a business can transform its original e-business into a digital business. Originally, Netflix used technology to manage an inventory system and send DVDs for rental to customers. Although this was more convenient for customers, it ultimately resembled a digital version of video rental. By offering video-on-demand streaming services, Netflix has broken their own business model and changed the way people use movie and television media. Consuming films and TV shows anytime and anywhere would not be possible today without access to the internet, which is almost universal for consumers.
- Disney is a traditional company that uses digital technologies to enhance theme park experiences. Visitors to the park can now enjoy an extraordinary park visit with the MagicBand, a wristband that uses RFID and radio with sensors in the park. Five basic things have been improved: visiting an attraction, hotel accommodation, dining out, taking pictures and sending them to friends or family, and buying souvenirs. Guests receive the MagicBand a few weeks before the visit, allowing them to enter the park, buy food or groceries, book attractions, and see waiting times in real time. At the end of the day, guests feel they have moved fluidly between the digital and physical worlds.

4.2.3 Thinking about Future Possibilities for Digital Businesses

The value proposition and the business model of a digital solution depend on the quality of the services provided to the users of the solution. The three horizons model [BaCoWh1999] can be used as a tool for thinking systematically about the scope of the digital business, about the insight

maturity of a digital solution, and for defining the level of business transformation. It structures potential ideas for digital businesses into three horizons (Figure 52).

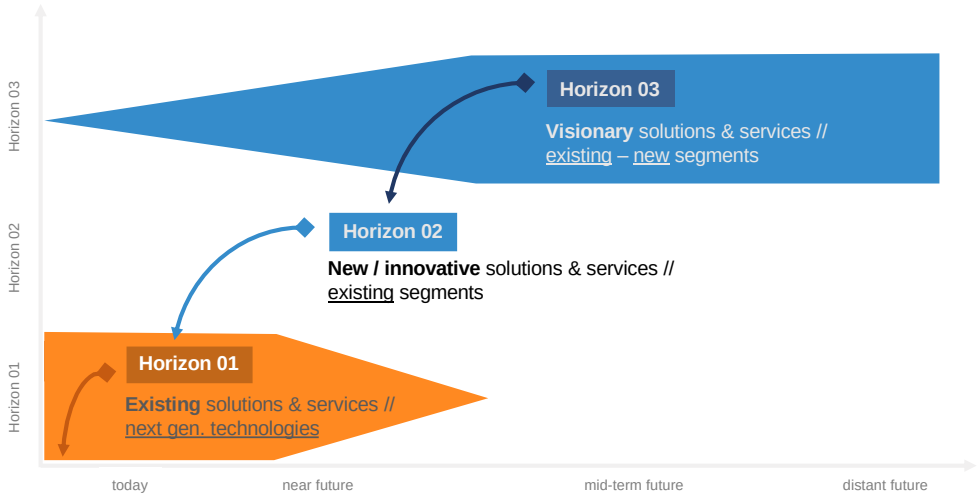


Figure 52 – The three horizons of business

Horizon 1 is about existing businesses of an organization. The existing business is driven by existing solutions that provide important revenue as part of the business model of an organization. In terms of Digital Design, these existing solutions can be improved by means of innovative digital technology. Such improvements are typically a short-term project that can be realized to the market rather fast and requires only short-term building processes. These improvements can be planned forward directly and are normally considered a tame problem (see Section 2.1). The timescale from today to the distant future depends on the domain and many other fields. For example, the smartphone industry is a very fast domain.

YPRC example. Assume that the runner's watch with the pulse sensor is a successful product on the market and that the remote coaching service is also successful. Further assume that technical advances in the area of smartwatches allow cheap smartwatches with built-in mobile internet connection to be built. With such a technology, it would be possible to provide a smartwatch-only solution that no longer needs the smartphone as a connection to the runner's portal.

Horizon 2 is about innovative solutions/services in the domain of the organization. Such solutions can extend the business model of an organization in already known domains. Creating such solutions can be a tame or even a wicked problem. It is a mid-term building process and requires a certain investment in terms of conceptual work to minimize the risk involved in building a successful solution.

YPRC example. In addition to the remote running coach service, a remote coaching service for athletic sports (e.g., weightlifting) or cycling could be created and offered. With such an extension, the YPRC company could extend its customer base to other segments of athletes.

Horizon 3 is about future opportunities that aim at visionary ideas in terms of solutions and/or technologies. Such ideas can significantly extend (and challenge) the business model of an organization. An example of such a horizon 3 is the Netflix case (see Section 4.2.2). Approaching such ideas is a wicked problem and a rather long-term project that typically requires significant effort in terms of scoping and conceptual work. If the idea also involves emerging technology,

there is a further risk that the emerging technology will not reach a mature enough state to allow productive application.

> **YPRC example.** Currently, the remote coaching service is provided by a human being. Assuming that artificial intelligence technology is developing to a stage that a real-time analysis of the runner's health is possible, the human coach could be replaced by a virtual AI-based coach that uses voice technology to guide the runner.

If you choose to create a digital solution that can be categorized as horizon 2 or even horizon 3, a completely different working mode is required. The planning and thinking direction of horizon 3 or 2 is reversed from the future to the present.

As highlighted in Section 4.3, a broad, abstract thinking mindset of the building team is necessary in this situation. With such a mindset, it is possible to envision an inspiring future by means of future scenarios. Such scenarios are necessary to shape the future solution by envisioning a future context and future customer needs in the far future. Such future scenarios can then be nailed down to more and more concrete stories closer to the present. If a common understanding of the scope, its risks, and the solution idea is stable, the planning direction follows the planning type of a building process: the scoping, conceptual, and development and operations steps.

4.3 People Management

So far, we have been looking at the building process from a methodological (Chapter 2 and Section 4.1) and technical (Chapter 3) perspective, as well as a business perspective (Section 4.2). This section introduces a further important perspective: the people perspective of the building process for digital solutions. This perspective is important because people play different roles in building digital solutions:

- People are the future users/customers of the digital solution
- People as clients that place additional requirements on the digital solution
- People who execute the building process

People, as clients, users, customers, building team members, and stakeholders, are at the core of Digital Design and have been addressed explicitly in this handbook. The techniques presented in this handbook are intended to support people (e.g., team members, clients, and sponsors) who execute the building process, especially the Digital Design part.

However, there is an additional dimension to people in the building process. Good Digital Design is achieved by people working together in interdisciplinary situations (see also Chapter 6). At foundation level, the DDP needs to name key indicators for understanding people and team dynamics. This understanding is necessary to characterize interpersonal dynamics in interdisciplinary teams and to explain the necessity for different leadership types in the building process for a digital solution. Like Section 4.1 on human factors, which aims to create awareness of the human dimension in the use of digital solutions, this section aims to create awareness of the human dimension in the building process.

A common misunderstanding sees the building process as a mechanical process; instead, it should be considered as a social process that takes place between the people who participate in the building process (cf., e.g., [VPGV2008]). This includes the people who actually build the digital solution and the people who provide input to the building process (i.e., future users and other stakeholders). The social process perspective considers the individual people, their relationships,

their understanding of the digital solution, and the communication and collaboration between the people involved. Furthermore, the social perspective means understanding how the personal background and profession of an individual can influence communication and team dynamics. The broad scope of these perspectives underlines the importance of understanding the building process as a social process. Dealing with all these aspects goes far beyond the foundation level. Nevertheless, a DDP at foundation level should have an awareness of all these perspectives.

4.3.1 Understanding the Building Process as a Social Process

The following aspects are useful for understanding the building process as a social process:

- People and organizations prefer different means of communication.
- People and organizations have different prerequisites (e.g., education, personal experience, origin, and learning style) for perceiving and understanding a subject matter.
- People and organizations prefer different ways of working to accomplish a task depending on their personality.
- The different stages of the building process provide different challenges for the people and the organization in terms of team dynamics and leadership.
- The working environment (e.g., time or expectations) has a decisive influence on people's actions and behavior.

Digital solutions are always built in a cultural environment

The aspects above can be summarized as the cultural environment of the group of people involved in the building process. Culture, like the culture of a building process, refers to the beliefs and behaviors that determine how team members and management interact. Often, culture is implied, not expressly defined, and develops organically over time from the cumulative traits of the people involved.

For a building team, culture can be influenced by national cultures and traditions, economic trends, company size, and products. Cultures, whether shaped intentionally or grown organically, reach to the core of a company’s ideology and practice and affect every aspect of a business [TARV2019]. Therefore, the personality of the people involved in the building process and the corporate culture of the building organization have a major impact on the efficiency and effectiveness of the building process.

The building process for a digital solution provides orientation and an understanding of the desired future (see Section 2.1). This understanding should not be considered a by-product of the process; it is often a key value for the people involved. Understanding the desired future is a key driver in engaging in change because people can clearly communicate the desired future and can thereby act as ambassadors for the idea. Furthermore, clear communication of the desired future ensures that people share the same goal. This increases people’s willingness to cooperate in teams in a spirit of trust. In Chapter 5, we present tools and methods for increasing content understanding as part of the building process.

Challenges of the building process from a social perspective

People involved in the building process (the building team) have to deal with the following challenges:

- *Building up the team:* People from different disciplines or business partners come together in the building process. Sometimes they already know each other, sometimes they are working together for the first time. A new relationship leads to challenges of understanding with regard to their different terminologies and different backgrounds and experience.
- *Mutual team understanding:* Translation gaps and misunderstandings not only arise because people do not sufficiently understand their client's business and project backgrounds, but also because people are different! In fact, a project or a company is a living ecosystem; it consists of the interaction of different people who react individually within their comfort zone. Under stress, people may act completely irrationally even though they are aware of a clearly defined building process.
- *Fitness for the building process step:* Teams have a different aptitude and mental fitness for each step of the building process. This aptitude and fitness depend on a clear and agreed role definition, responsibilities, and an intelligent mixture of team members with respect to cognitive style, mental health, and leadership potential.
- *Expectation management:* Time and expectation pressure in a building process with simultaneously decreasing transparency of content, a different mutual understanding of the persons involved, and an increasing disorientation about the relevance of digital trends by the customer leads to divergent expectations.
- *Expectation evolution:* Users' experience of new digital solutions leads to new user demands in the future. This change in demand takes place at increasingly shorter intervals compared to the non-digital age. The limited understanding of attractive future scenarios requires permanent interaction with the customer within the design process to figure out which solution will work.
- *Creative Tension Engine* [NIJS2019]: At a certain point, attention fades time and again in teamwork, things do not work out in the way expected, projects fail, or conflicts arise. This can be prevented by creating an attractive mission statement (e.g., as part of the future press release during the scoping step) to transport the sense of the project to external stakeholders as well as to keep the team's positive energy level high to avoid stress and to allow team members to become re-engaged in the project.

People management as a systematic approach

The conclusion from these challenges is that customer/user acceptance and acceptance of people's own organization do not just happen but must be systematically created and managed.

People management means managing three aspects in parallel:

- Managing process structures for communication through rules and procedures
- Managing the creation of the digital solution within the process structures
- Managing the experience of the people involved in terms of expectations and emotions within the process structures

Process structures (e.g., definition of process models, roles, and templates) are necessary to provide an explicit way of working during the building process. However, they are not sufficient for dealing with the afore-mentioned challenges adequately.

People management must integrate the understanding of goals and solutions, the perception, and the emotions of those involved. The explicit management of expectations allows threats to customer/user acceptance to be identified. Clients can thereby be protected from unpleasant surprises at an early stage.

Understanding the expectations of people is a question of communication. First of all, a common language is needed that conveys a clear and pictorial understanding of the content in such a way that translation gaps between different people are closed, reservations with regard to certain solution ideas are defused, and individual motives with respect to the new solution can be addressed.

Finding this common language is not only a matter of syntax (words, concepts, models, prototypes) and semantics (meaning), it is in particular a matter of the people involved (pragmatic). In Section 4.3.2, we see that people are different in various dimensions. These different dimensions have a significant impact on how people communicate and work with each other. Understanding these different dimensions of the people involved in the building process is a great benefit for people management in terms of finding a proper common language.

The basis for this is the understanding of the whole person with his or her attitudes and behavior in the comfort zone or under pressure. The DDP can handle the complexity and instability of interpersonal relationships in the building process if they have knowledge of the effectiveness of personality models and personal indicators:

- Clarifying people assignments
- Understanding future value creation
- Problem solving by managing stakeholders and designing communication measures

In the following Section 4.3.2, we introduce a model for understanding people that explains the three points. In Section 4.3.3, we use this model to discuss the team dimension of the building process.

4.3.2 Understanding People through Personality Models

People are different in their thoughts, values, languages or terminologies used, behaviors, or how they deal with stress. Each person thinks and communicates differently depending on their activity context, personality, individual life track, or cultural background (for example, nationality, relatives, education, brands). Even though a person can often acquire any expertise, individuals find it easy or difficult in different ways to learn and apply things, which has a significant impact on their stress levels and behavior.

An important example for the building process is the following: people see and understand the world from different perspectives. Some prefer to look more at details; others have a better chance of seeing the big picture; some rely on knowledge from the past by using an incredible memory but have real difficulty in developing their own imagination; others find it easier to look into the future [Keir1998].

Personality indicators are a good tool for understanding people’s behaviors and do identify preferences that cause little energy expenditure and thus little stress. The knowledge of these indicators is used in the following sections to lead communication and to understand team dynamics and leadership within the building process. Personality indicators are used in different personality models.

Personality models—like every other model—are useful but limited

However, before we go into the personality models in more detail, we must discuss the use of such models in the building process: despite our need for predictability, human personalities and the complex reality of interpersonal relationships cannot be captured even with the best

instrument in the world. All models carry the danger of putting people in drawers. Whether the model uses a color, a combination of letters, or something similar does not make any difference. We see the benefit of those models in gaining initial access to different personality patterns in order to make differences between people clear. Really good teams, departments, or companies have recognized the value of human individuality. They use it for more innovation and performance.

However, this can only succeed if we become aware of the differences between people. And in this context, becoming aware does not mean that we can classify team members into categories based on different numbers/letters/colors, but rather that we get an initial idea of the respective personality pattern, a first orientation to understand ourselves, to understand others, and our own interaction with others which the DDP can examine and deepen in discussions and joint actions.

The Keirsey Temperament Sorter is a good starting point

The Keirsey Temperament Sorter (KTS) is a concrete personality model that is closely associated with the Myers–Briggs Type Indicator (MBTI). The MBTI is an introspective self-report questionnaire that indicates differing psychological preferences in how people perceive the world and make decisions. The four pairs of preferences or "dichotomies" are: Extraversion/Introversion, Sensing/Intuition, Thinking/Feeling, Judging/Perception. The MBTI is based on the conceptual theory proposed by Swiss psychiatrist Carl Gustav Jung, who had speculated that people experience the world using four principal psychological functions—sensation, intuition, feeling, and thinking—and that one of these four functions is dominant for a person most of the time.

Keirsey has defined the KTS as a model that consists of four basic temperaments: the Artisans, Guardians, Idealists, and Rationals [Keir1998]. Gunter Dueck draws on Keirsey's books in some of his works and defines in his character model based on the KTS more descriptive temperament names: Go West, Citizen, Blue Helmet, Star Trek [Duck2013]. These names represent archetypes that allow an easy understanding of each temperament. Table 19 summarizes the four temperaments and gives advice from a people management perspective.

The communication styles of the four temperaments can be understood by comparing them to the following rings of a tree:

- *Inner ring*: abstract versus concrete (understanding, perceiving, and learning)
- *The second ring*: cooperative versus pragmatic (action focus)
- *The third ring*: directive versus informative (communication style)
- *The fourth ring*: expressive versus attentive (energy, impulsivity)

4.3.2.1 The Inner Ring: Abstract versus Concrete (Understanding, Perceiving, and Learning)

People naturally think and talk about what they are interested in. By listening carefully to people's conversations, we can understand their way of understanding and perceiving information. The different learning styles make it easy in different ways and at different levels for them to understand things [BeBa2007]. Two broad but distinct areas of subject matter are found for communication [Wiki2020c]:

- *Abstract*: Some people take in information through symbolic representation or abstract conceptualization and are introspective (they perceive abstractly or intuitively). People who are generally introspective are more *head in the clouds*. They are more abstract in

their world view and tend to focus on global or theoretical issues—all the why, if, and what-might-be aspects of life.
Important for the building process: they have a better chance of seeing the big picture and find it easier to look into the future.
Temperaments that perceive abstractly are Idealists (Blue Helmet) or Rationals (Star Trek).

- *Concrete*: Other people take information in through direct sensation; they are observant (perceive concretely or sensitively). People who are generally observant are more *down to earth*. They talk primarily about the external, concrete world of everyday reality: facts and figures, work and play, home and family—all the who-what-when-where-and how much aspects of life.
  Important for the building process: they look more at details and rely on knowledge from the past.
  Temperaments that perceive concretely are Artisans (Go West) or Guardians (Citizen).

At times, of course, everyone addresses both sorts of topics, but in their daily lives, and for the most part, concrete people talk about reality, while abstract people talk about ideas and the future. Each of these diametrically opposed sets of approaches presents a choice that an individual must make, and over time, individuals gravitate to a preferred style. The extent to which people are more observant (concrete) or introspective (abstract, intuitive) directly affects their behavior.

Table 19 – Characterization of the four basis temperaments

Temperament	Motivation	Time-frame	Syntax	Hints for people management	Archetype
Artisans are **concrete** and adaptable. Seeking stimulation and virtuosity, they are concerned with making an impact. Their greatest strength is tactics. They excel at troubleshooting, agility, and manipulating tools, instruments, and equipment. They behave in a process-orientated way. As Concrete Utilitarians, Artisans speak mostly about what they see right in front of them, what they can get their hands on, and they will do whatever works, whatever gives them a quick, effective payoff, even if they have to bend the rules.	Freedom of action, stimulation	Present	Descriptive	• Highlight personal gain, facts and figures • Interact generously • Offer tolerance & freedom • Praise actions, not results • Avoid rules & routines Typical roles: Developer, team lead implementation, communicator	**S**ensitive perceiving, **P**erceiving The **Go West** pioneers in America are an architype for this temperament.
Guardians are **concrete** and organized (scheduled). Seeking security and belonging, they are concerned with responsibility and duty. Their greatest strength is logistics. They excel at organizing, facilitating, checking, and supporting. They behave in a result-orientated way. As Concrete Cooperatives, Guardians speak mostly of their duties and responsibilities, of what they can keep an eye on and take good care of, and they are careful to obey the laws, follow the rules, and respect the rights of others.	Sense of duty, social position, control	Past	Com-parative	• Highlight facts and figures, personal gain • Respect responsibility, function & experience • Show gratitude • Confirm: “It works” • Certify ordinary work • Avoid superficial implementation Typical roles: System admin, support services, CFO	**S**ensitive perceiving, **J**udging The **Citizen** of emerging cities is an archetype for this temperament.
Idealists are **abstract** and compassionate. Seeking meaning and significance, they are concerned with personal growth and finding their own unique identity. Their greatest strength is diplomacy. They excel at clarifying, individualizing, unifying, and inspiring. They behave in a person-orientated manner. Idealists generally like to deal with people and emotions instead of logic and detail. They may be overly emotional (positive or negative) in their responses to information delivered and get highly frustrated when they do not get to express that emotion. When it is necessary to deliver non-positive news or criticism to an Idealist, it is important to frame such things positively.	Self-realization, identity, meaning, authenticity	Future	Meta-phorical	• Show enthusiasm • Highlight special characteristics • Address in person & bring up the relationship level • Demonstrate understanding • Praise, personal confirmation • Avoid disagreements • Avoid conformity & ignoring values Typical roles: Scrum master, product owner, coach, CMO	i**N**tuitive perceiving, **F**eeling The **Blue Helmets** of the United Nations are an archetype of this temperament.
Rationals are **abstract** and objective. Seeking mastery and self-control, they are concerned with their own knowledge and competence. Their greatest strength is strategy. They excel in any kind of logical investigation such as engineering, conceptualizing, theorizing, and coordinating. Rationals like to have lively discussions, argue their positions, and use logic to communicate points. They communicate relevant factors in a technical manner to get their points across. They value concise, to the point criticisms and get irritated with grandiose explanations for information that is being presented.	Power over nature, competence, knowledge	Interval: past, present, future	Con-junctive, categorical	• Highlight knowledge & alternative perspectives • Allow freedom of conception • Offer professional dialog • Discuss instead of praise • Avoid faulty principles & incompetence Typical roles: Designer, architect, data scientist, CIO, CEO	i**N**tuitive perceiving, **T**hinking The **Star Trek** characters are an archetype for this temperament.

Considerations for daily work

The building process and especially the scoping step needs above all a selection of people who can easily think abstractly, who are able to imagine possible futures in an abstract way. However, this step also needs people who can turn an abstract vision into a concrete future scenario (see Section 4.3.3.1).

4.3.2.2 The Second Ring: Cooperative versus Pragmatic (Action Focus)

There are two fundamentally opposite types of action to accomplish a goal:

- *Pragmatic*: Some people act primarily in a utilitarian or pragmatic manner; that is, they do what gets results, what achieves their objectives as effectively and efficiently as possible, and only afterwards do they check to see if they are observing the rules or going through proper channels. Pragmatic temperaments are Artisans (Go West) or Rationals (Star Trek).
- *Cooperative*: Other people act primarily in a cooperative or socially acceptable manner; that is, they try to do the right thing, in keeping with agreed social rules, conventions, and codes of conduct, and only later do they concern themselves with the effectiveness of their actions. Cooperative temperaments are Idealists (Blue Helmet) or Guardians (Citizen).

These two ways of acting can overlap, certainly, but as they lead their lives, utilitarian people instinctively, and for the most part, do what works, while cooperative people do what is right.

Considerations for daily work

Check whether the situation requires rationality, such as solution development, or a focus on personal interests, such as joint cooperation, acceptance of the solution, or enforcement in terms of the client's interests. Then assign people in the lead with sufficient preferences.

4.3.2.3 The Third Ring: Directive versus Informative (Communication Style)

The third ring distinguishes between people who generally communicate by informing others versus people who generally communicate by directing others. This is also known as proactive versus reactive, and describes how people communicate, either by informing others or directing others to action.

Considerations for daily work

Each situation requires a different behavior. Pay attention to those who pursue a natural claim to leadership regardless of their position and role—either to be perceived more intensively in their role or to consciously withdraw themselves in their personality.

4.3.2.4 The Fourth Ring: Expressive versus Attentive (Energy, Impulsivity)

The fourth ring describes how people interact with their environment and how they get energy.

- *Expressive*: Individuals who prefer more overt action (saying and doing, extraversion) during covert acting (conception and perception), (observing or introspecting) are described as expressive. Some associative words for *expressive* are: impulsive, active, chatty, conversant, effusive, fluent, profuse, verbose. Extraverts give width to life.

- *Attentive*: People who prefer more covert acting during overt (or inactive, introversion) action are described as attentive. Some associative words for *attentive* are: alert, all eyes, all ears, aware, chary, circumspect, heedful, reflective, wary, watchful. Introverts give depth to life.

Extraverts draw energy from being together with other people; introverts draw energy from being alone. The expressive versus attentive dichotomy is the most contextual.

Considerations for daily work

Try to stimulate people with impulsiveness to generate spontaneous ideas or to activate the team, for example. Try to stimulate people with reflection when concentrating on details or gaining deeper insights on a topic.

4.3.2.5 Conclusion on Working with Personality Models

If we combine the four rings, we have 16 archetypes that can be used to categorize people. The assumption of these archetypes is that each person has one preferred quality from each category, producing 16 unique types. The four rings relate to one another and to the various temperaments included in the 16 associated MBTI types.

The main lesson for people management for a DDP at foundation level is to select appropriate communication styles/forms, to:

- Understand people's characteristics and individual driver with their congenital and learned behavioral patterns in different situations (e.g., in comfort zones or under stress).
- Step into the shoes of the other person and look at the world from their eyes. For example, create a persona description or run field analysis in a customer role. It is not necessary to approve of what the other person thinks or feels, but rather to take a neutral interest in it.
- Listen to the other person without distraction and give them your full attention. Ask questions to further explore what is going on in the other person without becoming judgmental; ask about motives and motivations. It can be helpful to rephrase the understanding to get confirmation.

Considerations for daily work

Applying this tool in daily work, however, requires considerable practice and further training. As further reading, in addition to personal indicators, psychometrics indicators can support the identification of individual and team stability and provide a deep understanding of your colleagues' conditioning and characteristics. This understanding helps you to avoid risk and personnel health issues.

4.3.3 The Building Process from a Group Dynamic Perspective

In the following, we use the model introduced to view the building process from another perspective. We look at the group dynamics that shape the behavior and psychological processes of the people involved in the building process.

In order to build a great digital solution, a building team has to consider the area of conflict between the dimensions customer/user needs, economic feasibility, as well as the technical

possibilities during the building process (see Chapter 1). The core characteristics that are necessary to deal with this area of conflict are (cf. [BeBa2007]):

- *Tolerance for ambiguity:* Team members must tolerate phases during the building process where ideas and directions are not 100% certain. The chaotic process described by the design squiggle (see Section 2.1) is a good analogy for this ambiguity.
- *Abstract conceptualization:* Team members must develop abstract concepts of their ideas as part of the conceptual work (see Section 2.2) in order to bring them to a level of detail that can be discussed in terms of the three dimensions customer/user needs, economic feasibility, and technical possibilities.
- *Need for closure:* Discussion about ideas cannot go on forever in a building process: at some point in time, the team must decide in favor of or against an idea in order to approach the next step of the building process and finally the actual realization of the digital solution.

In order to unleash the full collective intelligence of the team during the building process, the organization must perceive people management in the building process as a success factor, especially in self-organizing building teams. It must actively authorize a responsibility to keep an eye on team fitness and to pay attention to which persons in the corresponding situation can unfold the team's potential particularly well with their personality.

Incorrect personnel dispositions in the team lead to an inadequate vision of the future, premature selection of ineffective implementation variants, a frittering away of time in finding solutions, and individual stress. Personality patterns support the people manager in setting up the building team and identifying risks of passive, hidden resistance or in discovering hidden economic potential at an early stage.

We distinguish between and elaborate these aspects in the following sections in terms of people management in the building process:

- Managing perception and learning potential
- Thinking into the future
- Managing working style and role assignment
- Managing leadership during the building process
- Considering the working environment and subject matter understanding

4.3.3.1 Managing Perception and Learning Potential during the Building Process

People management can make use of the dimension abstract (perceiving and learning/open minded) and concrete (perceiving and learning/need for closure) to manage the working mode and interpersonal relationships during the building process (see Table 20 and Section 4.3.3.3).

Figure 53 shows the idealized building process introduced in Section 2.1.5 and illustrates the distribution of temperaments along the different steps of the building process. In the scoping step, an abstract and open thinking mindset is needed to explore different ideas and to design powerful solution ideas. In the conceptual step, the need for concrete decisions and closure increases to identify a good solution candidate. The need for concrete decisions and closure further increases in the development/operation of the solution. Nevertheless, during operation, the need for abstract and open thinking rises again, when the existing solution has to be reconsidered or adapted.

