



SLOVENSKI STANDARD

SIST-TP CEN/TR 17603-40-01:2022

01-september-2022

Vesoljska tehnika - Priročnik o spreminjajočem se razvoju programske opreme

Space engineering - Agile software development handbook

Raumfahrttechnik - Handbuch zur agilen Softwareentwicklung

Ingénierie spatiale - Guide de développement logiciel en mode agile

Ta slovenski standard je istoveten z: CEN/TR 17603-40-01:2022

<https://standards.iteh.ai/catalog/standards/sist/f4f0c2b8-a485-46f3-94aa-865fb672c57d/sist-tp-cen-tr-17603-40-01-2022>

ICS:

35.080	Programska oprema	Software
49.140	Vesoljski sistemi in operacije	Space systems and operations

SIST-TP CEN/TR 17603-40-01:2022 **en,fr,de**

TECHNICAL REPORT

CEN/TR 17603-40-01

RAPPORT TECHNIQUE

TECHNISCHER BERICHT

June 2022

ICS 49.140; 35.080

English version

Space engineering - Agile software development handbook

Ingénierie spatiale - Guide de développement logiciel
en mode agile

Raumfahrttechnik - Handbuch zur agilen
Softwareentwicklung

This Technical Report was approved by CEN on 20 April 2022. It has been drawn up by the Technical Committee CEN/CLC/JTC 5.

CEN and CENELEC members are the national standards bodies and national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and United Kingdom.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST-TP CEN/TR 17603-40-01:2022

<https://standards.iteh.ai/catalog/standards/sist/f4f0c2b8-a485-46f3-94aa-865fb672c57d/sist-tp-cen-tr-17603-40-01-2022>



CEN-CENELEC Management Centre:
Rue de la Science 23, B-1040 Brussels

Table of contents

European Foreword.....	7
Introduction.....	8
1 Scope.....	9
2 References	10
3 Terms, definitions and abbreviated terms.....	11
3.1 Terms from other documents.....	11
3.2 Terms specific to the present document	11
3.3 Abbreviated terms.....	15
4 Introduction to the Agile software development approach	17
4.1 Introduction to Agile	17
4.1.1 General	17
4.1.2 Agile characteristics (as derived from the manifesto)	18
4.1.3 Lean management	20
4.2 General issues implementing Agile.....	21
5 Guidelines for Agile life cycle selection	24
5.1 Selecting Agile.....	24
5.2 Analysis of key factors for Agile selection	24
5.2.1 General	24
5.2.2 Customer context.....	26
5.2.3 Supplier context	27
5.2.4 Project context	27
5.2.5 Team context	29
5.2.6 Key Factors Summary.....	30
5.3 Agile assessment process	31
5.4 Selecting agile or waterfall	32
6 Reference models for Scrum-like Agile software life cycle	34
6.1 Introduction.....	34
6.2 Roles and competences	34

CEN/TR 17603-40-01:2022 (E)

6.2.1	Overview.....	34
6.2.2	Scrum master.....	34
6.2.3	Product owner.....	35
6.2.4	Development team.....	35
6.2.5	SCRUM team.....	36
6.2.6	Agile coach.....	36
6.2.7	Training and competencies.....	36
6.3	Exemplary Agile activities.....	37
6.3.1	Distinction between meeting or activity.....	37
6.3.2	Planning I – What will be delivered.....	38
6.3.3	Planning II – How will it be delivered.....	38
6.3.4	Sprint backlog management.....	39
6.3.5	Product backlog refinement.....	39
6.3.6	Progress tracking.....	40
6.3.7	Product backlog update.....	40
6.3.8	Coding, testing and documenting.....	40
6.3.9	User feedback.....	41
6.3.10	Review preparation.....	41
6.3.11	Sprint review.....	41
6.4	Meetings.....	42
6.4.1	Daily meeting.....	42
6.4.2	Management meeting.....	42
6.4.3	Retrospective.....	42
6.5	Organising the Agile activities and meetings in a project to create a life-cycle compliant to ECSS-E-ST-E-40.....	43
6.5.1	Preliminaries.....	43
6.5.2	Product releases.....	44
6.5.3	Start of the project: Sprint#0.....	44
6.5.4	Development phase: Sprints #1 - #N.....	45
6.5.5	Acceptance phase.....	45
6.6	Software lifecycle definition.....	46
6.6.1	ECSS-E-ST-40 reviews.....	46
6.6.2	Organising the ECSS-E-ST-40 reviews in an Agile software approach.....	47
6.6.3	Selecting the right model.....	56
7	Guidelines for software project management.....	57
7.1	Introduction.....	57
7.2	Software Project Management approach.....	57

