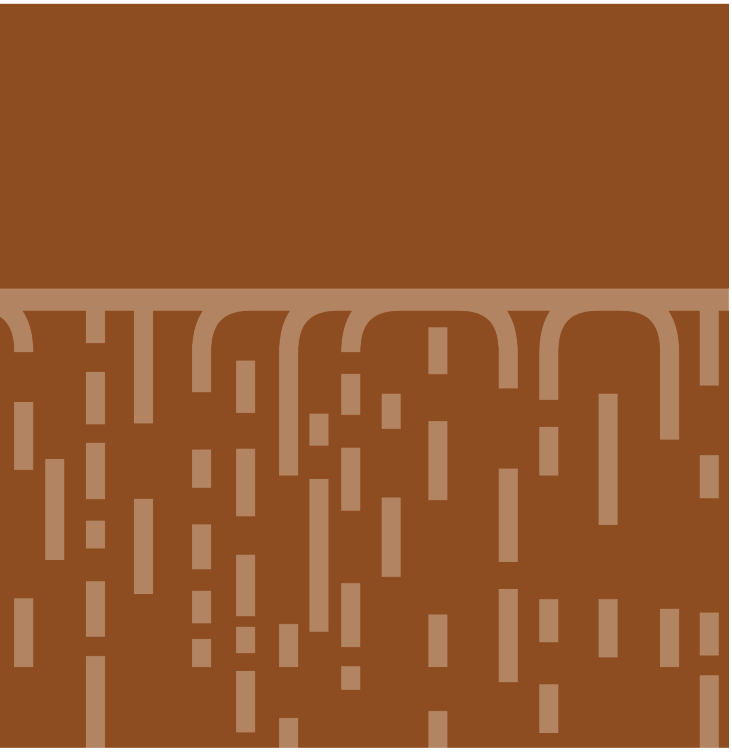
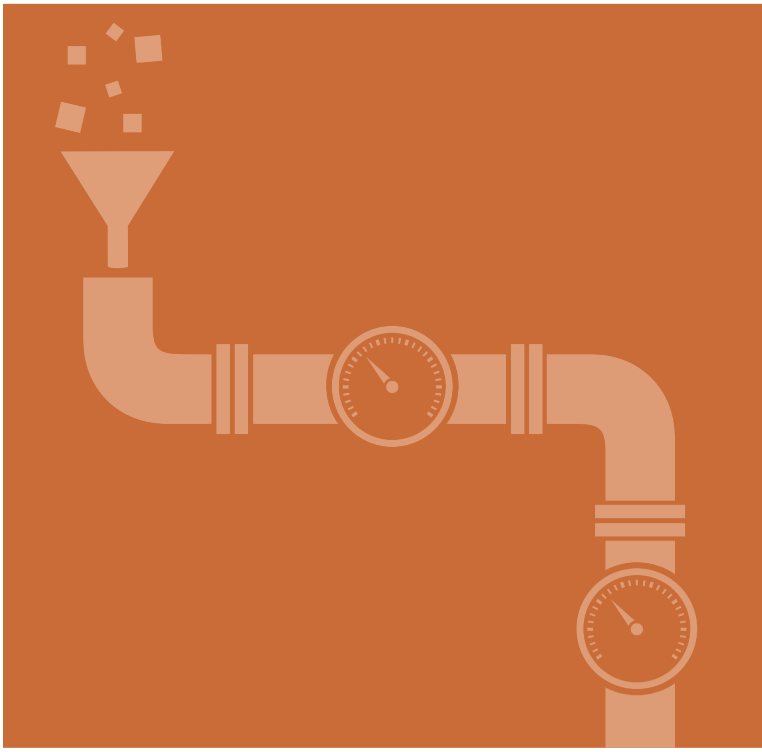
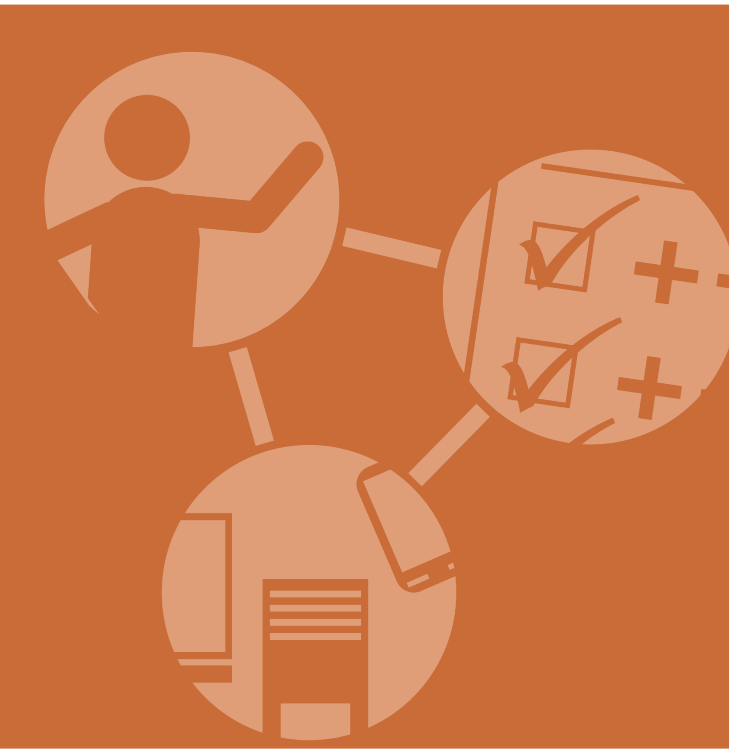
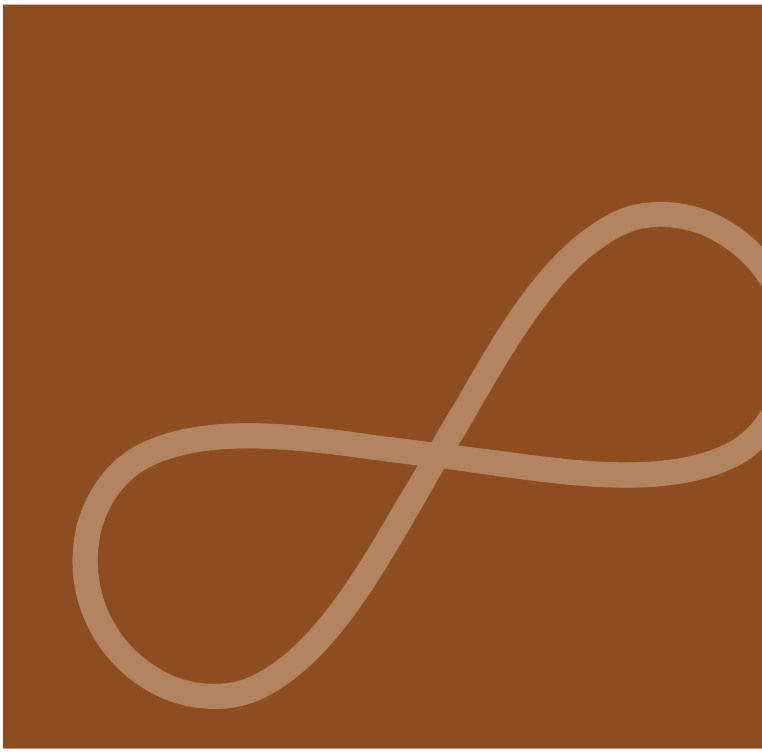
CONTINUOUS DESIGN

A publication in the **Continuous Everyting** series









BART DE BEST

db metrics

DevOps
Continuous Design
Best Practices

A publication in the Continuous Everything series

Bart de Best

Edited by
Louis van Hemmen

Colophon

More information about this and other publications can be obtained from:

Leonon Media

(0)572 - 851 104

Common questions : info@leonon.nl
Sales questions : verkoop@leonon.nl
Manuscript / Author : redactie@leonon.nl

Cover design : Eric Coenders, IanusWeb, Nijmegen
Production : Printforce B.V., Culemborg

Title : DevOps Continuous Design

Subtitle : A publication in the Continuous Everything series

Date : 3 December 2022

Author : Bart de Best

Publisher : Leonon Media

ISBN13 : 978 94 92618 702

Edition : First press, seventh edition, 3 December 2022

"We build our computer (systems)
the way we build our cities:
over time, without a plan, on top of ruins."

by Ellen Ullma

Table of Contents

1 INTRODUCTION 1
1.1 OBJECTIVE 1
1.2 TARGET GROUP 1
1.3 BACKGROUND 1
1.4 STRUCTURE 4
1.4.1 CHAPTER 2: BASIC CONCEPTS AND BASIC TERMS 4
1.4.2 CHAPTER 3: CONTINUOUS DESIGN DEFINITION 4
1.4.3 CHAPTER 4: CONTINUOUS DESIGN ANCHORING 4
1.4.4 CHAPTER 5: CONTINUOUS DESIGN ARCHITECTURE 4
1.4.5 CHAPTER 6: CONTINUOUS DESIGN DESIGN 4
1.4.6 CHAPTER 7: BUSINESS VIEW 4
1.4.7 CHAPTER 8: SOLUTION VIEW 4
1.4.8 CHAPTER 9: DESIGN VIEW 4
1.4.9 CHAPTER 10: REQUIREMENT VIEW 4
1.4.10 CHAPTER 11: TEST VIEW 4
1.4.11 CHAPTER 12: CODE VIEW 4
1.4.12 CHAPTER 13: CONTINUOUS DESIGN AT ASSURITAS 4
1.4.13 CHAPTER 14: CONTINUOUS DESIGN ASSESSMENT 5
1.5 APPENDICES 5
1.6 READING GUIDELINES 5
2 BASIC CONCEPTS AND BASIC TERMS 7
2.1 BASIC CONCEPTS 7
2.1.1 CONTINUOUS DESIGN PYRAMID 7
2.1.2 NON-CONTINUOUS DESIGN PYRAMID 7
2.2 BASIC TERMS 8
2.2.1 BUSINESS VIEW 8
2.2.2 SOLUTION VIEW 9
2.2.3 DESIGN VIEW 9
2.2.4 REQUIREMENT VIEW 9
2.2.5 TEST VIEW 9
2.2.6 CODE VIEW 9
3 CONTINUOUS DESIGN DEFINITION 11
3.1 BACKGROUND 11
3.2 DEFINITION 11
3.3 APPLICATION 11
3.3.1 PROBLEMS TO BE SOLVED 11
3.3.2 THE ROOT CAUSE 12
4 CONTINUOUS DESIGN ANCHORING 13
4.1 THE CHANGE PARADIGM 13
4.2 VISION 14
4.2.1 WHAT DO WE WANT? 14
4.2.2 WHAT DO WE NOT WANT? 15
4.3 POWER 15
4.3.1 WHAT DO WE WANT? 15
4.3.2 WHAT DO WE NOT WANT? 17
4.4 ORGANISATION 18
4.4.1 WHAT DO WE WANT? 18
4.4.2 WHAT DO WE NOT WANT? 18

4.5 Resources 19
4.5.1 What do we want? 19
4.5.2 What do we not want? 20
4.6 Waterfall versus Continuous Design 20
4.6.1 Vision Waterfall approach 21
4.6.2 Power Waterfall approach 22
4.6.3 Organisation Waterfall approach 22
4.6.4 Resources Waterfall approach 22
5 CONTINUOUS DESIGN ARCHITECTURE 23
5.1 Architecture principles 23
5.1.1 General 23
5.1.2 People 23
5.1.3 Process 23
5.1.4 Technology 25
5.2 Architecture models 25
5.2.1 Continuous Design Pyramid model 25
5.2.2 Continuous Design Planning model 26
6 CONTINUOUS DESIGN DESIGN 27
6.1 Continuous Design Value Stream 27
6.2 Continuous Design Use Case Diagram 27
6.3 Continuous Design Use Case 28
7 BUSINESS VIEW 33
7.1 Introduction 33
7.2 System Context Diagram 33
7.2.1 Objective 33
7.2.2 Content 34
7.2.3 Questions to answer 34
7.2.4 Template 34
7.2.5 Tips and Tricks 34
7.2.6 Example 35
7.3 Value Stream Canvas 36
7.3.1 Objective 36
7.3.2 Content 36
7.3.3 Questions to answer 36
7.3.4 Template 36
7.3.5 Tips and Tricks 38
7.3.6 Example 38
8 SOLUTION VIEW 41
8.1 Introduction 41
8.2 Use Case Diagram 41
8.2.1 Objective 41
8.2.2 Content 42
8.2.3 Questions to be answered 42
8.2.4 Basis Template 42
8.2.5 Template with an Include Use Case 42
8.2.6 Template with an Extend Use Case 43
8.2.7 Tips and Tricks 43
8.2.8 Example 44
8.3 System Building Blocks 45

8.3.1 Objective 45
8.3.2 Content 45
8.3.3 Questions to be answered 45
8.3.4 Template System Building Blocks – General 46
8.3.5 Tips and Tricks General 46
8.3.6 Template System Building Blocks – Information 47
8.3.7 Tips and Tricks SBB-I 49
8.3.8 Template System Building Blocks – Application 50
8.3.9 Tips and Tricks SBB-A 51
8.3.10 Template System Building Blocks – Technology 52
8.3.11 Tips and Tricks SBB-T 53
8.3.12 Example SBB-I, SBB-A and SBB-T 54
8.4 Value Stream Mapping 56
8.4.1 Objective 56
8.4.2 Content 56
8.4.3 Questions to be answered 56
8.4.4 Template 56
8.4.5 Tips and Tricks 57
8.4.6 Example 57

9 DESIGN VIEW 59
9.1 Introduction 59
9.2 Use Case 60
9.2.1 Objective 60
9.2.2 Content 60
9.2.3 Questions to be answered 61
9.2.4 Template Use Case Narrrative 61
9.2.5 Tips and Tricks Use Case Narrative 62
9.2.6 Example Use Case Narrative 62
9.2.7 Template Use Case Scenario 65
9.2.8 Tips and Tricks Use Case Scenario 65
9.2.9 Examples Use Case Scenario 66

10 REQUIREMENT VIEW 67
10.1 Introduction 67
10.1.1 Behaviour Driven Development 67
10.1.2 What is BDD? 68
10.1.3 What is Gherkin? 68
10.2 Behaviour Driven Development 68
10.2.1 Objective 68
10.2.2 Content 68
10.2.3 Questions to be answered 68
10.2.4 Template 68
10.2.5 Tips and Tricks 69
10.2.6 Example 69

11 TEST VIEW 71
11.1 Introduction 71
11.2 Test Driven Development 72
11.2.1 Objective 72
11.2.2 Content 72
11.2.3 Questions to be answered 72
11.2.4 Template 72

11.2.5 Tips and Tricks 73
11.2.6 Example 73

12 CODE VIEW 75
12.1 Introduction 75
12.2 Continuous Documentation Development 76
12.2.1 Objective 76
12.2.2 Content 76
12.2.3 Questions to be answered 76
12.2.4 Template 77
12.2.5 Tips and Tricks 78
12.2.6 Example 78

13 CONTINUOUS DESIGN AT ASSURITAS 83
13.1 Assuritas 83
13.1.1 The organisation 83
13.1.2 The vision 83
13.1.3 The goal 83
13.1.4 The strategy 83
13.1.5 The Ist situation 83
13.1.6 The Soll situation 84
13.1.7 The migration path 84
13.2 Continuous Design 85
13.2.1 The question 85
13.2.2 FSA Vision 85
13.2.3 FSA power 87
13.2.4 FSA organisation 87
13.2.5 FSA resources 90