Research on temperaments and cognitive style identifies similar characteristics (tolerance for ambiguity and need for closure) as being important to design. The generic building process is grounded in models of how people learn and make decisions. An individual with low tolerance for ambiguity, for example, sees ambiguous situations as threatening and tends to seek certainty, sometimes, for example, clinging to old information in the face of new information, as it is more certain. This is similarly true for the need for closure, which can be a personality trait as well as situation induced. Need for closure means that an individual seeks certainty, often grasping the first available information and locking onto it rather than remaining open to new information that might become available [BeBa2007].

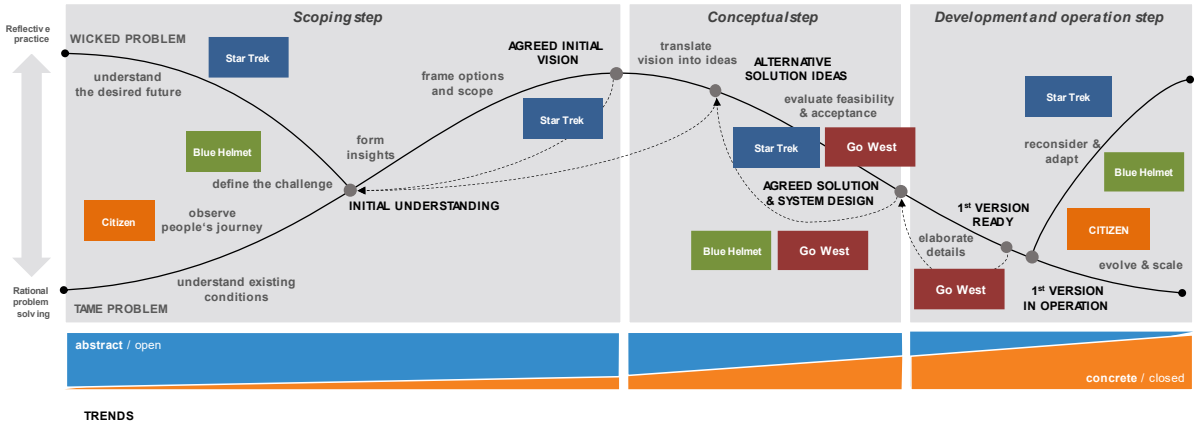


Figure 53 – Effective distribution of temperaments for the interdisciplinary collaboration involved in building a digital solution [KEMP2017]

Considerations for daily work

How to perceive information (concrete or abstract) depends on the working style due to specific disciplines. Strategic or future thinking require system and abstract context understanding (Star Trek). Implementing working styles often need a detail-focusing characteristic (Go West). People with high empathy scores (Blue Helmet or supporting citizen) are able to guide or support interdisciplinary teams. The selection of the right people is crucial, particularly in the scoping step. The participants with which the future can be designed or validated are decisive. Only a few are able to abstract from current attractors (current demand) to future attractors (attractive future and future demand) (Star Trek). At this point, it is also important to have the permanent feedback available in the customer system because the future demand is unknown. This requires cooperative people with a future mindset (Blue Helmet) to find out, together with the customer or customer representatives, what can be attractive in the future. We detail the ability to think into the future in the following section.

4.3.3.2 Managing the Thinking into the Future

In essence, every building process works on creating a desired future (see Chapter 1). In Section 4.2, we introduced the three horizons model to describe potential time horizons for thinking into the future. With the understanding of temperaments, we can identify temperaments that are especially suited to working within a certain time horizon. Abstract thinking characteristics and skills are needed if the digital solution envisions a new future that is far away, or its understanding is still unclear (horizon 2 & 3).

Figure 54 shows the three horizons and the corresponding temperaments:

- *Horizon 1 (improving existing business)* requires temperaments that are oriented towards concrete (Go West, Citizen) to focus on the situation today and to improve the things that exist.
- *Horizon 2 (innovative solutions/services in the domain of the organization)* requires a combination of temperaments that can work on abstract ideas (Star Trek) and at the same time, focus on the customer/user (Blue Helmet). This combination allows you to develop new ideas and at the same time, focus on the needs of the existing customers/users.
- *Horizon 3 (visionary ideas in terms of solutions and/or technologies)* requires temperaments that can work on future ideas (Star Trek).This temperament allows you to open up the solution space and search for visionary ideas in various directions.

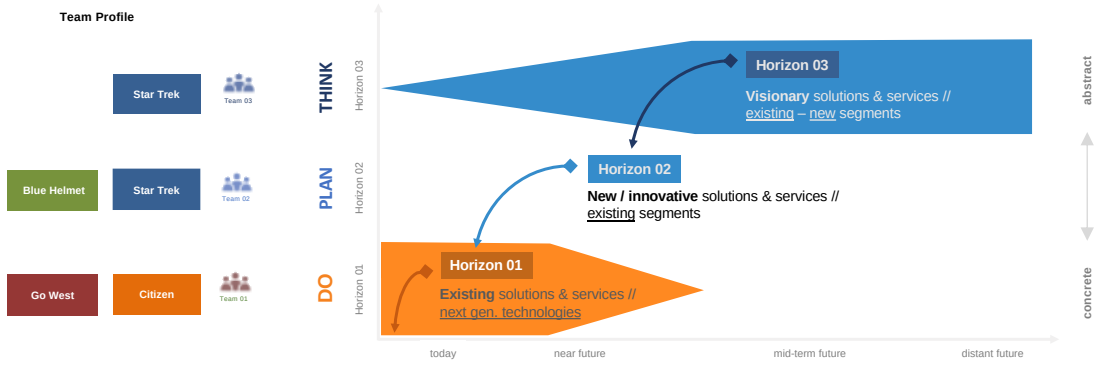


Figure 54 – Three horizon model and supportive team profiles

Here, it is important to recognize that all other temperaments are useful as well. The main message of Figure 54 is that the temperaments presented are better suited to dealing with specific challenges of each horizon.

Considerations for daily work

Understanding the profile of an existing team is a first step for working on team diversity. A team profile can be used as a tool to visualize the temperament diversity within a building team. The team profile is created from the cumulative number of temperaments of the team members and can be visualized with a net diagram.

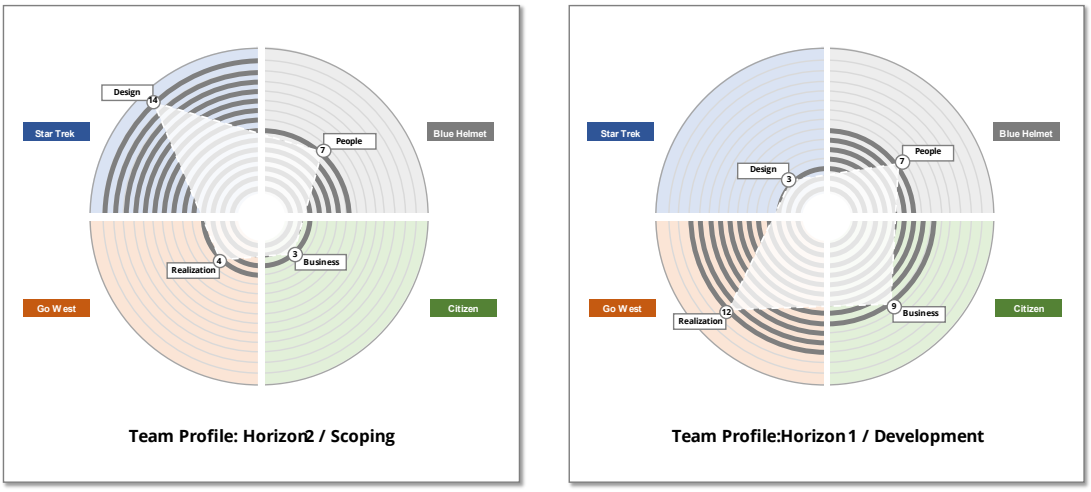


Figure 55 – Team profile examples

Figure 55 shows two team profile examples:

- Horizon 02 case (left diagram): mainly abstract idea-orientated (Star Trek) and future customer/user-orientated (Blue Helmet) team member. This profile can be suitable for the beginning of the scoping step.
- Horizon 01 case (right diagram): mainly concrete-orientated (Go West, Citizen) and cooperative (Citizen and Go West) team member. This profile can be suitable for the beginning of the development step.

4.3.3.3 Managing Working Style, Key Competences, and Role Assignment in the Building Process

From a practical perspective, three aspects are important for people management during the building process:

- Understanding the general challenges of each step of the building process in relation to people's temperaments (Section 4.3.3.3.1)
- Understanding the important skills for the digital age and their relationship to temperament (Section 4.3.3.3.2)
- Understanding typical roles in the building process and their relationship to the steps of the building process (Section 4.3.3.3.3)

Table 20 – Temperament suitability across the building process

<table>
<tr><th>Step</th><th>Main challenge</th><th colspan="2">Suitability of the temperament</th></tr>
<tr><td rowspan="2">Scoping</td><td rowspan="2">Vague future understanding, need for orientation, finding a vision, deciding a direction</td><td><u>Rational problem solving</u>
Guardians or artisans describe the concrete problem.</td><td><u>Reflective perspective</u>
● With their imaginative power, rationals design attractive future scenarios.
● Rationals discover, with the sponsor, the narrative for the case for action is translated and communicated by idealists.</td></tr>
<tr><td colspan="2">Provide input for defining the Digital Design brief:
● Idealists collect stories, perceive cultural forces, and filter relevant topics.
● Rationals and idealists design an acceptable vision, a design framework, and identify future stakeholders. They orientate the team to get an understanding of the scope.
● Guardians or artisans describe their understanding of stakeholder, context, and culture to date as an indicator of the change readiness of the people system.</td></tr>
</table>

Step	Main challenge	Suitability of the temperament
Conceptual	Explore solution ideas, frame the context, clarify the vision, build a community	• ***Rationals*** (outside of the industry) offer new ideas and challenge them together with laterally thinking *rational* process experts who are willing to break current rules. Together, they clarify the context and requirements. • ***Idealists*** define stories, collect demands, engineer requirements, find fans in the community, ensure understanding, evaluate commitment, and facilitate the team process.
Development & operations	Elaborate the details, build, and improve	• ***Idealists*** and ***artisans*** define stories, elaborate requirements, and ensure understanding. • ***Guardians*** manage the building process. • ***Idealists*** facilitate the team process. • ***Guardians*** and ***artisans*** implement the solution. • ***Rationals*** verify the quality of the overall architecture. • ***Idealists*** scale the community, check the acceptance by the decision makers, and recommend measures for continuation.

4.3.3.3.1 Managing Working Style from a People Management Perspective

We have seen that there is a relationship between the temperaments of the people involved and their suitability for different steps of the building process. People management in this aspect means identifying team members who are particularly suitable for the challenges of the step of the building process and transferring responsibility for the work to this person in order to unleash the individual potential.

Consideration for daily work

Table 20 summarizes the steps of the building process, including their main challenges (see Section 2.1.2), and discusses the suitability of the different temperaments in relation to the challenges. The table can be used as an overview and as a tool for team assignment and for evaluating the profile of existing teams.

Recent research shows that teams with higher diversity in the need for closure and tolerance for ambiguity outperform those with lower diversity. In other words, successful innovation requires both individuals with high tolerance for ambiguity and those with low tolerance for ambiguity to be on the same team [BeBa2007]. Building teams must therefore make a conscious choice at a given point in time related to the cognitive style it will allow to dominate its activities and working mode. For example, people who are natural designers are often not natural supervisors.

This choice of working mode should be guided by the stage of the building process, since each stage has preferable cognitive styles that correspond to the particular challenges of the stage. Role assignments in the building team are optimal if every team member takes the role they are best suited for. For example, during scoping, the suitable temperament has higher relevance than personal experience in order to explore alternative and potentially radical solution ideas.

4.3.3.3.2 Skills for the Digital Age and Their Relationship to Temperament

The World Economic Forum has defined a number of skills that are especially important in the digital age (see Table 21, cf. [WEF2016], [WEF2018]).

Table 21 – Temperaments and skills for the digital age

Skills for the digital age ([WEF2016], [WEF2018])	Temperament
• Analytical thinking/innovation • Active learning • Technology design • Creativity • Critical thinking • Complex problem solving	Star Trek (NT)
• People management • Originality	Blue Helmet (NF)
• Programming • Coordinating with others	Go West (SP)
• Critical thinking • Evaluation • Coordinating with others	Citizen (SJ)

These skills can be considered especially important when building and designing a digital solution. From a people management perspective, they can be associated with temperaments as well.

Considerations for daily work

Table 21 shows the corresponding temperament for each skill. The table can be used as a tool for identifying temperaments for each skill that will find learning these skills particularly easy.

4.3.3.3.3 Role Assignment from a People Management Perspective

Roles are a common tool for assigning responsibilities to people in a building process. Assigning roles to people can be supported by understanding their skills and experiences, but also by understanding their individual temperament. Many role names and titles are used in the digital industry. Table 22 is intended as a people management tool to support role assignment in the building process. It shows typical project roles, the main step of the building process where they contribute, a metaphorical title, a description of their responsibility, and the temperaments suitable.

Considerations for daily work

Table 22 can be used to support role assignment for a building process and to clarify the responsibilities of a person within a role. For example, if a person takes the role of a solution architect, an important responsibility of that person is to generate external impulses that stimulate the team to find new solution ideas. Furthermore, Table 22 can be used to check the fitness of a person that has the role in an existing building process. Finally, it is important to recognize that the DDP does not occur in Table 22 because we do not consider the DDP a role in the building process (see Section 1.4.3).

Table 22 – Roles, temperaments, and assignments

Typical project role	Main step focus	Metaphorical title	Responsibility	Suitable temperament
Project lead Project manager Product manager	All steps	**Guide**	Intermediary "I check business, people, context, culture, and playground!"	Star Trek, Blue Helmet
Client	Scoping	**Sponsor**	New business "I've got something on my mind."	Star Trek, Blue Helmet
			Tame problem "I've got a problem!"	Citizen
Inventor Designer Solution architect	Scoping, conceptual	**Inventor**	External impulse generators "We can do it this way!"	Star Trek
Business designer Process experts	Scoping, conceptual	**Domain experts**	Internal lateral thinkers "I break the rules!"	Star Trek, Go West
Business analyst Requirements Engineer UX designer Interaction designer Consultant Promoter	Conceptual, development & operations	**Bridge builder**	"Abstract/concrete" translator "I collect stories, demands and translate ideas & concepts. I'll get you fans!"	Go West, Blue Helmet
Developer Software architect Data scientist Analyst	Development & operations	**Implementer**	Practitioner "Let's do the things now!"	Go West, Citizen
User	Scoping, conceptual	**Customer**	Feedback providers "AHA!"	

4.3.3.4 Managing Leadership during the Building Process

Good teams rotate leadership as needed according to where they are in the building process (cf. [Pari2021]). Leadership should not go to the person whose turn it is next, but rather to the person most skilled and most suitable for the current step of the building process in terms of personality structure.

In this sense, good teams behave like bicycle racing teams, where individuals are assigned to positions during the race because of their strengths, not because of seniority or some other such measure. In these teams, everyone is in effect the boss at some point in time and is respected as a leader for that point in time when their skills are most needed.

Leading through the building process means understanding the building process and the need to move between the abstract and concrete and between analysis and synthesis to execute an efficient and effective process. This includes assembling the right mix of people on the team to execute the process and providing a leader for the building team who not only has the classic leadership skills, but who also understands the process and who is able to smoothly leverage and integrate the diverse ways of thinking that are represented on the team [BeBa2007]. The broad range of the building process and the associated personality requirements makes it clear that it is almost impossible for one person alone to lead the whole building process; it is a team leadership challenge, as described above. Figure 57 highlights suggested temperaments in the leading position at that specific point of time in the process.

Finally, every building process is also an orientation and a continuous understanding process in an interdisciplinary setting. In order to achieve this, the building team must also be able to engage people who can understand the future in order to create a logic of success that is acceptable to the team and the relevant stakeholders.

Considerations for daily work

To perform certain tasks to lead an interdisciplinary building team, the leader needs a mindset grounded in their own personality [Hüth2009]:

- To invite people to join a design process, personalities that like people and that are people persons are helpful.
- To inspire people to shape an attractive future, personalities that are fascinated by new things themselves are helpful.
- To encourage people to act against common rules and to think differently, personalities that are not afraid to act and to think for themselves are helpful.

The core characteristics necessary for these tasks are empathy and a certain maturity of self-confidence.

In addition, it is important to clarify roles and responsibilities, provide structure, and highlight the meaning and impact of work. The key is to create a team environment with psychological safety, where team members can take risks and question, for example, decisions, ideas, and directions, without feeling insecure or embarrassed [GOOG2005].

An efficient leadership team profile combination over the whole building process, but especially in the scoping and conceptual steps, can be suggested by:

- Introverted Star Treks/extraverted and decisive Blue Helmets

These profiles provide an environment for innovation, future mindset, gaining knowledge and insights, big picture, and active involvement of team members and customers/users.

The more stable the solution becomes, the more intensive the need for closure is and the sooner the leadership can be handed over to the following team profiles:

- Extraverted and decisive Star Treks/extraverted, feeling Citizen/extraverted, thinking Go West

These profiles provide an environment for implementation, need for closure, and operational support (see Figure 53).

4.3.3.5 Considering the Working Environment and the Context Understanding

The building organization (i.e., the organization executing the building process) has a major direct and indirect influence on the digital solution. The people within the organization must also accept and support the digital solution so that the building process and therefore the digital solution can become a success. Otherwise, the organization may hinder the building process or even cause it to fail [VPGV2008]. The working environment (e.g., time, workspace, psychological safety, leadership style, or clarity on expectations) has a decisive influence on people's actions and behavior and therefore their willingness to engage.

Another driver for engagement is an understanding of the larger context and the personal impact: the content dimension deals with the information needs of the people involved in the building process (see Section 2.2).

Considerations for daily work

Take two perspectives on the building process into account to identify a common understanding:

- The ongoing change of perspective between the building organization and its partners
- The future customers/users affected

Participants of the building organization influence the design and realization capability of a solution, customers and users influence acceptance. Every stakeholder of these two groups (participants and customers/users) has a different understanding of the vision of the solution, pursues different motives, and thus emotional needs. Check the understanding and commitment with a stakeholder list and be aware of and interested in the needs of every stakeholder in order to shape communication strategies and to modify the team setup.

The involvement of future customers/users is of great importance in achieving customer/user acceptance, especially for the design of digital solutions. The following aspects are intended as a checklist for the involvement of future customers/users:

- Customers *affected by the solution* need the solution to provide a clear added value to their needs in the corresponding situation. However, even during the building process there are anonymous customers (e.g., persona) or representatives who support the co-design process. They expect an early experience and an environment in which their criticism is heard and considered.
  Task for people management: Take the customer's perspective into account and understand context-related customer motives, establish a feedback culture.
- Create customer acceptance for the procedure. This is important for co-design so that the customer is also honestly involved.
  Task for people management: Explain the whole process to the customers who are part of the co-design process. Explain the responsibility, the type of participation, and the value contributed by the customer to the team and the solution maturity.
- *Understanding impacts*: For the customers affected, understanding the future context—challenges, trends, or solutions—reduces the feeling of concern .
  Task for people management: Depending on the duration of the building process, the future customer group affected notices that something is going on. As early as possible, organize a communication plan with persons responsible for transformation and offer space for participation and dialog.

4.3.4 Conclusion

People management is an important cross-cutting competence for a DDP. For a DDP at foundation level, the most important conclusion from this section is that people management is an important success factor for building a digital solution: it is not sufficient to manage people's skills, experiences, and availability—the DDP must consider personality indicators, an effective team diversity, and team fit.

This section is intended as a first introduction to the world of people management. We have used the Keirsey model as the foundation for this section. Experts from psychology will be aware of other models that could be applied to people management as well (e.g., the MBTI or the Big Five personality dimensions). We have chosen the Keirsey model since we believe that this model is a good starting point for beginners and provides good insights into the challenges and tasks of people management. The key takeaways from this model for people management in the building process are the different communication styles and their associated motives and behavior patterns:

- Understanding: future-oriented theoretician vs. concrete implementer; abstract vs. concrete perceiving and learning
- Action focus: cooperative, people-orientated vs. subject-orientated
- Leadership claim, communication style: directive vs. informative
- Impulsivity, energy: expressive vs. attentive, reflective

A further conclusion is that there are several challenges for the building process that can be addressed by taking people management seriously.

- A good building team requires roles that provide the bridge, that are able to translate between abstract thinker or designer and concrete implementer. A guide is needed to connect the people who are focused on a business, a solution, or a digital system.
- People need a working environment that supports the accomplishment of the tasks: apart from the necessary basic conditions, importance should be given to a suitable corporate culture of the building organization and a set of rules that provide psychological safety.
- A common context understanding among all stakeholders is an underestimated success factor in engaging for change.
- The targeted horizon of the digital solution and the step in the building process requires a suitable team profile that can change over the course of the building process—from an abstract-thinking, more open-minded team in the scoping step, to a team that becomes concrete and focuses on the closure.
- Leadership should go to the person most skilled and most suitable for the current step of the building process in terms of personality structure. Good teams rotate leadership as needed according to where they are in the building process.

A final conclusion for the DDP at foundation level is that people management starts with the DDP: the DDP must know their own personality, motives, and values. They have to know how these affect others in order to recognize and differentiate individuality in the building process and to be able to deal with otherness.

5 A Building Process for Beginners

This chapter provides guidelines for a complete building process for a digital solution (see Section 2.1.2). These guidelines originate from the experience of the authors of this handbook. Beginners should be aware that there are many other approaches for building a digital solution.

Studying other approaches is important and studying other approaches with practical experience is even better. The intention of these guidelines is to provide a starting point for beginners to gain their own experience and develop their own building processes. Experienced readers will be familiar with other approaches and will be able to define a completely different process or will even apply the techniques presented here in other steps or situations.

The guidelines assume that the reader is the person responsible for the digital solution and the management of the building process. Following the terminology from agile development, we call this person the product owner: the product owner is the accountable person authorized by the client to decide all matters related to the digital solution in consultation with the building team, the client, and other stakeholders.

The guidelines further assume that the product owner is working with a building team (in short: team) that has all the necessary skills to build the digital solution. A member of this team is called a building team member. Since the guidelines cover the whole building process, they go beyond the scope of Digital Design. This broader perspective is necessary to understand the integration of Digital Design into the building process.

We conclude this chapter in Section 5.4 with a discussion of *lean startup* as another prominent approach for starting a new digital solution.

5.1 Guidelines for the Scoping Step

The goal of the scoping step is to define the vision, the context, the scope, and the general terms of the digital solution (see Section 2.1).

In Section 2.1, we introduced the notion of wicked problems for characterizing two categories of design problems (wicked vs. tame problems). Although this notion is very useful for an initial understanding of the spectrum of design problems, it provides only little guidance for practical work.

A tool for measuring understanding and commitment

Building on the categories of wicked and tame problems, Figure 56 introduces a tool for measuring the understanding of a digital solution during the building process. A product owner can apply this tool to assess the understanding of stakeholders and building team members during the whole building process.

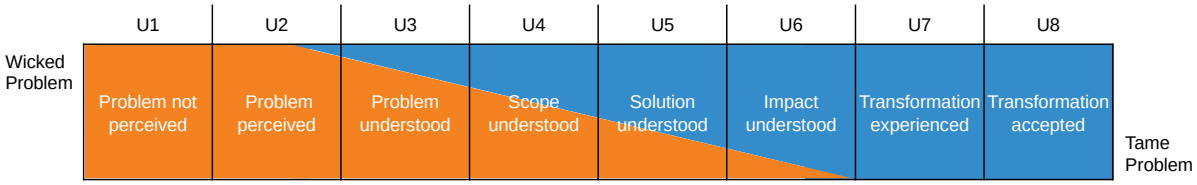


Figure 56 – A tool for measuring the degree of understanding of a digital solution

On the very left-hand side is a wicked problem. The problem is ill-defined and not even perceived by many people. On the right-hand side there is a tame problem that everybody accepts and that must be solved by the digital solution. Moving from left to right, the wickedness of the problem reduces and the problem becomes more and more tame.

In addition to understanding a digital solution, it is important to understand the level of commitment of the stakeholders. Figure 57 introduces a tool for measuring the level of commitment of stakeholder: A stakeholder in category C++ works proactively for the new digital solution and wants to bring it to a success. A stakeholder in category C-- works proactively against the digital solution.

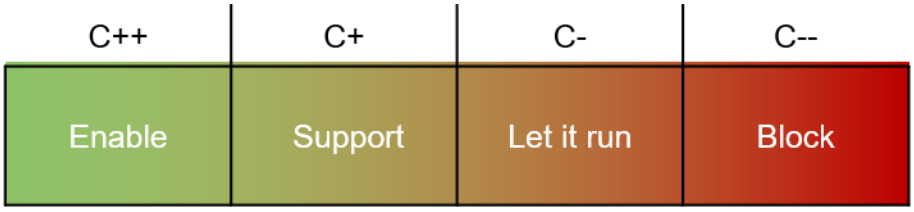


Figure 57 – A tool for measuring the level of commitment

Choosing an approach for coping based on understanding and commitment

In the following, we use both tools described above to explain different procedures during the building process.

When starting the scoping step for the new digital solution, we recommend two activities:

1) Get an understanding of the team you are working with in terms of temperaments and personality. Use the ideas presented in Section 4.3 and try to make use of the advice for the building process.
2) Get a clear picture of the level of understanding of all relevant stakeholders. If the understanding of relevant stakeholders is between level U1 and U5 (see Figure 56), follow our guidelines for a wicked problem (Section 5.1.1). Otherwise, the guidelines for a tame problem should be followed (Section 5.1.2)

As a final step, we provide guidelines for defining the general terms that are part of the Digital Design brief.

5.1.1 Scoping a Wicked Problem

The task of the scoping step for a wicked problem is to improve the understanding of the problem and to explore potential solution directions.

Design thinking as an approach for scoping wicked problems

We recommend design thinking [Brow2009] for this situation. Design thinking is a popular approach and consists of an iterative process with clear rules that emphasizes the importance of testing/validating ideas with early prototypes. Because of the broad range of resources already available, we do not provide the details of design thinking. Instead, we focus only on the particular aspects that are important for the application of design thinking in the scoping step.

The rules of design thinking further recommend that an interdisciplinary team performs the design thinking process. The result of a design thinking process is threefold:

- Documentation of different facets of the wicked problem
- A set of solution ideas validated by means of early prototypes
- A team of people with a detailed understanding of the problem (including the client's motivation) and the solution ideas

The last result (people with new understanding) is often underestimated in practice. Participants of a design thinking process gain significant knowledge that is very useful during the building process as a whole. We therefore highly recommend keeping the design thinking team as stable as possible during the whole building process to benefit from this understanding.