14 CONTINUOUS DESIGN ASSESSMENT 91
14.1 What is the CE-model? 91
14.2 Maturity dimensions 94
14.3 DevOps CE model, CN 94

APPENDIX A, LITERATURE LIST 99

APPENDIX B, GLOSSARY 103

APPENDIX C, ABBREVIATIONS 119

APPENDIX D, WEBSITES 123

APPENDIX E, INDEX 125

Figures

FIGURE 1-1, DEVOPS LEMNISCAAT. 1
FIGURE 1-2, SOR, SOE AND SOI (SOURCE HSO THE RESULT COMPANY). 3
FIGURE 2-1, CONTINUOUS DESIGN PYRAMID. 7
FIGURE 2-2, NON CONTINUOUS DESIGN PYRAMID. 8
FIGURE 4-1, CHANGE PARADIGM. 13
FIGURE 4-2, CHANGE PARADIGM - VISION. 14
FIGURE 4-3, CHANGE PARADIGM - POWER. 15
FIGURE 4-4, CHANGE PARADIGM - ORGANISATION. 18
FIGURE 4-5, CHANGE PARADIGM - RESOURCES. 19
FIGURE 4-6, COMPLETED CHANGE PARADIGM FOR CONTINUOUS DESIGN CHARACTERISTICS. 21
FIGURE 4-7, COMPLETED CHANGE PARADIGM FOR WATERFALL DESIGN CHARACTERISTICS. 21
FIGURE 5-1, CONTINUOUS DESIGN PYRAMID WITH DELIVERABLES AND QUESTIONS TO BE ANSWERED. 25
FIGURE 5-2, CONTINUOUS DESIGN PLANNING MODEL. 26
FIGURE 6-1, CONTINUOUS DESIGN VALUE STREAM. 27
FIGURE 6-2, USE CASE DIAGRAM FOR CONTINUOUS DESIGN. 28
FIGURE 7-1, BUSINESS VIEW. 33
FIGURE 7-2, SYSTEM CONTEXT DIAGRAM TEMPLATE. 34
FIGURE 7-3, SYSTEM CONTEXT DIAGRAM EXAMPLE. 35
FIGURE 7-4, VALUE STREAM CANVAS TEMPLATE. 37
FIGURE 7-5, VALUE STREAM CANVAS EXAMPLE. 39
FIGURE 8-1, SOLUTION VIEW. 41
FIGURE 8-2, USE CASE DIAGRAM BASIS TEMPLATE. 42
FIGURE 8-3, USE CASE DIAGRAM TEMPLATE WITH A INCLUDE USE CASE. 43
FIGURE 8-4, USE CASE DIAGRAM TEMPLATE WITH EXTEND USE CASE. 43
FIGURE 8-5, 16-CELL MODEL. 44
FIGURE 8-6, EXAMPLE USE CASE DIAGRAM 'COFFEE SERVICE'. 45
FIGURE 8-7, TEMPLATE SYSTEM BUILDING BLOCKS – GENERAL. 46
FIGURE 8-8, TEMPLATE SYSTEM BUILDING BLOCKS – INFORMATION. 48
FIGURE 8-9, TEMPLATE SYSTEM BUILDING BLOCKS – APPLICATION. 50
FIGURE 8-10, TEMPLATE SYSTEM BUILDING BLOCKS – TECHNOLOGY. 53
FIGURE 8-11, EXAMPLE OF INFORMATION SYSTEM BUILDING BLOCKS OF A COFFEE SERVICE. 54
FIGURE 8-12, EXAMPLE OF APPLICATION SYSTEM BUILDING BLOCKS OF A COFFEE SERVICE. 55
FIGURE 8-13, EXAMPLE OF TECHNOLOGY SYSTEM BUILDING BLOCKS OF A COFFEE SERVICE. 55
FIGURE 8-14, VALUE STREAM MAPPING TEMPLATE. 57
FIGURE 8-15, EXAMPLE-1 VALUE STREAM MAPPING COFFEE SERVICE. 58
FIGURE 8-16, EXAMPLE-2 VALUE STREAM MAPPING COFFEE SERVICE. 58
FIGURE 9-1, DESIGN VIEW. 59
FIGURE 9-2, TEMPLATE USE CASE SCENARIO. 65
FIGURE 9-3, EXAMPLE OF TWO USE CASE SCENARIOS. 66
FIGURE 10-1, REQUIREMENT VIEW. 67
FIGURE 11-1, TEST VIEW. 71
FIGURE 12-1, CODE VIEW. 75
FIGURE 13-1, BEMORE FOR AN FSA AT ASSURITAS. 85
FIGURE 13-2, FSA VISION AT ASSURITAS. 86
FIGURE 13-3, THE CURRENT AND DESIRED DOCUMENTATION METHOD AT ASURITAS. 86
FIGURE 13-4, FSA POWER AT ASSURITAS. 87
FIGURE 13-5, FSA ORGANISATION AT ASSURITAS. 88
FIGURE 13-6, IMPLEMENTATION CONTINUOUS DESIGN AT ASSURITAS. 88
FIGURE 13-7, FSA TEMPLATE. 89
FIGURE 13-8, FSA USAGE IN THE DEVELOPMENT PROCESS. 89

FIGURE 13-9, FSA RESOURCES. 90
FIGURE 14-1, DEVOPS CE-SPIDER MODEL. 93
FIGURE 14-2, DEVOPS CN-SPIDER MODEL. 96

Tables

TABLE 1-1, CONTINUOUS EVERYTHING ASPECTS. 2
TABLE 1-2, APPENDICES. 5
TABLE 3-1, COMMON PROBLEMS WHEN USING A DESIGN. 12
TABLE 4-1, DIFFERENCES BETWEEN A WATERFALL DESIGN AND A CONTINUOUS DESIGN. 20
TABLE 6-1, USE CASE TEMPLATE. 29
TABLE 6-2, USE CASE FOR CONTINUOUS DESIGN. 31
TABLE 9-1, USE CASE NARRATIVE TEMPLATE. 62
TABLE 9-2, EXAMPLE OF A USE CASE NARRATIVE. 65
TABLE 10-1, GHERKIN FEATURE FILE EXAMPLE. 70
TABLE 11-1, PYTHON UNIT TEST TEMPLATE. 73
TABLE 11-2, PYTHON UNIT TEST EXAMPLE. 73
TABLE 12-1, PYTHON HEADER ANNOTATION EXAMPLE. 80
TABLE 12-2, C++ QT STYLE SCRIPT ANNOTATION EXAMPLE. 81
TABLE 12-3, APPLICATION DOCUMENTATION GENERATED BY DOXYGEN. 82
TABLE 14-1, CE MATURITY MODEL. 91
TABLE 14-2, CONTINUOUS EVERYTHING. 92
TABLE 14-3, CMMI LEVELS FOR CONTINUOUS EVERYTHING. 93
TABLE 14-4, PRINCIPLE OF MATURITY LEVELS. 94
TABLE 14-5, CN MATURITY CHARACTERISTICS. 96

Appendices

APPENDIX A, LITERATURE LIST 99
APPENDIX B, GLOSSARY 103
APPENDIX C, ABBREVIATIONS 119
APPENDIX D, WEBSITES 123
APPENDIX E, INDEX 125

Preface

This book has been compiled based on my experiences designing information systems in a DevOps environment. It's a snapshot of the best practices I'm using now. Given the speed at which the world of DevOps is developing and the need to give you as many images as possible of using a Continuous Design with as little text as possible, I have decided to keep this book Agile. This means that it describes very briefly what are important insights that I have gained during my role as a consultant, trainer, coach and examiner with regard to Continuous Design related work. Where appropriate, I refer to sources that I myself have consulted for further training. I hereby realise that these best practices will not apply to all information systems and that the approach is a snapshot that may be outdated due to the increasing speed of innovation.

I have already shared many of my experiences in the articles on www.ITpedia.nl. I have also translated the knowledge and skills into various training courses that I provide. These can be found at www.dbmetrics.nl.

I would like to express my sincere thanks to the following people for their inspiring contribution to this book and the great collaboration!

• D. (Dennis) Boersen	Argis IT Consultants
• F. (Freek) de Cloe	smartdocs.com
• J.A.E. (Jane) ten Have	-
• Dr. L.J.G.T. (Louis) van Hemmen	BitAll B.V.
• J.W. (Jan-Willem) Hordijk	Cloud Advisor - Nordcloud, an IBM company
• W. (Willem) Kok	Argis IT Consultants
• N (Niels) Talens	www.nielstalens.nl
• D. (Dennis) Wit	ING

I wish you a lot of fun reading this book and, above all, much success in applying Continuous Design within your own organisation.

If you have any questions or comments, please don't hesitate to contact me. A lot of time has gone into making this book as complete and consistent as possible. Should you nevertheless find shortcomings, I would appreciate it if you would inform me, so that these matters can be incorporated in the next edition.

Bart de Best, Zoetermeer.
bartb@dbmetrics.nl

1 Introduction

Reading Guide:
This chapter describes the objective of this book (1.1) the intended target group (1.2), the background of this book (1.3), the structure (1.4) the appendices (1.5) and finally some tips for handling this book (1.6).

1.1 *Objective*

The objective of this book is to provide basic knowledge of Continuous Design and tips and tricks for applying this aspect area of Continuous Everything.

1.2 *Target group*

The target audience of this book are all involved functions in the DevOps teams. This includes the architects, Dev engineers, Ops engineers, Product owners, Scrum masters, Agile Coaches and representatives of the user organisation. This book is of course also very suitable for line managers, process owners, process managers, etc. who are involved in the creation of information provision through a DevOps method. Finally, there is a target group that does not develop or manage, but that determines whether the information provision meets the required criteria. This target group includes quality employees and auditors. They can use this book to identify risks that need to be taken or controlled.

1.3 *Background*

This book contains various techniques to continuously give substance to a design that grows with the information system to be developed. Continuous Design is part of the DevOps Lemniscate as shown in Figure 1-1.

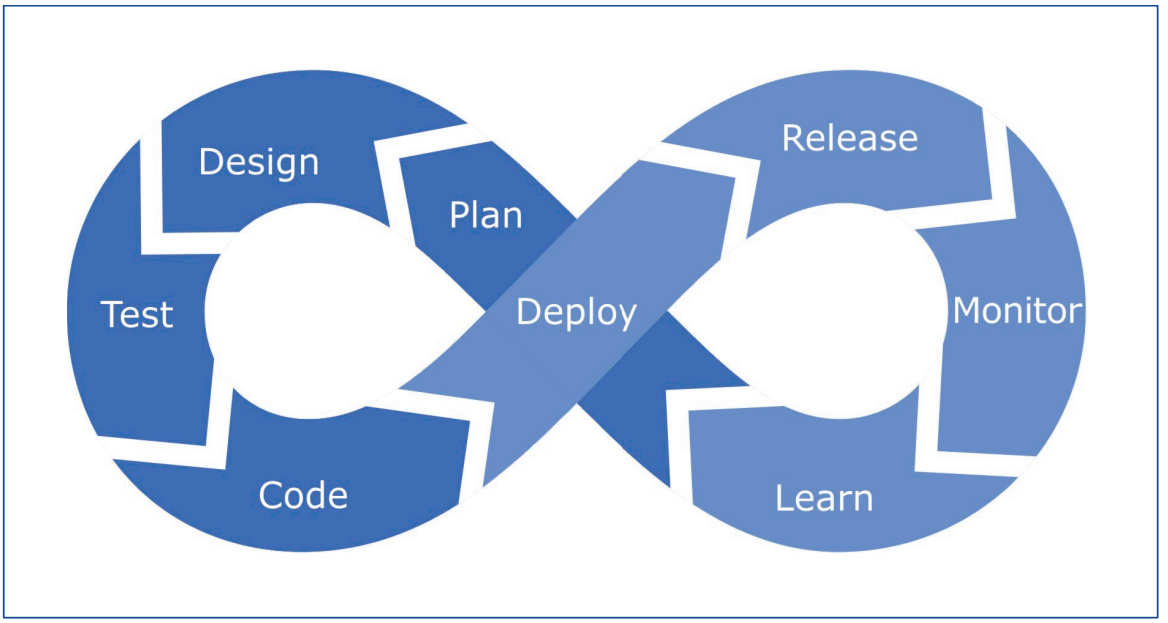


Figure 1-1, DevOps Lemniscaat.

The DevOps Lemniscate provides an overview of the phases to be followed to continuously produce software. The DevOps Lemniscate is therefore a good basis for defining the concept of Continuous Design.

Continuous Design is one of the aspects of the Continuous Everything concept. The CE concept describes all phases of the DevOps Lemniscate in the form of activities to be performed continuously. Table 1-1 shows the relationship between the steps of the DevOps Lemniscate and the Continuous Everything aspect areas.

Development		Operations	
1	Continuous Planning (Plan)	6	Continuous Deployment (Release)
2	Continuous Design (Design)	7	Continuous Monitoring (Monitor)
3	Continuous Testing (Test)	8	Continuous Learning (Learn)
4	Continuous Integration (Code)	9	Continuous Auditing (-)
5	Continuous Deployment (Deploy)	10	Continuous Assessment (-)

Table 1-1, Continuous Everything aspects.

Continuous Auditing (9) and Continuous Assessment (10) are not represented in the DevOps Lemniscate, as are other Continuous aspect areas such as Continuous Documentation for the sake of simplicity of the DevOps Lemniscate.

The word "Continuous" refers to the incremental and iterative development of software that creates a "stream" of code that is continuously transferred to production through the CI/CD secure pipeline. This stream is nothing but a value stream that needs to be optimised.

Continuous Design is one of the aspect areas of the Continuous Everything concept. Continuous Design is an Agile way of designing an information system by defining the behaviour, functionality, and quality of the information system at the right time. All aspects of the DevOps Lemniscate have a direct or indirect relationship with Continuous Design because it is designed holistically. This implies that Continuous Design relates to both the information system (Technology) and the production process (Process) as well as knowledge and skills (People). Thus, Continuous Design gives a new design on PPT.

The existence of a design for an information system has been questioned by many organisations in recent years. The classic justification of bundling information about an information system and involving all stakeholders is seen as outdated by the Agile way of working and the idea of the three-amigo development strategy. This strategy implies that from three disciplines: business, development and testing, an increment that needs to be built is looked at in advance. In this way, the 'how' and the 'what' question are better worked out and consensus can be reached about the Definition of Done (DoD) of the increment. However, this ignores the other classic justification for a design, which is that a design is also intended for knowledge transfer, support for service management and compliance with laws and regulations.

From the point of view of service management, the developments taking place within the DevOps world are therefore very important to be followed. While on the one hand there are still organisations that work with waterfall projects that require a large design effort, there are also organisations that experience that working with user stories alone is not the best solution and that some form of design is indeed necessary. And so, the world of system development comes into balance again.

The question is, of course, whether the same structure of work should apply to all types of information systems. With the arrival of Gartner's BI model, it has become clear that a distinction must be made between the System of Records (SoR) and the System of Engagement (SoE) information systems. In addition to the SoE, people nowadays also speak of the System of Intelligence (SoI). Figure 1-2 provides an overview of the relationship between the three types of information systems (SoR, SoE and SoI).

System of Records

The SoR are information systems of the back office that fulfil the finance, logistics, inventory and Human Resource Management (HRM) tasks. These systems are supervised by the government and must meet many legal and regulatory requirements, such as those of the tax authorities, De Nederlandse Bank (DNB) and the Netherlands Authority for the Financial Markets (AFM). But the Sarbanes Oxley (SoX) legislation and the General Data Protection Regulation (GDPR) are also sources of requirements that justify a design of an information system. The financial information systems must also comply with the General IT Controls (GITC) of the accountants in order to receive a signature under the annual accounts. This means, among other things, that designs are required that indicate how the financial data is generated and what the interfaces are of the different involved information systems.

In general, they are information systems that are part of a chain of information systems. These systems require a well-considered approach and therefore a design.

System of Engagement

The SoE information systems are aimed at sales channels to consumers, especially web shops and apps for smartphones. These applications are easy to provide with a new release, version, and patch. These information systems are usually not an integral part of a chain, but rather the end points of a chain.
These are often also the examples from the publications about Agile and Development & Operations (DevOps). For these information systems it is clear that a design that has been thought through in advance (upfront design) is less necessary and can often suffice with a growing design (emerging design). It does appear that for these SoE information systems it is useful to have more than just a collection of user stories. All the more so because user stories are often placed on the sprint backlog and archived after the sprint has been completed. Also, the individual user stories do not form an accessible description of an information system. There is therefore still a need for a design that provides an overview and insight into the functionality, quality and operation of the information system.

System of Intelligence

In addition, there are Business Intelligence (BI) solutions. These are the reports, data analysis tools and the like. The same applies to these applications as the SoE information systems. They are the representation of information from the SoR and are easier to modify.

Figure 1-2, SoR, SoE and SoI (source HSO the result company).

Necessity of Continuous Design

There is therefore a need for a Continuous Design for all three types (SoR, SoE and SoI) of information systems. More than just a set of user stories is needed to gain and maintain an overview and insight into the information system. Otherwise, the risks and impact of adapting and expanding the information system cannot be recognised in a timely and correct manner.

However, it must be prevented that the design does not destroy the Agility of the production process. This means that the design must be designed incrementally and iteratively (Continuous Design). The degree to which the design must be defined in advance is shifting from a lot with SoR to less with SoE and almost nothing with SoI. But also, with a SoR, the design can be divided into layers that are defined more upfront (in advance) and emerging (in sprints). To explain this, this book first discusses the Continuous Design Pyramid that divides a Continuous Design into layers (views) and then each view is explained in a separate chapter.

1.4 *Structure*

This book includes six views to shape Continuous Design. Before these are discussed, the definitions, anchoring and architecture of Continuous Design are first elaborated. This is followed by a discussion of the six views for each chapter.

1.4.1 *Chapter 2: Basic concepts and basic terms*

This chapter discusses the basic concepts and the basic terms.

1.4.2 *Chapter 3: Continuous Design definition*

It is important to have a common definition of Continuous Design. Therefore, this chapter defines this concept and discusses the problems and possible causes of improper information system design.

1.4.3 *Chapter 4: Continuous Design anchoring*

This chapter discusses how Continuous Design can be anchored through the change paradigm. The following questions will be answered.

- What is the vision on Continuous Design (Vision)?
- Where do the responsibilities and authorities lie (Power)?
- How can Continuous Design be applied (Organisation)?
- Which profiles of people and which resources are needed (Resources)?

1.4.4 *Chapter 5: Continuous Design architecture*

This chapter describes the architectural principles and models for Continuous Design. The architectural models relate to the 'Continuous Design Pyramid' model and the 'Continuous Design Planning' model.

1.4.5 *Chapter 6: Continuous Design design*

The design of Continuous Design defines the Continuous Design Value Stream and Use Case Diagram.

1.4.6 *Chapter 7: Business View*

The business view includes the description of the business processes based on a 'System Context Diagram' and a 'Value Stream Canvas model'.

1.4.7 *Chapter 8: Solution View*

The solution view comprises the outline description of the information system and also the way in which the information system is related to the business processes and the stakeholders. Three models are used for this purpose, namely the 'Use Case Diagram', the 'System Building Blocks' and the 'Value Stream Mapping'.

1.4.8 *Chapter 9: Design View*

The design view is based on the use of Use Cases.

1.4.9 *Chapter 10: Requirement View*

The requirement view uses the Behaviour Driven Development (BDD) approach to describe the behaviour of the information system.

1.4.10 *Chapter 11: Test View*

The test view is part of the Continuous Design because at this level the technical specifications are determined at unit test case level, using Test Driven Development (TDD).

1.4.11 *Chapter 12: Code View*

The code view is also part of the Continuous Design because it is at this level that decisions are made about how the technical requirements (unit test cases) should be programmed. Based on the source code and all other deliverables of the deployment pipeline such as infrastructure as code, test cases, deployment scripts, log files and the like, an application design can be generated using specialised tools. Because this implies that the documentation is continuously created and maintained, it is also known as Continuous Documentation.