5.1.1.1 Phase 1: Setting up a design thinking team

To perform a design thinking process successfully, it is important to have an interdisciplinary team (8-10 people). A systematic stakeholder analysis is a good starting point for setting up the team. The onion model [Alex2005] is a simple tool for this initial analysis. It distinguishes between direct stakeholders of the system (e.g., users, operators), stakeholders of the containing system (the digital solution in Digital Design terminology, e.g., owners, sponsors), and the wider environment (e.g., legal entities, standardization bodies). Furthermore, the guidelines from Section 4.3 on temperaments should be considered and the understanding and level of commitment (see Figure 57) should also be evaluated (e.g., by means of short interviews).

The result of this stakeholder analysis should be documented in the Digital Design brief (see Section 2.2.2).

5.1.1.2 Phase 2: Performing the design thinking process

To perform a design thinking process at foundation level, the following recommendations are useful as a starting point:

- Set up a team of 8-10 people, make use of the results of the stakeholder analysis. Try to find a good mixture of people in terms of understanding. Try to avoid people that might block the digital solution (commitment C--). The product owner is part of the team. An unbiased moderator to guide the process is recommended.
- Try to include people with construction and realization experience as well. An optimal team includes at least one participant that is planned for the subsequent development and operations team.
- Define a clear timebox for the process. Plan time for post-processing the results. If timeboxing is unclear, plan for two weeks of design thinking and three days for follow up. If people are not available full time, provide a clear schedule (e.g., half days timeboxed) so that people can arrange their other duties properly.
- Determine which time horizon the problem is concerned with (use the horizon model, see Section 4.2). Open up the solution space as much as possible depending on the time horizon you want to address. Nurture a future mindset as a core culture of the group. Collect all input necessary for the time horizon (e.g., case studies on future trends and emerging technologies). Invite experts in innovative digital technology (e.g., artificial intelligence or blockchain experts), analog technology (e.g., material scientists), and others from sciences (e.g., social science) to give talks on their subjects in order to inspire the design thinking team.

- Include techniques for emphasizing with future customers, users, and further stakeholders. For example, create different personas with the team or develop a customer journey (see Section 2.2.3).
- Give authorization for open thinking and a future mindset. Scoping a digital solution requires open thinking and anticipating the impact of various digital materials. Create an environment in which people feel free to think openly and far into the future (cf., e.g., [GOOG2020]).
- Use simple prototypes such as paper prototypes and storyboards (see Section 2.3).
- Make use of the future press release (see Section 2.2.3) as a temporary artifact to develop different visions in your team.
- Search for competitors, related solutions, and patterns that change whole industries as a source of inspiration. Document them in the Digital Design brief as well.
- To evaluate the ideas developed, invite a broad range of stakeholders in terms of understanding and commitment. It may be difficult to convince stakeholders with a negative commitment (C-, C--) to participate in an evaluation effort. However, from our experience, they may provide critical input that will help to improve the ideas and the evaluation of the ideas may help to convince these stakeholders to support the new solution.

5.1.1.3 Phase 3: Documenting the results and iterating if necessary

At the end of the design thinking process, a lot of material will have been created. The product owner is responsible for evaluating this material. If the results are not convincing, plan for another design thinking process. From the point of view of Digital Design, design thinking must not only enable team building and the training of a desired working attitude but must also be consistently understood as a results-oriented approach.

The Digital Design brief template serves as guidance for documenting the results (see Section 2.2.2). A future press release (see Section 2.2.3) is a good tool for describing the vision of a digital solution in the sense of a desired future. The design thinking process may produce different solution ideas. All ideas should be documented in the Digital Design brief.

5.1.2 Scoping a Tame Problem

A wicked problem is a challenge for every building process. However, a problem that appears to be tame at first sight should not be underestimated. Our own experience has taught us that many problems appear tame initially and that their wicked nature becomes apparent when we look at the details and the contextual area of conflict (see Section 2.1.2). Every beginner in Digital Design should take this as a serious warning.

Systematic analysis as an approach for scoping a tame problem

The task of the scoping step for a tame problem is therefore to challenge the problem understanding (and tameness of the problem) and to define a clear vision for the digital solution. We recommend an analysis-oriented approach consisting of a series of interviews and a compact scoping workshop format for this situation.

Even if potential stakeholders are known, a systematic stakeholder analysis (see Section 5.1.1) should still be conducted. This allows important stakeholders that have been overlooked so far to be identified and invited to the workshop. Try to identify important stakeholders with a low level of understanding (U1–U3) of the solution and invite them to the workshop. Such

stakeholders will make the step more challenging but will definitely improve the outcome in terms of detailed understanding.

5.1.2.1 Phase 1: Interview important stakeholders

All important stakeholders identified should be interviewed by the product owner to collect their understanding of the problem and their commitment. Chapters 1–3 of the Digital Design brief can serve as a rough interview guideline. If the problem requires it, further experts should be consulted. For beginners, the maxim should always be to conduct all interviews yourself in order to deepen your own understanding of the problem. We further recommend an additional person to help document the interview.

Start with a client interview to identify the client‘s motivation for the project, the level of problem understanding, and the case for action experienced.

The results of this series of interviews can be documented in the Digital Design brief. If you do not obtain a clear picture of the problem from this process, we recommend shifting to the approach for wicked problems (see Section 5.1.1).

If the results show a clear picture of the problem, we recommend a scoping workshop to discuss the results from the interviews with the relevant stakeholders identified. The goal of the scoping workshop is to question the problem understanding in detail. For this purpose, a good mixture of stakeholders with positive and negative commitment is key to getting realistic feedback.

5.1.2.2 Phase 2: Scoping workshop

In order to prepare the scoping workshop, the product owner elaborates a future press release, a story board, and paper prototypes to visualize the overall vision of the digital solution. When preparing the material, keep in mind the level of understanding and the commitment of the stakeholders. Include a reasonable description and explanation of the material presented so that stakeholders with a low level of understanding have a chance to catch up and get a good understanding of the material.

This material can be sent to the workshop participants beforehand for preparation but this is not essential. Both options are possible and depend on the situation at hand:

- We advise sending out preparation material if the topic is rather complicated and requires in-depth study by the participants. The participants are invited to send feedback to the product owner before the workshop. Finally, the product owner collects and analyzes the feedback.
- If you want to see the immediate reaction of those involved, you should not send the material for preparation but instead, present it directly in the workshop. This allows you to capture immediate reactions and spontaneous impressions and discuss them straight away.

We recommend the following steps as an agenda for a scoping workshop:

Step 1: The product owner presents the future press release

All participants discuss the feedback and work and agree on a final future press release. The discussion should focus especially on the downsides of the future described. Is the future described really desirable for people, business, and society? How might the future described

be perceived from a negative point of view? Here, stakeholders with a negative commitment can be a valuable source of insights.

Consider the YPRC future press release as an example. A negative point of view on YPRC may come from professional running coaches. They might consider YPRC as a threat to their own business. How might they react when the solution is presented to the market?

Step 2: The product owner presents the storyboard, the paper prototypes, and the feedback

All participants discuss the feedback and decide on potential modifications/additional input. The discussion should especially try to focus on the exceptional situations and their impact on the solution. The group thereby identifies key stories (for example, exceptional situations) and relevant themes (for example, topics that are related to the solution) that frame the scope of the solution in a more detailed way.

An exceptional situation might be a remote coach that puts a runner under so much stress that the runner collapses during a training session. How does this feel to you if you put yourself into the role of the people designing YPRC?

Step 3: The product owner presents the Digital Design brief elaborated, including the feedback

All participants discuss the feedback and decide on potential modifications/additional input. Here, the discussion should especially focus on the details of the budget allocation and the timeframe. The participants should especially focus on critical aspects of the general terms. What happens if the budget is not sufficient or if the timeframe is too short?

The concrete timeline depends on the amount of material and level of common understanding (see Section 4.3). Up to three days of workshop time are appropriate. If this timeframe seems too short, several workshops on dedicated sub-themes should be planned.

5.1.2.3 Phase 3: Document the results and iterate if necessary

At the end of the workshop, the product owner updates the Digital Design brief according to the feedback to create a final version. If it is not possible to create a final version, a second round of interviews and a second workshop should be scheduled to clarify the open issues. If the second round does not produce a clear result either, we recommend changing the procedure to a wicked problem approach (see Section 5.1.1).

5.1.3 Defining the General Terms for Building a Digital Solution

The general terms in the Digital Design brief separate schedule, budget, and available resources (see Section 2.2.2). The following guidelines discuss each of these parts.

5.1.3.1 Guidelines for Defining the Schedule

It is not usually realistic to expect to be able to define precise schedules during the scoping step. However, without an initial schedule, it is difficult to keep track of the progress. At foundation level, the following guidelines are useful for defining a schedule:

- Plan the conceptual step iteratively and with timeboxes. If there is a low level of understanding among all relevant stakeholders, shorter timeboxes should be preferred to obtain regular feedback and to improve the level of understanding among all relevant stakeholders.
- Arrange fixed deadlines with your sponsor to review the results.

- Plan for a maximum of three months for the conceptual step. Longer conceptual phases increase the risk of undesirable developments, as many questions of detail arise only during realization. In this case, consider whether the solution can be broken down into parts so that implementation can start more quickly.
- If you are dealing with large teams, plan slack time for communication and shared understanding.
- If time for the conceptual step is fixed (because the start of development is fixed), plan for at least three iterations: use 25% of the time for a first and fast iteration, 50% for the second, and again, 25% for the final iteration.
- Be prepared to kill the solution idea early if you do not make progress or get commitment from the stakeholders for the solution idea. Go back to the scoping step.
- If the delivery date of the digital solution is already fixed, plan to deliver a first version of the digital solution after 50% of the time. Make your sponsors aware that a fixed date means that the scope of the digital solution must be adjusted to meet the deadline.
- Analyze how people indicators such as temperament, understanding, and commitment, as well as insight maturity of the digital solution, can impact the timeframe. Plan for additional time to take care of important stakeholders.

5.1.3.2 Guidelines for Defining the Mode of Cooperation

The building team does not develop the digital solution for itself, but for the client, the customers and the user. Therefore, the mode of cooperation between the building team and all relevant stakeholders is of particular importance for the success of a building process and for clarifying the role of all participants. At foundation level, the following guidelines are useful for defining the mode of cooperation:

- Schedule regular meetings with the client to update them on progress and to make important decisions regarding the digital solution and document them. At the beginning, the intervals of these exchanges should be rather shorter in order to react quickly. Extend the interval of meetings when the topics become less urgent and shorten it when the topics become more urgent again.
- Explicitly define the rights and obligations of the client. Important rights and obligations of the client can be: Short-term information about critical topics (e.g., scope, budget, or schedule); insight into all concepts; comprehensible justification of the building team's decisions.
- Define rights and obligations of the building team explicitly. Important rights and obligations of the building team can be: Quick decisions by the client and other stakeholders; Explicitly defined scope for decision-making about the design of the digital solution (when must the client be consulted, when does the team decide);
- Explicitly define the rights and obligations of customers and users in the building process. Important rights and obligations can be: Scope for decision-making on the design of the digital solution (when does the building team have to implement the requirements of the customer and / or user, when can the building team decide for itself); right to get information about the digital solution (which information may customers and / or users get, which not).
- Define an escalation instance for the event of a crisis. If an unresolvable conflict arises between different parties, then a higher-level authority is needed to resolve this conflict and bring about a decision. This authority can be, for example, a representative of the

client or a neutral authority. It is important that all parties agree on this authority and respect the decision.

5.1.3.3 Guidelines for Defining the Budget

Budgeting is as difficult as defining the schedule. However, without a budget, it is difficult to keep track of the progress and the costs. At foundation level, the following guidelines are useful for defining a budget for the conceptual step.

- Plan explicit budgets for the team that is working on the conceptual step.
- If the team includes part-time members (a situation often faced in larger organizations), plan part-time members in timeboxes on a full-day schedule.
- If necessary, plan extra budget for additional experts (e.g., software developers for prototyping, interaction designers, user researchers, software architects).
- If necessary, plan extra budget for incentives (e.g., money for user tests).
- If you expect a substantial number of relevant stakeholders with negative commitment, plan for extra budget in order to perform measures that help to convince relevant stakeholders.

Always plan for a development and an operation budget with your client/sponsor. Even if this step is difficult for your client/sponsor, these budgets are an important reference point for the building process to recognize that the initial budget is not sufficient.

5.1.3.4 Guidelines for Defining Potential Revenue Streams

Just like the budget and schedule, it is difficult to define potential revenue streams during this step of the building process. However, if it is not possible to define at least a rough idea of the revenue stream, the chance of defining a strong business model is rather low. At foundation level, you should consider the different digital business models presented in Section 4.1. Try to apply each digital business model to your solution.

We further recommend trying to estimate the maximum number of customers that might use your solution (or any other measure for growth). A good way of thinking about this is the *invisible asymptote* [Wei2018], which is an estimate of your maximum level of growth. From such a number, you can work backward to calculate your potential income. Keep in mind that the number does not need to be perfect at this stage of the building process. More details will be elaborated during the conceptual step.

5.1.3.5 Guidelines for Defining Available Resources

Available resources cover personnel and technical resources that are necessary for the conceptual step or the development and operations step. An explicit definition of these resources is necessary so that they are available when they are needed.

The concrete type of resources needed depends on the digital solution. Common resources for the conceptual step are:

- Additional experts for the conceptual work (e.g., requirements engineers, business analysts, interaction designers, software architects, industrial designers, usability engineers)
- Availability of stakeholders for reviews, workshops, and evaluation
- Availability of external moderators (e.g., for workshops)
- Appropriate workshop rooms and offices for teamwork

- Compliant infrastructure for distributed working (e.g., file sharing, video conference)
- Special software and hardware for conceptual work and prototypes (see Section 2.3, e.g., mock-up software, development environment, frameworks)
- Workshop materials (e.g., cards, post-its, pens, pins, etc.)

Common resources for the development and operations step are:

- Software/hardware development team
- Software and hardware for developing the digital solution
- Hardware for operating the solution under development
- Test environments for quality assurance
- Hardware for operating the digital solution
- Personnel to support the transformation process (if a transformation process is necessary)
- Other personnel necessary to operate the digital solution (e.g., support)

5.2 Guidelines for the Conceptual Step

The conceptual step is an upfront step that takes place before the development of the digital solution starts. The goal of this step is to gain a sufficient understanding of the intended digital solution before taking the risk of starting development. Two results are created for this goal:

- The initial solution design concept
- The initial system-level design concept

These concepts are called initial because they are further refined and revised during the development and operations step (see Section 5.3).

The conceptual step is about shared understanding

A typical misunderstanding of conceptual work is that it is directed towards the creation of documents. This misunderstanding originates from the document structure of concepts (see Section 2.2.2) and the belief that the documents must be processed sequentially like a kind of questionnaire or checklist. Quite the opposite is true: conceptual work is directed towards shared understanding and is a highly iterative process that integrates an analytical, creative, and heuristic mindset.

Human-centered design process as a working mode in a long-term workshop

A general process model for beginners in the conceptual step is the human-centered design process [ISO2019]. It consists of four iterative activities of equal importance: understand, define, design, evaluate.

A good working mode for beginners in the conceptual step is that of a long-term workshop of several weeks with the product owner, the building team, and other experts in which the client, potential customers, potential users, and other relevant stakeholders take part when needed.

A permanent workshop room as home base for the team

We recommend setting up a permanent workshop room in which conceptual work can take place (cf., e.g., [GBGSV2013]). The walls of the room can be used to visualize the various results of the conceptual work. One wall can be dedicated to the solution design concept, one wall can be dedicated to the system design concept, and one wall can be used for the

construction perspective. The remaining fourth wall can be used for collecting material, for managing work, or for alternative perspectives (for example, change management). An alternative approach is to dedicate three walls of the room to alternative solution ideas and one wall to work management.

With such an approach, the room itself becomes a medium on its own that can be used for creative work and also for presenting the ideas developed to the client and other relevant stakeholders. In the following, we describe four phases as a work structure for the conceptual step.

5.2.1 Phase 1: Explore the solution space for the digital solution from the customer perspective

Beginners in Digital Design should always start with an exploration of the solution space as phase 1. The value proposition, the customer journey, and the customer personas of the digital solution (see Section 2.2.3) are the work products that we elaborate in this phase. In a permanent workshop room, all three work products (value proposition map, customer journey map, and personas) can be created on the wall with post-its and tape.

The vision and situation description from the Digital Design brief are the starting point for the work in this phase. The following sequence of five activities should be performed.

Activity 1: Develop empathy and understand the situation of potential customers and users

As a starting point, the team should identify real persons who can become potential users or are already customers and/or users. The team should interview such persons to get a deep understanding of their situation, their jobs, needs, and desires. Besides personal interviews, field research is an important source of information here. The team can observe people in their daily work, for example, or can try to take over the jobs of the persons concerned (e.g., do the work themselves).

Activity 2: Define the core insights into customer jobs, pains, and gains from today's perspective

When the team has gathered sufficient understanding of potential customers and users, the team should collect the findings and define the core insights from activity 1. For this activity, the team can use persona templates, stakeholder lists, and the customer profiles from the value proposition canvas and the customer journey map (see Section 2.2.3).

Start the work in the team by defining an initial set of customer/user profiles and elaborate a persona template for each customer profile. Prioritize the customer personas defined to understand their importance for the solution.

As soon as the personas have been defined and prioritized, work on the customer profile of the value proposition canvas:

- Create a note in the customer job(s) box for every major and ancillary job you intend to help your customer get done.
- Create a note in the pains box for every pain your customer experiences or could experience before, during, and after getting the job done.
- Create a note in the gains box for every benefit your customer expects, desires, or would be surprised by.
- Create a note on the customer journey map for existing activities and touchpoints.

Within this activity, try to integrate the personas defined, the value map, and the existing customer journeys to get a consistent understanding of the current situation of the customer:

- Evaluate current experiences in the customer journey map: how does the customer experience the touchpoint today? Do expectations fail to materialize, are they fulfilled, or does something happen that the customer did not expect?
- Identify trust points and relevant needs: what is the relevance of a touchpoint in the respective situation?
- Transfer relevant needs and pains into an initial value proposition: what are the key needs and pains in the moment of truth?

It is completely normal to switch between the work on persona templates, the customer profiles, and customer journey maps since working on one work product improves the team's understandings of the others.

Activity 3: Define and frame the value proposition together with the customer journey from a future perspective

As soon as the team has reached a good consensus on the current situation of the customer using personas and customer profiles, the team can approach the future perspective of the digital solution. This future perspective should not be underestimated in Digital Design since the potential of innovative digital technologies can only be evaluated from a future perspective as they are often not in place today.

The team should start by exploring potential trends for future experiences and future needs. For example, the availability of reliable voice interaction technology can create a different interaction and service experience in a digital solution. Another approach is to transfer existing services (e.g., real-time parcel tracking) to the domain of the digital solution.

The needs identified can be integrated into the existing value map. A pragmatic approach is to document the needs with notes in a different color to indicate that they address future needs. In a second step, the team can evaluate the future needs identified in the customer journey map to finally identify issues and opportunities that enhance the solution space.

Activity 4: Explore the solution directions and create solution ideas

With this enhanced understanding of issues and opportunities in the solution space, the team can start to explore alternative solution directions and create solution ideas.

Start the work with the value map:

- List all the product and service ideas your value proposition is built around by creating a note for each element in the products and services box of the value map.
- Describe how your digital solution creates value within a customer journey map by either eliminating customer pains or creating customer gains.

Alternative solution ideas can be defined within a single value map or with several value maps. Here, it is important to recognize that a product/service idea defined in the value map does not necessarily define the whole digital solution. A digital solution consists of several product/service ideas.

With a good team understanding of the value map, the team can start to work on customer journey maps. Elaborate customer journey maps that put the product/service ideas into the

real-life situation of potential customers. Define customer journey maps for each persona to explore their personal journey and experience within your solution idea. Keep in mind that the customer journey map is a very detailed work product that allows you to describe very concrete situations and experiences of the customer. When defining the customer journey, also consider different technologies that you have defined in the Digital Design brief.

Similar to activity 2, the work on the value map and the customer journey maps are closely related to each other. You will therefore definitely switch back and forth between both work products.

Activity 5: Lightweight evaluation of solution ideas

As soon as the team has reached a good understanding of alternative solution ideas, the team should evaluate their ideas. Since we are in an early stage of the process here, we recommend a lightweight evaluation approach.

The team can invite the client, potential customers/users, and further stakeholders to the permanent workshop room. There, the team can present the solution ideas and obtain immediate feedback. In this form of early evaluation, it is important that the team creates a relaxed and open workshop atmosphere so that open feedback can be given.

Perform the five activities at least three times before approaching a detailed evaluation

These five activities should be performed at least three times to further improve the understanding and the value propositions, customer journeys, and personas defined. At the end of this process, a clear understanding of the value propositions should be achieved. If this is not the case, the process should be repeated. A clear understanding of the value proposition has been achieved if the team can present a clear value proposition that is easy for external people to understand.

Table 23 – Structure of a pitch presentation with example content from YPRC

Topic	Example from YPRC
Who is the customer that will benefit from the digital solution?	YPRC addresses beginners in long-distance running.
What is the value that customers expect from the solution?	Our customers expect professional coaching for making a good start with long-distance running.
What is the main reason for the customer to choose our solution?	YPRC delivers an affordable and individual personal coaching experience.
What is the digital solution about and what value proposition does the solution offer?	YPRC provides a remote coaching service that connects the runner with a remote running coach through a digital voice connection.
How does the solution achieve the value proposition for the customer?	Using a smartwatch and a smartphone app, the running coach can monitor the runner's performance and health data. With this data, the coach can give coaching advice through a voice connection.

How does a customer experience our digital solution?	Our customers appreciate the individuality of our coaching service. They do not need to rely on fixed running groups. Instead, they can book an individual coaching appointment when it suits them best.
How does the client experience our digital solution?	The YPRC company is proud to be the first company on the market that is able to offer such an affordable and individual coaching solution.
What is the roadmap for realizing the digital solution?	The first version of YPRC will use a white label smartwatch to capture health data. The development of the smartphone app and the coaching portal will take approximately 9 months. We expect to start the service with friendly customers within 12 months.
What is the ultimate vision for the digital solution?	The intention is for the YPRC to become a market-leading solution for coaching services in running. In addition to personal coaches, the vision is to offer a digital voice coach that uses artificial intelligence.

After three iterations, the two or three best solution ideas should be prepared for more detailed evaluation by means of a pitch presentation for the client and a fictitious video prototype (see Section 2.3).

The pitch presentation is a format that is common in the startup community (cf., e.g., [Ries2011]). It is a short presentation of the whole idea behind a startup and is used to convince investors to support the startup. Within the building process for a digital solution, we recommend using the pitch presentation as a tool for giving a short and crisp presentation of the digital solution. The presentation should be a maximum of 10 minutes long and should cover the topics shown in Table 23.

The commercial video should be presented in addition to the pitch to various potential customers/users and to further stakeholders to gather their feedback. The concrete content of the video, together with the evaluation goals, should be described briefly in order to create a simple solution evaluation concept. In addition, the team should define a short list of questions that the stakeholders should answer after watching the video. If the solution ideas are accepted, the next phase can be approached. If not, the process starts again.

5.2.2 Phase 2: Elaborate and evaluate solution candidates from a business perspective

The second phase deals with solution candidates. We recommend focusing on the business model to keep an appropriate level of detail. The business model canvas (see Section 2.2.3) and a system design canvas serve as a guideline for this task.

Create a separate business model canvas for each solution idea

We recommend creating a separate business model canvas for each solution idea. In a permanent workshop room, the business model canvas can be created on the wall with post-its and tape.

The business model canvas (see Figure 18 and Figure 58) is an important tool for getting an initial and deep understanding of the solution ideas. The team describes all views on the canvas: customers, technology, and environment. They collect information, interview experts, and study potential customers and barriers or potential of the development organization.

There is no particular order to follow for the canvas. A good place to start is the customer point of view since the business is centered around the customers. A good sequence is as follows:

- Fill the customer segments and value proposition elements with the results from phase 1 (value proposition).
- Define a clear name for the business to address the client's perspective as well. The name should reflect the case for action defined by the motivation for the digital solution as part of the solution design concept.

The other elements must be elaborated by the team. If necessary, external experts should be involved:

- Describe channels and customer relationships
- Evaluate the revenue stream
- Describe key resources, key activities, and key partners
- Evaluate the cost structure

Iterate the elements of the business model canvas several times

A good business model canvas is created by iterating the different elements several times. It is common for the whole content on the canvas to be thrown away in this process. During these iterations, the overall value proposition of the business model (the *story* for the customer) should be challenged again and again. The business model can only deliver good value to the customer if the story is clear for the customer.

Figure 58 summarizes a good working sequence for and the relationships between the elements of the business model canvas. Certain types of digital solutions (e.g., internal systems within a company) do not require a fully elaborated business model canvas. Nevertheless, we recommend staying with the full canvas and filling the parts in the best way possible.

The business model canvases created must be evaluated together with the client/sponsor. Like the evaluation of the value proposition canvas, we recommend defining and documenting clear evaluation goals in the solution evaluation concept (e.g., evaluation of the price for a certain service). The feedback from the evaluation should be incorporated into the business model canvas.

Create the business model canvases in three rounds with different stakeholders

We recommend creating the business model canvases in three rounds. The first round should be quick and only a rough sketch that can be discussed with the client/sponsor and relevant stakeholders. The second and third rounds can be more detailed. During this process, it is important to continuously discuss the case for action of the customer/client and to monitor whether the case for action changes. In our experience, the case for action of the client/customer will change and become more concrete during early conceptual work.

We further recommend keeping an eye on the problem understanding (see Figure 56), the case for action, and the commitment of the relevant stakeholders (see Figure 57). An improvement in the problem understanding and the level of commitment is a good sign that the solution idea is developing in a good direction. If no improvement is observed, the building team should try to understand the reasons for this and incorporate the insights from this analysis into the further development of the digital solution.