1.4.12 *Chapter 13: Continuous Design at Assuritas*

This chapter gives an example of the application of Continuous Design.

1.4.13 *Chapter 14: Continuous Design assessment*

The maturity of Continuous Design is made measurable in this chapter on the basis of a Continuous Design assessment.

1.5 *Appendices*

The appendices contain important information that helps to better understand Continuous Design.

Appendix	Subject	Explanation
A	Literature references	In this book reference is made to consulted literature in the form of: [Author Year]. In the appendix, the full name of the author, the title and the ISBN number are given.
B	Glossary	Only the main concepts are explained in this appendix.
C	Abbreviations	Within the world of DevOps many abbreviations are used. Frequently used terms have been abbreviated for the readability of this book. The first time an abbreviation is used, it is spelled out.
D	Websites	A number of relevant websites are included in this appendix. In this book, these websites are referred to by the reference: [http Name].
E	Index	The index includes references to terms used in this book.

Table 1-2, Appendices.

1.6 *Reading guidelines*

The number of abbreviations in this book is limited. However, terms that keep coming back are represented as abbreviations to increase readability. Appendix C lists these abbreviations.

Appendices

Appendix A, Literature list

Table A-1 provides an overview of books that are directly or indirectly related to DevOps.

References	Publications
Best 2011a	B. de Best, "SLA best practice", Dutch language, Leonon Media 2011, ISBN13: 978 90 71501 456.
Best 2011b	B. de Best, "ICT Performance-Indicatoren", Dutch language, Leonon Media 2011, ISBN13: 978 90 71501 470.
Best 2012	B. de Best, "Quality Control & Assurance", Dutch language, Leonon Media 2012, ISBN13: 978 90 71501 531.
Best 2014a	B. de Best, "Acceptatiecriteria", Dutch language, Leonon Media, 2014, ISBN 13: 978 90 71501 784.
Best 2014c	B. de Best, "Cloud SLA, English language, Leonon Media, 2014 ISBN13: 978 90 9261 8009.
Best 2017a	B. de Best, "Beheren onder Architectuur", Dutch language, Leonon Media, 2017, ISBN13: 978 90 71501 913.
Best 2017c	B. de Best, "SLA Templates", English language, Leonon Media, 2017, ISBN13: 978 94 92618 030.
Best 2018a	B. de Best, "Agile Service Management with scrum", English language, Leonon Media, 2018, ISBN13: 978 94 9261 8085.
Best 2018b	B. de Best, "Agile Service Management with Scrum in Practice", English language, Leonon Media, 2018, ISBN13: 978 94 9261 8177.
Best 2018c	B. de Best, "DevOps best practice", English language, Leonon Media, 2018, ISBN13: 978 94 92618 078.
Best 2019	B. de Best, "DevOps Architecture", English language, Leonon Media, 2019, ISBN13: 978 90 71501 579.
Best 2021b	B. de Best, "Basiskennis IT", Dutch language, Leonon Media, 2021, ISBN13: 978 94 92618 573.
Best 2022 CA	B. de Best, "Continuous Auditing", English language, Leonon Media, 2022, ISBN13: 978 94 92618 757.
Best 2022 CD	B. de Best, "Continuous Deployment", English language, Leonon Media, 2022, ISBN13: 978 94 92618 733.
Best 2022 CI	B. de Best, "Continuous Integration", English language, Leonon Media, 2022, ISBN13: 978 94 92618 689.
Best 2022 CL	B. de Best, "Continuous Learning", English language, Leonon Media, 2022, ISBN13: 978 94 92618 740.
Best 2022 CM	B. de Best, "Continuous Monitoring", English language, Leonon Media, 2022, ISBN13: 978 94 92618 719.
Best 2022 CN	B. de Best, "Continuous Design", English language, Leonon Media, 2022, ISBN13: 978 94 92618 702.
Best 2022 CP	B. de Best, "Continuous Planning", English language, Leonon Media, 2022, ISBN13: 978 94 92618 726.
Best 2022 CS	B. de Best, "Continuous Assessment", English language, Leonon Media, 2022, ISBN13: 978 94 92618 696.
Best 2022 CT	B. de Best, "Continuous Testing", English language, Leonon Media, 2022, ISBN13: 978 94 92618 672.
Best 2022 CY	B. de Best, "Continuous Security", English language, Leonon Media, 2022, ISBN13: 978 94 91480 188.
Best 2022a	B. de Best, "Continuous Development", English language, Leonon Media, 2022, ISBN13: 978 94 92618 764.

References	Publications
Best 2022b	B. de Best, "Continuous Operations", English language, Leonon Media, 2022, ISBN13: 978 94 92618 771.
Best 2022c	B. de Best, "Continuous Control", English language, Leonon Media, 2022, ISBN13: 978 94 91480 201.
Best 2022d	B. de Best, "Continuous Everything", English language, Leonon Media, 2022, ISBN13: 978 94 92618 665.
Bloom 1956	Benjamin S. Bloom, "Taxonomy of Educational Objectives (1956)", Allyn and Bacon, Boston, MA. Copyright (c) 1984 by Pearson Education.
Boehm 1981	Boehm B. Software Engineering Economics, Prentice Hall, 1981
Caluwé 2011	L. de Caluwé en H. Vermaak, "Leren Veranderen", Kluwer, 2011, tweede druk, ISBN13: 978 90 13016 543.
Davis 2016	Jennifer Davis, Katherine Daniels, "Effective DevOps Building a Culture of Collaboration, Affinity, and Tooling at Scale", O'Reilly Media; 1 edition, 2016, ISBN-13: 978 14 91926 307.
Deming 2000	W. Edwards Deming, "Out of the Crisis. MIT Center for Advanced Engineering Study", 2000, ISBN13: 978 02 62541 152.
Downey 2015	Allen. B. Downey, "Think Python", O'Reilly Media, Inc, Usa; Druk 2, 2015, ISBN-13: 978 14 91939 369.
Galbraith 1992	Galbraith, J.R. "Het ontwerpen van complexe organisaties", Alphen aan de Rijn: Samson Bedrijfsinformatie, 1992.
Humble 2010	Jez Humble, David Farley "Continuous Delivery Reliable Software Releases through Build, Test, and Deployment Automation", Addison-Wesley Professional; 1 edition, 2010, ISBN-13: 978 03 21601 919.
Kim 2014	Gene Kim, Kevin Behr, George Spafford "The Phoenix Project", IT Revolution Press, 2014, ISBN-13: 978 09 88262 508.
Kim 2016	Gene Kim, Jez Humble "The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations, Patrick Debois, John Willis", 2016, IT Revolution Press, ISBN-13: 978 19 42788 003.
Kotter 2012	John P. Kotter, "Leading Change", Engels 1e druk, November 2012, ISBN13: 978 14 22186 435.
Kaplan 2004	R. S. Kaplan en D. P. Norton, "Op kop met de Balanced Scorecard", 2004, Harvard Business School Press, ISBN13: 978 90 25423 032.
Layton 2017	Mark C. Layton Rachele Maurer, "Agile Project Management for Dummies", tweede druk, John Wiley & Sons Inc, 2017, ISBN13: 978 11 19405 696.
Looijen 2011	M. Looijen, L. van Hemmen, "Beheer van Informatiesystemen", zevende druk, Academic Service, 2011, ISBN13: 978 90 12582 377.
MAES	R. Maes, "Visie op informatiemanagement", www.rikmaes.nl.
McCabe	McCabe T. "A Complexity Measure" in: IEEE Transactions on Software Engineering 1976, vol. 2, nr. 4.
Michael Porter 1998	M.E. Porter '"acceptance criteria Advantage: Creating and Sustaining Superior Performance, Simon & Schuster, 1998, ISBN13: 978 06 84841 465.
Oirsouw 2001	R.R. van Oirsouw, J. Spaanderman, C. van Arendonk, "Informatiserings-economie", 2001, ISBN 90 395 1393 7.
scrum	Ken Schwaber and Jeff Sutherland, "The Scrum Guide™", 2017, www.scrumguides.org.

References	Publications
Schwaber 2015	K. Schwaber, “Agile Project Management with scrum”, Microsoft Press, ISBN13: 978 07 35619 937.
Toda 2016	(Luke) Toda, President Strategic Staff Services Corporation and Director of TPS Certificate Institution Nobuyuki Mitsui, CTO of Strategic Staff Services Corporation, “Success with Enterprise DevOps Koichiro” “White Paper”, 2016.

Table A-1, Literature list.

Appendix B, Glossary

A glossary of terms is included in Table B-1.

Term	Meaning
5S	Japan's principle of order and cleanliness. These Japanese terms with their Dutch equivalent are: Seiri (整理): Sort Seiton (整頓): Arrange Seisō (清掃): Cleaning Seiketsu (清潔): Standardise Shitsuke (躾): Hold or Systematise [Wiki]
A/B testing	A/B testing means that two versions of an application or webpage are taken into production to see which performs better. Canary releasing can be used, but there are also other ways to perform A/B testing.
Acceptance test	For DevOps engineers the acceptance testcases gives the answer "How do I know when I am done?". For the users the acceptance testcases gives the answer "Did I get what I wanted?". Examples of acceptance testcases are Functional Acceptance Testcases (FAT), User Acceptance Testcases (UAT) and Production Acceptance Testcases (PAT). The FAT and UAT should be expressed in the language of the business.
Affinity	DevOps is about collaboration and affinity. Where collaboration is focused on the relationship between individuals in a DevOps team, affinity goes one step further. This DevOps pillar is about shared organisational goals, empathy and learning between different groups of people by sharing stories and learn from each other.
Agile Infrastructure	Within DevOps both Development and Operations work in an Agile way. This requires an Agile Infrastructure that can be changed with the same pace as the application is changed through the deployment pipeline. A good example of an Agile Infrastructure is the use of Infrastructure as Code.
Alternate path	See happy path.
Andon cord	In the Toyota manufacturing plant, above every work centre a cord is installed. Every worker and manager are trained to pull when something goes wrong; for example, when a part is defective, when a required part is not available, or even when work takes longer than planned. When the Andon cord is pulled, the team leader is alerted and immediately works to resolve the problem. If the problem cannot be resolved within a specified time (e.g., fifty-five seconds), the production line is stopped so that the entire organisation can be mobilised to assist with problem resolution until a successful countermeasure has been developed [Kim 2016].
Anomaly detection techniques	Not all data that needs to be monitored has a Gaussian (normal) distribution. The anomaly detection techniques make it possible to find noteworthy variances using a variety of methods for data that has no Gaussian distribution. These techniques are either used in monitoring tools or require people with statistical skills.
Anti-pattern	An anti-pattern is an example of the wrong interpretation of a pattern. The anti-pattern is often used to explain the value of the pattern.

Term	Meaning
Antifragility	This is the process of applying stress to increase resilience. This term is introduced by author and risk analyst Nassim Nicholas Taleb.
Artefact	An artefact is a product that is manufactured. Within DevOps the output of the commit phase are binaries, reports and meta data. These products are also referred to as artefacts.
Artefact repository	The central storage of artefacts is called the artefact repository. The artefact repository is used to managed artefacts and their dependencies.
Automated tests	Testcases should be automated as much as possible to reduce waste and to increase velocity and quality of the products that are to be delivered.
Bad apple theory	People that believe in the 'Bad Apple Theory' think that a system is basically safe if it were not for those few unreliable people in it. By removing these people, the system will be safe. This results in the anti DevOps pattern of 'name, blame, shame'.
Bad paths	A 'bad path' is a situation where the application does not follow the 'happy path' or 'the alternate' path. In other words, something goes wrong. This exception must be handled and should be monitorable.
Behavior Driven Development (BDD)	The development of software requires that the users are asked to define the (non) functional requirements. Behavior driven development is based on this concept. The difference however is that the acceptance criteria of these requirements should be written in the customer's expectation of the behavior of the application. This can be accomplished by formulating the acceptance criteria in the Given – When – Then format.
Binary	A compiler is used to transform source code to object code. The object code is also known as a binary. The source code is readable for human being, the object code however is only readable for computers since they have been written in hexadecimals.
Blameless post-mortem	Blameless post-mortem is a term coined by John Allspaw. It helps to examine "mistakes in a way that focuses on the situational aspects of a failure's mechanism and the decision-making process of individuals proximate to the failure." [Kim 2016].
Blamelessness	This approach is about learning rather than punishing. Within DevOps this is one of the basic ideas of learning from mistakes. The energy of the DevOps team is spending on learning from the mistake, rather than on finding the one to blame.
Blue-Green deployment pattern	Blue and green refer to two identical production systems. One is used for the final acceptance of a new release. If this acceptance is successful, then this environment becomes the new production environment. In case of a failure of the production system, the other system can be used instead. This mitigates the risk of downtime since the switchover is likely to be less than a second.
Broken build	A build that fails due to an error in the application source code.
Brown field	There are two scenarios' for applying DevOps best practices: green field and brown field. In case of a green field scenario the whole DevOps organisation has to be established from scratch. The opposite scenario is where there is already a DevOps organisation, but improvements are needed. The colour green refers to the situation that a factory is built on a clean grass field.

Term	Meaning
	The colour brown refers to the situation that a factory is to be built on a place where there has already been a factory that poisoned the ground. In order to build on a brown field, the poison needs to be removed.
Business value	Applying DevOps best practices results in increasing the business value. Research of Puppet Labs (State Of DevOps Report) proofs that high-performing organisations using DevOps practices are outperforming their non–high performing peers in many following areas [Kim 2016].
Canary releasing pattern	Normally a release is offered to every user at once. Canary releasing is the approach in which a small set of users is receiving the new release. If this small scope release works fine than the release can be deployed to all users. The term canary refers to the old habit to have a canary in the coal mines to detect toxic gas.
Change categories	Changes can be categorised into standard changes, normal changes and urgent changes.
Change schedules	Changes can be scheduled in order to defined in which order they have to be applied.
Cloud configuration files	Cloud configuration files are used to initiate a cloud service before using it. In this way cloud service providers enable customers to configure the cloud environment for their needs.
Cluster immune system release pattern	The cluster immune system expands upon the canary release pattern by linking our production monitoring system with our release process and by automating the roll back of code when the user-facing performance of the production system deviates outside of a predefined expected range, such as when the conversion rates for new users drops below our historical norms of 15%–20% [Kim 2016].
Code branch	See branching.
Code review methods	Code review can be performed in several ways like "over the shoulder", pair programming, email pass-around and tool-assisted code review.
Codified NFR	A list of Non-Functional Requirements (NFR) that are categorised in categories like availability, capacity, security, continuity et cetera.
Collaboration	One of the four pillars of DevOps is collaboration. Collaboration refers to the way the individuals of a DevOps team works together to achieve the common goal. There are many forms in which this collaboration comes to expression like: • peer to peer programming; • demonstrating weekly progress; • documentation; et cetera.
Commit code	Committing code is the action in which the DevOps engineer adds the changed source code to the repository, making these changes part of the head revision of the repository [Wiki].
Commit stage	This is the phase in the CI/CD secure pipeline where the source code is compiled to the object code. This includes the performance of the unit testcases.
Compliance checking	The manual action of a security officer to make sure that the system is built in accordance with the agreed standards.

Term	Meaning
	This is the opposite of security engineering where the DevOps teams works together with the security officer in order to embed the agreed standards in the deliverables and enable continuous monitoring of the standard in the whole lifecycle of the product.
Compliancy officer	The compliancy officer is a DevOps role. The compliancy officer is responsible for ensuring compliance with agreed standards throughout the whole life cycle of a product.
Configuration management	Configuration Management refers to the process by which all artefacts, and the relationships between them, are stored, retrieved, uniquely identified and modified.
Containers	A container is an isolated structure that is used by DevOps engineers to build their application independently from the underlying operating system or hardware. This is accomplished by interfaces in the container that are used by DevOps engineers. Instead of installing the application in an environment, the complete container is deployed. This saves a lot of dependencies and prevents configuration errors to occur.
Conway's law	The following statement of Melvin Conway is called the Conway's law: "organisations which design systems ... are constrained to produce designs which are copies of the communication structures of these organisations." [Wiki].
Cultural debt	There are three forms of debt. Cultural debt, technical debt and information debt. This form of debt refers to the decision to keep flaws in the organisation structure, hiring strategy, values et cetera. This debt costs interest and will result in less maturity growth of the DevOps teams. Cultural debt can be recognised by the exitance of extensive silos, workflow constraints, miscommunications, waste et cetera.
Culture, Automation Measurement, Sharing (CAMS)	CAMS is the abbreviation for Culture, Automation, Measurement and Sharing. • Culture: Culture relates to the people and process aspects of DevOps. Without the right culture, automation attempts will be fruitless. • Automation: Release management, configuration management, and monitoring and control tools should enable automation. • Measurement: 'If you can't measure it, you can't manage it.' & 'If you can't measure it, you can't improve it'. • Sharing: Culture of sharing ideas and problems is critical to help organisations to improve. Creates feedback loop.
Cycle time (flow time)	Cycle time measures more the completion rate or the work capability of a system overall, and a shorter cycle time means that less time is being wasted when a request has been made but no progress or work is getting done.
Cycle time (lean)	The average time between two successive units leaving the work or manufacturing process.
Declarative programming	This is a programming paradigm that expresses the logic of a computation without describing its control flow. An example are the database query languages for example TSQL and PSQL.