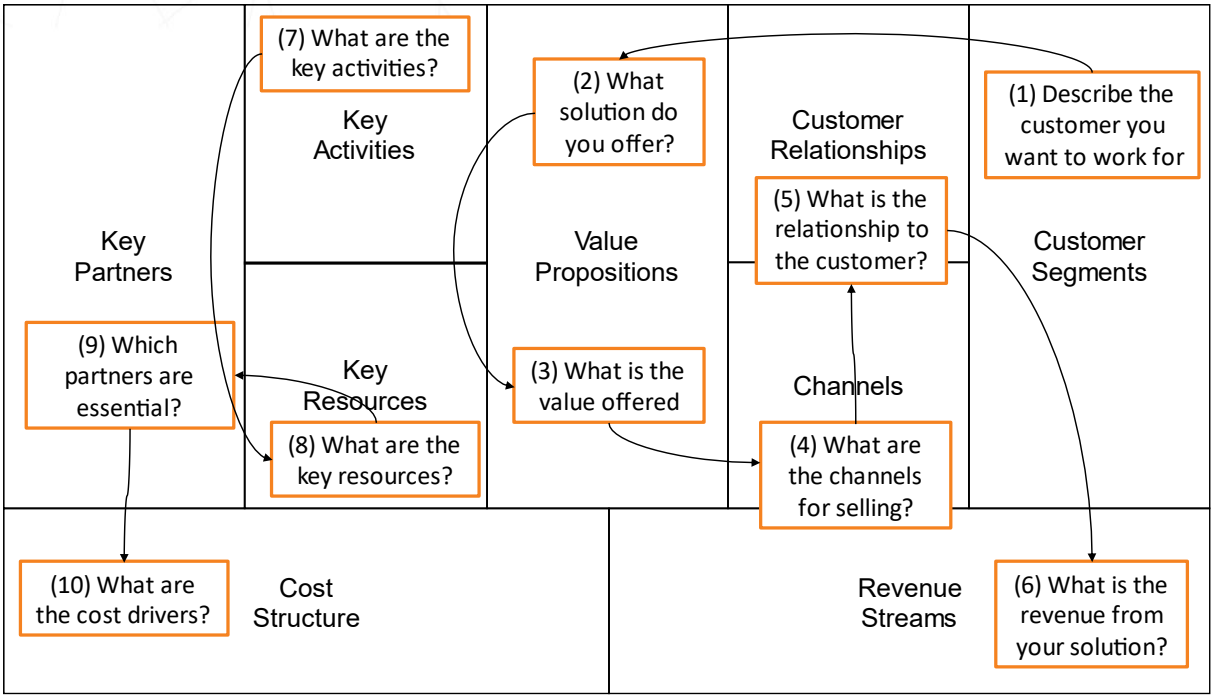


Figure 58 – Relationships that support the development of a good business model canvas

Elaborate solution candidates with a system design canvas

With the information from the business model canvases, the team can start to elaborate an initial system-level design concept for each idea.

In this phase of the process, alternative solution ideas are desired and valuable. Hence, the team should elaborate alternative and lightweight system-level design concepts.

In a permanent workshop room, the system design concept template (see Section 2.2.2, Table 5) can be used as a canvas structure instead of a document structure. We recommend starting with the description of the purpose (goals) and the function of the system (scenarios). Other elements of the system design concept can be derived based on the scenarios. The following guidelines further support the creation of a system design concept from the solution design concept:

- Think of user types that will support the value proposition. User types can be derived from the customer segments if the customer is also a user. Start with one user type for your customers. Keep in mind that there might be additional user types that are important for running the business (e.g., the coach from YPRC). Other possible user types cover customer support or administrative personnel.
- Existing objects depend on the solution idea. Typical examples of existing objects are devices that the customer will use to run a digital solution (e.g., a smartphone, a tablet, or a personal computer).
- Existing systems depend on the solution idea and the business model. Ask yourself what services are necessary to create the solution? Typical examples of existing services are payment providers and systems that provide additional services or data (e.g., map data).
- Defining elements of the digital solution is difficult. In a connected world, most digital solutions consist of at least two elements: one element for the direct user and one element that represents the server or backend of the digital solution.

- Quality requirements and constraints can occur at any time during the process. Whenever an element of the solution is defined, ask yourself the following: what are important qualities of the new element? What are constraints that apply to the new element?

Guidelines for finding the right level of detail for system design of a solution candidate

A typical beginner's mistake in creating a system design concept is overcomplicating things and trying to build everything yourself. The following guidelines will support you in finding the right level of detail for the system design:

- Focus on a simple solution that supports the core process of the digital solution and define the details in the element design concepts. Remember, the system design concept is about defining the overall structure of the digital solution.
- Administrative aspects of the solution (registration, login, logout) are a matter for the element design concepts. Special requirements on safety and security can be documented by means of short textual quality requirements. For example, writing a scenario that describes a two-factor authentication of the user in detail is a waste of time in the system design concept. Instead, a short quality requirement is sufficient (e.g., *The app must provide a two-factor authentication for login*).
- Avoid defining exceptions and alternative behaviors at the system level. Describe scenarios that illustrate the general function of the solution (sometimes called the *happy path*) and leave the details to the element level (and the use cases).

During this phase, additional input from external experts will be necessary. Additional interviews and workshops with potential users may also be necessary.

In order to evaluate the different system level concepts with users, the team should create simple interactive mock-ups as prototypes of the solution. Dedicated tools exist for this purpose and allow fast and lightweight creation of such interactive mock-ups. Use the scenario description as a script for the prototype and prepare the mock-ups in such a way that they illustrate the interaction with the digital solution during the scenario. Furthermore, the team should prepare interview questions that the users should answer after using the prototype. All these aspects should be documented in the system evaluation concept.

With this approach, you can also create *Wizard of Oz* prototypes that simulate the digital solution for potential users and further stakeholders. These prototypes allow you to gather feedback from the potential users and further stakeholders and to identify early usability issues (see Section 4.1.4).

5.2.3 Phase 3: Approach a promising solution candidate from a feasibility perspective

With the insights from the interactive mock-ups, a larger iteration cycle can start in phase 3 to examine the feasibility of the solution candidates. The feedback may result in modifications to the solution design concept (value proposition, customer journey, and business model) or the system-level concept.

The team should iterate both concepts and again evaluate them by means of pitch presentations, simple video prototypes, or interactive mock-ups. You can even combine both elements: the interactive mock-ups can be used to represent the digital system in the video.

The primary goal is to get a deep understanding of the feasibility of a solution candidate in terms of technology (is it realizable?) and operability (can we operate the solution?).

The intensity of evaluation activities should be defined in line with the level of problem understanding. As a rule of thumb, a lower level of problem understanding requires more concrete and realistic prototypes to improve the understanding of relevant stakeholders. In addition to the prototypes, good stories should be prepared that explain the application of the prototype in its later context. Another approach for evaluating the overall solution idea is the lean startup approach (see Section 5.4).

After a certain number of iterations, a promising overall idea of the digital solution should emerge. If the iterations do not lead to a stable idea, the team should consider going back to the scoping step.

5.2.4 Phase 4: Final evaluation of the solution candidate with the client

Once a stable idea for the digital solution and the underlying system has been defined, the team should elaborate the solution design and system-level concepts in detail according to the templates defined (see Section 2.2.2).This is where the real document work starts.

During the elaboration of the detailed system-level concept, important questions related to the feasibility of details may arise. Simple functional prototypes should be created by realization experts to evaluate the feasibility of these details.

YPRC example. There are two good examples in the YPRC case study for such feasibility questions: the feasibility of the remote coaching function and the feasibility of the artificial intelligence coaching (please read the case study for further details). Such early prototypes should generally focus on the technical feasibility. The user is not the focus during this stage.

Once the detailed solution design and system-level design concepts are available, a final evaluation is necessary:

- Review the concepts with your sponsor/clients and relevant stakeholders.
- Use interactive mock-ups and/or a commercial video for potential users/customers.

The concrete procedures (e.g., review checklist, feedback questions, prototypes used) and review results should be documented in the solution and/or system evaluation concept. The documentation of this information is particularly important for traceability reasons and to document the common understanding and feedback from all stakeholders. Documenting these details may become especially useful as a reference point during the development step.

In addition to the evaluation of the details of the digital solution, it is also necessary to finally evaluate the problem understanding and the level of commitment of the relevant stakeholders. The evaluation can be a byproduct of the evaluation of the details of the digital solution. At this stage of the building process, it is of great importance that all relevant stakeholders have at least understood the solution in general. Furthermore, no relevant stakeholder should be in a block level (C--, see Figure 57).

If the results from this evaluation are positive, the development and operations step can start. Otherwise, further iterations are recommended, especially if there are still stakeholders with a negative commitment.

5.3 Guidelines for the Development and Operations Step

The goal of the development and operations step is to bring the digital solution to life and to maintain it. The main results of this step from the design perspective are:

- The digital solution in operation
- Element-level design concepts that describe each element implemented (see Section 2.2.2)
- A revised design brief, solution and system-level design concept that reflect the detailed decisions from the development and operations step

There are many process models for developing a digital solution

Industrial practice has developed various approaches, methods, and process models for developing a digital solution. Experts will be aware of the broad range of approaches (e.g., V-model, Scaled Agile Framework, Rational Unified Process, and lean startup). Beginners in Digital Design often find this broad range overwhelming. The important message for beginners is that there is no single ready-to-use approach for developing a digital solution. All approaches are a kind of framework that must be tailored according to the specific situation.

A beginner's process inspired by Kanban and Scrum

The process presented in this section was inspired by Kanban and Scrum and uses elements from both sources that we believe are particularly suitable for getting started in the building process for digital solutions. Such elements were adapted to best fit to Digital Design. The process is therefore neither Kanban nor Scrum in its purest form.

Kanban is a lean approach for managing software development processes [Ande2010]. Scrum is a framework for developing and sustaining complex products [ScSu2020]. Both are valuable and widely adopted; they have large communities that can provide beginners with various resources for tailoring a development process. A practical aspect of using Kanban and Scrum as a foundation is that their mechanics are well supported by several software tools.

Expectation management on the process presented

Before we get into the details, we would like to do a little expectation management for the reader. The development and operations step of the building process is by far the most complicated and challenging step. The following section presents the various aspects of this step in detail.

The goal of the DDP foundation level is to teach the basic mechanics of this step and its relationships to Digital Design. This understanding is the foundation for learning to work with such a process. The additional explanation is intended to provide the complete picture of the building process.

A reader with no experience in building a digital solution will have to take a significant learning curve in order to understand the whole process. In addition to the following introduction, the material of the DDP foundation level contains a complete case study that shows the process presented in action. We strongly recommend examining the case study in detail and using it as a reference source for your own work.

5.3.1 Overview of the Process Structure

For the sake of simplicity, we assume that the elements of the digital solution can be realized by a single team. The development of larger digital solutions can be scaled from the foundations presented in this handbook. However, this requires additional management skills that go beyond the scope of this foundation level.

Managing the work at all three abstraction levels

Following the abstraction levels introduced in Section 1.2, we manage the work during the development and operations step at three levels:

- The *solution level* process focuses on the client and aims to achieve the objectives of the client through the digital solution. This process is about communication and coordination between the client and the product owner.
- The *system level* process focuses on the customer perspective and aims to realize value for the customer through the system. This process is about communication and coordination between the client and the product owner.
- The *element level* process focuses on the user perspective and aims to realize value for the user. This process is about communication and coordination between the building team and the product owner.

When talking about the process level, the difference between clients, customers, and users becomes important again (see Section 1.2.3), since each level focuses on a particular perspective.

Work product types for each level

At all three levels, the process uses the following work product types to manage the work:

- Work item: a coherent and documented unit of work
- Backlog: an ordered list of work items that represents the sequence in which work items shall be processed
- Kanban board: a visualization of the various stages of a process. Cards represent work items and columns represent the stage of the process. Work-in-progress limits can be introduced to limit the amount of parallel work in one column. Furthermore, rows can be used to assign a work item to a certain role.

Two additional work products are defined that apply to work items at all three levels:

- Definition of ready: definition of general criteria that must be met to consider a work item type ready for work on it to start
- Definition of done: definition of general criteria that must be met to consider a work item type completed

The details on managing the work at the three levels is presented in Section 5.3.2 together with an overview of the relevant work items. The details of the particular work items (content and rules for writing, including definition of done and definition of ready) are given in Section 5.3.2.3. In order to conclude the general mechanics of the process, Section 5.3.3.1 summarizes the relationships between the different work items and the design concepts.

Four phases of the development and operations step

In order to structure the process, we define four phases:

- Phase 1: Initial release planning and backlog preparation. In this phase, the product owner and the building team create an initial backlog that is sufficient for starting the development of the initial release.
- Phase 2: Development of the initial release. In this phase, the building team and the product owner work on the initial release.
- Phase 3: Further evolution during operation. In this phase, the initial release of the digital solution is in operation and the building team maintains it and works on the evolution by creating further releases.
- Phase 4: Retirement. In this phase, the solution is taken off the market.

Important events for structuring the work

For the development of the initial release (phase 2) and the further evolution (phase 3), the following events are used to structure the work:

- Daily: a short and timeboxed meeting on a daily basis to inspect the progress since the last daily
- Release planning: a timeboxed meeting on a regular basis in which the client, the product owner, and the team work on maintaining the backlogs and work items at the solution and system levels
- Iteration planning: a timeboxed meeting on a regular basis in which the product owner and team work on maintaining the backlog and the work items at the element level
- Iteration: a fixed period of time in which the building team works on the work items
- Retrospective: a regular meeting for self-improvement to inspect and adapt the process
- Solution review: a presentation and review of the results of an iteration to the client and other relevant stakeholders

The backlog preparation is a special phase. The details of this phase are presented in Section 5.3.4. Further details on the development of the initial release (phase 2) are discussed in Section 5.3.5. Section 5.3.6 discusses the further evolution (phase 3) and Section 5.3.7 concludes the process for the development and operations step with the retirement of the digital solution (phase 4).

5.3.2 Managing Work at the Three Levels

In the following, we present the management of the process and the necessary work products at the three abstraction levels of the building process. We also introduce the work items for each level and their individual purpose. Further details on elaborating the work items in relation to the design concepts are presented in Section 5.3.2.3. We start at the element level and then move upward.

5.3.2.1 Managing the Work at the Element Level

The element level process focuses on the user perspective and aims to realize value for the user; it is used for communication and coordination between the building team and the product owner.

5.3.2.1.1 Work Items at the Element Level for Documenting Work to be Done

Work at the element level has a wide spectrum that we capture with the following system work item types that realize the elements of the digital system:

- User story
- Technical work item
- Concept work item
- Prototype work item
- Evaluation work item
- Defect

User story: A description of a need from a user's perspective together with the expected benefit when this need is satisfied

The user story is the core work item since it represents implementation work that will lead to the realization of a user's need. The user story is a very popular concept (cf. [Cohn2004]) and is used in several approaches. The benefit of structuring implementation work by means of user values is that the outcome of a user story implemented is directly visible and can be experienced by users. This allows users and other stakeholders to give immediate feedback on the functionality realized and allows fast evaluation and feedback loops.

Technical work item: Work item for the elaboration/realization of a technical prerequisite for realizing a user story

However, user stories are not enough on their own since there is also implementation work that is not directly related to a user need (e.g., the implementation of technical interfaces). Including such work within a user story would lead to very large work items that are difficult to estimate and manage. To capture implementation work that is not directly related to a user need, we define the technical work item as a work item type. A technical work item can then become a prerequisite for starting the work on a user story.

Concept work item: Work item for the elaboration of conceptual details as a prerequisite for realizing a user story

The details on what is implemented in a user story or technical work item are captured by design concepts and/or realization concepts. These concepts are not necessarily elaborated in detail at the time of defining user stories or technical work items. In order to capture and manage the work to elaborate the concepts, we define concept work items. A concept work item can then become a prerequisite for starting the work on a user story or a technical work item.

Prototype work item: Work item for the creation of a prototype of an aspect of the digital solution

In addition to implementation work, the building team often also works on prototypes. In order to distinguish the implementation work on the solution from the work on prototypes, we define dedicated prototype work items. The concrete content of a prototype work item depends on the type of prototype. The outcome of a prototype work item is a particular prototype or an extension of an existing prototype.

Evaluation work item: Work item for the evaluation of a prototype or an aspect of the digital solution that has already been realized

Prototypes are often used to evaluate a certain aspect of the digital solution. To capture evaluation work, we define the evaluation work item. Evaluation work items can further be used to evaluate a particular aspect of the digital solution that has already been implemented.

Defect: Work item for a defect of the solution that has to be analyzed and fixed

The last work item type at the element level is the defect. Defects are a normal part of every building process and should be managed explicitly. The work item type defect captures the work for analyzing and understanding the cause or causes of a particular defect.

5.3.2.1.2 System Backlog and System Board to Visualize the Progress and the Work to Be Done

All work items defined on the basis of the work item types introduced above are managed within the system backlog. This is a list of work items (also called backlog items) with the order of the items defined by the product owner and building team. The order of the backlog items represents the priority and work order in which the items are processed by the building team and the product owner. The order of the elements is therefore an important management tool for controlling the work at element level.

To visualize and manage the progress at the element level, we use the system board as a Kanban board. In addition to the columns, the system board should have a row for each member of the building team and for the product owner. These rows visualize which person is working on which work item. Furthermore, a row with the title *Fast track* is useful to indicate that a particular work item is critical and should be processed with priority.

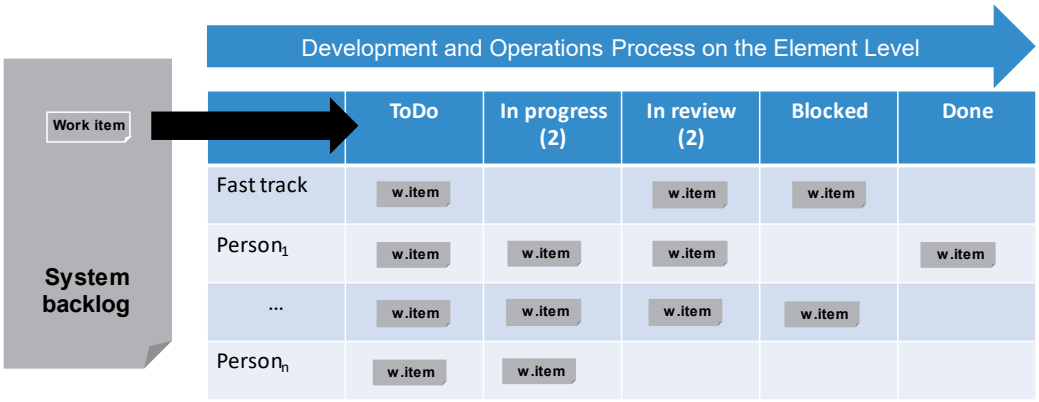


Figure 59 – System backlog and system board

Figure 59 shows an abstract system backlog together with a system board. The work with work items at the element level is as follows:

- *ToDo*: Every user story that meets the definition of ready and that is assigned to a particular person or is so important that it is assigned to the fast track (see above) is put on the system board.
- *In progress*: When a person is working on a user story, the user story is put into the *In progress* column.

- *In review*: When a user story meets the definition of done, it can be moved to the *In review* column, which means that another team member can review the result of the user story.
- *Blocked*: When the work on a user story cannot proceed due to external reasons, it is moved to the *Blocked* column.
- *Done*: When a user story has been successfully reviewed, it can be moved to the *Done* column. When an iteration is over, all user stories in this column can be removed.

5.3.2.2 Managing the Work at the System Level

The system level process focuses on the customer perspective with the aim of realizing value for the customer through the system. It is used for communication and coordination between the client and the product owner.

5.3.2.2.1 Epic as a Work Item at the System Level

At the system level, the work is managed by means of epics. The epic is a core element of this step of the building process and is defined as follows:

Epic: A work item that describes a characteristic of a digital system that provides value for stakeholders.

For the process described in this section, we focus primarily on the customer's value to be realized by the system as part of a release. An epic therefore refers to a particular goal that the digital system shall achieve as part of the value proposition of the digital solution.

5.3.2.2.2 Story Map as a Tool for Structuring and Visualizing Epics and User Stories

In order to define, understand, and manage the details of an epic, we use a technique called *story mapping* [Patt2014]. The main structure of a story map is a two-dimensional arrangement of user stories. A user story represents a particular user need and is also used at the element level (see Section 5.3.2.1). The horizontal dimension of the story map focuses on the backbone, meaning the narrative flow of the system (or the overall process provided by the system). Narrative flow here means the various steps or activities that a user can perform with the system in an end-to-end flow. The use cases and scenarios of the digital system are a good perspective on this flow. The vertical dimension provides details for each part of the narrative flow as well as a separation of items according to priority of the user stories from a function perspective (the order of the stories defines a priority, stories on top are more important than the ones below in the same column).

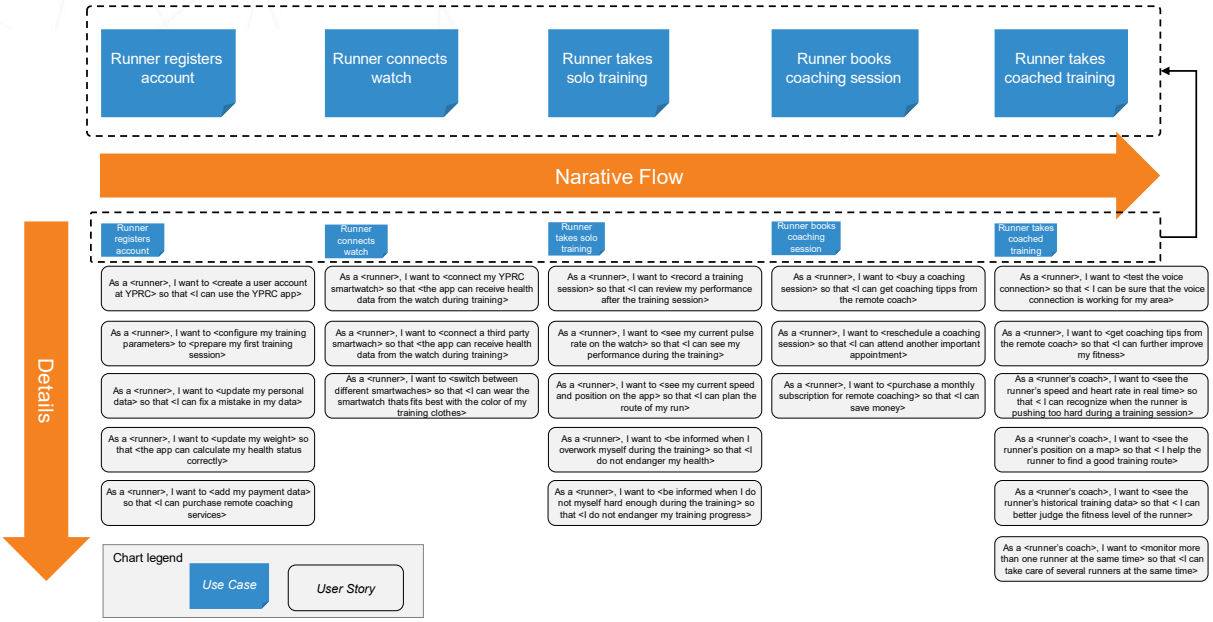


Figure 60 – Example of a story map from the YPRC case study

Figure 60 shows a miniature story map from the YPRC case study. The user stories are not relevant here and are not intended to be readable. The use cases are magnified for better readability of the narrative flow. The full-scale version can be found in the case study material.

Figure 61 shows an intermediate version of the YPRC story map (see Figure 60) where the middle is enlarged so that the user story titles are readable

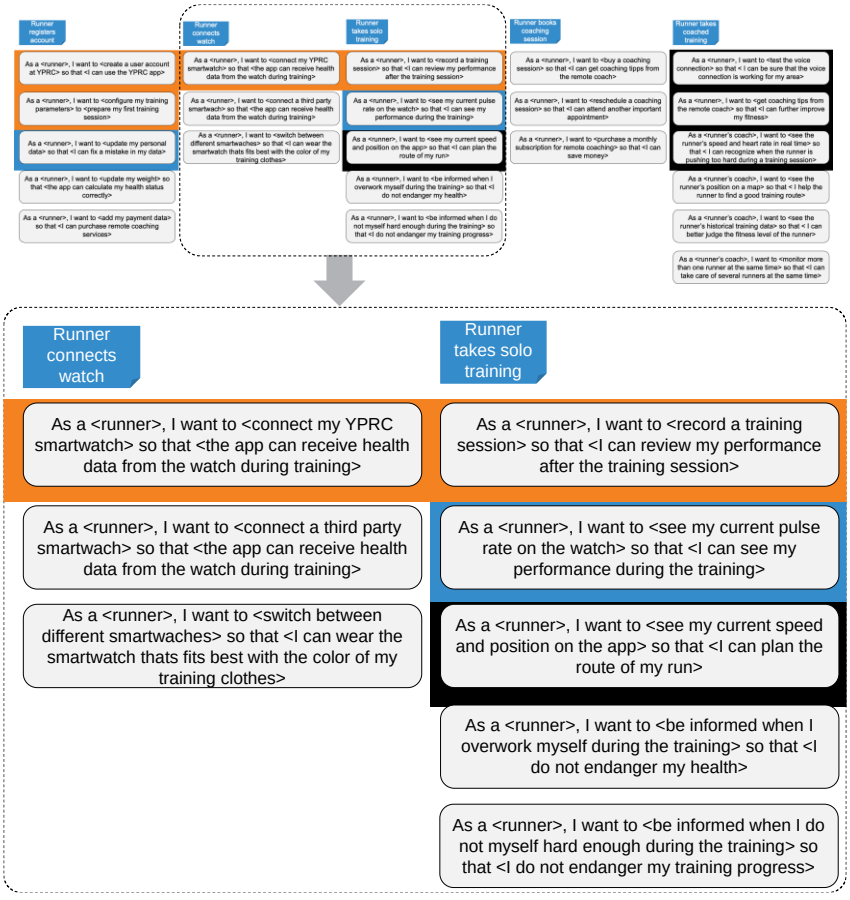


Figure 61 – Using a story map for prioritizing user stories

In this example, the user stories are separated into three epics:

- Orange epic: The stories with an orange background define a first and very simple functionality of YPRC that consists of four user stories that describe the registration process, the connection of the smartwatch, and the possibility of recording a training session and reviewing the training data afterwards.
- Blue epic: The stories with a blue background define further functionality of YPRC consisting of two additional user stories. The epic shall provide the ability for the runner to modify personal data in the app and to see their current pulse rate on the smartwatch.
- Black epic: The stories with a black background define further functionalities of YPRC consisting of four user stories. The epic shall provide the ability to view the current speed and position of the runner on a map and an initial version of the remote coaching feature.

The three epics are of course very small from a functional perspective but could be combined into one release. This example is not about a realistic amount of functionality—it is about understanding the value of the story map for defining epics. The exemplary story map is rather small. This does not mean that story maps are not useful for larger scale solutions. It is possible to create really large story maps with several use cases. In such a situation, the story map can easily cover large walls of a room.

The story map can be used for the initial definition of user stories (see Section 5.3.4) but it is also a technique for documenting and managing development during the whole development process. We therefore recommend maintaining the story map during the whole development process. With this perspective, the story map is an important tool that helps you to focus on the solution perspective and to complement the detailed view of system and element design concepts.

5.3.2.2.3 Epic Board for Visualizing the Progress and the Work to Be Done

Figure 62 shows the structure of an epic board. The story map serves as a tool for maintaining and defining the epics during the building process and the epic board is used to manage the work during the development and operations process.

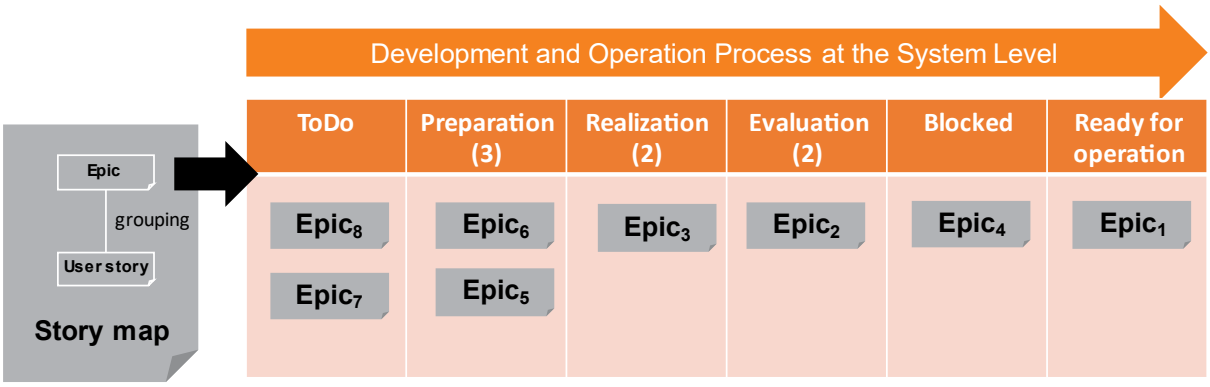


Figure 62 – Story map and epic board

In the following, we discuss the columns of the epic board:

- *ToDo*: Every epic that meets the definition of ready (see above) is put on the epic board in the *ToDo* column. The order of the epics in this column can be used to show the priority of the epics and their related work items.

- *Preparation*: As soon as the building team starts to work on conceptual work items or technical work items that belong to an epic, the epic is moved to the *Preparation* column. This is an indication for the product owner and the client that the details of an epic have been elaborated and that the preparation of the realization is underway.
- *Realization*: When the building team starts to work on user stories related to a particular epic, the epic is moved to the *Realization* column. It is not necessary that all conceptual work items have been done for this particular epic.
- *Evaluation*: When all user stories, technical work items, and conceptual work items have been done, the epic is moved to the *Evaluation* column and the final evaluation of a particular epic should start. It is important to recognize that this criterion for putting an epic into the column is different from the other columns. For this stage of the building process of an epic, dedicated evaluation work items should be defined that evaluate whether the digital solution meets the acceptance criteria defined in the epic (see above).
- *Ready for operation*: As soon as the definition of done for epics is met (i.e., all evaluation work items have been done and the results of the evaluation show that the acceptance criteria are met), the epic can be moved to the *Ready for operation* column. As soon as the epic is in operation, the epic is removed from the board (see Section 5.3.5). Removing the epic from the board is mainly done to keep the board as lean as possible.

Besides the columns described above, the epic board has the *Blocked* column. An epic is put into this column if a work item related to this epic at the element or solution level is blocked.

This way of working with epics makes the epic the central management tool that integrates the various other work items. Maintaining these various relationships between epics and work items is part of the activity area management and not the activity area design. However, from a Digital Design perspective, it is important to understand the relationship to and the impact of defining epics on design concepts as highlighted in Figure 67.

5.3.2.3 Managing the Work at the Solution Level

The solution level process focuses on the client and aims to achieve the objectives of the client through the digital solution; it is used for communication and coordination between the client and the product owner.

Managing the work at the solution level consists of two parts: managing the release plan and managing work at the solution level.

5.3.2.3.1 Release Plan as a High-Level Management Tool

The release plan is a high-level management tool that defines the releases planned for the digital solution (the releases already created can also be maintained in the release list). As defined above, a release is simultaneously a particular instance of the digital solution and a period of time in which the particular instance of the digital solution is realized. Work on the release plan takes place in the release planning (see Section 5.3.1). Details on how to define a particular release in terms of functionality (which epics are expected to be part of the release) and in terms of time and budget (what is the expected budget and timeframe in which the release shall be available) are discussed in Section 5.3.4.

5.3.2.3.2 Work Items at the Solution Level

The concrete work at the solution level depends on the particular type of digital solution and can vary significantly. We therefore define only abstract solution work items. A solution work item is a work item that provides a resource or any other means necessary for the development or the operation of the digital solution. Examples of such solution work items are:

- Organizing a customer or user test for the digital solution
- Procuring personnel or material that is necessary to build the digital solution
- Obtaining technical resources (e.g., data centers) that are necessary for operating a digital solution

5.3.2.3.3 Solution Backlog and Solution Board for Visualizing and Managing the Work to Be Done

To manage the work, a solution backlog and a solution board are used. The solution backlog is a list of work items that have to be processed to realize a release of the digital solution. The product owner defines the order of the work items in the list.

The solution board is a Kanban board for managing the work items at the solution level and has the following columns:

- *ToDo*: work items that meet the definition of ready (see below)
- *In progress*: work items that are in progress
- *In review*: work items that are under review
- *Blocked*: work items that are blocked
- *Done*: work items that meet the criteria from the definition of done (work items are removed at the end of a release)

5.3.2.4 Big Picture on Managing Work at the Three Levels

In this section, we have introduced several work products for managing the work at the three levels. Figure 63 shows the work products introduced at the three levels.

One aspect that may be confusing at first reading is the fact that the work products are not defined symmetrically. For example, we defined a solution and a system backlog but no element backlog. We also did not define an element board. The reason for this is that there is no need to separate the work at the system and the element levels since the system consists of nothing but elements. Working on a particular element therefore means working on the system. Nevertheless, it is necessary to distinguish between managing the detailed work on the elements (system backlog and system board) and the long-term perspective on the development of the system as a whole. For the long-term perspective, we have introduced the story map and the epic board at the system level.

Story Map as the Backbone for Managing the Development and Operation

The story map is the central tool for managing the development and operations of a digital solution—it relates customer values (defined by epics) to user values (defined by user stories). With the story map, the product owner gets on overview of what the digital solution aims to achieve and how the digital system will contribute to achieving the customer value.

The work on the digital solution and the digital system is prioritized in relation to the epics. From the solution level perspective, a release of the digital solution is defined by a number of

epics. From the system-level and element-level perspectives, the work on user stories and related work items is prioritized according to the priority of the epics. The prioritization process should by no means be understood as a top-down process. The priorities at all three levels may be changed for various reasons. When deciding on priorities, the product owner should always consider the solution backlog, the story map, and the system backlog together in order to find the optimal priority.

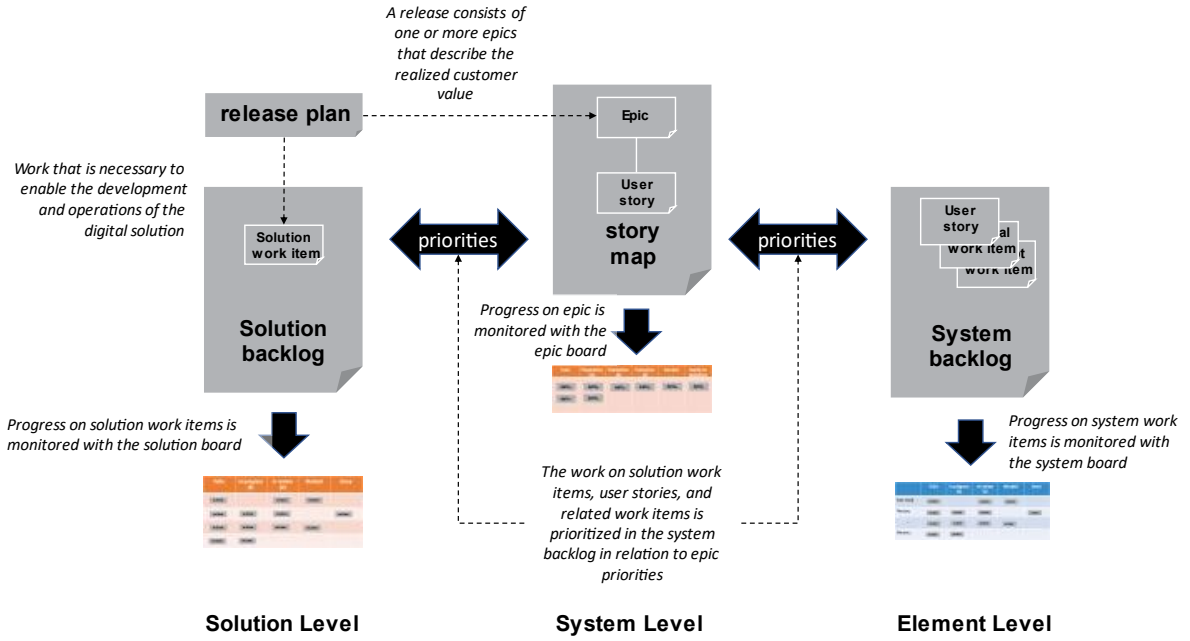


Figure 63 – Relationships between work products at the three levels

One final remark on the work products shown in Figure 63: the work products defined in this section are defined with the assumption that a single team can work on the digital solution together with one product owner. When the size and complexity of a digital solution requires the coordination of several teams and several product owners, we recommend that you do not use the process presented here since scaling a building process with several teams requires additional management structures that go beyond a foundation level building process.

5.3.3 Guidelines for Defining Work Items in the Building Process

In the following, we provide detailed guidelines for writing each work item type, including a detailed template, writing rules for the work item, and the definition of ready and done. These guidelines apply for the initial creation of work items and for the creation of work items during the whole development.

Besides the work item types relevant from a Digital Design perspective, this section also provides brief guidelines on further work item types. They go beyond the scope of the DDP foundation level but are important for understanding the complete building process.

Before we go into the details of the different templates for the work items, in Section 5.3.3.1 we give an overview of the relationships between work item types and design concepts. We then introduce a general template for work items in Section 5.3.3.2 and then continue with the details of the individual work item types.

5.3.3.1 Overview of Relationships between Work Products and Design Concepts

In the previous section, we defined work products at the three levels. Before we go into the detailed guidelines, let us look at the Digital Design perspective on these work products and discuss the overall relationships between the work products and the design concepts on a general level. Understanding these relationships is important to understanding how Digital Design is integrated into the management of the realization work.

Figure 64 shows two viewpoints: the viewpoint on design concepts and the viewpoint on work products. Furthermore, the figure shows the relationships between the work products and the design concepts.

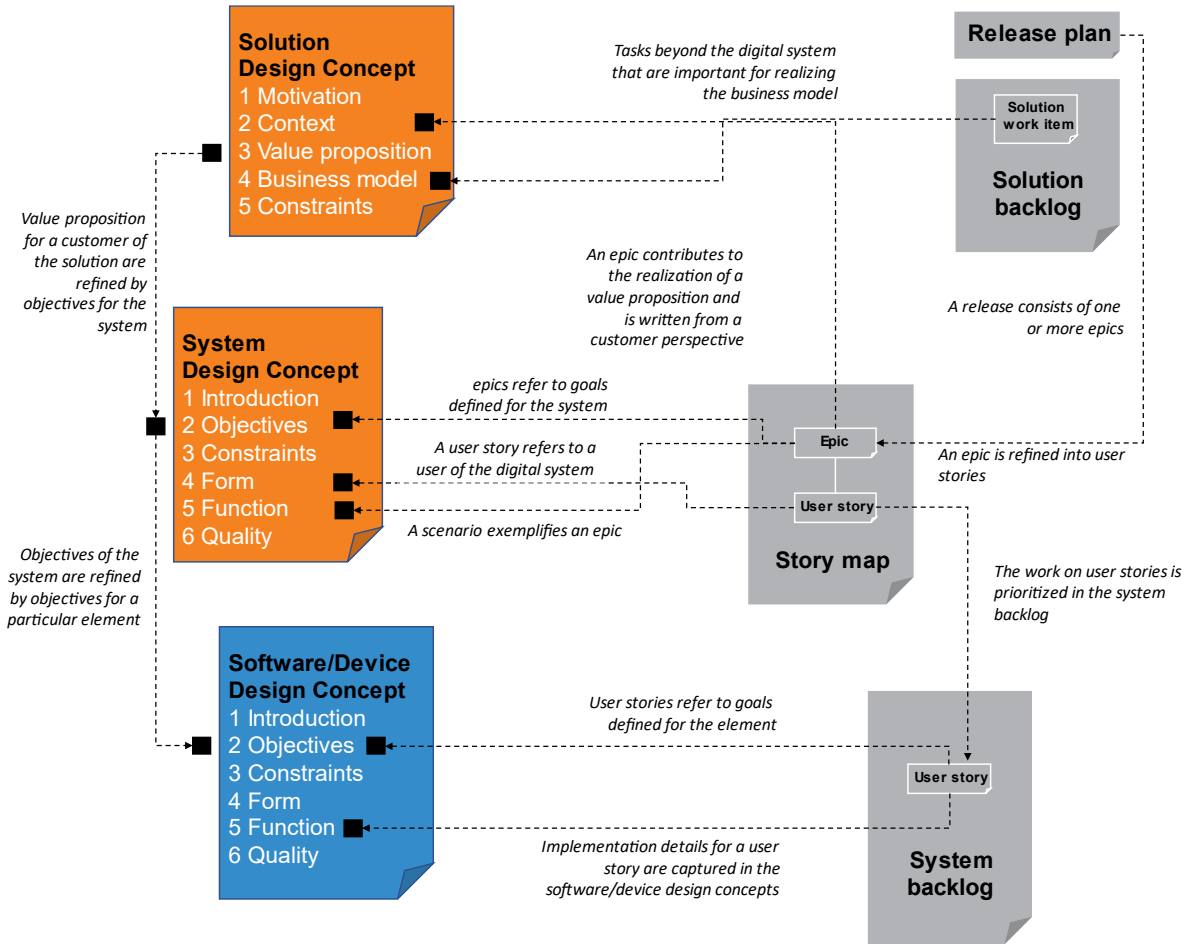


Figure 64 – Relationships between work item management and design concepts

We start with the perspective on the work products. A release defines the long-term perspective on the building process. A release consists of one or more epics and an epic is further refined into user stories. This means that a particular release creates the customer value that is defined by the epics and the corresponding user stories. A similar hierarchy is defined by the design concepts. The value proposition defined in the solution design concept is refined by goals defined in the system design concept, which are further refined by goals at the element level.

These relationships imply that working on releases, epics, and user stories means working on value proposition, system goals, and element goals. This means, for example, that refining user stories means refining goals of an element and refining the use case of an element means modifying one or more user stories. A DDP should keep these dependencies in mind during

the realization of a digital solution otherwise they will lose connection with the realization activities. These relationships are explained in Section 5.3.3 and support maintaining consistency between work products and design concepts. Understanding these relationships is the prerequisite for keeping both perspectives consistent with each other.

We want to emphasize that maintaining consistency between design concepts and work products is not an end in itself—it is a tool for integrating the design perspective and the management perspective of the building process. Furthermore, we want to emphasize that it is impossible to achieve complete consistency because design concepts and process artifacts are continuously revised and reworked. Nevertheless, the effort spent for keeping both perspectives as consistent as possible is not wasted time because it is the proper tool for keeping the management perspective on the building process consistent with the design perspective.

In the YPRC case study, we introduce further tool support that supports maintaining consistency between both perspectives.

5.3.3.2 General Template for Work Items

The work items are created for the backlogs and to manage the work at the levels of the building process. In general, a work item template consists of the following sections:

- Identifier and title
- Section with detailed information of the work to be done
  - Task description
  - Prerequisites
  - Acceptance criteria: definition of individual criteria that must be met to consider the particular work item completed
  - Relevant elements
  - Epic reference: reference to the epic to which the work item belongs
- Sections with management information
  - Assignee
  - Creator
  - Release
  - Estimated effort
  - Effort spent
  - Remaining effort

Similar to the building blocks of design concepts, we recommend using unique identifiers for each work item to support traceability between work items. Defining an adequate title that gives a crisp description of the work will further support the management of work items.

Detailed information of the work to be done

The detailed information for a work item consists of the task description, which depends on the particular work item type, and four general sections.

The *prerequisites* are used to document other tasks that have to be done before the current task can be processed. The *acceptance criteria* are used to define individual criteria that must be met to consider the work item done. Acceptance criteria complement the general criteria provided by the definition of done and are a useful tool for defining clear criteria that have to be met by the work item. The *relevant elements* provide reference to the elements of the digital

solution that are assumed to be affected by the task. This reference is useful for keeping track of the work items that belong to a particular element of the digital solution. The *epic reference* is used to relate a work item to a particular epic. This reference enables an overview of all work items that contribute to a particular epic.

Management information

The management information is important for managing the work during the building process. It is useful for distinguishing between the *assignee* and the *creator* to keep track of the person who created the work item (e.g., for asking questions). The *release* reference is important for documenting the release that a task contributes to. This information can, for example, be used to control the amount of work for a particular release. The *estimated effort,* the *effort spent*, and the *remaining effort* are useful for keeping track of the amount of work (see Section 5.3.4.3).

The list of sections for a work item described above is a rather minimal list and experts in management will be aware of several other aspects. Therefore, experts should keep in mind that the building process described here is for beginners.

Reminder: Work items are temporary

Since the work item is a temporary work product, the detailed information of the work item should contain only information that is of particular importance for the task at hand. All information on the digital solution that has to remain should be documented in the concepts. References between work items and concepts are used to achieve this separation and to provide further details.

In the following, we provide detailed guidelines for the task description of each work item type.

5.3.3.3 Solution Work Items

Solution work items are a tool for capturing all work at the solution level. The scope of work at the solution level is very broad (see 5.3.2.3), and we therefore define a very general template for a solution work item. The task description of the solution task should provide sufficient detail for the assignee to work on the task. Acceptance criteria, definition of done, and definition of ready have to be defined depending on the particular digital solution.

Obtain test environment for payment provider “pay friend”	
Prerequisites	• None
Task description	• Negotiate a test environment for the payment provider “pay friend” to evaluate the coaching purchase use case UC-10 in the development environment
Acceptance criteria	• The IP address for the test environment is available • Test accounts for three users are available • Administrative account for checking the payment is avaiable
Relevant elements	• ESys-2 Payment Provider
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 65 – Example of a solution work item from the YPRC case study

Although this is only a very abstract description of a solution work item, solution tasks can be very important for the building process for a digital solution. Figure 65 shows an example of a solution task from the YPRC case study that deals with a particular payment provider that shall be included in YPRC.

The example is about organizing a test environment for the payment provider. Such a work item is typical for the solution level during the building process since it deals with resources that are necessary to work on the digital solution.

5.3.3.4 Epics in Relation to the Solution and System Design Concept

The epic serves two purposes in our building process. It is:

- A short and simple description of a value that is offered by the digital system to a customer of the digital solution
- A management tool for organizing and prioritizing the development work at the system level (see Section 5.3.2.2)

Beginners in Digital Design often find the combination of these two aspects (mixture of content and management) difficult to understand. The strength of this combination is that at the system level, the building process is managed according to real customer value. With this approach, it is possible to discuss priorities of the development work with the client based on customer value. This means that the client and the product owner can decide which customer value has more importance and should be realized first by the team.

In our building process, epics are defined by grouping a number of user stories to provide a meaningful customer value (see Section 5.3.2.2). This means that the details of an epic are defined by means of user stories.

Advice for defining the title

In order to identify an epic, we need a proper title that reflects the customer value. We recommend using the following template for the title of an epic:

As a <type of customer>, I want <some system function> so that <some system goal>.

The part <type of customer> refers to the customer segments that we have defined in the solution design concept. The part <some system function> refers to a function that the digital system will perform to achieve a goal for the customer that is described by <some system goal>. Figure 66 shows an exemplary epic of the YPRC case study.

As an <athlete>, I want to <record my training data> so that <I can analyze my performance after the training>	
Prerequisites	• None
System objective	• G-4 Analyze the performance after the training session
Scenario	• Scen-3 The runner does a training and reviews the data after the training
Acceptance criteria	The athlete can review the following data after the training session • Minimum, average, and maximum pulse rate • Minimum, average, and maximum speed • Running distance • Highest and lowest altitude • The data is stored in the app but not in the portal
Relevant elements	• DDev-1 Runner's watch • DDSys-1 Runner's app

Figure 66 – Example of an epic from the YPRC case study

In order to maintain consistency between the epics and the design concepts, we recommend the following guidelines with respect to the epic title:

- The <type of customer> part should refer to a customer segment from the solution design concept.
- The <some system goal> part of the epic should be documented as a goal in the system design concept.
- The <some system function> part of the epic should be documented as a scenario in the system design concept.

The references visualized in Figure 67 allow the task description to be documented in a compact way since the important details of the epic are provided in the design concepts.

Scenario from system design concepts as a source for acceptance criteria

Furthermore, the scenario is an important source for defining the *acceptance criteria* of the epic. The following guidelines are useful for defining acceptance criteria for an epic:

- Make use of quality requirements or constraints at the system level.
- Define only those acceptance criteria whose fulfillment contributes to the goal referred to in the system design concept.
- Name observable behavior or functions that are described in the scenario.
- Use the same wording in the scenario and the acceptance criteria.

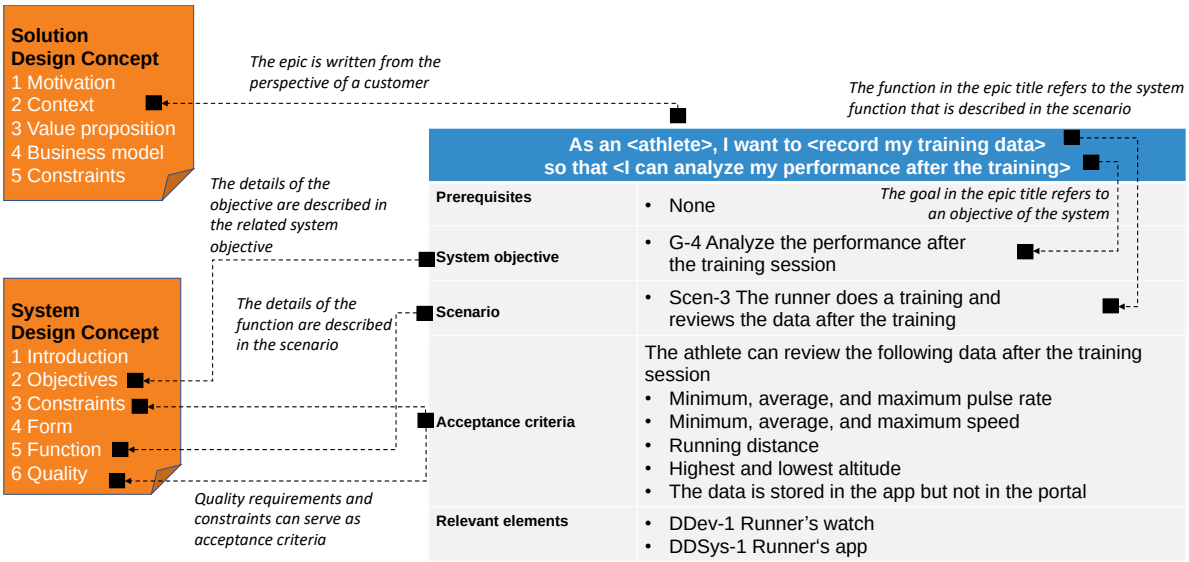


Figure 67 – Relationship between epic and design concepts

Advice for the definition of ready

For the definition of ready of an epic, we recommend the following guidelines:

- The epic description is consistent with the design concepts.
- The epic has been refined by user stories and the user stories meet the definition of ready.
- The client agrees with the modification of the design concepts (if changes were necessary).

Advice for the definition of done

For the definition of done of an epic, we recommend the following guidelines:

- The fulfillment of the acceptance criteria has been demonstrated (e.g., by dedicated evaluation work items, see below).
- All work items related to the epic meet the definition of done.
- The digital solution realized is consistent with the design concepts.

Considerations for daily work

Documenting and maintaining the relationships between epics and design concepts has an important benefit. It allows you to reconsider the details and the structure of the goals and scenarios in the system design concept. Often, the goals and functions defined in the epic creation do not fit 1:1 with the goals and scenarios in the system design concept. They are often cut differently or described from a different perspective. This is not an error or flaw in the conceptual step in which the solution and system design concepts have been initially created, it simply reflects the growing understanding of digital solutions. We therefore recommend that the goals and scenarios in the system design concept are adapted to the new state of knowledge and the structure of the epics. In this way, the epics and the system design concept are consistent, and the new state of knowledge based on the epics is also documented in the design concepts.

For beginners in Digital Design, this work seems to create a lot of overhead and effort. In particular, for people who are working with the epics and the design concepts on a daily basis, the update work appears to be a waste of time, since they know the details and they know what to do. Here, you should keep in mind that design concepts are the long-term memory of the building process (see Section 2.2.1) so they should be kept up to date. Further practical advice from our experience is that proper software support is the key to minimizing the effort for maintenance of the design concept. In the YPRC case study, we provide details on possible approaches for tool support in situation.

5.3.3.5 User Stories in Relation to Element Design Concepts

The user story serves two purposes in our building process. It is:

- A short and simple description of a function that a user wants to have; and
- A management tool for organizing and prioritizing the development work at the element level (see Section 5.3.2.1)

Just like for epics, beginners in Digital Design find the combination of these two aspects (mixture of content and management) in a user story difficult to understand. The strength of this combination is that at the element level, the building process is managed according to real user needs. This way of managing the development work with user stories means that the benefit/value for the user is always in sight during the whole building process.

Advice for defining the title of user stories

We recommend using the following template for a user story title:

As a <type of user>, I want <some function> so that <some goal>.

As with epics, the parts of a user story should reference design concepts (see Figure 68):

- The part <type of user> should refer to a user type defined in the system design concept.
- The part <some function> should refer to a particular function of an element that is related to achieving the goal.
- The part <some goal> should refer to a user goal that is documented for the corresponding function in the element design concept.