Term	Meaning
Defect tracking	Defect tracking is the process of tracking the logged defects in a product from beginning to closure and making new versions of the product that fix the defects [Wiki].
Development	Development is an activity that is performed by the DevOps role 'DevOps engineer'. A DevOps engineer is responsible for the complete lifecycle of a configuration item. Within DevOps there is no difference anymore between designer, builder or tester.
Development rituals	The Agile Scrum rituals of development are the sprint planning, daily stand-up, sprint execution, review and the retrospective.
Downward spiral	Gene Kim explains in his book [Kim 2016] that the downward spiral in Information Technology (IT) has three acts. • The first act begins in IT Operations where technical debt results in jeopardising our most important organisational promises. • The second act starts with compensating the latest broken promise by promising a bigger, bolder feature or an even larger revenue target. As a result, Development is tasked with another urgent project which results in even more technical debt. • The third stage is where the deployments are getting slower and slower, and outages are increasing. The business value continuously decreases.
E-mail pass-around	E-mail pass-around is a review technique where the source code management system emails code to reviewers automatically after the code is checked in [Kim 2016].
Error path	See happy path.
Fast feedback	Fast feedback refers to the second way of the three ways of Gene Kim. The second way is about having feedback on the functionality and quality of the product that is created or modified as soon as possible in order to maximise the business value.
Feature toggles	A feature toggle is a mechanism that makes it possible to enable or disable a part of the functionality of an application released in production. Feature toggles enables testing the effect of changes on users in production. Feature Toggles are also referred to as Feature Flags, Feature Bits or Feature Flippers.
Feedback	Feedback within the context of DevOps is the mechanism by which errors in the value stream are detected as soon as possible and is used to improve the product and if necessary to improve the value stream as well.
Feedforward	Feedforward within the context of DevOps is the mechanism by which experiences in the present value stream are used to improve the future value stream. Feed forward is the opposite of feedback since feedback is focused on the past and feed forward on the future.
Gaussian distribution	In probability theory, the normal (or Gaussian) distribution is a very common continuous probability distribution. Normal distributions are important in statistics and are often used in the natural and social sciences to represent real-valued random variables whose distributions are not known. A random variable with a Gaussian distribution is said to be normally distributed and is called a normal deviate [Wiki].

Term	Meaning
Given-When-Then	The Given-When-Then format is used to define acceptance criteria in a way that the stakeholders understand how the functionality actually will work. GIVEN – the fact that… WHEN – I do this… THEN – this happens…
Green field	See brown field.
Hand-off Readiness Review (HRR)	The HRR term is introduced by Google. An HRR is set of safety checks for a critical stage of releasing new services. HRR is performed when a service is transitioned from a developer-managed state to an OPS-managed state (usually months after the LRR). HRR makes service transition easier and more predictable and helps create empathy between upstream and downstream work centers.
Happy path	An application supports a business process by receiving, editing, storing and providing information. The assumed steps in which the information processing is performed is called the happy path. The steps in alternate ways are called the alternate path. In that case, the same result will be achieved via another navigation path. The crawl of the application that causes an error is called an error path.
Holocracy	In this type of organisation all decisions are made through self-organising teams rather than through a traditional management hierarchy.
Horizontal splitting of features	A feature can be splitted into stories. Horizontal splitting refers to the result of a feature splitting in which more DevOps teams must work tightly together. They have to align their work continuously in order to deliver together the feature.
I-shaped, T-shaped, E-shaped	I-shaped, T-shaped, E-shaped are the categories to indicate the knowledge and special skills of a person. An I-shaped person is a pure specialist in one area. The T-shaped person has special skills in one field and broad general knowledge. The E-shaped person has special skills in more than one field and broad general knowledge.
Idempotent	Continuous delivery requires that a component can always to be brought fully automatically to the desired status regardless of the component's initial state and regardless of the number of times the component is configured. The characteristic of a component to always be able to get back into the desires is called idempotent.
Imperative programming	This is a programming paradigm that uses statements that change a program's state. Imperative programming focuses on how a program should operate and consists of commands for the computer to perform. Examples are COBOL, C, BASIC et cetera. The term is often used in contrast to declarative programming, which focuses on what the program should accomplish without specifying how the program should achieve the result.
Independent, Negotiable, Valuable, Estimable, Small, and Testable (INVEST)	Independent, Negotiable, Valuable, Estimable, Small, and Testable. • Independent: The product backlog item should be self-contained, in a way that there is no inherent dependency on another product backlog item. • Negotiable: Product backlog items, up until they are part of an iteration, can always be changed, rewritten or even discarded. • Valuable: Product backlog item must deliver value to the stakeholders.

Term	Meaning
	• Estimable: The size of a product backlog item must always estimable. • Small: Product backlog items should not be so big as to become impossible to plan / task / prioritise with a certain level of certainty. • Testable: The product backlog item or its related description must provide the necessary information to make test development possible.
Information radiators	An Information Radiator is a visual display that a team places in a highly visible location so that all team members can see the latest information at a glance.
Infosec	A team that is responsible for securing systems and data.
Infrastructure as Code (IaC)	Normally infrastructure components have to be configured in order to perform the requested functionality and quality for example a rule set for a firewall or the allowed IP addresses for a network. These configurations normally are stored in configuration files which enable the operators to manage the functionality and the quality of the infrastructure components. Infrastructure as code (IaC) makes it possible to programme these infrastructure component settings and deploy these settings through the CI/CD secure pipeline by the use of machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.
Infrastructure as Code (IaC)	Infrastructure as code (IaC) is a software-based approach to the ICT infrastructure, whereby the systems can be rolled out and adapted in a consistent manner through templates. If a change has to be made, it is implemented in the template which is then rolled out again.
Infrastructure management	Infrastructure management consists of the lifecycle management of all infrastructure products and services in order to support the correct working of the applications that run on top of the infrastructure.
Ji-Kotei-Kanketsu (JKK)	JKK which means 100% completion of an item. This quality way of working means: • clear understanding of the goals; • understanding the right way to work; • ensure high quality of work; • getting the work right for 100% completion, never pass defects to the next process; • Definition of Done (DoD) is vital; and then maintaining the required quality without inspections.
Just In Time (JIT)	JIT means building up a stream-lined supply chain with one-piece flow.
Kaizen	Kaizen is Japanese for “improvement”. Kaizen is used to improve production systems. The goals of kaizen are: • elimination of waste (muda's); • JIT; • standardisation of production; • cycle of continuous improvements. Continuous improvement means circulate the Plan-Do-Check-Act (PDCA) cycle daily, weekly. This can be accomplished by finding the root cause of a failure by asking "Why" 5 times. The following steps can be followed: • defining problems with supporting data; • making sure everybody recognises the problems clearly;

Term	Meaning
	• setting a hypothesis on the problems found; • defining countermeasure actions to verify the hypothesis; • defining countermeasure actions be in daily based activities; • measuring a weekly KPI so people can feel a sense of accomplishment.
Kaizen Blitz (or Improvement Blitz)	A Kaizen Blitz is a rapid improvement workshop designed to produce results / approaches to discrete process issues within a few days. It is a way for teams to carry out structured, but creative problem solving and process improvement, in a workshop environment, over a short timescale.
Kaizen in advance	Kaizen in advance goes one step further than Kaizen. Not only the own activities are improved but also the activities that are performed upstream and that lead to problems downstream. In this way a feedback loop of problems is created which improves the system as a whole.
Kanban	This is system to signal when something is needed. Kanban is a system for managing the logistics production chain. Kanban was developed by Taiichi Ohno, at Toyota, to find a system that made it possible to achieve a high level of production. Kanban is often used for application management. One of the characteristics of Kanban is that it is pull oriented which means that there is not stock of material to be used during the production. Kanban can be used to implement JIT in production systems.
Kata	A kata is any structured way of thinking and acting (pattern of behavior) that is practiced until the pattern becomes a second nature. Four steps can be recognised to accomplish this second nature: • direction (target); • current condition (IST situation); • target condition (SOLL situation); • PDCA (Deming wheel). From an architectural viewpoint the migration path might be added to Kata as well. The migration path shows the way to go in order to achieve the SOLL situation.
Kibana dashboards	A Kibana dashboard displays a collection of saved visualisations.
Latent defects	Problems that are not visible yet. Latent defects can be made visible by injecting faults into the system.
Launch Readiness Review (LRR)	The LRR term is introduced by Google. An LRR is a set of safety checks for a critical stage of releasing new services. It is performed and signed off before a service is made publicly available and receive live production traffic. LRR is self-reported by the project teams. LRR is used in the development-managed state.
Launching guidance	To prevent the possibility of problematic, self-managed services going into production and creating organisational risk, launch requirements may be defined that must be met in order for services to interact with real customers and be exposed to real production traffic [Kim 2016].
Lead Time (LT)	Lead time is the time from when a request is made to when the final result is delivered, or the customer's point of view on how long something takes to complete.
Lean tools	• A3 thinking (problem solving) • Continuous flow (eliminates waste)

Term	Meaning
	• Kaizen • Kanban • KPI (Key Performance Indicator) • Plan Do Check Act (PDCA) • Root cause analysis • Specific, Measurable, Accountable, Realistic, Timely (SMART) • Value stream mapping (depict the flow) • JKK (No defects are passed to next process)
Learning culture	A learning culture is a collection of organisational conventions, values, practices and processes. These conventions encourage employees and organisations to develop knowledge and competence. An organisation with a learning culture encourages continuous learning and believes that systems influence each other. Since constant learning elevates an individual as a worker and as a person, it opens opportunities for the establishment to transform continuously for the better.
Light weight ITSM	This variant of Information Technology (IT) Service Management (ITSM) is strictly focused on business continuity with a set of Minimum Required Information (MRIs). The MRI set for each organisation depends on their business.
Logging levels	Within monitoring systems there are several levels of logging recognised: • Debug level: Information at this level is about anything that happens in the program, most often used during debugging. • Info level: Information at this level consists of actions that are user-driven or system specific. • Warn level: Information at this level tells us of conditions that could potentially become an error. • Error level: Information at this level focuses on error conditions Fatal level: Information at this level tells us when we must terminate.
Loosely coupled architecture	Loosely coupled architectures enables that changes can be made safely and with more autonomy, increasing developer productivity.
Micro service	Microservices are a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services. In a microservices architecture, services should be fine-grained, and the protocols should be lightweight [Wiki].
Micro service architecture	This architecture consists of a collection of services where each service provides a small amount of functionality, and the total functionality of the system is derived from composing multiple versions of a service in production simultaneously and to roll back to a prior version relatively easily.
Mini pipeline	In rare cases more than one deployment pipeline is required in order to produce the entire application. This can be accomplished by the use of a pipeline per application component. All these components are then assembled in a central pipeline which puts the entire application through acceptance tests, non-functional tests, and then deploys the entire application to testing, staging, and production environments.

Term	Meaning
Monitoring Framework	A framework of components that together form a monitor facility that is capable to monitor business logic, applications, and operating systems. Events, logs and measures are routed by the event router to destinations [Kim 2016].
Monolithic	A monolithic architecture is the traditional programming model, which means that elements of a software program are interwoven and interdependent. That model contrasts with more recent modular approaches such as a micro service architecture (MSA).
MTTR	Mean Time To Repair (MTTR) is a basic measure of the maintainability of repairable items. It represents the average time required to repair a failed component or device.
Muda	This is a Japanese word for waste. It is used in relationship to production systems.
Non-Functional Requirement (NFR)	NFR are requirements that define the quality of a product like maintainability, manageability, scalability, reliability, testability, deploy ability and security. NFR are also referred to as operational requirements.
Non-Functional Requirement (NFR) testing	NFR testing is the testing aspect that focusses on the quality of the product.
Obeya	Obeya is a war room which serves two purposes: • information management; • and on-the-spot decision making.
One piece flow	The Lean approach means that the DevOps team only works at one item at a time as a team with a fast pace and smooth flow. This is also used in the first way of the three ways of Gene Kim.
Operations	Operations is the team often responsible for maintaining the production environment and helping to ensure that required service levels are met [Kim 2016].
Operations stories	The work that has to be done by Ops can be written in stories. In that way that can be prioritised and managed.
OPS liaison	An OPS liaison is an operation employee who is assigned to a development team in order to facilitate the development team for their infrastructural demands.
Organisation archetypes	There are three organisation archetypes: functional, matrix, and market. They are defined by Dr. Roberto Fernandez as follows: • Functional: Functional-oriented organisations optimise for expertise, division of labour, or reducing cost. • Matrix: Matrix-oriented organisations attempt to combine functional and market orientation. • Market: Market-oriented organisations optimise for responding quickly to customer needs.
Organisational typology model	This a model of Dr. Ron Westrum in which he defined three types of culture: 'pathological', 'bureaucratic', 'generative'. These organisation types can be recognised by the following characteristics: • Pathological organisations are characterised by large amounts of fear and threat. • Bureaucratic organisations are characterised by rules and processes. • Generative organisations are characterised by actively seeking and sharing information to better enable the organisation to achieve its mission.

Term	Meaning
	Dr. Westrum observed that in healthcare organisations, the presence of "generative" cultures was one of the top predictors of patient safety.
Over-the-shoulder	This is a review technique where the author walks through his code while another developer gives feedback.
Packages	A set of individual files or resources which are packed together as a software collection that provides certain functionality as part of a larger system.
Pair-programming	This is review technique where two developers work together using one computer. While one developer writes the code the other reviews it. After one hour they exchange their role.
Peer review	This is a review technique where developers review each other's code.
Post-mortems	After a major incident a post-mortem meeting can be organised in order to find out what the root-cause is of the incident and how to prevent it in the future.
Product owner	The Product Owner is a DevOps role. The Product Owner is the internal voice of the business. The Product Owner is the owner of the product backlog and determines the priority of the product backlog items in order to define the next set of functionalities in the service.
Programming paradigm	A style of building the structure and elements of computer programs.
Pull request process	This is a form of peer review that span Dev and Ops. It is the mechanism that lets engineers tell others about changes they have pushed to a repository.
Quality Assurance (QA)	Quality Assurance (QA) is the team responsible for ensuring that feedback loops exist to ensure the service functions as desired [Kim 2016].
Reduce batch size	The size of a batch has an influence on the flow. Small batch sizes results in a smooth and fast flow. Large batch sizes results in high Work In Progress (WIP) and increases the level of variability in flow.
Reduce number of handoffs	In terms of a software process a handoff means that the work that is performed in order to produce software is stopped and handed over to another team. Each time the work passes from one team to another team, this requires all sorts of communication using different tools and filling up queues of work. To less handoffs the better.
Release managers	This a DevOps role. The release manager is responsible for managing and coordinating the production deployment and release processes.
Release patterns	There are two patterns of releases to be recognised [Kim 2016]: • Environment-based release patterns: In this pattern there are two or more environments that receive deployments, but only one environment is receiving live customer traffic. • Application-based release patterns: In this pattern the application is modified in order to make selectively releases possible and to expose specific application functionality by small configuration changes.
Sad path	A specific type of a 'bad path' is called a 'sad path'. This is the case if the 'bad path' results in a security-related error condition.
Safety checks	Safety checks are performed during a release of a product. They are typical part of an HRR of an LRR.

Term	Meaning
SBAR	This technique offers guidelines for making sure concerns or critiques are expressed in a productive manner. In this situation the people who concerns it have to follow the following steps: • situational information to describe what is happening; • background information or context; • an assessment of what they believe the problem is; • recommendations for how to proceed.
Security testing	Security testing is one of many types of tests. Within DevOps security testing is integrated in the deployment pipeline by using automated tests as early as possible in the flow.
Self service capability	One way of integrating Ops in Dev is the usage of infrastructure self-services.
Shared goals	Delivering value to the customer requires that Dev and Ops are working together in value streams and have shared goals and practices.
Shared Operations Team (SOT)	A SOT is a team that is responsible for managing all the DTAP environments performing daily deployments into those development and test environments, as well as doing periodically production deployments. The reason to use a SOT is to have a team that focusses only on deployments. This results in automation of repeatable work and learning how to fix occurring problems very fast.
Shared version control repository	In order to be able to use trunk-based development DevOps engineers need to share their source code. The source code must be committed into a single repository that also supports version control. Such a repository is called a shared version control repository.
Simian army	Simian Army consists of services (Monkeys) for generating various kinds of failures, detecting abnormal conditions, and testing the ability to survive them. The goal is to keep the cloud service safe, secure, and highly available. Currently there are 3 Monkeys in the Simian Army: • Janitor Monkey (unused resources); • Chaos Monkey (try to shut down a service); • Conformity Monkey (non-conformance to rules).
Single repository	A single repository is used to facilitate trunk-based development.
Smoke testing	Smoke testing is one of the test types that is used to determine whether or not the basics of a new or adjusted service works. Only a few testcases are needed to indicate whether or not at least the most important functions are working properly. This test type origins from the hardware manufacturers where engineers tested circuits by powering on the system and checking for smoke which was an alarm of malfunctioning hardware.
Standard deviation	In statistics, the standard deviation (SD, also represented by the Greek letter sigma σ or the Latin letter s) is a measure that is used to quantify the amount of variation or dispersion of a set of data values. A low standard deviation indicates that the data points tend to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the data points are spread out over a wider range of values [Wiki].
Standard operations	The standard operations is the situation in which the system performs as designed. Deviations of the standard operations need to be detected as early as possible.

Term	Meaning
Static analysis	Static analysis is a type of testing that is performed in a non-runtime environment, ideally in the deployment pipeline. Typically, a static analysis tool will inspect program code for all possible run-time behaviours and seek out coding flaws, back doors, and potentially malicious code [Kim 2016].
Swarming	David Bernstein explains how swarming helps to build an effective team which is able to focus and solve complex problems: ”When swarming, the whole team works together on the same problem. It helps to know each other and work well together. Generally, groups need to go through the phases of forming (getting to know each other) and storming (having conflicts and resolving them) before they get to performing (being a highly functional team), so give everyone the space to become a team.”. According to Dr. Spear, the goal of swarming is to contain problems before they have a chance to spread, and to diagnose and treat the problem so that it cannot recur. “In doing so,” he says, “they build ever-deeper knowledge about how to manage the systems for doing our work, converting inevitable up-front ignorance into knowledge.” [Kim 2016].
System of Engagement (SoE)	SoE’s are decentralised Information Communication Technology (ICT) components that incorporate communication technologies such as social media to encourage and enable peer interaction [What-is].
System of Information (SoI)	The term SOI includes are all the tools that are used to process and visualise information from SoR systems. Typically, examples are Business Intelligence (BI) systems.
System of Records (SoR)	A SoR is an ISRS (information storage and retrieval system), that is the authoritative source for a particular data element in a system containing multiple sources of the same element. To ensure data integrity, there must be one -- and only one -- system of record for a given piece of information [What-is].
Technology adaption curve	It takes time for new technology to get adapted in the market. The technology adaption curve indicates the stages of market penetration in time.
Technology executives	This is a DevOps role also named ‘value stream manager’. The value stream manager is someone who is responsible for “ensuring that the value stream meets or exceeds the customer (and organisational) requirements for the overall value stream, from start to finish” [Kim 2016].
Test Driven Development (TDD)	Test driven development is the approach in which the source code is written after the completion of the test case definition and execution. The source code is written and adjusted until the test case conditions are met.
Test harness	Software constructed to facilitate integration testing. Where test stubs are typically components of the application under development and are replaced by working components as the application is developed (top-down integration testing), test harnesses are external to the application being tested and simulate services or functionality not available in a test environment.
The Agile Manifesto	The Agile Manifesto (Manifesto for Agile Software Development) was set up during an informal meeting of seventeen software DevOps engineers. This meeting took place from 11 to 13 February 2001 at "The Lodge" in Snowbird, Utah.

Term	Meaning
	The charter and the principles formed an elaboration of ideas that had arisen in the mid-nineties, in response to methods traditionally classed as waterfall development models. Those models were experienced as bureaucratic, slow, and narrow-minded and would hinder the creativity and effectiveness of DevOps engineers. The seventeen people who have drawn up the Agile Manifesto together represented the various Agile movements. After the publication of the charter, several signatories set up the "Agile Alliance" to further convert the principles into methods [Wiki].
The ideal testing automation pyramid	The ideal testing automation pyramid is a way of testing that can be characterised as follows: • Most of the errors are found using unit tests as early as possible. • Run faster-running automated tests (e.g., unit tests) before slower-running automated tests (e.g., acceptance and integration tests), which are both run before any manual testing. • Any errors should be found with the fastest possible category of testing.
The Lean movement	An operating philosophy that stresses listening to the customer, tight collaboration between management and production staff, eliminating waste and boosting production flow. Lean is often heralded as manufacturers' best hope for cutting costs and regaining their innovative edge.
The non-ideal testing automation inverted pyramid	The non-ideal testing automation pyramid is a way of testing that can be characterised as follows: • Most of the investment is in manual and integration testing. • Errors are found later in the testing. • Slower running automated tests are performed first.
The Simian Army	The Simian Army is a collection of open-source cloud testing tools created by the online video streaming company, Netflix. The tools allow engineers to test the reliability, security, resiliency and recoverability of the cloud services that Netflix runs on Amazon Web Services (AWS) infrastructure [Whatis]. Within this Simian Army the following monkeys are recognised: Chaos Gorilla, Chaos Kong, Conformity Monkey, Doctor Monkey, Janitor Monkey, Latency Monkey and Security Monkey.
The three ways	The three ways are introduced in 'The Phoenix Project: A Novel About IT, DevOps, And Helping Your Business Win' by Gene Kim, Kevin Behr and George Spafford. The Three Ways are an effective way to frame the processes, procedures and practices of DevOps, as well as the prescriptive steps. • The first way – flow understand and increase the flow of work (left to right); • The second way – feedback create short feedback loops that enable continuous improvement (right to left); • The third way – Continuous Experimentation and Learning (continuous learning).
Theory of constraints	This is a methodology for identifying the most important limiting factor that stands in the way of achieving a goal and then systematically improving that constraint until it is no longer the limiting factor.
Tool-assisted code review	This is a review technique where authors and reviewers use specialised tools designed for peer code review or facilities provided by the source code repositories [Kim 2016].

Term	Meaning
Toyota Kata	Toyota Kata is a management book by Mike Rother. The book explains the Improvement Kata and Coaching Kata, which are a means for making the Continual improvement process as observed at the Toyota Production System teachable [Wiki].
Transformation team	Introducing DevOps requires a defined transformation strategy. Based on their research, Dr. Govindarajan and Dr. Trimble assert that organisations need to create a dedicated transformation team that is able to operate outside of the rest of the organisation that is responsible for daily operations (which they call respectively the "dedicated team" and "performance engine"). The lessons learned from this transformation team can be used to apply in the rest of the organisation.
Value stream	The process required to convert a business hypothesis into a technology-enabled service that delivers value to the customer [Kim 2016].
Value Stream Mapping (VSM)	Value stream mapping is a Lean tool that depicts the flow of information, materials, and work across functional silos with an emphasis on quantifying waste, including time and quality.
Vertical splitting of features	A feature can be splitted into stories. Vertical splitting refers to the result of a feature splitting in which more DevOps teams can work independently on their own stories. Together they realise the feature. See also Horizontal splitting of features.
Virtualised environment	An environment that is based on virtualisation of hardware platforms, storage devices and network resources. In order to create a virtualised environment usually VMware is used.
Visualisation	In computing, virtualisation refers to the act of creating a virtual (rather than actual) version of something, including virtual computer hardware platforms, storage devices, and computer network resources. Virtualisation began in the 1960s, as a method of logically dividing the system resources provided by mainframe computers between different applications. Since then, the meaning of the term has broadened [Wiki].
Walking skeleton	Walking skeleton means doing the smallest possible amount of work to get all the key elements in place.
Waste	Waste comprises the activities that are performed in the manufacturing process that are not adding value to the customer. Examples in the context of DevOps are: • Unnecessary software features. • Communication delays. • Slow application response times. • Overbearing bureaucratic processes.
Waste reduction	Minimisation of waste at its source is to minimise the quantity required to be treated and disposed of, achieved usually through better product design and/or process management. Also called waste minimisation [Businessdictionary].
WIP limit	This is a Key Performance Indicator (KPI) that is used in the Kanban process to maximise the number of items that has been started but that is not completed. Limiting the amount of WIP is an excellent way to increase throughput in your software development pipeline.
Work In Progress (WIP)	Material that has entered the production process but is not yet a finished product.

Term	Meaning
	Work in progress (WIP) therefore refers to all materials and partly finished products that are at various stages of the production process.

Table B-1, Glossary.

Appendix C, Abbreviations

Abbreviation	Meaning
%C/A	Percent Complete / Accurate
AWS	Amazon Web Services
BDD	Behavior Driven Development
BI	Business Intelligence
BOK	Body of Knowledge
BSC	Balanced Score Card
BVS	Business Value System
CA	Competitive Advantage
CA	Continuous Auditing
CAB	Change Advisory Board
CAMS	Culture, Automation, Measurement and Sharing
CD	Continuous Deployment
CE	Continuous Everything
CEM	Central Event Monitor
CEMLI	Configuration, Extension, Modification, Localisation, Integration
CEO	Chief Executive Officer
CFO	Chief Finance Officer
CI	Configuration Item
CI	Continuous Integration
CIA	Confidentiality, Integrity & Accessibility (or Availability)
CIO	Chief Information Officer
CL	Continuous Learning
CM	Continuous Monitoring
CMDB	Configuration Management DataBase
CMMI	Capability Maturity Model Integration
CMS	Configuration Management System
CN	Continuous desigN
CO	Continuous dOcumentation
CoC	Code of Conduct
CoP	Communities of Practice
CP	Continuous Planning
CPU	Central Processing Unit
CR	Competitive Response
CRAMM	CCTA Risk Assessment Method Methodology
CRC	Cyclic Redundancy Check
CS	Continuous aSsessment
CSF	Critical Success Factor
CT	Continuous Testing
CTO	Chief Technical Officer
CY	Continuous securitY
DevOps	Development & Operations
DML	Definitive Media Library

Abbreviation	Meaning
DNS	Domain Name System
DoD	Definition of Done
DoR	Definition of Ready
DTAP	Development, Test, Acceptance and Production
DU	Definitional Uncertainty
DVS	Development Value System
E2E	End-to-End
ERD	Entity Relation Diagram
ERP	Enterprise Resource Planning
ESA	Epic Solution Approach
ESB	Enterprise Service Bus
ETL	Extract Transform & Load
EUX	End User eXperience Monitoring
FAT	Functionele AcceptatieTest
FSA	Feature Solution Approach
GCC	General Computer Controls
GDPR	General Data Protection Regulation
GIT	Global Information Tracker
GSA	Generic & Specific Acceptatiecriteria
GUI	Graphical User Interface
GWT	Given-When-Then
HRM	Human Resource Management
HRR	Hand-off Readiness Review
IaC	Infrastructure as Code
ICT	Information Communication Technology
ID	Identifier
INVEST	Independent, Negotiable, Valuable, Estimatable, Small and Testable
IPOPS	Information assets, People, Organisation, Products, and services, Systems and processes
IR	Infrastructure Risk
ISAE	International Standard On Assurance Engagements
ISMS	Information Security Management System
ISO	Information Standardisation Organisation
ISVS	Information Security Value System
IT	Information Technology
ITIL	Information Technology Infrastructure Library
ITSM	Information Technology Service Management
JIT	Just In Time
JKK	Ji-Kotei-Kanketsu
JVM	Java Virtual Machine
KPI	Key Performance Indicator
LAN	Local Area Network
LCM	LifeCycle Management
LDAP	Lightweight Directory Access Protocol

Abbreviation	Meaning
LRR	Launch Readiness Review
LT	Lead Time
MASR	Modify, Avoid, Share, Retain
MFA	Multi Factor Authentication
MI	Management Information
MOF	Microsoft Operations Framework
MRI	Minimum Required Information
MT	Module Test
MTBF	Mean Time Between Failure
MTBSI	Mean Time Between System Incidents
MTTR	Mean Time To Repair
MVP	Minimal Viable Product
NC	Non-Conformity
NFR	Non-Functional Requirement
OAWOW	One Agile Way of Working
OLA	Operational Level Agreement
PAAS	Platform As A Service
PAT	Production Acceptance Test
PBI	Productie Backlog Item
PDCA	Plan Do Check Act
PESTLE	Political, Economic, Sociological, Technological, Legislative, Environmental
POR	Project or Organisational Risk
PPT	People, Process & Technology
PST	Performance StressTest
PT	Processing Time
QA	Quality Assurance
QC	Quality Control
RACI	Responsibility, Accountable, Consulted, and Informed
RASCI	Responsibility, Accountable, Supporting, Consulted and Informed
RBAC	Role Based Access Control
REST API	REpresentational State Transfer Application Programming Interface
ROI	Return On Investment
RUM	Real User Monitoring
S-CI	Software Configuration Item
SA	Strategic IS Architecture
SAFe	Scaled Agile Framework
SAT	Security AcceptatieTest
SBAR	Situation, Background, Assessment, Recommendation
SBB	System Building Block
SBB-A	System Building Block Application
SBB-I	System Building Block Information
SBB-T	System Building Block Technology

Abbreviation	Meaning
SIT	System Integration test
SLA	Service Level Agreement
SM	Strategic Match
SMART	Specific, Measurable, Accountable, Realistic, Timely
SME	Subject Matter Expert
SNMP	Simple Network Management Protocol
SoA	Statement of Applicability
SoE	System of Engagement
SoI	Systems of Information
SoR	System of Records
SoX	Sarbanes Oxley
SQL	Structured Query Language
SRG	Standards Rules & Guidelines
SSL	Secure Sockets Layer
ST	System test
SVS	Service Value System
TCO	Total Cost of Ownership
TCP	Transmission Control Protocol
TDD	Test Driven Development
TFS	Team Foundation Server
TISO	Technical Information Security Officer
TOM	Target Operating Model
TPS	Toyota Production System
TTM	Time To Market
TU	Technical Uncertainty
UAT	User Acceptance Test
UML	Unified Modeling Language
UT	Unit Testing
UX design	User eXperience Design
VOIP	Voice over Internet Protocol
VSM	Value Stream Mapping
WAN	Wide Area Network
WIP	Work In Progress
WMI	Windows Management Instrumentation
WoW	Way of Working
XML	eXtensible Markup Language
XP	eXtreme Programming

Table C-1, Abbreviations.

Appendix D, Websites

bigpanda	[Bigpanda]	https://www.bigpanda.io/blog/event-correlation/
Bullseye	[Bullseye]	https://www.bullseye.com/minimum.html
Businessdictionary	[Businessdictionary]	http://www.businessdictionary.com
Collabnet	[CollabNet]	https://www.collab.net
CleanArchitecture	[CleanArchitecture]	https://www.freecodecamp.org/news/a-quick-introduction-to-clean-architecture-990c014448d2/
CleanCode	[CleanCode]	https://cvuorinen.net/2014/04/what-is-clean-code-and-why-should-you-care/
dbmetrics	[dbmetrics]	http://www.dbmetrics.nl
dbmetrics	[dbmetrics publicaties]	https://www.dbmetrics.nl/wp-content/uploads/2021/07/dbmetrics_best-practice-publications_2021-07-22_900.pdf
De Caluwé	[De Caluwé]	https://www.agile4all.nl/het-kleurenmodel-van-de-caluwe-en-vermaak/
DevOps	[DevOps]	http://DevOps.com
DDD	[DDD]	https://www.slideshare.net/skillsmatter/ddd-in-agile
doxygen	[doxygen]	http://www.doxygen.nl/manual/docblocks.html
doxygen example	[doxygen example]	http://www.doxygen.nl/manual/examples/qtstyle/html/class_q_tstyle___test.html#a0525f798cda415a94fedeceb806d2c49
EXIN	[Exin]	http://www.exin.nl
Gladwell	[GLADWELL]	http://www.gladwill.nl
IIR	[IIR]	http://www.IIR.nl
Investopedia	[Investopedia]	https://www.investopedia.com
ITMG	[ITMG]	http://www.ITMG.nl
ITPedia	[ITPEDIA]	http://www.itpedia.nl
Patrick Cousot	[Patrick Cousot]	https://www.di.ens.fr/~cousot/abstract_interpret.shtml
Porter	[Porter]	https://medium.com/@sniloy/value-chain-analysis-value-stream-mapping-and-business-process-mapping-what-is-the-difference-431589d27ea8
Sneider	[Schneider]	https://shift314.com/are-you-using-the-right-culture-model/
Tiobe	[Tiobe]	www.tiobe.com/content/paperinfo/DefinitionOfConfidenceFactor.html
UnitTest	[UnitTest]	https://docs.python.org/3/library/unit test.html
Westrum	[Westrum]	https://www.delta-n.nl/het-belang-van-cultuur-in-devops/
Wiki	[Wiki]	http://nl.wikipedia.org / wiki / Cloud_computing
Wiki docgen	[Wiki docgen]	https://en.wikipedia.org/wiki/Comparison_of_documentation_generators

Table D-1, Websites.

Appendix E, Index

%

%C/A · 119

A

A/B testing · 103
acceptatiecriterium · 104, 108
acceptatietest · 103
actor · 28, 29, 30, 42, 61, 62, 63
added value · 24, 33, 41, 43, 56, 60, 61, 67
affinity · 103
Agile · 103, 115, 116
Agile infrastructure · 103
Agile project · 13
Agile Scrum · 16, 17, 107
Agile Scrum framework · 16
alternate path · 103
Amazon Web Services · See AWS
Andon cord · 103
anomaly detection technique · 103
antifragility · 104
anti-pattern · 7, 14, 15, 17, 18, 19, 20, 103
applicatiebeheer · 110
applicatiecomponent · 111
application architecture · 51
application business rule · 77
architecture model · 4, 14, 19, 23, 25, 33, 41, 49, 78
architecture principle · 23, 25
artefact · 104, 106
artefact repository · 104
assessment · 29, 114
automated test · 104
availability · 105
AWS · 119

B

backlog item · 113
bad apple theory · 104
bad path · 104
Balanced Score Card · See BSC
Balanced scorecard · 91
BDD · 4, 26, 67, 68, 69, 71, 72, 76, 77, 78, 90, 94, 104, 119
Behavior Driven Development · See BDD
best practice · 105
BI · 119
binary · 104
black box · 54
blameless post mortem · 104
blamelessness · 104
blue/green deployment · 104
Body of Knowledge · See BOK
BOK · 119
bottleneck · 37, 57
boundary · 8, 36, 38, 95
branching · 105
broken build · 104
brown field · 104
BSC · 119
build · 104, 105, 106, 115
business case · 11, 13, 37, 46, 85
Business Intelligence · See BI
business value · 105, 107
Business Value System · See BVS
business view · 7
BVS · 119

C

C/A · 37
CA · 119
CAB · 119
CAMS · 106, 119
canary releasing · 105
capability · 106
Capability Maturity Model Integration · See CMMI
capaciteit · 105
CCTA Risk Assessment Method Methodology · See CRAMM
CD · 92, 105, 109, 119
CE · 119
CEM · 119
CEMLI · 119
CE-model · 91
Central Event Monitor · See CEM
Central Processing Unit · See CPU
CEO · 119
CFO · 119
chain · 3, 16, 17, 52, 89, 92, 95
Change Advisory Board · See CAB
change category · 105
change paradigm · 4, 13, 14, 15, 18, 19, 20, 21, 23, 83, 85, 143
change schedule · 105
Chief Executive Officer · See CEO
Chief Finance Officer · See CFO
Chief Information Officer · See CIO
Chief Technology Officer · See CTO
CI · 92, 105, 109, 119
CI/CD secure pipeline · 16, 19
CIA · 119
CIO · 119
CL · 92, 119
cloud · 105
cloud configuration file · 105
cloud service · 105
cluster immune system release pattern · 105