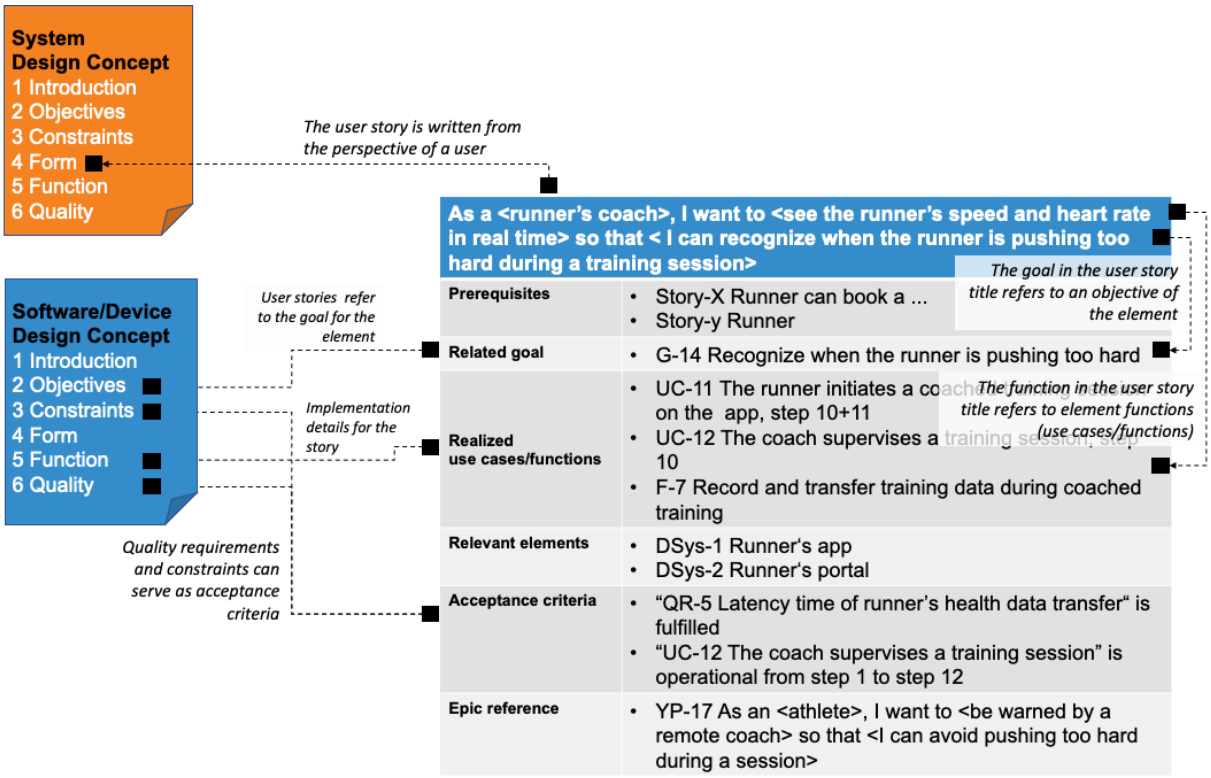


Figure 68 – Relationship between user story and design concepts

YPRC example. The following user story is an example from the YPRC case study (see Figure 69 for the full story):

As a <runner's coach>, I want to <see the runner's speed and heart rate in real time> so that <I can recognize when the runner is pushing too hard during a training session>

As a <runner's coach>, I want to <see the runner's speed and heart rate in real time> so that < I can recognize when the runner is pushing too hard during a training session>	
Prerequisites	• Story-X Runner can book a ... • Story-y Runner
Related goal	• G-14 Recognize when the runner is pushing too hard
Realized use cases/functions	• UC-11 The runner initiates a coached training session on the app, step 10+11 • UC-12 The coach supervises a training session, step 10 • F-7 Record and transfer training data during coached training
Relevant elements	• DSys-1 Runner's app • DSys-2 Runner's portal
Acceptance criteria	• "QR-5 Latency time of runner's health data transfer" is fulfilled • "UC-12 The coach supervises a training session" is operational from step 1 to step 12
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Advice for the task description

With the title, an initial idea of part of the digital solution emerges. However, for the actual development work, further details are needed, and these are given in a task description. We recommend providing a very brief explanation in the user story that refers to the element design concepts for further details. The description should provide the following information on the use case (steps)/functions realized, the references to the use cases/functions, and relvant use case steps, or function (parts) that are realized by this user story.

The main reference points for the task description of user stories are goals, use cases, and functions in the element design concept. The reason for this is that the realization of a user story has to provide added value to the user defined by the goal. The added value in terms of the element design concepts is the perceivable or underlying function that is described by means of use cases and functions. It is important to keep in mind that the relationship between user stories and the design concepts is bi-directional. During the elaboration of a user story and the discussion with the building teams, new insights will occur that must be reflected in the corresponding design concepts and user stories. This leads to regular updates of the element design concepts.

The relationship to other aspects of form defined in the element design concepts (e.g., user interfaces, software interfaces, or entities) is defined by the references from use cases/functions to these elements (constructive relationships, see Section 2.2.6). These references do not have to be introduced in user stories because the whole building team works

with the element design concepts and can collect this information from the element design concepts.

This does not mean that the actual implementation of form aspects of a digital solution (e.g., software interfaces) is not part of the backlog. For this purpose, we introduce the technical work item (see Section 5.3.3.9).

Advice for the acceptance criteria

Acceptance criteria in user stories describe conditions that must be demonstrated by the building team when the user story has been implemented. They also serve as an abstract list of test cases that the product owner can use to determine whether the user story has been implemented properly. Good candidates for acceptance criteria are quality requirements and constraints from the system and element design concepts.

Other good sources for acceptance criteria are references to use cases that must work when the user story has been implemented. The example in Figure 69 mentions the use case UC-12 with steps 1 to 12 as part of the acceptance criteria. This means that after the implementation of this user story (which focuses on step 10), the coaching process from step 1 to 12 must work. This is possible since the other steps have already been implemented by preceding user stories.

The epic reference of the user story is used to indicate the epic that the realization of the user story contributes to. Through the epic reference, an indirect reference to the system design concept is created since the epic refers to a goal from the system design concept. For reasons of consistency, the system goal should also be referenced by the element goal that the user story refers to.

Advice for the definition of ready

A good practice for the *definition of ready* for user stories is the INVEST acronym [Wake2003]:

- I – Independent: user stories should be as independent from each other as possible. This makes parallel implementation in one or more teams easier.
- N – Negotiable: a user story does not represent a fixed contract—it leaves room for discussion of the details.
- V – Valuable: a user story brings real value to the user/customer.
- E – Estimable: the description is at a level of detail that allows for proper estimation of the realization effort.
- S – Small: a user story is small enough to allow implementation in one iteration.
- T – Testable: the level of detail is sufficient to allow for testing of the implementation.

Meeting these criteria is a real challenge for beginners in Digital Design. Finding the right level of independence, size, and detail requires cooperation with the building team and is an iterative process that goes back and forth between the product owner and the building team.

Advice for the definition of done

The *definition of done* for a user story should include criteria from a design, construction, and realization perspective. From a design perspective, criteria for done should include some quality assurance measures that demonstrate that the acceptance criteria are fulfilled (e.g., test cases or explicit references to use case steps, see Figure 69). Criteria for done from a construction and realization perspective are defined by experts from these fields. They can

include the implementation of unit tests or the documentation of technical details (e.g., architecture documentation).

5.3.3.6 Concept Work Items for Working on Design Concepts

Concept work items are used to plan for conceptual work that is necessary for future iterations and for achieving the definition of ready for user stories and other work items. Typical conceptual tasks are:

- Elaboration of new user interface designs
- Elaboration of new entities
- Elaboration of use cases
- Refinement of existing design concepts

Concept work items therefore refer to the building block or building blocks of the design concept that has to be elaborated and to other building blocks that prove useful input for the concept work item (see Figure 70).

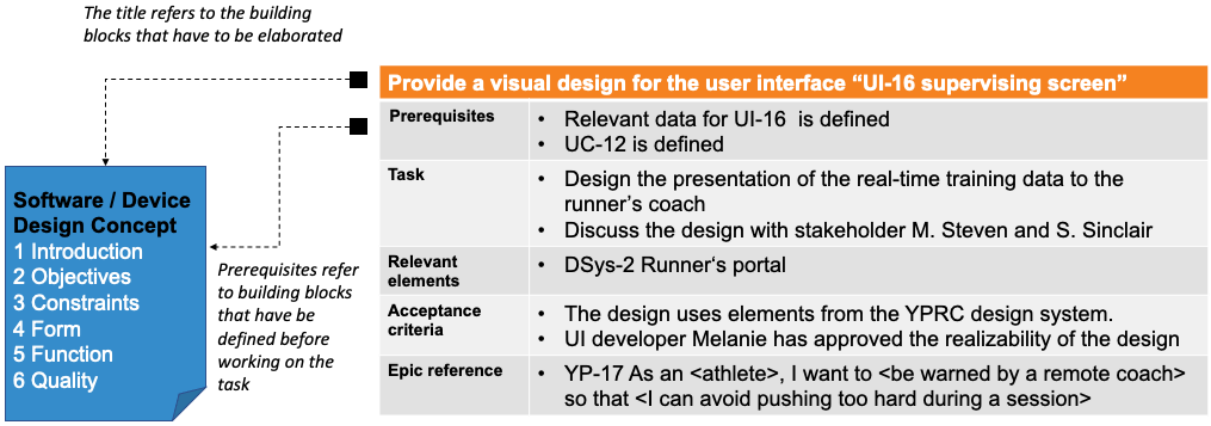


Figure 70 – Relationship between concept work items and design concepts

Advice for defining the title and the task description

The title of a concept work item should give a brief summary of the task—if possible, the concept work item should already refer to the particular building block(s) that should be elaborated. The example shown in Figure 71 refers to the user interface UI-16 that should be elaborated. The example is a very fine-grained concept work item (the design of a particular user interface screen).

Provide a visual design for the user interface "UI-16 supervising screen"	
Prerequisites	• Relevant data for UI-16 is defined • UC-12 is defined
Task	• Design the presentation of the real-time training data to the runner's coach • Discuss the design with stakeholder M. Steven and S. Sinclair
Relevant elements	• DSys-2 Runner's portal
Acceptance criteria	• The design uses elements from the YPRC design system. • UI developer Melanie has approved the realizability of the design
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 71 – Example of a concept work item from the YPRC case study

Concept work items can of course be defined with a much broader scope (e.g., elaborate a whole process). However, keep in mind that our building process assumes that the work on the concept work items takes place within the frame of the iteration together with the whole building team. We therefore recommend defining concept work items with this in mind. Beginners should start by defining rather fine-grained tasks in order to get a feeling of how much work can be done in one concept work item.

Advice for the acceptance criteria

Acceptance criteria for a concept work item always refer to the design concepts created. Good candidates for acceptance criteria are reviews by team members (e.g., from a technical perspective) or by relevant stakeholders.

Advice for the definition of ready

The definition of ready for a concept work item is mainly determined by the prerequisites defined for the task. Keep in mind that it is not necessary to fulfill the definition of ready immediately after the concept work item has been written. The definition of ready must be fulfilled when the work on the task is about to start.

Advice for the definition of done

The definition of done for a concept work item can include the following aspects:

- Product owner has reviewed and accepted the concept elaborated.
- All traceability relationships between the new concept part and the existing parts have been checked.
- The impact of the new content on user stories already defined has been checked.
- The creation of new user stories that reflect the content has been checked.
- Checking for necessary updates of the system design concept, of the solution design concept, or even of the Digital Design brief (if necessary).

With these aspects, the definition of done for concept work items can be used to maintain high-quality design concepts and to define a common standard for working with these design concepts.

5.3.3.7 Prototype Work Item

Prototype work items are used to manage prototyping work as part of the iteration. This way of working with prototype work items follows the same idea as introduced for concept work items.

Advice for defining the title and the task description

If prototyping is necessary during the development of a digital solution, work on the prototypes must be aligned with the other tasks. Keep in mind that prototyping is a powerful tool (see Section 2.3) that saves resources since it may prevent developing the solution in the wrong direction. By using prototype work items, the product owner and the team understand the impact on the progress of the development work since working on a prototype will consume the resources of the building team but can save resources in the long run.

Create an interactive high-fidelity mock-up of a graph-based visualization of the runner's data that shows the current data and the history during the training session	
Prerequisites	• Three data sets of running sessions are available (minimum 5 minutes per training data set)
Task	• Agree on a technical framework for graph visualization together with the building team • Design a graph-based visualization of the training data flow • Realize a demonstration that uses the three data sets in real-time
Relevant elements	• DSys-2 Runner‘s portal
Acceptance criteria	• The three data sets are visualized in a demonstration in real-time
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 72 – Example of a prototype work item from the YPRC case study

Figure 72 shows an exemplary prototype work item from the YPRC case study. The example presents a user interface prototype that concretizes an alternative visualization idea for the visualization of the runner's data for the coach in order to understand whether the graph-based visualization approach is more valuable for the coach than the textual one. This prototype will be created by a design expert (here, probably interaction design) from the team. However, prototype work items depend on the particular objective of the prototype (see Section 2.3.2). It can also include, for example, functional prototypes that focus on the feasibility of a function of a digital solution. Such prototypes will of course be created by construction/realization experts from the building team.

Advice for the acceptance criteria

Acceptance criteria for a prototype work item should refer to the quality requirements that the prototype should fulfil. Keep in mind that the quality requirements for a prototype can differ depending on the objective of the prototype. The prototype objectives and categories introduced in Section 2.3 provide good guidelines for defining appropriate quality requirements for a prototype work item.

In addition to the quality requirements, the results of prototyping (according to the tasks defined in the prototype work item) should be reviewed by team members (e.g., from a technical perspective) or by relevant stakeholders.

Advice for the definition of ready and the definition of done

The definition of ready and the definition of done depend on the concrete prototype work item. Therefore, it is difficult to define general rules for both. As a rule of thumb, the prerequisites should be formulated in such a way that they define the readiness for working on the prototype work item. The definition of done should include the fulfillment of the acceptance criteria.

5.3.3.8 Evaluation Work Items

Evaluation work items are used to manage the evaluation of a certain aspect of the digital solution (e.g., a usability test of parts of the solution) or of a prototype as part of the iteration work. Evaluation work items focus on dedicated questions and do not replace systematic quality assurance measures of the whole digital solution (e.g., system tests or user acceptance

tests). These tests have to be planned independently. The procedures should be documented in the corresponding evaluation concept.

Evaluate the graph-based visualization of the runner's data with stakeholder X,Y, and Z to decide on the implementation	
Prerequisites	• Graph-based visualization prototype is available
Task	• Plan a short usability test for the prototype • Visit X, Y, and Z and present the prototype, perform the usability test • Invite X,Y, and Z to a joint workshop with PO to discuss the results from the usability test and to decide on the implementation of the prototype
Relevant elements	• DSys-2 Runner‘s portal
Acceptance criteria	• The workshop provides a clear recommendation for or against implementing the graph-based visualization
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 73 – Example of an evaluation work item from the YPRC case study

Figure 73 shows an exemplary evaluation work item from the YPRC case study. The example presents the subsequent evaluation work item that follows the creation of the user interface prototype from Figure 72.

Advice for the task description

In terms of the element evaluation concept, the details of an evaluation work item should refer to the element evaluation concept. We recommend documenting important evaluation work items (e.g., usability tests or detailed user acceptance tests) not only by means of backlog items but also as part of the solution, system, or element evaluation concept. For example: a system test case that includes different elements of the system belongs in the system evaluation test, and a detailed usability test of an element belongs to the particular element evaluation concept.

Advice for the acceptance criteria

Acceptance criteria for evaluation work items always have a different focus compared to other work items because the evaluation work items focus on insights on an existing artifact. We therefore recommend defining acceptance criteria for evaluation work items based on the expected outcome quality of the evaluation task. For example, Figure 73 demands a clear recommendation for or against the implementation of the graph-based visualization.

Advice for the definition of ready and the definition of done

The definition of ready and the definition of done depend on the concrete evaluation work item. Therefore, it is difficult to define general rules for both. As a rule of thumb, the prerequisites should be formulated in such a way that they define the readiness for performing the evaluation (e.g., required test data available). The definition of done should include the fulfillment of the acceptance criteria.

5.3.3.9 Technical Work Items

Technical work items belong in the domain of construction and realization. We therefore introduce them only briefly here.

Examples of technical work items are:

- Elaboration of a realization concept (or part of it)
- Implementation of technical interfaces
- Implementation of data structures
- Revision of software elements already implemented

Technical work items are used to manage the development work that is necessary as a prerequisite for implementing a user story. Although technical work items do not belong to the core domain of Digital Design, technical work items can have relationships to design concepts (see Figure 74).

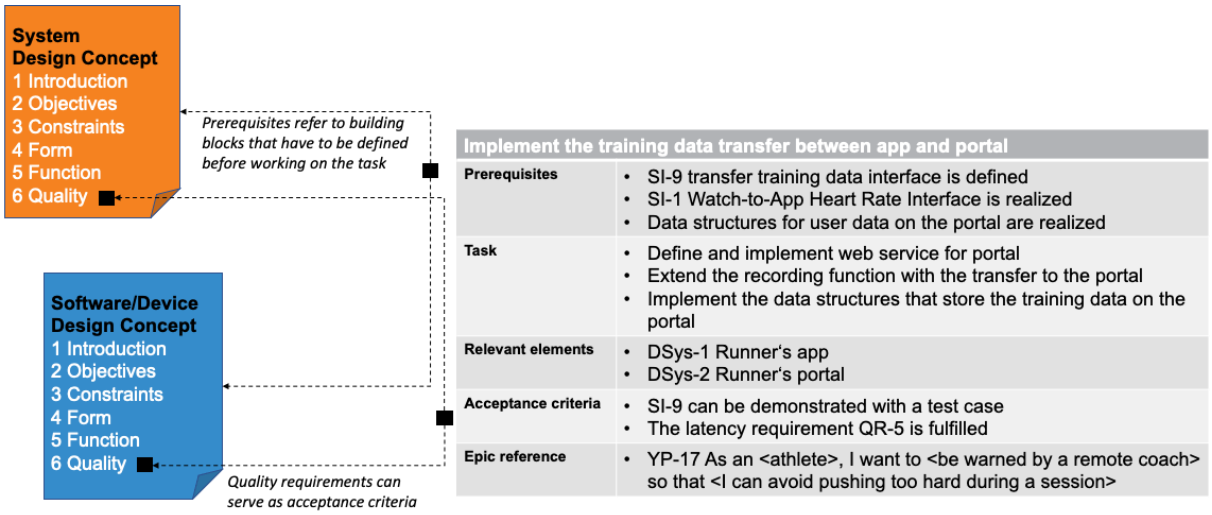


Figure 74 – Relationship between technical work items and design concepts

For example, acceptance criteria of a technical work item can refer to quality requirements at the system or element level or to test cases that demonstrate the result at a technical level. The prerequisites of a technical work item can refer to building blocks of design concepts as input for the technical work item.

Implement the training data transfer between app and portal	
Prerequisites	• SI-9 transfer training data interface is defined • SI-1 Watch-to-App Heart Rate Interface is realized • Data structures for user data on the portal are realized
Task	• Define and implement web service for portal • Extend the recording function with the transfer to the portal • Implement the data structures that store the training data on the portal
Relevant elements	• DSys-1 Runner's app • DSys-2 Runner's portal
Acceptance criteria	• SI-9 can be demonstrated with a test case • The latency requirement QR-5 is fulfilled
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 75 – Example of a technical work item from the YPRC case study

Furthermore, the role of the product owner requires a basic understanding of technical work items in order to allow effective communication with team members working on the

construction and realization of the digital solution. Figure 75 shows an example of a technical work item that deals with the implementation of a particular interface.

The definition of ready and the definition of done must be defined together with construction and realization experts. The definition of done of a technical work item will include some quality assurance measures that demonstrate that the acceptance criteria are fulfilled (e.g., by a set of test cases). We further recommend also using references to the design concepts in technical work items as reference points where possible. This way of working supports an alignment of the technical work with the design concepts.

5.3.3.10 Writing Defects

Regardless of the specific development approach, defects will inevitably be identified in the digital solution. We recommend managing defects as work item types. Figure 76 shows an example from the YPRC case study that can serve as a template.

Pulse data above 255 causes the runner's app to freeze	
Prerequisites	• The runner's watch measures a pulse above 255 during a training session
Defect description	• The runner's watch shows a pulse rate above 255 (a sensor malfunction?) • The runner's app training screen (UI-5) shows an empty pulse value and freezes • The runner must shut down and restart the app in order to continue the training. • The previously recorded training data is lost
Affected elements	• DSys-1 Runner's app • UI-5 training status screen
Expected behavior	• The app should show and record this exceptional pulse value without freezing
Epic reference	• YP-17 As an <athlete>, I want to <be warned by a remote coach> so that <I can avoid pushing too hard during a session>

Figure 76 – Example of a defect from the YPRC case study

Advice for defining the title and the description

The title should provide a short description of the defect in order to provide an idea of the defect. The description consists of prerequisites that are necessary to reproduce the defect. This information is especially important for understanding and analyzing the cause of the defect. Furthermore, the defect description should provide a clear explanation of the defect. The *Affected elements* part should provide some reference to design concepts that may be related to the defect.

Considerations for daily work

In the example above, a screen of the runner's app freezes. Therefore, the app and the concrete user interface are mentioned. Like the prerequisites, this information can support the analysis of the defect. Finally, the last part *Expected behavior* gives a short description of the behavior that would be expected instead of the defect. This information is necessary to give an indication of how the defect should be repaired.

No additional tasks need to be created for defects that can be corrected easily. However, if the analysis of a defect reveals that the defect is complex, new work items (for example, technical work item or user story) are created from the analysis.

5.3.4 Phase 1: Initial release planning and backlog preparation

Before the actual development can start, an initial list of releases and initial backlogs are required. The starting points for this work are the solution design concept and the system design concept created in the conceptual step (see Section 5.2). A naïve approach for Phase 1 would be a complete elaboration of the element design concepts as introduced in Section 2.2.2. This does not make sense because the detailed creation of element design concepts requires a lot of effort and delays the start of the implementation.

From our experience, the following procedure is more practical for beginners in Digital Design:

1) Elaborate element design canvases for each element defined by the system design concept.
2) Elaborate a story map with user stories.
3) Prioritize and group the user stories into epics that define a distinguishable and realizable customer value.
4) Elaborate all work items necessary to achieve the definition of ready for all user stories of the two to three most relevant epics.

As a rule of thumb, the work items for the two to three most important epics should be elaborated. This should create an element design backlog with a sufficient amount of work for a typical building team size (6–7 people). Further epics can be elaborated in parallel with the other development work.

5.3.4.1 Elaborate Element Design Canvases for Each Element

Like the work with the value proposition and business model canvases in the conceptual step, the work on element design concepts can start in a canvas-oriented working mode. Figure 77 shows a simple structure for such a canvas.

Advice for working on element design canvases

The canvas is a temporary artifact and is used for structuring and discussing the ideas for a particular element. For beginners, we suggest defining the elements on the canvas in the following order:

1. Define the main goals that the element shall achieve.
2. Define the initial function of the element:
    - Name use cases (sketch the main scenario and critical alternative scenarios in case the use case is unclear).
    - Name the functions that are necessary for the use cases (sketch function details if the function is not clear from the name).
3. Define the initial form of the element based on the use cases and functions:
    - Name data structures in terms of entities (attributes are created during the sprints).
    - Name important user interfaces (if applicable).
    - Name important technical interfaces (if applicable).
    - Name parts of a device (if applicable).
4. Visualize important data flows between interfaces, use cases, functions, and entities.
5. Create initial lists of quality requirements and constraints for the element.

6. Iterate over all elements until the team agrees on an initial common understanding of the element, include the solution and system design concept as a reference point (for example, check the element against the proposed values from the value proposition canvas).

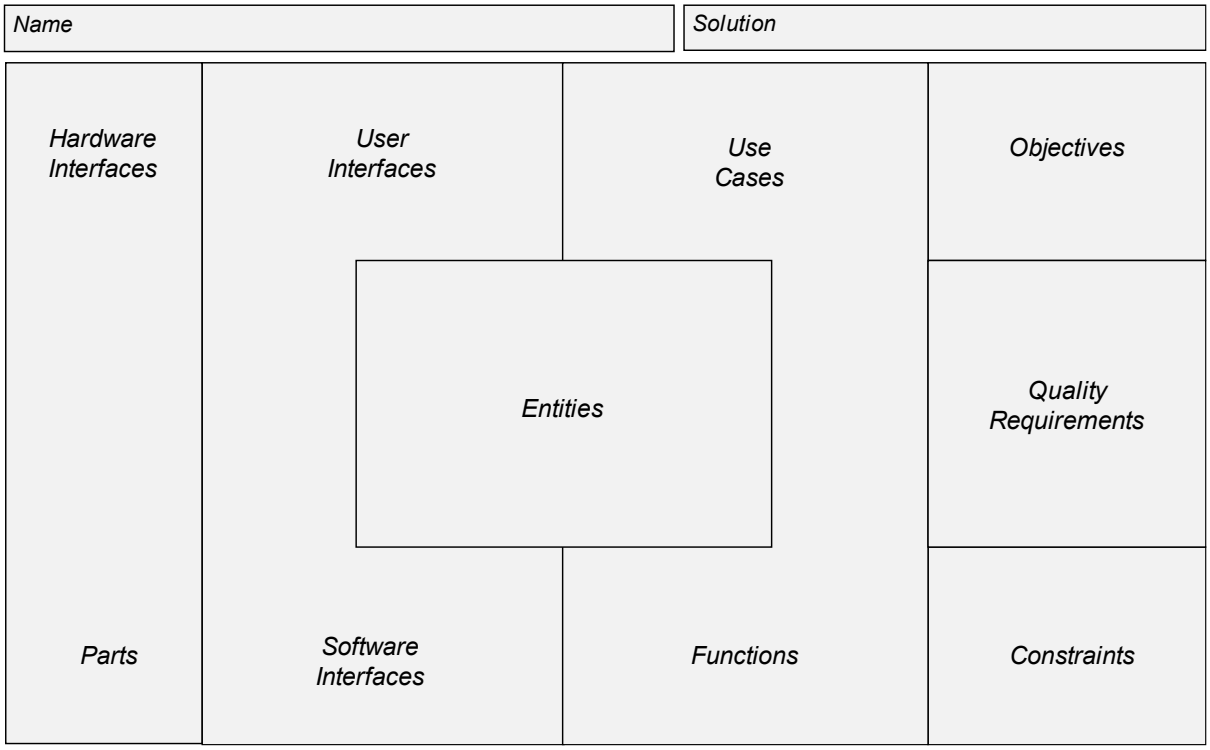


Figure 77 – A simple structure for an element design canvas

It is important to recognize that such a canvas is not intended to be a complete and consistent description of the element considered; it is a starting point for discussion.

Further details (for example, attributes of the entities, detailed shape of a user interface) are defined as part of the sprints and during the elaboration of user stories. With this work, a fully detailed element design concept is created as part of the sprint work (see below).

We recommend that the whole building team works with the product owner on the element design canvas. Additional realization concepts will be created (e.g., an initial technical architecture). Their structure and content depend on the technical details of the digital solution. The important point here is that the product owner, together with the building team, prepares the backlog and all other necessary foundations for the start of the development work (e.g., setup of development environments).

YPRC example. Figure 78 shows an exemplary canvas for the runner's app from the YPRC case study. At first sight, the canvas seems quite chaotic and confusing, especially because of the many elements and connections. As with any canvas technique, however, note that the canvas is a thinking tool for a group of people who create and edit the canvas together in workshop style working mode. A closer look at the canvas reveals important structures for the app. For example, the entity "Training Data" is integrated into various functions and the "Internet Access" is a central interface to fulfill the goals of the app.

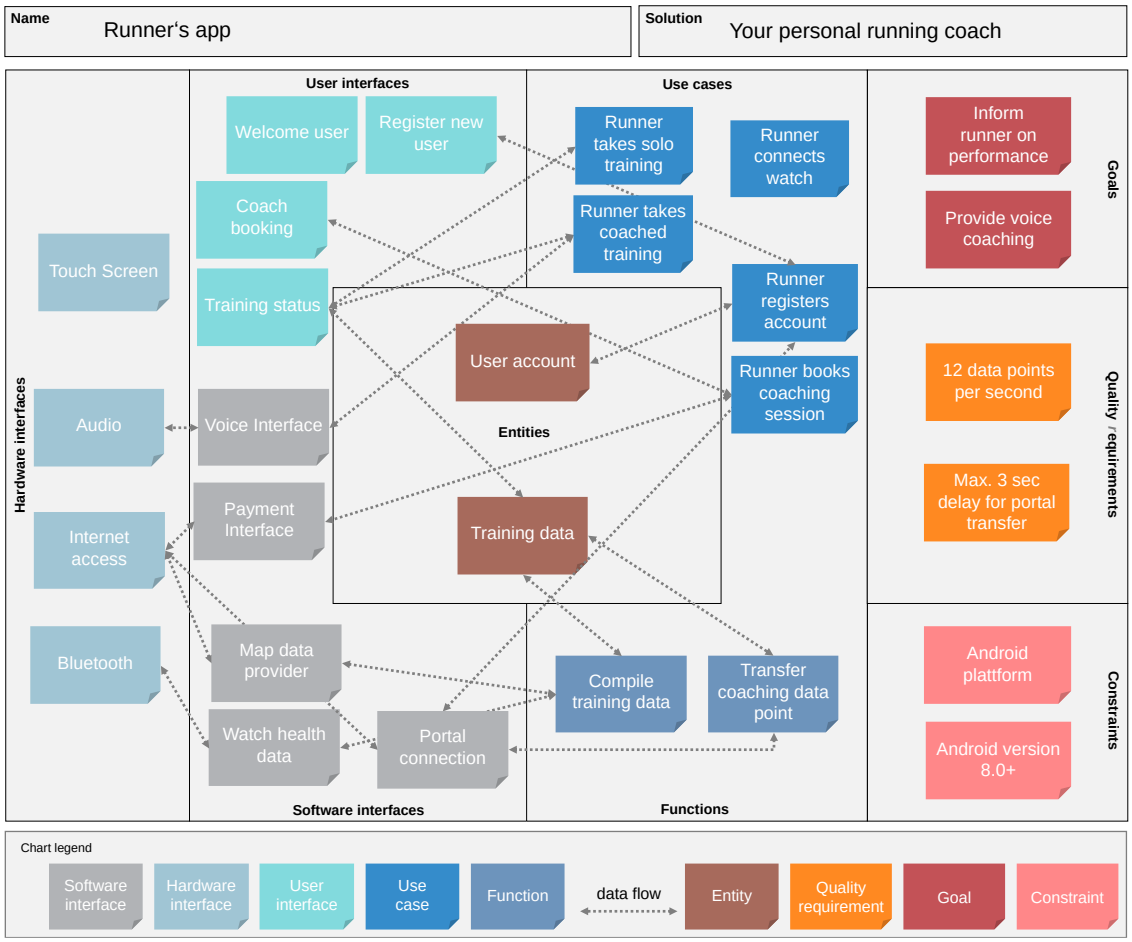


Figure 78 – An exemplary element design canvas from the YPRC case study

Consider the validation of important ideas of an element

It is possible that during the conceptual work on the initial concepts, an urgent need to validate an idea (e.g., a function or usability aspect) occurs.

The following prototypes (see Section 2.3) are recommended as validation techniques for beginners:

- Interactive mock-ups to validate alternative ideas for interaction
- High-fidelity mock-ups to validate alternative ideas for the visual design of the user interface
- Functional prototypes to validate critical technical aspects

However, in this step of the building process, the development of dedicated prototypes requires careful consideration in terms of cost and benefits. Development of a dedicated prototype may be more expensive than the actual implementation and real-life validation of a function. A

general rule for such situations is difficult to define—it is a matter of risk tolerance. Therefore, the product owner should discuss this issue in detail with the building team to identify a good approach.

5.3.4.2 Creating a Story Map for Defining User Stories

User stories can be derived from the goals, use cases, and functions that are described in the element design canvas by means of a story map (see Section 5.3.2.2). We recommend creating an initial story map from the knowledge of the people involved and without considering the existing concepts (solution design concept, system design concepts, and element design canvases). The reason for this is simple; in this stage of the building process, the team is developing an initial understanding of the elements of the solution. By not using the concepts in this step, the team can test its level of understanding of the elements.

This way of working may appear unsystematic at first sight because it does not make use of the existing concepts. Remember that concepts serve as an external memory and this situation (the definition of the story map) is a good point in time to get a fresh view on the building team's understanding of the digital solution. This fresh look may offer new insights into and new ideas on how the digital solution should work. Keep in mind that new insights and ideas should be evaluated.

Once an initial story map has been created, the stories defined—including the flow of stories—can be compared to the existing concepts while working on the details of the user stories. In this step, the team and the product owner can work on the story map, the system backlog, and all concepts in order to achieve a consistent and coherent understanding of the digital solution. Once this understanding has been achieved at a sufficient level of detail, it is important to step back to look at the whole solution by using, for example, the future press release as a reference point and to plan for an evaluation of new insights and ideas.

5.3.4.3 Define Work Items and Estimate, Prioritize, and Manage the System Backlog

As soon as the initial system backlog is defined, it must be prioritized.

DEEP as a tool for prioritization

A good acronym that describes the characteristics of the backlog is *DEEP* [Cohn2004]:

- D – Detailed appropriately means that the backlog items at the top of the backlog must be detailed appropriately. Backlog items with lower priority may require further elaboration work.
- E – Estimated means that effort for working on backlog items should be estimated. Below, we provide advice for estimating backlog items by means of T-shirt sizes.
- E – Emergent means that the backlog is a changing artifact that will be revised continuously.
- P – Prioritized means that the backlog always represents a priority. The easiest way of defining the priority is to understand the backlog as a to-do list where the order of the elements represents the priority.

All these characteristics are closely related to each other. For example, estimation requires a certain level of detail. Furthermore, it is important to recognize that the prioritization of the backlog must align user stories as well as design, technical, prototyping, and evaluation work items in a way that allows proper implementation. For example, if the implementation of a user

story requires details from an element level concept that are not yet finished, the corresponding design task must have a higher priority in the backlog compared to the user story.

Advice for effort estimation

Effort estimation for backlog items is a real challenge, not only for working on user stories, but also for all other types of backlog items introduced above. There are many techniques for effort estimation (cf. [McCo2014]). From our experience, estimation based on T-shirt sizes (cf. [McCo2014]) is a good starting point for getting a feeling for effort estimation. With this technique, every backlog item is categorized by a T-shirt size (S, M, L, XL) that indicates the effort for this task. With each size step, the assumed effort roughly doubles, i.e., an S user story requires half the work of an M user story. Comparing the efforts of backlog items is only advisable for items of the same type (e.g., user stories or concept work items). If a budget-oriented estimation is mandatory (e.g., for budget planning), the team should agree on a baseline budget for the S size for each type. The team can thereby calculate a rough budget estimation.

YPRC example. Figure 79 shows an excerpt of the system backlog from the YPRC case study that is prioritized by using the story map. At the top of the backlog, four user stories are presented that deal with the development of the runner's app initial features (see Figure 61, orange epic). From the backlog, we can conclude that all necessary prerequisites for these user stories have been fulfilled, i.e., the technical basis for the app has been created and the team can start working on the functions and use cases defined by the user stories.

The user story at position 19 is the user story that deals with the runner's coach and the real-time data visualization (black group in Figure 61). In order to implement this user story, conceptual and technical work still needs to be done. This can be concluded from the priority of the two related tasks at position 12 (realization of the interface) and position 13 (visual design of the user interface).

Pos	ItemID	Type	Summary	T-Shirt
1	US-103	User Story	As a <runner>, I want to <register on the portal> to <use the app>	L
2	US-107	User Story	As a <runner>, I want to <configure my training parameters> to <prepare my first training>	XL
3	US-113	User Story	As a <runner>, I want to <connect my YPRC smartwatch> so that <the app can receive health data from the watch during training>	L
4	US-125	User Story	As a <runner>, I want to <record a training session> so that <I can review my performance after the training session>	XL
…				
12	T-2	Technical	Implement the training data transfer between app and portal	M
13	C-7	Concept	Provide a visual design for the user interface "UI-16 supervising screen"	L
..				
19	US-243	User Story	As a <runner's coach>, I want to <see the runner's speed and heart rate in real time> so that < I can recognize when the runner is pushing too hard during a training session>	M
…				

Figure 79 – Excerpt of the system backlog from the YPRC case study

The important benefit of this approach is that the building team does not have to discuss concrete numbers and can instead discuss the relative size of work. When working with this technique, the building team will sooner or later calibrate their understanding of the work and will develop a feeling of how much time is really needed for a particular T-shirt size.

Considerations for daily work

Together with the story map, the system backlog is a powerful tool for managing the development of a digital solution at the system and element levels in detail. Nevertheless, this integrated management of all activity areas during the development and operations step requires training and experience. Our experience shows that this way of working with a story map to maintain the big picture and detailed tasks in a single backlog is a real benefit for multidisciplinary teams. At every point in time, the backlog provides a clear and transparent picture of the work to be done for the whole team and the story map provides a good overview of the longer-term development perspective. And finally, the references to the detailed design concepts take care of documenting the important details of the digital solution.

Managing the system backlog and the story map becomes a challenging task when the backlog grows beyond a number of 20-30 backlog items. Such a number of items will be reached easily, even for very simple digital solutions. We therefore recommend using dedicated software tools to manage the product backlog together with the different concepts. From our experience, a good starting point is issue-tracking tools for managing the backlog and wikis for managing the concepts. The YPRC case study shows you one possible approach for tool support.

5.3.5 Phase 2: Developing the first release of the digital solution

With the initial prioritized backlogs, the first iteration can start to initialize the series of iteration for developing the particular elements and this first release of the digital solution.

Advice for iteration length

We recommend an iteration length of three weeks as a starting point for beginners. We further recommend always starting an iteration on Monday and ending an iteration on Friday. This gives a clear structure to the process and allows long-term reliable scheduling for the team and other stakeholders.

Timebox recommendation for each event

Within an iteration, we recommend the following timeboxes for the events:

- A 15 minute daily (see Section 5.3.1) on every day of the iteration except the last Friday. A good time slot for the daily is right before lunch time.
- On the first and third Wednesday afternoon, the team and the product owner meet in a four-hour timeboxed meeting for the iteration planning.
- On the second Wednesday afternoon, the team, the product owner, and the client meet for the release planning.
- The last Friday of the iteration is reserved for the solution review and the retrospective:
    - The solution review takes place in the morning as a two-hour timeboxed meeting. Scheduling the solution review right before lunch allows some preparation time in the morning.
    - The retrospective takes place in the afternoon as a three-hour timeboxed meeting.
    - The daily is skipped on this day since the whole team is blocked with the solution review and the retrospective.

The timebox recommendations given above are a starting point for beginners. It is important to recognize that a timebox defines a fixed end of the event; it does not mean that the timebox must be used completely at every event. The retrospective is the proper event for reviewing the timebox of other events and the length of the iteration. If the building team feels the need to extend or reduce the timebox of a particular event, the team should discuss this together with the product owner.

Considerations for daily work

The work mode now follows the events described above. The building team works on the items from the system backlog and uses the daily as a collaboration and coordination meeting. The product owner works on the other work items at the same time together with the building team.

At the end of an iteration, the solution review is used to present the results to the client, product owner, and other important stakeholders. The product owner decides whether the implementation presented is acceptable and should become part of the product increment. The guidelines for this decision are of course the list of acceptance criteria. The results from the other tasks (concept, prototyping, technical, etc.) should also be presented briefly during the solution review in order to inform the whole team about the progress that has been made.

After the iteration review has taken place, the retrospective takes place. The concrete form depends on various factors. The retrospective is especially useful for allowing beginners to learn from each other and to plan for improvements for the next sprints.

With the results of the solution review and the retrospective, the next iteration planning can be approached and the iteration starts again.

The story map from the preparation phase (see Section 5.3.4) can be used to maintain the roadmap for the development of the whole digital solution and to align the priorities in the product backlog from the solution perspective.

When an acceptable first release of the solution has been created through this process, the release is put into production. The concrete procedure depends on the type of solution and must be defined with the relevant stakeholders.

5.3.6 Phase 3: Further evolution during operation

As soon as the digital solution is in operation, the working mode changes, since the activities of fixing bugs, incorporating feedback from stakeholders, and the further evolution of the digital solution must be prioritized against each other.

New work item types for managing work during operation

We recommend introducing new work item types for this purpose:

- External bugs: description of defects in the digital solution that have been reported by customers/users
- Improvement ideas: description of an improvement that users/stakeholders want to include in the digital solution

For beginners, we recommend capturing bugs in the digital solution in production as dedicated items (external bugs) in the product backlog. They can be documented in the same way as defects (see Section 5.3.3.10). The separate name helps to distinguish them from internal defects.

Additional feedback from stakeholders can be captured as an improvement task. These tasks can be described in the same way as other concept work items (see Section 5.3.3.6). Here, we also recommend labeling these tasks as improvement ideas to distinguish them from internal conceptual tasks. External bugs and improvement ideas can then be prioritized in the backlog and processed as soon as possible and necessary.

Considerations for daily work

As a rule of thumb, a building team that works on a digital solution in production should reserve approximately 30% of their working capacity for fixing critical external bugs from production. If this capacity is not used during the sprint, additional tasks from the backlog can be worked on. However, our experience shows that such a capacity reserve is useful, especially when longer-term forecasts on the progress of the building team are necessary.

5.3.7 Phase 4: Retirement

At first glance, it may seem strange to talk about the retirement of a digital solution in a handbook on Digital Design. The retirement of a digital solution belongs to the operation of a digital solution and should be considered as well. We do not want to go into details in this section, we only want to create awareness that this phase also belongs to the building process.

Nowadays, a digital solution cannot simply be switched off and disappear from the market. Such a behavior would harm the customers and users and would harm the organization that has provided the digital solution as well.

Considerations for daily work

The retirement of a digital solution also has to be planned and from a Digital Design perspective, the following aspects are important:

- Find out who the actual users of the system are. Especially in large organizations, the number of unknown users of a system or service is constantly increasing.
- Prepare the customers and users for the retirement in advance. For example, inform the customers/users ahead of time that the solution will retire.
- Provide means for customers/users to take care of their data. Depending on the type of the digital solution, it may contain data that is of real importance for the customers (e.g., 10 years of health data when YPRC has been on the market for 10 years). Customers should be able to take their personal data out of the digital solution. This can mean that dedicated data export functionality has to be realized.
- If necessary, plan the migration of data to a new service or at least persist the existing data for later use.
- Make clear that both the service and the data will no longer be available/usable after the decommissioning.
- Potentially, plan a monitoring (or even parallel operation) period after the retirement of the product/service to see if there are unexpected side effects.

5.4 Lean Startup as an Alternative Approach

In the previous three sections, we have introduced a detailed building process for digital solutions. This process follows a structured approach for understanding and realizing a digital solution. A core characteristic of the process presented is that the digital solution will go

operational as soon as a first complete version has been realized. We believe that a DDP at foundation level should be aware of the fact that this is not the only approach for building a digital solution. In the following, we briefly introduce *lean startup* as an approach that follows a different philosophy.

Lean startup [Ries2011] is a methodological framework that aims to shorten product development cycles by adopting a combination of business-hypothesis-driven experimentation, iterative product releases, and validated learning. The central hypothesis is that if startup companies invest their time into iteratively building a product to meet the needs of early customers, they can reduce the market risks and sidestep the need for large amounts of initial project funding and expensive product launches and failures.

Speaking in terms of the building process introduced in Section 2.1, choosing a lean startup approach has a significant impact on all stages of the building process. The scoping and the conceptual step are reduced to a minimum in order to start the actual development of the digital solution as early as possible to realize a minimal viable product.

Minimal viable product as a core element of lean startup

A key challenge in lean startup is to define the *minimal viable product (MVP)* correctly. In our experience, a common misunderstanding when thinking about the MVP is to understand it as a stripped-down digital solution that must be implemented. Such an approach will lead to a weak and often unsuccessful MVP.

With the understanding that a digital solution can be created even without implementation efforts (see Chapter 1), a different approach towards an MVP can be chosen. The MVP can be created by reusing existing and available digital technologies. The YPRC case study can serve as a good reference point for making this clear.

YPRC example. During the whole story line, the team never questioned the fact that some software has to be implemented in order to realize YPRC. For the illustration purpose of the case study, this is fine, and we use this example to explain what an MVP for YPRC could look like. One main part of the YPRC business model is selling remote coaching via voice connection. A radical MVP for this part of the business model could look as follows:

- A simple website that allows users to book a remote coaching session. Payment for such a session can take place with the existing payment provider or via credit card.
- The actual remote coaching takes place with a cell phone call and without any additional digital data transfer.
- The runner can use an instant messenger with a location-sharing feature to share the position and the running route with the remote coach.
- The running coach can talk to the runner and determine their health status from the sound of their voice and the way the runner is breathing.

This MVP is really radical, especially since the digital part is quite primitive. The quality of the coaching might be limited due to a weak data basis. However, in the spirit of lean startup, this solution can be implemented and launched in a matter of days. After a few weeks in business, a lot of experience has been obtained that will be very useful for developing a next version of the MVP—this time maybe with real-time health data transfer.

Considerations for daily work

Lean startup is particularly useful if the chances of success of a digital solution cannot be reasonably estimated by other methods. In this case, the development is approached very quickly in order to bring an operational digital solution to the market as a minimum viable product with minimal resources. Feedback from real customers can then be used to estimate whether the solution and the business model can be successful and how the solution can be developed further.

5.5 Conclusions on the Building Process for Beginners

This section about the building process is certainly overwhelming at first reading. Building a digital solution is indeed a large and complicated undertaking.

The first step in mastering the building of digital solutions is to know what you want to build. This is precisely the competence of Digital Design. The second step is to implement the ideas. And this requires process competence, from the initial idea to the actual implementation. This is also part of Digital Design.

The process presented in this chapter is intended to show how design and actual implementation interact and how a possible process works. In particular, however, this chapter was intended to show that the process is complicated but manageable.

6 Achieving Good Digital Design

To conclude this handbook, we want to discuss how the principles of good Digital Design (see Section 1.4.2) relate to the contents of the handbook:

- P1: Good Digital Design is useful and usable
- P2: Good Digital Design is elegant and aesthetic
- P3: Good Digital Design is evolutionary
- P4: Good Digital Design is exploratory
- P5: Good Digital Design focuses on the person as a whole
- P6: Good Digital Design anticipates the effects of its results
- P7: Good Digital Design respects data protection and data security
- P8: Good Digital Design is sustainable and creates sustainability
- P9: Good Digital Design appreciates analog and digital means equally
- P10: Good Digital Design uses digital means only where this is necessary

This section summarizes the contents of the handbook and shows the reader how important the various contents are in achieving good Digital Design.

6.1 Contributions of the Digital Design Professional

Achieving good Digital Design should be part of the attitude of every DDP. Knowing and understanding the ten principles is therefore the foundation for achieving good Digital Design. The ideas presented in this handbook make an important contribution to the achievement of good Digital Design.

Good Digital Design in the building process

During the building process (see Section 2.1 and Chapter 5), the DDP uses the ten principles above to guide them in all decisions related to the digital solution. The following list is an overview of the steps of the building process in relation to the ten principles:

- The scoping step is very important for understanding the main motivation for building a new digital solution. Principles P9 and P10 should be used to guide the whole discussion in the scoping step. A valid answer of the scoping step, for example, could be that building a digital solution was a good idea but is not really necessary since the existing analog solution is sufficient. Furthermore, the effects of an intended digital solution should be discussed in advance in as much detail as possible to anticipate the effects that the solution will have (P6). However, some effects will only emerge later. A special aspect of this is data protection, which must be considered from the very beginning (P7).
- During the conceptual step, important decisions related to the usefulness and usability will be made (P1), since the main functions of the digital solution are defined together with the value proposition. Each function should be evaluated with principles P9 and P10 in mind in order to define only those functions that are really necessary. This means that the effects of defined functions should be considered (P6) together with data protection and security issues (P7). Important decisions related to the sustainability of a digital solution are made when defining the details of the business model (P8). Finally, in the conceptual step, the foundations for evolutionary and

exploratory development of the digital solution are defined. Here, the form and function of the digital solution are important factors as well as the decisions with regard to the main technology.

- During the development and operations step, usefulness, usability, elegance, and aesthetics (P1 and P2) as well as sustainability (P8) must take priority since the many small design and realization decisions made with respect to the form (e.g., user interface), function (e.g., stored data), and quality (e.g., reaction time) will have tremendous impact on these principles. Nevertheless, all other principles are important as well. In order to achieve good Digital Design, a DDP should not hesitate to question core decisions of previous steps. If new insights appear that mean that an important principle is violated, this decision must be questioned and potentially revised.

Good Digital Design and conceptual work

With the design concepts (see Section 2.2.2), the intended positive benefits, usefulness, usability, elegance, and aesthetics (P1 and P2) should be made explicit, as well as the directions for evolution (P3) and exploration (P4). The foundations for usefulness and usability are laid in the design concepts and the thoughts of the DDPs who shape the design concepts.

In well-defined design concepts, the above-mentioned aspects are already visible. Similarly, the elegance and aesthetics of a digital solution should become visible in the design concepts. This visibility may occur in various details—for example, in a carefully crafted user interface, a sophisticated form with minimal interaction, or in efficient data structures.

Data protection and security (P7) also begin in the design concepts. It is a common misunderstanding that they are only a technological issue. Every decision related to the form or function may have an impact on data protection and security. The main questions must be:

- Is this function or entity really necessary for my digital solution?
- Is this attribute important for the solution and do we really need to store it?
- If we have to store certain data, when do we plan to delete it?
- Is it possible to design the solution in such a way that this data is not necessary?

Good Digital Design and prototypes

The creation of prototypes (see Section 2.3) is an important technique for evaluating the solution defined against the ten principles. For example, usefulness and usability (P1) can be evaluated with user interface prototypes at a very early stage in the process. The same applies to sustainability (P8)—here, functional prototypes can be used to evaluate issues in this direction. Furthermore, prototypes allow exploration of (fundamentally different) alternative design directions and solutions. Even an evolutionary development approach can be based on prototypes.

Good Digital Design and digital technology

Digital technology (Chapter 3) has an impact on several principles. For example, evolutionary and exploratory development depends on the flexibility of the technology and requires a fast and flexible construction and realization process. Usefulness and usability (P1) have a direct relation to digital technology, especially when it comes to interaction technology. Finally, the decision for or against a certain technology should always be made with sustainability (P8) and data protection/security (P7) in mind.

Good Digital Design and human factors

Understanding and addressing human factors (see Section 4.1) is important for creating useful and usable digital solutions (P1), but also for understanding the impact of the digital solution on the behavior and experience of people (P5).

Good Digital Design and digital business models

The business model (see Section 4.2) is the foundation for creating an economically successful digital solution and for obtaining the necessary financial resources for development of a digital solution. However, the business model is also at the core when it comes to the effects of a digital solution (P6). It can also be the cause of negative developments (P5) since the new solution may impact other businesses or lead to unethical business behavior. For example, platform business models can lead to precarious employment or have a negative impact on the market.

The business model is also an important means for understanding whether a digital solution is really necessary (P10) and whether analog or digital means are better suited (P9). Carefully defined revenue streams and cost drivers will speak a clear language here. Finally, the business model should consider sustainability (P8). Even if there are no direct costs or gains created in terms of sustainability, the sustainability of a digital solution has now become an important asset for every digital solution. In addition, certain businesses are not legally responsible for and do not feel responsible for certain negative effects—for example, the nuclear industry or the mobile industry for expanding rare earth under bad conditions for the workers.

Good Digital Design and people management

Finally, people management (see Section 4.3) is important for involving all relevant stakeholders and for getting commitment to build the digital solution envisioned according to the ten principles. People management also becomes important when setting up the whole building team in order to cover the broad competence spectrum necessary to achieve good Digital Design.

6.2 The Importance of Practical Experience and Heuristics

The most important tool for a DDP is their practical experience and their own reflection of it. At this point, we would like to encourage every DDP to actively design their own individual learning path and give some impulses here in order to achieve good Digital Design.

For your own reflection, it can be helpful to compare your own experience as a designer with a theoretical description of design practice or to take a closer look at the social discourses surrounding your own actions. Concrete starting points for this would be:

- [Dors2003] gives a broad introduction to design as a profession.
- Critical text on the development of digital technology and its impact on society (e.g., [Lani2011]) raises awareness of the possible effects of a solution.

It is also valuable to look into the practical experience of established designers and to be inspired by the reflection of concrete design results. Examples of this are:

- Texts about the work of outstanding designers (e.g., [Rams2016]) improve your own aesthetic knowledge and can serve as a source of inspiration
- Forming opinions on winners of established design awards (e.g., Webby Awards: https://www.webbyawards.com/)
- Forming opinions on portfolios of renowned design agencies (e.g., Frog Design: https://www.frogdesign.com/work)

A special category for learning from the experiences of others is heuristics. Heuristics can be defined as a "strategy that ignores part of the information, with the goal of making decisions more quickly, frugally, and/or accurately than more complex methods" [GiGa2011]. For the DDP, there is the concrete possibility to use heuristics as concept modifiers that quickly lead to a potential solution, providing the opportunity for a novel design to occur. [Desi2018] gives an overview of such design heuristics and explains their use.

Furthermore, the DDP can benefit in practice from usability heuristics which, above all, make psychology of perception basics easy to handle. Two very valuable contributions are:

- [Wein2011] is a collection of 100 heuristics from human factors for designers.
- A collection of heuristics that improve the usability of software can be found in [Niel1994].