CM · 92, 96, 119
CMDB · 119
CMMI · 93, 119
CMS · 119
CN · 119
CO · 92, 119
CoC · 119
code branch · 105
Code of Conduct · See CoC
code review form · 105
code view · 7
codified NFR · 105
collaboratie tool · 20
collaboration · 105
commit code · 105
commit stage · 105
Communities of Practice · See CoP
competence · 103, 108
Competitive Advantage · See CA
Competitive Response · See CR
Completeness / Accurateness · See %C/A
compliance · 106
compliance checking · 106
compliancy · 106
compliancy officer · 106
component · 109, 112, 115
Confidentiality, Integrity & Accessibility · See CIA
Configuration Item · See CI
configuration management · 106
Configuration Management DataBase · See CMDB
Configuration Management System · See CMS
Configuration, Extention, Modification, Localisation, Integration · See CEMLI
container · 106
continuity · 105, 111
Continuous
- Assessment · 2, 11, 28, 29
- Auditing · 2
- Deployment · 2
- Design · 2
- Design Pyramid · 26
- Everything · 92, 93
- Improvement · 116
- Integration · 2
- Learning · 2, 116
- Monitoring · 2
- Planning · 2
- Testing · 2

Continuous aSsessment · See CS
Continuous Auditing · See CA
Continuous Deployment · See CD
Continuous desigN · See CN
Continuous dOcumentation · See CO
Continuous Everything · See CE
Continuous Integration · See CI
Continuous Learning · See CL
Continuous Monitoring · See CM
Continuous Planning · See CP
Continuous securitY · See CY
Continuous Testing · See CT
control · 106, 114
Conway's law · 106
CoP · 16, 17, 119
counter measure · 110
CP · 119
CPU · 119
CR · 119
CRAMM · 119
CRC · 119
Critical Success Factor · See CSF
CS · 119
CSF · 119
CT · 91, 119
CTO · 119
cultural debt · 106
Culture, Automation, Measurement and Sharing · See CAMS
current state · 36
CY · 119
cycle time · 106
Cyclic Redundancy Check · See CRC

D

debt · 106
declarative programming · 106
defect · 111
defect tracking · 107
Definition of Done · See DoD
Definition of Ready · See DoR
Definitional Uncertainty · See DU
Definitive Media Library · See DML
Demming wheel · 110
deployment · 103
deployment pipeline · 53, 105
design · 106, 117, 122
design view · 7
Dev engineer · 87
development · 103, 104, 107, 109, 110, 112, 114, 115, 116, 117
Development & Operations · See DevOps
development ritual · 107
Development Value System · See DVS
Development, Test, Acceptance and Production · See DTAP
DevOps · 99, 100, 103, 105, 107, 113, 117, 119, 133
- engineer · 12, 15, 16, 17, 19, 30, 85, 103, 105, 106, 107, 114, 115, 116
- Lemniscate · 2
- team · 103, 104, 105, 106, 108, 112, 117

DML · 119
DNS · 120
DoD · 2, 9, 29, 30, 72, 77, 92, 109, 120
Domain Name System · See DNS
DoR · 120
downward spiral · 107
DTAP · 114, 120
DTAP environments · 114
DU · 120
DVS · 120

E

E2E · 120
e-mail pass around · 107
emerging design · 3, 12, 15, 24, 89
End User eXperience Monitoring · See EUX
End-to-End · See E2E
enterprise architect · 90
enterprise architecture · 9, 56
enterprise business rule · 77
Enterprise Resource Planning · See ERP
Enterprise Service Bus · See ESB
Entity Relation Diagram · See ERD
epic · 14, 26, 31, 41, 60, 67, 87, 88, 89, 91, 120
Epic Solution Approach · See ESA
ERD · 120
ERP · 16, 48, 120
error path · 107
ESA · 120
ESB · 120
E-shaped · 108
ETL · 120
EUX · 120
event · 112
eXtensible Markup Language · See XML
Extract Transform & Load · See ETL
eXtreme Programming · See XP

F

failure · 104
fast feedback · 13, 15, 24, 83, 91
FAT · 103, 120
feature · 26, 60, 62, 67, 68, 69, 70, 71, 76, 77, 78, 85, 87, 88, 89, 107, 108, 117
Feature Solution Approach · See FSA
feature toggle · 107
feature-design · 87
feedback · 106, 107, 110, 113, 116
feedforward · 107
flow · 106, 109, 110, 111, 112, 113, 114, 116, 117
framework · 112
FSA · 86, 120
Functional Acceptance Test · See FAT

G

Gaussian distribution · 103, 107
GCC · 120
GDPR · 2, 120
Gene Kim · 107, 112, 116
General Computer Controls · See GCC
General Data Protection Regulation · See GDPR
Generic & Specific Acceptatiecriteria · See GSA
Gherkin · 69
Gherkin language · 9, 18, 24, 25, 68, 69, 72, 77
GIT · 120
GITC · 2
Given When Then · 108, See GWT
Global Information Tracker · See GIT
golden plated · 20
governance · 136
Graphical User Interface · See GUI
green build · 92
green field · 108
GSA · 120
GUI · 120
GWT · 26, 29, 31, 72, 78, 79, 94, 108, 120

H

Hand-off Readiness Review · See HRR
happy path · 103, 104, 108
hardware · 106, 109, 117
holocracy · 108
horizontal splitting of feature · 108, 117
HRM · 2, 19, 20, 23, 120
HRR · 120
Human Resource Management · See HRM

I

IaC · 103, 109, 120
ICT · 109, 120
ID · 120
ideal test pyramid · 116
idempodent · 108
IDentifier · See ID
impactanalyse · 35
imparative programming · 108
incident · 84
incidents · 94
Independent, Negotiable, Valuable, Estimatable, Small and Testable · See INVEST
information architecture · 49
Information assets, People, Organisation, Products and services, Systems and processes · See IPOPS
Information Communication Technology · See ICT
information radiator · 109
Information Security Management System · See ISMS
Information Security Value System · See ISVS
Information Standardisation Organisation · See ISO
Information Technology · See IT
Information Technology Infrastructure Library · See ITIL
Information Technology Service Management · See ITSM
Infosec · 109
Infrastructure as Code · See IaC

infrastructure component · 109
infrastructure management · 109
Infrastructure Risk · See IR
International Standard On Assurance Engagements · See ISAE
INVEST · 108, 120
IP address · 109
IPOPS · 120
IR · 120
ISAE · 120
I-shaped · 108
ISMS · 120
ISO · 120
IST · 110
ISVS · 120
IT · 107, 111, 116, 120
ITIL · 120
ITSM · 111, 120

J

Java Virtual Machine · See JVM
Ji-Kotei-Kanketsu · See, See JKK
JIT · 109, 110, 120
JKK · 109, 120
Just In Time · See JIT
JVM · 120

K

Kaizen · 109, 111
Kaizen Blitz (or Improvement Blitz) · 110
Kaizen in advance · 110
Kanban · 110, 111, 117
Key Performance Indicator · See KPI
kibana dashboard · 110
knowledge transfer · 2, 12
KPI · 9, 96, 110, 111, 117, 120

L

LAN · 120
latent defect · 110
Launch Readiness Review · See LRR
launching guidance · 110
LCM · 120
LDAP · 120
Lead Time · 110, See LT
Lean · 116, 117
Lean tool · 110
learning culture · 111
lifecycle · 107, 109
LifeCycle Management · See LCM
Lightweight Directory Access Protocol · See LDAP
limitation · 24, 36
Local Area Network · See LAN
log · 112
logging level · 111
loosely coupled architecture · 111
loosely coupled services · 111
LRR · 110, 121
LT · 37, 110, 121

M

Management Information · See MI
manufacturing process · 117
marker · 30
MASR · 121
maturity · 8, 16, 17, 91, 93, 94, 96
Mean Time Between Failure · See MTBF
Mean Time Between System Incidents · See MTBSI
Mean Time To Repair · See MTTR
meta-data · 104
metadata · 14, 15, 20, 36, 41, 45, 61, 62, 78, 95
MFA · 121
MI · 121
microservice · 111
microservice architecture · 111
Microsoft Operations Framework · See MOF
migration path · 84
mini pipeline · 111
Minimal Viable Product · See MVP
Minimum Required Information · See MRI
mission · 91
Modify, Avoid, Share, Retain · *See* MASR
modulair programming · 77
Module Test · See MT
MOF · 121
monitoring · 112
monolithic · 112
MRI · 111, 121
MT · 121
MTBF · 121
MTBSI · 121
MTTR · 112, 121
muda · 112
Multi Factor Authentication · See MFA
MVP · 31, 41, 121

N

NC · 121
NFR · 105, 112, 121
Non Conformity · See NC
Non Functional Requirement · See NFR

O

OAWOW · 121
obeya · 112
object code · 104
Oirsouw · 100
OLA · 121
One Agile Way of Working · See OAWOW

one piece flow · 112
Operational Level Agreement · See OLA
operations · 103, 107, 112, 114, 117
operations story · 112
Ops engineer · 87
Ops liaison · 112
organisation archetype · 112
organisational typology model · 112
over-the-shoulder · 113

P

PAAS · 121
package · 113
pair programming · 105, 113
PAT · 103, 121
pattern · 103, 113
PBI · 121
PDCA · 110, 111, 121
peer review · 113
peer to peer programming · 105
People, Process & Technology · See PPT
performance · 105, 111, 117, 121
performance indicator · 37, 56, 57, 62
Performance StressTest · See PST
PESTLE · 121
PI · 16
pipeline · 103, 109, 111, 114, 115, 117
Plan Do Check Act · See PDCA
Platform As A Service · See PAAS
Political, Economic, Sociological, Technological, Legislative, Environmental · See PESTLE
POR · 121
post mortem · 113
post-condition · 61
power · 15, 17
PPT · 2, 11, 23, 47, 121
pre-condition · 61
Processing Time · See PT
product
- backlog · 14, 15, 17, 23, 26, 29, 38, 52, 60, 91, 113
- backlog item · 17, 109
- owner · 113
- vision · 91

Product Backlog Item · See PBI
Production AcceptatieTest · See PAT
production environment · 111
programming paradigm · 113
Project or Organisational Risk · See POR
PSQL · 106
PST · 121
PT · 37, 121
pull request process · 113
Python · 73, 79

Q

QA · 113, 121
QC · 121
Quality Assurance · See QA
Quality Control · See QC

R

RACI · 121
RASCI · 17, 23, 121
RBAC · 121
Real User Monitoring · See RUM
reduce batch size · 113
reduce number of handoffs · 113
release · 2, 113
release manager · 113
release pattern · 113
release plan · 91
repository · 104, 105, 113, 114
REpresentational State Transfer Application Programming Interface · See REST API
requirement · 68, 104, 110, 112, 115, 121
requirement view · 7
Responsibility, Accountable, Consulted and Informed · See RACI
Responsibility, Accountable, Supporting, Consulted and Informed · See RASCI
REST-API · 121
retrospective · 107
Return On Investment · See ROI
review · 107
risico · 104, 110
ROI · 121
Role-based access control · See RBAC
rootcause analyse · 111
RUM · 121

S

SA · 121
sad path · 113
SAFe · 121
SAFe framework · 17
safety check · 113
Sarbanes Oxley · See SoX
SAT · 121
SBAR · 114, 121
SBB · 29, 41, 46, 47, 48, 49, 50, 51, 52, 53, 54, 57, 62, 63, 64, 65, 69, 70, 121
SBB-A · 121
SBB-I · 62, 63, 64, 69, 70, 121
SBB-T · 29, 62, 121
Scaled Agile Framework · See SAFe
S-CI · 121
Secure Sockets Layer · See SSL
security · 105, 112, 113, 114, 116
Security Acceptance Test · See SAT
security officer · 105
self service capability · 114
service · 121
Service Level Agreement · See SLA
Service Value System · See SVS
shared goals · 114
silo · 18, 117

Simian army · 114, 116
Simple Network Management Protocol · See SNMP
SIT · 122
Situation, Background, Assessment, Recommendation · See SBAR
skills · 108
SLA · 122
SM · 122
SMART · 111, 122
SME · 17, 34, 122
smoke testing · 114
SNMP · 122
SoA · 122
SoE · 2, 3, 16, 115, 122
software · 104, 115, 117
software configuration item · 53, 62
Software Configuration Item · See S-CI
SoI · 2, 3, 115, 122
SOLL · 110
solution view · 7
SoR · 2, 3, 16, 20, 52, 115, 122
sourcecode · 104, 105, 107, 114, 115, 116
SoX · 2, 122
Specific, Measurable, Accountable, Realistic, Timely · See SMART
Spotify model · 16, 84
sprint · 107
- planning · 90
sprint execution · 107
sprint planning · 107
SQL · 122
SRG · 122
SSL · 122
ST · 122
stakeholder · 2, 4, 8, 9, 11, 23, 26, 30, 33, 34, 35, 36, 38, 42, 43, 59, 68, 76, 85, 87, 96, 108
standard deviation · 114
standard operations · 114
Standard Rules & Guidelines · See SRG
stand-up · 107
Statement of Applicability · See SoA
static analysis · 115
story · 20
Story · 59
Strategic IS Architecture · See SA
Strategic Match · See SM
strategy · 2, 91, 95
Structured Query Language · See SQL
Subject Matter Expert · See SME
SVS · 122
System Building Block · See SBB
System Building Block Application · See SBB-A
System Building Block Infrastructure · See SBB-I
System Building Block Technology · See SBB-T
System Context Diagram · 29, 33
System Integration Test · See SIT
System of Engagement · See SoE
System of Records · See SoR
System Test · See ST
Systems of Information · See SoI

T

taak · 103
Target Operating Model · See TOM
task · 109
TCO · 122
TCP · 122
TDD · 4, 9, 71, 72, 76, 77, 78, 90, 94, 115, 122
Team Foundation Server · See TFS
technical debt · 13, 17, 94, 106, 107
Technical Information Security Officer · See TISO
Technical Uncertainty · See TU
technology adaption curve · 115
technology executive · 115
test
- case · 72, 103, 104, 105
- harness · 115
- tool · 74
- view · 7
Test Driven Development · See TDD
tester · 107
TFS · 122
The Agile Manifesto · 115
the ideal testing automation pyramid · 116
The Lean movement · 116
the non-ideal testing automation inverted pyramid · 116
The Three Ways · 112, 116
theme · 14, 26, 36, 60, 67
theory of constraints · 116
Time To Market · See TTM
TISO · 122
TOM · 122
tool-assisted code review · 116
Total Cost of Ownership · See TCO
Toyota Kata · 117
Toyota Production System · See TPS
TPS · 122
traceability · 11, 14, 15, 30, 44, 92
trage feedback · 11
transformation team · 117
Transmission Control Protocol · See TCP
trunk · 114
T-shaped · 108
TSQL · 106
TTM · 122
TU · 122

U

UAT · 122
UML · 122
Unified Modeling Language · See UML
Unit Test · See UT
unit test case · 30
upfront design · 3

Use Case · 9, 14, 27, 28, 29, 31, 37, 41, 42, 43, 44, 52, 56, 57, 59, 60, 61, 62, 63, 64, 65, 66, 77, 88, 89, 95
- Diagram · 4, 9, 27, 29, 41, 45
- Narrative · 60
User Acceptance Test · See UAT
User eXperience Design · See UX design
UT · 73, 122
UX design · 122

V

value chain · 91
value stream · 27, 33, 36, 37, 41, 107, 111, 114, 115, 117, 122
- canvas · 29, 33, 35, 37
- mapping · 29
value stream canvas · 91, 95
Value Stream Canvas model · 4, 8
Value Stream Mapping · See VSM
velocity · 104
vertical splitting of feature · 117
virtualised environment · 117
vision · 4, 13, 14, 15, 83, 91
visualisatie · 117
Voice over Internet Protocol · See VOIP
VOIP · 122
VSM · 41, 117, 122

W

walking skeleton · 117
WAN · 122
war room · 112
waste · 104, 106, 109, 110, 112, 116, 117
waste reductie · 117
waterfall approach · 21, 22
Way of Working · See WoW
Westrum · 112, 113
Wide Area Network · See WAN
Windows Management Instrumentation · See WMI
WIP · 122
WMI · 122
Work In Progress · See WIP
workflow · 106
WoW · 13, 16, 18, 122

X

XML · 122
XP · 122

Epilogue

My experience is that the ideas I capture in an article or book continue to evolve. If you are going to work with a certain topic from this book in your own DevOps organisation, I advise you to contact me. Perhaps there are additional articles or experiences in this area that I can share with you. This also applies inversely proportionally. If you have any experiences that complement what is described in this book, I invite you to share them with me. You can reach me via my e-mail address bartb@dbmetrics.nl.

About the author

Drs. Ing. B. de Best RI has been working in ICT since 1985. He has mainly worked in the top 100 of Dutch business and government. He has held positions in all phases of system development, including operation and management, for 12 years. He then focused on the service management field. Currently, as a consultant, he fulfils all aspects of the knowledge lifecycle of service management, such as writing and providing training to ICT managers and service managers, advising management organisations in directing the management organisation, management design, improving management processes, outsourcing (parts of) the management organisation and reviewing and auditing management organisations. He graduated in management field at both HTS level and University level.

Other books by this author

Basiskennis IT

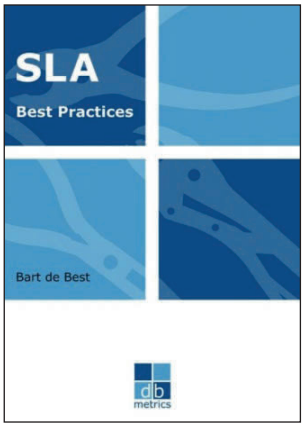
De eerste stap van een leven lang leren.