6.3 The Importance of Teamwork

The ideas presented in this chapter show that good Digital Design is an issue for the whole building process and requires close cooperation between management, design, construction, and realization. DDPs should therefore continuously assess their own competencies (and the competencies of their teams) against the ten principles and involve additional experts where necessary. Examples of experts are:

- Data protection experts to achieve data protection and privacy (P7)
- Visual design experts to achieve elegant and aesthetic visual design (P2)
- Ergonomics and usability experts to achieve enjoyable, useful, and usable digital solutions (P1)
- Requirements engineers to identify stakeholders and to understand stakeholder needs (P1 and P5)
- Social scientists and in particular ethnologists to anticipate the effects of a digital solution (P6)
- Sustainability experts to assess and improve the sustainability of a digital solution (P8)
- Experts in design and construction to achieve an elegant and aesthetic digital solution on the perceivable and underlying layers (P2)

Finally, one important observation remains. This handbook shows that Digital Design is a very diverse profession that requires various skills. We believe it is possible to understand the importance of and the relationship between all these skills. We further believe that it is possible to become a master in some of these skills. Becoming a master of the whole spectrum of skills is possible, but only for exceptional talents. For average people, such as we authors are, the following thought remains as a conclusion to this chapter and this handbook:

Good Digital Design can be achieved only through transdisciplinary teamwork with a team that can cover the diversity of Digital Design skills.

I. References

[Alex2005] Alexander, I. F.: A Taxonomy of Stakeholders: Human Roles in System Development. International Journal of Technology and Human Interaction, Vol 1, 1, 2005, pages 23-59.

[AlIS1977] Alexander, C., Ishikawa, S., Siverstein, M.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977.

[Ande2010] Anderson, D. J.: Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press, 2010.

[BaCoWh1999] Baghai, M., Coley, S., White, D.: The Alchemy of Growth – Kickstarting and Sustaining Growth in Your Company. Orion Business Books, 1999.

[Barn2011] Barnum, C. M.: Usability Testing Essentials. Ready, Set…Test! Morgan Kaufmann, 2011.

[BBKL1978] Boehm, B. W., Brown, J. R., Kaspar, H., Lipow, M., McLeod, G., Merritt, M.: Characteristics of Software Quality. North Holland, 1978.

[BCDE2015] Becker, C., Chitchyan, R., Duboc, L., Easterbrook, S., Prenzenstadler, B., Seyff, N., Venters, C.: Sustainability Design and Software: The Karlskrona Manifesto. International Conference on Software Engineering (ICSE), 467-476, 2015.

[BeBa2007] Beckman, S., Barry, M.: Innovation as a Learning Process: Embedding Design Thinking, California Management Review, 2007.

[BeHo1998] Beyer, H., Holtzblatt, K.: Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann, 1998.

[Bitk2017] Bitkom e. V.: Rollenideal Digital Design - Erfolgreiche Digitalisierung und Digitale Transformation erfordern ein Umdenken in der Softwareentwicklung, 2017, https://www.bitkom.org/Bitkom/Publikationen/Rollenideal-Digital-Design.html, last accessed 2021/03/11.

[Bloo2018] Bloomberg, J.: Digitization, Digitalization, And Digital Transformation: Confuse Them At Your Peril https://www.forbes.com/sites/jasonbloomberg/2018/04/29/digitization-digitalization-and-digital-transformation-confuse-them-at-your-peril/#6f4658632f2c, last accessed 2021/03/11.

[BrDP2005] Broy, M., Deißenböck, F., Pizka, M.: A Holistic Approach to Software Quality at Work. Third World Congress for Software Quality, Munich, Germany, 2005.

[Brow2009] Brown, T.: Change by Design. Harper Business, 2009.

[Budi2018] Budiu, R.: Change Blindness in UX: Definition. https://www.nngroup.com/articles/change-blindness-definition, 2018, last accessed 2021/05/03.

[Chou2013] Choudary, S. P.: Why Business Models Fail: Pipes vs. Platforms. Wired Magazine, 2013.

[Cohn2004] Cohn, M.: User Stories Applied: For Agile Software Development. Addison Wesley Professional, 2004.

[Coop2004] Cooper, A.: The Inmates Are Running the Asylum: Why High-Tech Products Drive Us Crazy and How to Restore the Sanity. 2nd Edition, Que, Indianapolis, 2004.

[CPRE2020] IREB Certified Professional for Requirements Engineering - Foundation Level Syllabus, Version 3.0.1 October 7, 2020.

[CPUX2018] UXQB Certified Professional for Usability and User Experience - Curriculum and Glossary Version 3.15, March 23, 2018.

[CRCN2014] Cooper, A., Reimann, R., Cronin, D., Noessel, C.: About Face: The Essentials of Interaction Design. 4th Edition, John Wiley & Sons, 2014.

[Cros2006] Cross, N.: Designerly Ways of Knowing. Birkhäuser, 2006.

[Demi2000] Deming, W. E.: Out of the Crisis. Reprint. MIT Press, 2018.

[Desi2018] Design Heuristics: Strategies to Inspire Ideas. https://www.designheuristics.com/, 2018, last accessed 2021/03/11.

[Dick2019] Dickel, S.: Prototyping Society. Bielefeld: transcript, 2019.

[Dors1997] Dorst, K.: Describing Design – A Comparison of Paradigms. Delft University Press, Delft, The Netherlands, 1997.

[Dors2003] Dorst, K.: Understanding Design: 150 Reflections on Being a Designer. BIS, 2003.

[Duck2013] Dueck G.: E-Man: Die neuen virtuellen Herrscher. Vieweg+Teubner Verlag, 2013

[ErMa2008] Erlhoff, M., Marshall, T. (Eds.): Design Dictionary: Perspectives on Design Terminology. Birkhäuser, 2008.

[Fla2018] Flaherty, K.: Why Personas Fail. https://www.nngroup.com/articles/why-personas-fail, last accessed 2021/05/02.

[Floy1984] Floyd, C.: A Systematic Look at Prototyping. In: Budde R., Kuhlenkamp K., Mathiassen L., Züllighoven H. (eds): Approaches to Prototyping. Springer, Berlin, Heidelberg, 1984.

[Fros2020] Frost, B.: Atomic Design. https://atomicdesign.bradfrost.com, last accessed 2021/03/11.

[FrPr2009] Freemann, S., Pryce, N.: Growing Object-Oriented Software, Guided by Tests. Addison-Wesley, 2009.

[Gass2013] Gassmann, O.: The St. Gallen Business Model Navigator, BMI Pattern Cards https://bmilab.com/resources, last accessed 2021/03/11.

[GBGSV2013] Grapenthin, S., Book, M., Gruhn, V., Schneider, C., Völker, K.: Reducing Complexity Using an Interaction Room: An Experience Report. SIGDOC 2013, pp. 71-76, 2013.

[GeSE2017] Geissdoerfer, M., Savaget, P., Evans, S.: The Cambridge Business Model Innovation Process. Procedia Manufacturing 8:262–269, 2017.

[GiGa2011] Gigerenzer, G., Gaissmaier, W.: Heuristic Decision Making. Annual Review of Psychology 62(1):451–482, 2011.

[Glas2006] Glass, R. L.: Software Creativity 2.0. developer.* books, 2006.

[Glin2007] Glinz, M.: On Non-Functional Requirements. 15th IEEE International Requirements Engineering Conference (RE'07), Delhi, India, 21–26, 2007

[Glin2008] Glinz, M.: A Risk-Based, Value-Oriented Approach to Quality Requirements. IEEE Software 25(2):34–41, 2008

[Glin2020] Glinz, M.: A Glossary of Requirements Engineering Terminology Version 2.0 October 2020. International Requirements Engineering Board (IREB).

[GOOG2005] Google re:Work, Foster psychological safety: https://rework.withgoogle.com/guides/understanding-team-effectiveness/steps/foster-psychological-safety/ last accessed 2021/03/11

[GOOG2020] Google: Google re:Work, https://rework.withgoogle.com, last accessed 2021/03/11.

[Grop1930] Gropius, W.: bauhausbauten dessau. bauhausbücher 12, albert langen verlag,1930.

[Grud1992] Grudin, J.: Utility and Usability: Research Issues and Development Contexts. Interacting with Computers 4(2):209–217, 1992.

[Hass2008] Hassenzahl, M.: User Experience (UX): Towards an Experiential Perspective on Product Quality. International Conference on Association Francophone d'Interaction Homme-Machine, 11-15, 2008.

[HaTr2006] Hassenzahl, M., Tractinsky, N.: User Experience – A Research Agenda. Behavior & Information Technology 25(2):91–97, 2006.

[Hewe1992] Hewett, T. et al.: ACM Curricula for Human-Computer Interaction. ACM, 1992.

[Hinm2012] Hinman, R.: The Mobile Frontier: A Guide for Designing Mobile Experiences. Rosenfeld, 2012.

[Hüth2009] Hüther, G.: "Gelassenheit hilft - Anregungen für Gehirnbenutzer" (in German), https://www.youtube.com/watch?v=2XlJmew2lK4, last accessed 2021/03/11.

[IDSA2020] Industrial Designers Society of America, "How They Do It" http://www.idsa.org/education/how-they-do-it, last accessed 2021/03/11.

[IEEE2017] ISO/IEC/IEEE 24765. Systems and Software Engineering — Vocabulary. Second Edition. ISO/IEC/IEEE 24765:2017E). New York: The Institute of Electrical and Electronics Engineering, 2017.

[ISO2001] ISO/IEC 9126-1: Software Engineering — Product Quality — Part 1: Quality Model. International Organization for Standardization, 2001

[ISO2011] ISO/IEC 25010: Software Product Quality. https://iso25000.com/index.php/en/iso-25000-standards/iso-25010, 2011, last accessed 2021/03/11.

[ISO2018] ISO 9241-11: Ergonomics of Human-System Interaction — Part 11: Usability: Definitions and Concepts. 2018.

[ISO2019] ISO 9241-210:2019: Ergonomics of Human-System Interaction — Part 210: Human-Centred Design for Interactive Systems. ISO, 2019.

[ISO2020] ISO 9241-110: Ergonomics of Human-System Interaction — Part 110: Interaction Principles. 2020.

[Jera2016] Jerald, J.: The VR Book: Human-Centered Design for Virtual Reality (ACM Books). Morgan & Claypool Publishers, 2016.

[John2010] Johnson, J.: Designing with the Mind in Mind. Morgan Kaufmann, 2010.

[JPMLT2018] Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., Tilkov, S.: Microservices: The Journey So Far and Challenges Ahead. IEEE Software 35(3):24-35, 2018.

[Keir1998] Keirsey, D.: Please Understand Me II: Temperament, Character, Intelligence (1st ed.). Prometheus Nemesis Book Co. ISBN 1-885705-02-6.

[Kell2016] Kelly, K.: The Inevitable: Understanding the 12 Technological Forces that will shape our Future. Viking, 2016.

[KEMP2017] Kemper, M.: Teamsetup & interpersonal dynamics https://www.smarter-service.com/2017/11/28/persoenliches-im-digitalen-transformationsgetriebe/

[Lani2011] Lanier, J.: You Are Not a Gadget: A Manifesto. Penguin, 2011.

[LBGH2018] Lauenroth, K., Bramsiepe, H., Gilbert, D., Hartwig, R., Lehn, K., Schubert, U., Trapp, M.: The Digital Design Manifesto, https://www.digital-design-manifest.de/wp-content/uploads/2021/03/Bitkom_LF_Digital_Design_Manifest_EN.pdf, last accessed 2021/05/02.

[LRHV2009] Law, E. L. C., Roto, V., Hassenzahl, M., Vermeeren A. P., Kort, J.: Understanding, Scoping and Defining User Experience: A Survey Approach. Proceedings of the Conference on Human Factors in Computing Systems, 719-728, 2009.

[LWLB2017] Lee, J. D., Wickens, D., Liu, Y., Boyle, L.: Designing for People: An Introduction to Human Factors Engineering. CreateSpace Independent Publishing, 2017.

[MaLa2015] Margolis, E., Laurence, S. (Eds): The Conceptual Mind: New Directions in the Study of Concepts. MIT Press, 2015.

[Mcco2004] McConnell, S.: Code Complete: A Practical Handbook of Software Construction. Microsoft Press, 2004.

[McCo2014] McConnell, S.: Software Estimation: Demystifying the Black Art. Microsoft Press, 2014.

[McEl2017] McElroy, K.: Prototyping for Designers. O'Reilly, 2017.

[MCPKV2006] McCurdy, M., Connors, C., Pyrzak, G., Kanefsky, B., Vera, A. H.: Breaking the fidelity barrier: an examination of our current characterization of prototypes and an example of a mixed-fidelity success. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06). ACM, New York, 2006.

[McRW1977] McCall, J. A., Richards, P. K., Walters, G. F.: Factors in Software Quality. Rome Air Development Center, 1977.

[Meye2014] Meyer, B.: Agile: The Good, the Hype and the Ugly. Springer, 2014.

[MiKi1994 Milgram, P., Kishino, F.: A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information and Systems. Vol. 77, 1994.

[Ming2020] Minge, M.: meCUE 2.0. http://mecue.de/english/index.html, 2020, last accessed 2021/03/11.

[MoJo2016] Moazed, A., Johnson, N. L.: Modern Monopolies: What It Takes to Dominate the 21st Century Economy. Griffin Publishing, 2016.

[Newm2020] Newman, D.: The Process of Design Squiggle. thedesignsquiggle.com, last accessed 2021/03/11.

[Niel1994] Nielsen, D.: 10 Usability Heuristics for User Interface Design: https://www.nngroup.com/articles/ten-usability-heuristics/. Last accessed August 2019.

[NIJS2019] Nijs, D.(Eds.): Advanced Imagineering. Edward Elgar Publishing, 2019.

[Nobl1996] Noblet, J. de: Industrial Design: Reflection of a Century. Flammarion 1996.

[OKKP2015] Obbink, H., Kruchten, P., Kozaczynski, W., Postema, H., Ran, A., Dominick, L., Kazman, R., Hilliard, R., Tracz, W., Kahane, E.: Software Architecture Review and Assessment Report, https://pkruchten.files.wordpress.com/2011/09/sarav1.pdf, last accessed 2021/03/11.

[OPBS2014] Osterwalder, A., Pigneur, Y., Bernarda, G., Smith, A.: Value Proposition Design: How to Create Products and Services Customers Want. Wiley, 2014.

[Orio2018] Oriol, M. et al.: FAME: Supporting Continuous Requirements Elicitation by Combining User Feedback and Monitoring. 26th IEEE International Requirements Engineering Conference (RE), 2018.

[OsPi2010] Osterwalder, A., Pigneur, Y.: Business Model Generation. Wiley, 2010.

[Pari2021] Paris-Bicking, Christel "DICP" www.keypeoplecheck.com, last accessed 2021/05/02

[Patt2014] Patton, J.: User Story Mapping: Discover the Whole Story, Build the Right Product. O'Reilly, 2014.

[PaYC2010] Paul, D., Yeates, D., Cadle, J. (Eds): Business Analysis. Second Edition, BCS, 2010.

[PeWo1992] Perry, D. E., Wolf, A. L.: Foundations for the Study of Software Architecture. ACM SIGSOFT Software Engineering Notes 17(4), 1992.

[PoLR2013] Polaine, A., Løvlie, L., Reason, B.: Service Design: From Insight to Implementation. Rosenfeld Media, 2013.

[Rams2016] Rams, D.: Less but besser. 6th Edition. Jo Klatt Design+Design Verlag, 2016.

[Rein1997] Reinertson, D.: Managing the Design Factory. Simon and Schuster, 1997.

[Ries2011] Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Books, 2011.

[RiWe1973] Rittel, H. W. J., Webber, M. M.: Dilemmas in a General Theory of Planning. In: Policy Sciences 4(2):155–169, 1973.

[RoCa2003] Rosson, B. M.; Carroll, J. M. (2003): Scenario-Based design. In: Jacko J. A. (ed.); The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications. L. Erlbaum Associates Inc., USA, 1032–1050.

[Ross2019] Rossmann, J.: Think Like Amazon: 50 ½ Ideas to Become a Digital Leader. McGraw-Hill, 2019.

[Royce1970] Royce, W.: Managing the Development of Large Software Systems: Concepts and Techniques. In: Proceedings of IEEE WESCOM. IEEE Computer Society Press, Los Alamitos, 1970.

[SaLe2016] Sauro, J., Lewis, J. R.: Quantifying the User Experience: Practical Statistics for User Research. Morgan Kaufmann, 2016.

[ScHo2016] Schmalstieg, D., Höllerer, T.: Augmented Reality: Principles and Practice. Addidon-Wesley, 2016.

[ScKr2010] Schulze, K., Krömker, H.: A framework to measure user eXperience of interactive online products. International Conference on Methods and Techniques in Behavioral Research, 1–5, 2010.

[ScSu2020] Schwaber, K., Sutherland, J.: The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game. November 2020.

[SeRP2017] Sedano, T., Ralph, P., Péraire, C.: Software Development Waste. 39th International Conference on Software Engineering (ICSE), 2017.

[ShRP2019] Sharp, H., Rogers, Y., Preece, J.: Interaction Design: Beyond Human-Computer Interaction. Fifth edition. Wiley, 2019.

[SlWi1997] Slater, M., Wilbur, S.: A Framework for Immersive Virtual Environments (FIVE): Speculations on the Role of Presence in Virtual Environments. PRESENCE: Virtual and Augmented Reality 6(6):603–616, December 1997.

[Snow2005] Snowden, D.: Multi-Ontology Sense Making: A New Simplicity in Decision Making. Informatics in Primary Health Care 13(1):45-54, 2005.

[Snyd2003] Snyder, C.: Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces. Morgan Kaufmann, 2003.

[SSRW2013] Stanton, N. A., Salmon, P. M., Rafferty, L. A., Walker, G. H., Baber, C., Jenkins, D. P.: Human Factors Methods. A Practical Guide for Engineering and Design. Taylor & Francis, 2013.

[Steu1992] Steuer, J.: Defining Virtual Reality: Dimensions Determining Telepresence. Journal of Communication 42(4):73–93, 1992.

[StSc2011] Stickdorn, M., Schneider, J.: This is Service Design Thinking: Basics - Tools – Cases. BIS Publishers, 2011.

[TARV2019] Tarver, E.: Corporate Culture. https://www.investopedia.com/terms/c/corporate-culture.asp, 2019, last accessed 2021/03/11.

[ThMa2007] Thüring, M., Mahlke, S.: Usability, Aesthetics and Emotions in Human–Technology Interaction. International Journal of Psychology 42(4):253–264, 2007.

[TuAl2013] Tullis, T.; Albert, B.: Measuring the User Experience. Collecting, Analyzing, and Presenting Usability Metrics. Morgan Kaufmann, 2013.

[User2020] User Interface Design GmbH: AttrakDiff. http://attrakdiff.de/index-en.html, 2020, last accessed 2021/03/11.

[VPGV2008] Van de Ven, A., Polley, D., Garud, R., Venkataraman, S.: The Innovation Journey. Oxford University Press, 2008.

[Wake2003] Wake, B: INVEST in Good Stories, and SMART Tasks. 2003, https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/, last accessed 2021/03/11.

[WaMe1986] Ward, P. T., Mellor, S. J.: Structured Development for Real-Time Systems. Pearson, 1986.

[WEF2016] World Economic Forum: The 10 skills you need to thrive in the Fourth Industrial Revolution, 2016, https://www.weforum.org/agenda/2016/01/the-10-skills-you-need-to-thrive-in-the-fourth-industrial-revolution/, last accessed 2021/03/11.

[WEF2018] World Economic Forum: 5 things to know about the future of jobs, 2018, https://www.weforum.org/agenda/2018/09/future-of-jobs-2018-things-to-know/, last accessed 2021/03/11.

[Wegn2005] Wegner, I.: Complexity Theory. Exploring the Limits of Efficient Algorithms. Springer, 2005.

[Wei2018] Wei, E.: Remains of the Day: Invisible Asymptotes. 2018 https://www.eugenewei.com/blog/2018/5/21/invisible-asymptotes, last accessed 2021/03/11.

[Wein1971] Weinberg, G. M.: The Psychology of Computer Programming. Van Nostrand Reinhold, 1971.

[Wein2011] Weinschenk, S. M.: 100 Things Every Designer Needs to Know about People. New Riders, 2011.

[WHBP2016] Wickens, C. D., Holland, J. G., Banbury, S., Parasuraman, R.: Engineering Psychology and Human Performance. Taylor and Francis, 2016.

[Wik2020a] Wikipedia: Databases. https://en.wikipedia.org/wiki/Database, last accessed 2021/03/11.

[Wik2020b] Wikipedia: Outline of databases. https://en.wikipedia.org/wiki/Outline_of_databases, last accessed 2021/03/11.

[Wiki2020c] Wikipedia: Keirsey Temperament Sorter. https://en.wikipedia.org/wiki/Keirsey_Temperament_Sorter, last accessed 2021/03/11.

[WiSh2011] Williamson, D. P., Shmoys, D. B.: The Design of Approximation Algorithms. Cambridge University Press, 2001.

[XDF2020] Xue, H., Desmet, P. M. A., Fokkinga, S. F.: Mood Granularity for Design: Introducing a Holistic Typology of 20 Mood States. International Journal of Design, 14(1):1–18, 2020.

II. List of Figures

Figure 1 – Simplified form of the digital solution YPRC 9
Figure 2 – Simplified form and function of the digital solution YPRC 10
Figure 3 – Intersection between the core activity areas of the building process 16
Figure 4 – Overview of the abstraction levels solution, system, and element 23
Figure 5 – Activities of the building process work in parallel at system and element level 27
Figure 6 – The π-shaped competence profile of a DDP at foundation level 28
Figure 7 – The design squiggle [Newm2020] 34
Figure 8 – The dual-mode model of design [Dors1997] 35
Figure 9 – The three essential steps and work products of the building process 37
Figure 10 – The context scoping perspectives of Digital Design 38
Figure 11 – Adapted ISO/IEC 25010 quality characteristics [ISO2011] 49
Figure 12 – An idealized model of the building process 51
Figure 13 – The end of the building process is a new beginning 53
Figure 15 – A future press release 65
Figure 15 – A persona template 66
Figure 16 – A value proposition canvas template 67
Figure 17 – A customer journey map template 68
Figure 18 – A business model canvas template 69
Figure 19 – A goal template/example from the YPRC case study 76
Figure 20 – A constraint template 77
Figure 21 – A user template 78
Figure 22 – An existing object template 79
Figure 23 – An existing system template 80
Figure 24 – A device template 81
Figure 25 – A software template 82
Figure 26 – A scenario template 83
Figure 27 – A hardware interface template 85
Figure 28 – A user interface template 87
Figure 29 – A software interface template 88
Figure 30 – An entity template 90
Figure 31 – A physical part template 91
Figure 32 – A function template 92
Figure 33 – A use case template 93
Figure 34 – A quality requirement template 95
Figure 35 – Overview of the YPRC digital system with its elements 96

Figure 36 – Building blocks of a software/device design concept 97
Figure 37 – Example of a use case framing other elements 99
Figure 38 – Building blocks of a system design concept 100
Figure 39 – Overview of design concept types in the different stages of the building process 103
Figure 40 – Level of interaction 110
Figure 41 – One example display screen from prototypes of the YPRC case study showing different visual quality levels contributing to the fidelity level of the prototype dimension sensory refinement. Map within the rightmost image: © OpenStreetMap-Mitwirkende, license: www.openstreetmap.org/copyright. 114
Figure 42 – Typical prototype use (activities) in the course of a Digital Design project 119
Figure 43 – Exemplary industrial design prototypes (copyright by Neuland/GENERATIONDESIGN) 123
Figure 45 – Example screens of a paper prototype for the YPRC case study 125
Figure 46 – Example paper prototype for the YPRC case study with several sketched screens 125
Figure 47 – Example cardboard prototype of the runner's watch of the YPRC case study 126
Figure 48 – Example storyboard of the YPRC case study 127
Figure 48 - Examples of technology categorized into (1) perceivable form and function for hardware and software (upper part) and (2) underlying form and function for hardware and software (lower part) 130
Figure 49 – The virtuality continuum according to Milgram and Kishino [MiKi1994] 138
Figure 50 – Simplified model of sensation, attention, and perception 156
Figure 51 – The Component model of User Experience (CUE) (adapted from [ThMa2007][Ming2020]) 160
Figure 52 – The three horizons of business 165
Figure 53 – Effective distribution of temperaments for the interdisciplinary collaboration involved in building a digital solution [KEMP2017] 176
Figure 54 – Three horizon model and supportive team profiles 177
Figure 55 – Team profile examples 177
Figure 56 – A tool for measuring the degree of understanding of a digital solution 185
Figure 57 – A tool for measuring the level of commitment 186
Figure 58 – Relationships that support the development of a good business model canvas 199
Figure 59 – System backlog and system board 206
Figure 60 – Example of a story map from the YPRC case study 208
Figure 61 – Using a story map for prioritizing user stories 208
Figure 62 – Story map and epic board 209
Figure 63 – Relationships between work products at the three levels 212
Figure 64 – Relationships between work item management and design concepts 213
Figure 65 – Example of a solution work item from the YPRC case study 215

Figure 66 – Example of an epic from the YPRC case study 216
Figure 67 – Relationship between epic and design concepts 217
Figure 68 – Relationship between user story and design concepts 219
Figure 70 – Example of a user story from the YPRC case study 220
Figure 70 – Relationship between concept work items and design concepts 222
Figure 71 – Example of a concept work item from the YPRC case study 222
Figure 72 – Example of a prototype work item from the YPRC case study 224
Figure 73 – Example of an evaluation work item from the YPRC case study 225
Figure 74 – Relationship between technical work items and design concepts 226
Figure 75 – Example of a technical work item from the YPRC case study 226
Figure 76 – Example of a defect from the YPRC case study 227
Figure 77 – A simple structure for an element design canvas 229
Figure 79 – An exemplary element design canvas from the YPRC case study 230
Figure 80 – Excerpt of the system backlog from the YPRC case study 232

III. List of Tables

Table 1 – Terminology in building architecture and Digital Design 4
Table 2 – Exemplary relationships between a design concept and a realization concept 17
Table 3 – Document template for a Digital Design brief 58
Table 4 – Document template for a solution design concept 59
Table 5 – Document template for a system design concept 60
Table 6 – Document template for a software design concept 61
Table 7 – Document template for a device design concept 63
Table 8 – Overview of documentation techniques for the solution level 64
Table 9 – Relationships between system level and element level 101
Table 10 – Examples of relationships between solution level and system/element level 101
Table 11 – Concept types in Digital Design 102
Table 12 – Criteria for categorizing prototypes 110
Table 13 – Fidelity profile of the YPRC prototype for the coach's perspective with regard to the five dimensions of a prototype 113
Table 14 – Fidelity profile range of typical prototypes in Digital Design 117
Table 15 – Relationship between prototype dimensions and immersion elements. "X" indicates a strong and "~" a weak relationship. 121
Table 16 – User interface paradigms 135
Table 17 – Characteristics of different application development approaches 144
Table 18 – The three error types 159
Table 19 – Characterization of the four basis temperaments 172
Table 20 – Temperament suitability across the building process 178
Table 21 – Temperaments and skills for the digital age 180
Table 22 – Roles, temperaments, and assignments 181
Table 23 – Structure of a pitch presentation with example content from YPRC 196