Het boek Basiskennis IT geeft een goede impressie wat dit vakgebied omvat. Zonder dat vele details worden besproken krijgt de lezer een uitleg van de meest essentiële begrippen en concepten van de IT. De doelgroep van dit boek zijn studenten, schoolverlaters en mensen die zich willen laten omscholen tot een beroep in de IT. Daartoe is het een heel nuttig middel als voorbereiding op IT trainingen.

De content bestaat uit het behandelen van IT begrippen uit vier perspectieven te weten het IT landschap, het ontwikkelen van software, het beheren van software en trends in de IT.

Hierbij worden tal van begrippen en concepten behandeld op het gebied van informatie, maatwerkprogrammatuur, systeemprogrammatuur, softwarepakketten, middleware, hardware, netwerk, processen, methoden en technieken. Op deze wijze bent u snel uw weg vinden in de wereld van IT, het begin van een leven lang leren.

Auteur	: Bart de Best
Uitgever	: Leonon Media, 2021
ISBN (NL)	: 978 94 92618 573



SLA Best Practices

Het volledige ABC van service level agreements.

Het belangrijkste bij het leveren van een service is dat de klant tevreden is over de geleverde prestaties. Door deze tevredenheid verkrijgt de leverancier heraankopen, wordt hij gepromote in de markt en is de continuïteit van het bedrijf geborgd. Wellicht nog het belangrijkste aspect van deze klanttevredenheid voor een leverancier is dat de betrokken medewerkers een drive krijgen om hun eigen kennis en kunde verder te ontwikkelen om nog meer klanten tevreden te stellen. Dit boek beschrijft de best practices om erachter te komen wat de Prestatie-Indicatoren (PI's) zijn die gemeten moeten worden om de tevredenheid van de klant te borgen.

Het tweede deel beschrijft de documenten die van toepassing zijn om de afspraken in vast te leggen. Het opstellen, afspreken, bewaken en evalueren van serviceafspraken is een vak op zich. Het derde deel geeft de gereedschappen om hier adequaat invulling aan te geven. De werkzaamheden rond serviceafspraken herhalen zich in de tijd. Deel vier van dit boek beschrijft hoe deze werkzaamheden in een proces gevat kunnen worden en hoe dit proces het beste in de organisatie kan worden vormgegeven. Tot slot geeft bespreekt dit boek een aantal raakvlakken van serviceafspraken en een tweetal artikelen met SLA best practices.

Auteur	: Bart de Best
Uitgever	: Leonon Media, 2011
ISBN (NL)	: 978 90 7150 1456

Cloud SLA
The best practices of cloud service level agreements

More and more organisations are opting to replace traditional ICT services with cloud services. Drawing up effective SLAs for traditional ICT services is a real challenge for many organisations. With the advent of cloud services, this initially seems much simpler, but soon the difficult questions such as data ownership, information links and security are addressed. This book describes what cloud services are. The risks that organisations run when entering into contracts and SLAs are discussed. Based on a long list of risks and countermeasures, this book also provides recommendations for the design and content of the various service level management documents for cloud services.
This book first defines the term 'cloud' and then describes various aspects such as cloud patterns and the role of a cloud broker. The core of the book concerns the discussion of contract aspects, service documents, service designs, risks, SLAs, and cloud governance. To enable the reader to immediately get started with cloud SLAs, the book also includes checklists of the following documents: Underpinning Contract (UC), Service Level Agreement (SLA), File Financial Agreements (DFA), Dossier Agreements and Procedures (DAP), External Spec Sheets (ESS) and Internal Spec Sheets (ISS).

Author : Bart de Best
Publisher : Leonon Media, 2014
ISBN (NL) : 978 90 7150 1739
ISBN (UK) : 978 94 9261 8009

SLA Templates
A complete set of SLA templates

The most important thing in providing a service is that the customer is satisfied with the delivered performance. With this satisfaction, the supplier gets re-purchasing's, promotions in the market and is the continuity of the company ensured. Perhaps the most important aspect of this customer satisfaction for a supplier is that the employees in question get a drive to further develop their own knowledge and skills to satisfy even more customers. This book describes the templates for Service Level Agreements in order to agree with the customer on the required service levels. This book gives both a template and an explanation for this template for all common service level management documents.

The following templates are included in this book:

- Service Level Agreement (SLA)
- Underpinning Contract (UC)
- Operational Level Agreement (OLA)
- Document Agreement and Procedures (DAP)
- Document Financial Agreements (DFA)
- Service Catalogue
- External Spec Sheet (ESS)
- Internal Spec Sheet (ISS)
- Service Quality Plan (SQP)
- Service Improvement Program (SQP)

Author : Bart de Best
Publisher : Leonon Media, 2017
ISBN (UK) : 978 94 92618 030
ISBN (Pocket Guide) : 978 94 92618 320

ICT Prestatie-indicatoren

De beheerorganisatie meetbaar gemaakt.

De laatste jaren is het maken van concrete afspraken over de ICT-serviceverlening steeds belangrijker geworden. Belangrijke oorzaken hiervoor zijn onder meer de stringentere wet- en regelgeving, de hogere eisen die gesteld worden vanuit regievoering over uitbestede services en de toegenomen complexiteit van informatiesystemen. Om op de gewenste servicenormen te kunnen sturen, is het belangrijk om een Performance Measurement System (PMS) te ontwikkelen. Daarmee kunnen niet alleen de te leveren ICT-services worden gemeten, maar tevens de benodigde ICT-organisatie om de ICT-services te verlenen.

Het meten van prestaties is alleen zinvol als bekend is wat de doelen zijn van de opdrachtgever. Daarom start dit boek met het beschrijven van de bestuurlijke behoefte van een organisatie en de wijze waarop deze vertaald kunnen worden naar een doeltreffend PMS. Het PMS is hierbij samengesteld uit een meetinstrument voor de vakgebieden service management, project management en human resource management. Voor elk van deze gebieden zijn tevens tal van prestatie-indicatoren benoemd. Hiermee vormt dit boek een onmisbaar instrument voor zowel ICT-managers, kwaliteitsmanagers, auditors, service managers, project managers, programma managers, proces managers, als human resource managers.

Auteur	: Bart de Best
Uitgever	: Leonon Media, 2011
ISBN (NL)	: 978 90 7150 1470

Quality Control & Assurance

Kwaliteit op maat.

De business stelt steeds hogere eisen aan de ICT-services die ICT-organisaties leveren. Niet alleen nemen de eisen van de overheid toe in de vorm van wet- en regelgeving, ook de dynamiek van de markt wordt hoger en de levenscyclus van business producten korter. De reactie van veel ICT-organisaties hierop is het hanteren van kwaliteitsmodellen zoals COBIT, ITIL, TOGAF en dergelijke. Helaas verzandt het toepassen van de best practices van deze modellen vaak omdat het model als doel wordt verklaard, hierdoor ontstaat veel overhead. Nut en noodzaak worden niet onderscheiden. In het beste geval is de borging van kwaliteit een golfbeweging met pieken en dalen waarop maar weinig grip op te krijgen is. Dit boek bespreekt op welke wijze de keuze voor kwaliteit concreet en kwantitatief gemaakt kan worden alsmede hoe de kwaliteit in de ICT-organisatie verankerd kan worden. De voorgestelde aanpak omvat zowel Quality Control (opzet en bestaan) als Quality Assurance (werking) voor ICT-processen. Hierbij worden de eisen die aan de ICT-organisatie worden gesteld vertaald naar procesrequirements (opzet) en worden deze binnen ICT-processen geborgd (bestaan). Periodiek worden deze gemeten (werking). Door requirements te classificeren naar tijd, geld, risicobeheersing en volwassenheid kan het management een bewuste keuze maken voor de toepassing van requirements. Hierdoor wordt kwaliteit meetbaar en blijft de overhead beperkt. Dit boek is een onmisbaar instrument voor kwaliteitsmanagers, auditors, lijnmanagers en proces managers.

Auteur	: Bart de Best
Uitgever	: Leonon Media, 2012
ISBN (NL)	: 978 90 7150 1531

Acceptatiecriteria

Naar een effectieve en efficiënte acceptatie van producten en services in de informatietechnologie.

Acceptatiecriteria zijn een meetinstrument voor zowel gebruikers als beheerders om te bepalen of nieuwe of gewijzigde informatiesystemen voldoen aan de afgesproken requirements ten aanzien van functionaliteit, kwaliteit en beheerbaarheid. Er komt heel wat bij kijken om acceptatiecriteria te verankeren in beheerprocessen en systeemontwikkelingsprojecten. Het opstellen en het hanteren van acceptatiecriteria voor ICT-producten en ICT-services geschiedt bij veel organisaties met wisselend succes. Vaak worden acceptatiecriteria wel opgesteld, maar niet effectief gebruikt en verworden ze tot een noodzakelijk kwaad zonder kwaliteitsborgende werking. Dit boek geeft een analyse van de oorzaken van dit falen van de kwaliteitsbewaking. Als remedie worden drie stappenplannen geboden voor het afleiden, toepassen en invoeren van acceptatiecriteria. De doelgroep van dit boek omvat alle partijen die betrokken zijn bij de acceptatie van ICT-producten en ICT-services: de klanten, de leveranciers en de beheerders. Ook is er nog een doelgroep die niet accepteert, maar vaststelt of correct is geaccepteerd; hiertoe behoren kwaliteitsmanagers en auditors die het boek als normenkader kunnen gebruiken. In dit boek is een aantal casussen opgenomen die diverse manieren laten zien voor het effectief en efficiënt omgaan met acceptatiecriteria.

Auteur : Bart de Best
Uitgever : Leonon Media, 2014
ISBN (NL) : 978 90 7150 1784

Beheren onder Architectuur

Het richting geven aan de inrichting van beheerorganisaties.

Veel organisaties zijn al jaren bezig met het vormgeven van de beheerorganisatie door vanaf de werkvloer te kijken wat er fout gaat en op basis daarvan verbetervoorstellen te formuleren. Hierbij wordt meestal gebruik gemaakt van beheermodellen, zoals ITIL, ASL en BiSL, omdat deze veel best practices bevatten. Deze bottom-up benadering werkt een lange tijd goed. De afstemming van de beheerorganisatie-inrichting op de behoefte van de business is daarmee echter nog geen feit. Het wezenlijke verschil met een top-down benadering is dat er eerst een kader gesteld wordt dat richting geeft aan de inrichting van de beheerorganisatie.

Dit kader bestaat uit beleidsuitgangspunten, architectuurprincipes en -modellen. Deze richtinggevendheid is ook van toe passing op de projectorganisatie waarin de producten en services worden vormgegeven die beheerd moeten gaan worden. Het eerste deel van dit boek positioneert dit gedachtegoed binnen de wereld van de informatievoorzieningsarchitectuur. Het tweede deel beschrijft een stappenplan om invulling te geven aan dit gedachtegoed aan de hand van vele best practices en checklists. Het derde deel beschrijft hoe beheren onder architectuur in de organisatie kan worden ingebed. Tot slot geeft het vierde deel een negental casussen van organisaties die het aangereikte stappenplan al hebben toegepast.

Auteur : Bart de Best
Uitgever : Leonon Media, 2017
ISBN (NL) : 978 90 7150 1913

Agile Service Management with scrum

Towards a healthy balance between the dynamics of development and the stability of information management.

The application of Agile software development is booming. The terms Scrum and Kanban are already established in many organisations. Agile software development sets different requirements for the implementation of software management. Many organisations are therefore busy considering this new challenge. Especially the interaction between the Scrum development process and the management of the software that the Scrum development process has produced is an important aspect area. This book discusses precisely this interaction.

Examples of topics that are discussed are the service portfolio, SLAs and the handling of incidents and change requests. This book first defines the risk areas when introducing Scrum and Kanban. After that, the various Agile concepts and concepts are discussed. The implementation of Agile service management is described at both organisational and process level. The relevant risks have been identified for each management process. It is also indicated how this can be implemented within the context of scrum.

Author	: Bart de Best
Publisher	: Leonon Media, 2014 (NL), 2018 (UK)
ISBN (NL)	: 978 90 7150 1807
ISBN (UK)	: 978 94 9261 8085

Agile Service Management with Scrum in Practice

Towards a healthy balance between the dynamics of development and the stability of information management.

Many companies are in the process of applying Agile software development in the form of Scrum or Kanban or have already started using the new development process. Sooner or later, the question arises as to how this development process relates to the management processes. This interface has already been examined in the book 'Agile Service Management with scrum' and a number of risks per management process have been identified. Countermeasures that can be taken are also defined. These risks were presented in a survey of ten organisations, and they were asked how they dealt with these risks.

It was also investigated which Agile aspects are applied and in particular those of Scrum or Kanban. Finally, each organisation performed a maturity assessment for both the Agile development process and the change management process. This book is the report on the research into the collaboration of Agile software development and management processes in practice. The target audience of this book includes all parties involved in the application of Agile software development and who would like to know how colleagues have designed this crucial interface for successful service provision. This book also provides a brief description of each organisation about the way in which the Agile development process is designed.

Author	: Bart de Best
Publisher	: Leonon Media, 2015 (NL), 2018 (UK)
ISBN (NL)	: 978 90 7150 1845
ISBN (UK)	: 978 94 9261 8177

DevOps Best Practices
Best Practices for DevOps

In recent years, many organisations have experienced the benefits of using Agile approaches such as Scrum and Kanban. The software is delivered faster whilst quality increases and costs decrease. The fact that many organisations that applied the Agile approach did not take into account the traditional service management techniques, in terms of information management, application management and infrastructure management, is a major disadvantage. The solution to this problem has been found in the Dev (Development) Ops (Operations) approach. Both worlds are merged into one team, thus sharing the knowledge and skills. This book is about sharing knowledge on how teams work together.

For each aspect of the DevOps process best practices are given in 30 separate articles. The covered aspects are Plan, Code, Build, Test, Release, Deploy, Operate and Monitor. Each article starts with the definition of the specifically used terms and one or more concepts. The body of each article is kept simple, short, and easy to read.

Author	: Bart de Best
Publisher	: Leonon Media, 2017 (UK), 2018 (UK)
ISBN (NL)	: 978 94 92618 078
ISBN (Pocket Guide)	: 978 94 92618 306

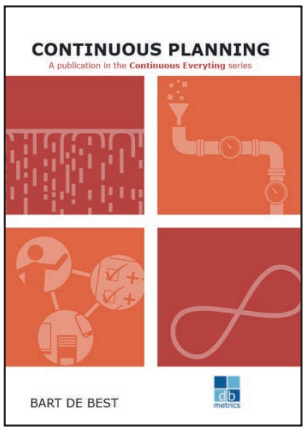
DevOps Architecture
DevOps Architecture Best Practices

The world of systems development is changing at a rapid pace. In addition, Development (Dev) and Operations (Ops) are increasingly integrated so that solutions can be offered to the customer faster and of better quality. The question is how within this new view of DevOps there room is for Agile architecture. This book answers this question by providing many examples of architectural principles and models that guide the organisation and operation of a DevOps organisation. Throughout the book, as much as possible per paragraph, an explanation is given based on an imaginary company Assuritas.

This book consists of several parts, which makes the book modular. So, it does not have to be read from A to Z. The brief outline of the case company is followed by a discussion of the DevOps organisation from an architectural perspective. Then the DevOps management facility is discussed. Both treatises are made transparent based on the case company. After discussing the integration of the Dev and Ops roles, there are two useful analysis tools to determine the maturity of DevOps. The book concludes with a case in which the choice for Agile documentation is made based on architectural principles and models. This work on DevOps architecture is an indispensable tool in the design and implementation of a DevOps service organisation.

Author	: Bart de Best
Publisher	: Leonon Media, 2019
ISBN (NL)	: 978 94 92618 061
ISBN (UK)	: 978 90 71501 579

Continuous Everything books



Continuous Planning

A publication in the Continuous Everything series.

Continuous Planning is an approach to get a grip on changes that are made in the information provision in order to realise the outcome improvement of the business processes and thus achieve the business goals. The approach is aimed at multiple levels, whereby an Agile planning technique is provided for each level that refines the higher-level planning. In this way, planning can be made at a strategic, tactical, and operational level and in an Agile manner that creates as little overhead as possible and as much value as possible. This book is a publication in the continuous everything series. The content consists of a discussion of planning techniques such as the balanced scorecard, enterprise architecture, product vision, roadmap, epic one pager, product backlog management, release planning and sprint planning. It also indicates how these techniques are related to each other. In addition, this book indicates how to set up continuous planning in your organisation based on the change manager paradigm and architecture principles and models. With this integral Agile approach to planning, you have a powerful tool at your disposal to systematically approach your organisation's strategy and thereby realise your business goals.

Author	: Bart de Best
Publisher	: Leonon Media, 2022
ISBN (NL)	: 978 94 92618 504
ISBN (UK)	: 978 94 92618 726

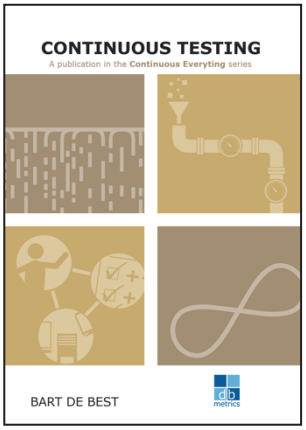
Continuous Design

A publication in the Continuous Everything series.

Continuous Design is an approach that aims to allow DevOps teams to briefly think in advance about the contours of the information system to be realised and to allow the design to grow during the Agile project (emerging design). This prevents interface risks and guarantees essential knowledge transfer to support management and compliance with legislation and regulations. Elements that guarantee the continuity of an organisation. This book is a publication in the continuous everything series. The content consists of the continuous design pyramid model in which the following design views are defined: business, solution, design, requirements, test, and code view.

The continuous design encompasses the entire lifecycle of the information system. The first three views are completed based on modern design techniques such as value stream mapping and use cases. However, the emphasis of the effective application of a continuous design lies in the realisation of the information system, namely by integrating the design in the Behavior Driven Development and Test-Driven Development as well as in continuous documentation. With this Agile approach to design you have a powerful tool at your disposal to get a grip on an Agile development project.

Author	: Bart de Best
Publisher	: Leonon Media, 2022
ISBN (NL)	: 978 94 92618 481
ISBN (UK)	: 978 94 92618 702

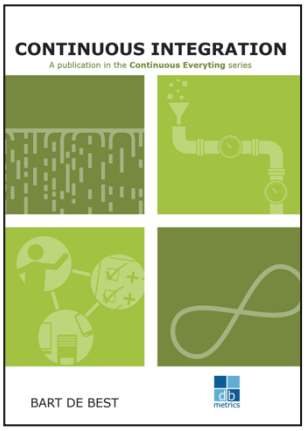


Continuous Testing

A publication in the Continuous Everything series.

Continuous Testing is an approach that aims to provide rapid feedback in the software development process by defining the 'what' and 'how' questions as test cases before starting to build the solution. As a result, the concepts of requirements, test cases and acceptance criteria are integrated in one approach. The term 'continuous' refers to the application of test management in all phases of the deployment pipeline, from requirements to production. The term 'continuous' also includes the aspects People, Process and Technology. This makes test management holistic. This book is a publication in the continuous everything series. The content consists of treating continuous testing based on a definition, business case, architecture, design, and best practices. Concepts discussed are: the change paradigm, the ideal test pyramid, test metadata, Behavior Driven Development (BDD), Test Driven Development (TDD), test policies, test techniques, test tools and the role of unit test cases in continuous testing. In this way you are quickly up to date in the field of DevOps developments and in the field of continuous testing.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 450
ISBN (UK) : 978 94 92618 672



Continuous Integration

A publication in the Continuous Everything series.

Continuous Integration is a holistic Lean software development approach that aims to produce and put into production continuous software in an incremental and iterative way, where waste reduction is of paramount importance.

The word 'holistic' refers to the PPT concepts: People (multiple expert), Process (knowledge of business and management processes) and Technology (application and infrastructure programming). The incremental and iterative method makes fast feedback possible because functionalities can be put into production earlier. This reduces waste because defects are found earlier and can be repaired faster. This book is a publication in the continuous everything series. The content consists of treating continuous integration based on a definition, business case, architecture, design, and best practices. Concepts discussed here are the change paradigm, the application of continuous integration, use of repositories, code quality, green code, green build, refactoring security-based development and built-in failure mode. In this way you are quickly up to date in the field of DevOps developments with regard to continuous integration.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 467
ISBN (UK) : 978 94 92618 689

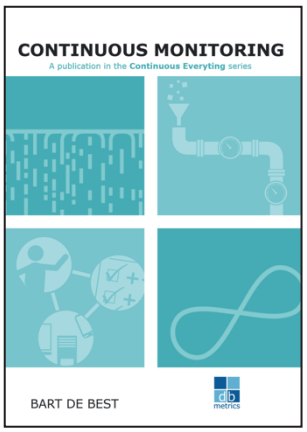
Continuous Deployment

A publication in the Continuous Everything series.

Continuous Deployment is a holistic Lean production approach that aims to deploy and release continuous software in an incremental and iterative way, where time to market and high quality are of paramount importance. The word 'holistic' refers to the PPT concepts: People (multiple expert), Process (knowledge of business and management processes) and Technology (application and infrastructure programming. The incremental and iterative deployments enable fast feedback because errors are more likely to are observed in production of the CI/CD secure pipeline, making recovery actions faster and cheaper, leading to a waste reduction.

This book is a publication in the continuous everything series. The content consists of treating continuous deployment based on a definition, business case, architecture, design, and best practices. Concepts that are discussed here are the change paradigm, the application of continuous deployment, a step-by-step plan for the systematic arrangement of continuous deployment and many patterns to allow deployments to take place. In this way you are quickly up to date in the field of DevOps developments in the field of continuous deployment.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 511
ISBN (UK) : 978 94 92618 733



Continuous Monitoring

A publication in the Continuous Everything series.

Continuous Monitoring is an approach to get a grip on both core value streams (business processes) and enable value streams that support these core value streams. Continuous monitoring differs from classical monitoring by its focus on outcome improvement and the holistic scope with which value streams are measured, i.e. the entire CI/CD secure pipeline for all three perspectives of PPT: People, Process and Technology.

The approach includes People, Process and Technology, which makes it possible to identify and eliminate or mitigate the bottlenecks in your value streams.

This book is a publication in the continuous everything series. The content consists of a discussion of the monitor functions defined in the continuous monitoring layer model. This layer model classifies the monitoring tools available on the market. Each monitor archetype is defined in this book in terms of definition, objective, measurement attributes, requirements, examples, and best practices. This book also indicates how to set up continuous monitoring in your organisation based on the change manager paradigm and architecture principles and models. With this integral agile approach to monitoring you have a powerful tool at your disposal to set up the controls for the control of your value streams.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 498
ISBN (UK) : 978 94 92618 719

Continuous Learning

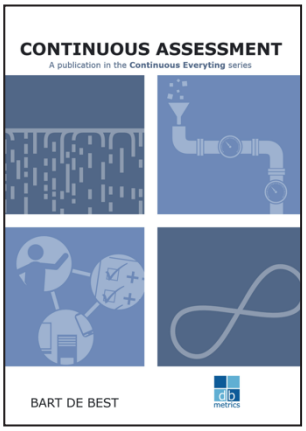
A publication in the Continuous Everything series.

Continuous Learning is an approach to get a grip on the competences needed to realise your organisation's strategy. To this end, continuous learning offers Human Resource Management an approach that explores the organisational needs and competences step by step and converts these needs into competency profiles.

A competency profile is defined here as the set of knowledge, skills and behavior at a certain Bloom level that produces a certain result. Competency profiles are then merged into roles that in turn form functions. In this way an Agile job house is obtained. This book is a publication in the continuous everything series.

The content consists of a discussion of the continuous learning model that helps you to translate a value chain strategy step by step into a personal roadmap for employees. This book also indicates how to organise Continuous Learning in your organisation based on the paradigm of the change manager and architecture principles and models. With this agile approach to HRM you have a powerful tool to get the competences to the desired level of your organisation.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 528
ISBN (UK) : 978 94 92618 740



Continuous Assessment

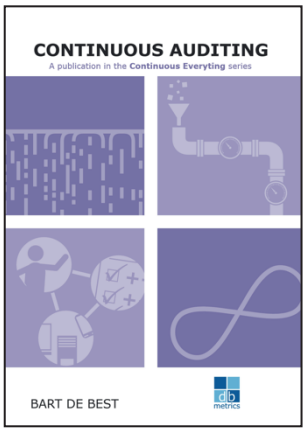
A publication in the Continuous Everything series.

Continuous Assessment is an approach that aims to allow DevOps teams to continuously develop in terms of knowledge and skills in the field of business, development, operations, and security. This book provides a tool to make the DevOps teams aware where they stand in terms of development and what next steps they can take to develop. This book is a publication in the continuous everything series.

The content consists of the business case for continuous assessment, the architecture of the two assessment models and the assessment questionnaires.

The DevOps Cube model is based on the idea that DevOps can be viewed from six different perspectives of a cube, namely: 'Flow', 'Feedback', 'continuous learning', 'Governance', 'Pipeline' and 'QA'. The DevOps CE model is based on the continuous everything perspectives, namely: 'continuous integration', 'continuous deployment', 'continuous testing', 'continuous monitoring', continuous documentation' and 'continuous learning'. This book is an excellent mirror for any DevOps team that wants to quickly form a complete picture of DevOps best practices to be adopted.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 474
ISBN (UK) : 978 94 92618 696



Continuous Auditing

A publication in the Continuous Everything series.

Continuous Auditing is an approach that aims to enable DevOps teams to demonstrate in a short cyclical way that they are in control when realising, putting into production, and managing the new or modified products and services at a rapid pace.

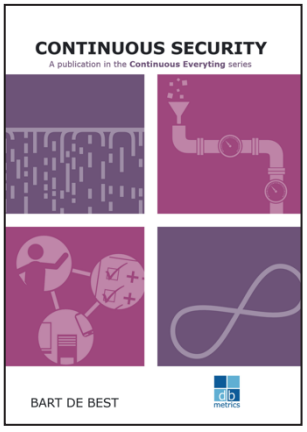
As a result, compliance risks are prevented by already thinking about which risks to mitigate or eliminate from the requirements and the design based on them.

This book is a publication in the continuous everything series.

The content consists of an explanation of the continuous auditing pyramid model that describes the six steps to give substance to continuous auditing, namely: determining scope, determining goals, identifying risks, realising controls, setting up monitoring facilities and demonstrating effectiveness of controls.

The Continuous Auditing concept thus encompasses the entire lifecycle of risk management. As a result, the risks are continuously under control. With this Agile approach of auditing, you have a powerful tool to get a grip on the compliancy of your Agile system development and management.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 542
ISBN (UK) : 978 94 92618 757



Continuous Security

A publication in the Continuous Everything series.

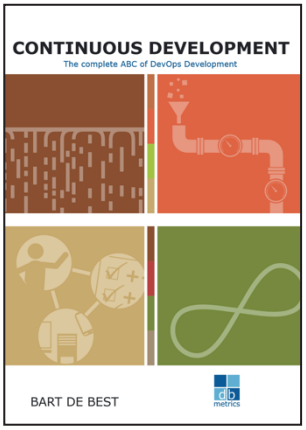
Continuous security is an approach that aims to keep an organisation in control from three perspectives:

- The business perspective: Business value streams are in control of the identified risks by continuously testing the effectiveness of the controls deployed and recording evidence.
- The development perspective: Development value streams are in control by integrally including the non-functional requirements for information security in the development.
- The operations perspective: Operations value streams are in control for the production of the new and changed ICT services through an adequate design of the CI/CD secure pipeline in which controls automatically test the non-functional requirements.

This book is a publication in the continuous everything series. The content consists of a discussion of the application of ISO 27001 on the basis of three sets of security practices, namely Governance, Risk and Quality. The practices are provided with a definition and objective. In addition, examples and best practices are given.

The continuous security concept is designed to be used in Agile Scrum (development) and DevOps (Development & Operations) environments. To this end, it connects seamlessly to common Agile management models. This Agile approach to information security provides you with a powerful tool to get a grip on the compliance of your Agile system development and management.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 91480 171
ISBN (UK) : 978 94 91480 188



Continuous Development

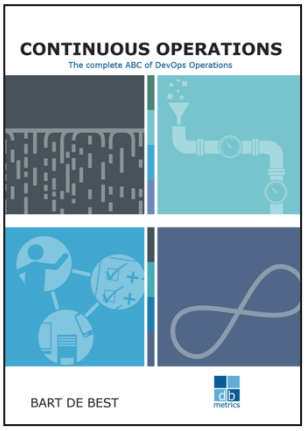
A publication in the Continuous Everything series.

Continuous Everything is the collective name for all Continuous developments that are currently going on in the DevOps world. By placing these under one heading, structure can be applied to individual developments and best practices can be defined on the basis of patterns.

The term 'Continuous' includes the terms: outcome driven development, incremental & iterative working, waste reduction through a Lean approach, holistic working by including people, process, partner & technology in the scope and giving continuous attention to a deliverable (product or service) across the entire lifecycle from an end-to-end approach.

This book is a collection of four Continuous Everything books, namely: Continuous Planning, Continuous Design, Continuous Testing and Continuous Integration. For each Continuous Everything aspect area it is indicated how to organise it in your organisation based on the change manager paradigm and architecture principles and models. The best practices are also discussed per aspect area. With this book in hand, you have a powerful tool to further your DevOps skills.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 641
ISBN (UK) : 978 94 92618 764



Continuous Operations

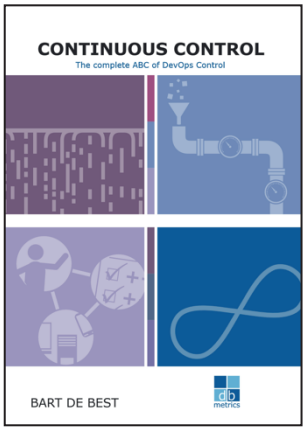
A publication in the Continuous Everything series.

Continuous Everything is the collective name for all Continuous developments that are currently going on in the DevOps world. By placing these under one heading, structure can be applied to individual developments and best practices can be defined on the basis of patterns.

The term 'Continuous' includes the terms: outcome driven development, incremental & iterative working, waste reduction through a Lean approach, holistic working by including people, process, partner & technology in the scope and giving continuous attention to a deliverable (product or service) across the entire lifecycle from an end-to-end approach.

This book is a collection of four Continuous Everything books, namely: Continuous Deployment, Continuous Monitoring, Continuous Learning and Continuous Assessment. For each Continuous Everything aspect area it is indicated how to organise it in your organisation based on the change manager paradigm and architecture principles and models. The best practices are also discussed per aspect area. With this book in hand, you have a powerful tool to further your DevOps skills.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 658
ISBN (UK) : 978 94 92618 771



Continuous Control

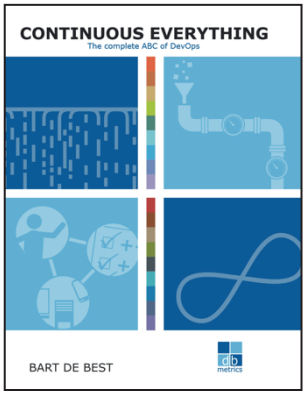
A publication in the Continuous Everything series.

Continuous Everything is the collective name for all Continuous developments that are currently going on in the DevOps world. By placing these under one heading, structure can be applied to individual developments and best practices can be defined on the basis of patterns.

The term 'Continuous' includes the terms: outcome driven development, incremental & iterative working, waste reduction through a Lean approach, holistic working by including people, process, partner & technology in the scope and giving continuous attention to a deliverable product or service across the entire lifecycle from an end-to-end approach.

This book is a collection of three Continuous Everything books, namely: Continuous Assessment, Continuous Security, Continuous Audit. For each Continuous Everything aspect area it is indicated how to organise it in your organisation based on the change manager paradigm and architecture principles and models. The best practices are also discussed per aspect area. With this book in hand, you have a powerful tool to further your DevOps skills.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 91480 195
ISBN (UK) : 978 94 91480 201



Continuous Everything

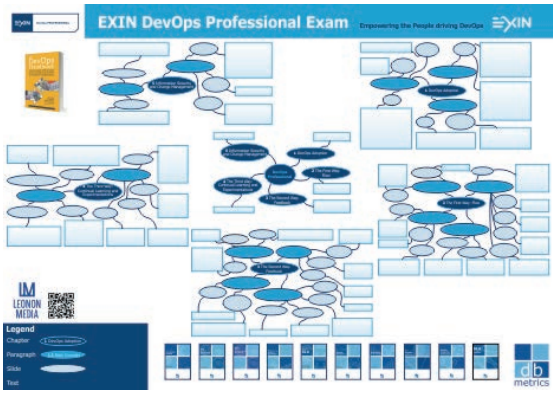
A publication in the Continuous Everything series.

Continuous Everything is the collective name for all Continuous developments that are currently going on in the DevOps world. By placing these under one heading, structure can be applied to individual developments and best practices can be defined on the basis of patterns.

The term 'Continuous' includes the terms: outcome driven development, incremental & iterative working, waste reduction through a Lean approach, holistic working by including people, process, partner & technology in the scope and giving continuous attention to a deliverable product or service across the entire lifecycle from an end-to-end approach.

This book is a collection of eight Continuous Everything books, namely: Continuous Planning, Continuous Design, Continuous Testing, Continuous Integration, Continuous Deployment, Continuous Monitoring, Continuous Learning and Continuous Assessment. For each Continuous Everything aspect area it is indicated how to organise it in your organisation based on the change manager paradigm and architecture principles and models. The best practices are also discussed per aspect area. With this book in hand, you have a powerful tool to further your DevOps skills.

Author : Bart de Best
Publisher : Leonon Media, 2022
ISBN (NL) : 978 94 92618 597
ISBN (UK) : 978 94 92618 665



DevOps Poster

DevOps Professional Exam Poster

This poster lists all the DevOps terms that a student must learn in order to pass the exam of DevOps Professional of Exin. This poster can be ordered at info@leonon.nl.

The subjects on the poster are based on the basic training material of Exin. Since there are many terms to be learned, this poster will help to learn them by reviewing them all at once daily.

Author : Bart de Best
Publisher : Leonon Media, 2018
Ordering : info@leonon.nl

CONTINUOUS DESIGN

A publication in the
Continuous Everyting
series

Bart de Best

Continuous Design is an approach that aims to allow DevOps teams to briefly think in advance about the contours of the information system to be realized and to allow the design to grow during the Agile project (emerging design).

This prevents interface risks and guarantees essential knowledge transfer to support management and compliance with legislation and regulations. Elements that guarantee the continuity of an organisation.

This book is a publication in the Continuous Everything series. The content consists of the Continuous Design Pyramid model in which the following design views are defined: business, solution, design, requirements, test, and code view.

The Continuous Design encompasses the entire lifecycle of the information system. The first three views are completed based on modern design techniques such as value stream mapping and use cases. However, the emphasis of the effective application of a Continuous Design lies in the realisation of the information system, namely by integrating the design in the Behaviour Driven Development and Test Driven Development as well as in Continuous Documentation.

With this Agile approach to design, you have a powerful tool at your disposal to get a grip on an Agile development project.

db metrics

ISBN 978-94-92618-70-2