7.2.1	Overview	57
7.2.2	Management objectives and priorities	57
7.2.3	Schedule management	61
7.2.4	Assumptions, dependencies and constraints.....	63
7.2.5	Work breakdown structure	64
7.2.6	Roles.....	64
7.2.7	Risk management	65
7.2.8	Monitoring and controlling mechanisms	66
7.2.9	Staffing Plan.....	70
7.2.10	Software procurement process.....	72
7.2.11	Supplier management.....	72
7.3	Software development approach	73
7.3.1	Strategy to the software development	73
7.3.2	Software project development lifecycle	73
7.3.3	Relationship with the system development lifecycle	73
7.3.4	Reviews and milestones identification and associated documentation	73
7.4	Software engineering standards and techniques	73
7.5	Software development and software testing environment	73
7.6	Software documentation plan	74
8	Guidelines for software engineering processes	75
8.1	Overview	75
8.2	Software related system requirements process	75
8.3	requirements and architectural engineering	75
8.3.1	Software requirements analysis	75
8.3.2	Software architectural design	81
8.4	Software design and implementation engineering.....	83
8.5	Software validation	85
8.6	Software delivery and acceptance	86
8.7	Software verification	88
8.8	Software operations.....	91
8.9	Software maintenance	91
8.9.1	Overview.....	91
8.9.2	Agile maintenance challenges.....	91
8.9.3	Tailoring Agile to Maintenance	92
8.10	Independent software verification and validation.....	95
9	Guidelines for software product assurance and configuration management	96

CEN/TR 17603-40-01:2022 (E)

9.1	Software product assurance	96
9.1.1	Introduction	96
9.1.2	Planning of software product assurance activities	97
9.1.3	Software product assurance reporting	97
9.1.4	Technical Debt and noncompliance of Quality Requirements	98
9.1.5	Software criticality	99
9.1.6	Software problem management	99
9.1.7	Control of non-conformances	100
9.1.8	Software development environment aspects	100
9.1.9	Summary of software product assurance activities in Agile	100
9.2	Software configuration management	102
9.2.1	Introduction	102
9.2.2	Agile software configuration management challenges	102
9.2.3	Agile methods for configuration management	104
9.2.4	Summary of software configuration activities in Agile	105

Figures

Figure 4-1: From Plan-driven approach to Value-driven approach	19
Figure 4-2: The Lean Thinking House (for details see LEAN-PRIMER)	21
Figure 5-1: Factors for adopting Agile process	25
Figure 5-2: Agile selection factors scale	26
Figure 6-1: Organisation of activities during a sprint	37
Figure 6-2: Exemplar Agile lifecycle	43
Figure 6-3: Model 1: Review driven lifecycle	51
Figure 6-4 Model 2: More flexible review driven lifecycle	53
Figure 6-5: Review driven lifecycle with full flexibility	54
Figure 6-6: Sprint driven lifecycle with formalisation	55
Figure 7-1: Project Management Triangle	58
Figure 7-2: Cost Management: change for free	59
Figure 7-3: Sample Burndown Chart for a Sprint	62
Figure 7-4: Example for an Agile work breakdown	64
Figure 7-5: Agile model supports risk management	65
Figure 7-6: Success of continuous integration tests	68
Figure 7-7: A team metric dashboard	68
Figure 7-8: Summary of performed work	69
Figure 7-9: Delivered business value in a project	70
Figure 8-1: Example of User Story and Tasks	77

Figure 8-2: Kano model showing means to ensure customer satisfaction.....	79
---	----

Tables

Table 5-1: Supplier context.....	27
Table 5-2: Project context.....	28
Table 5-3: Team context.....	29
Table 5-4: Key factors for selection of classical or agile lifecycle	30
Table 5-5: Aspects for the selection of agile or waterfall approach	32
Table 6-1 – Overview of the different models	49
Table 6-2 – Examples of selection of models based on project characteristics	56
Table 8-1: Mapping ECSS-E-ST-40 to Agile activities. Software Requirements Analysis	80
Table 8-2: Mapping ECSS-E-ST-40 to Agile activities. Software architectural design.....	82
Table 8-3: Mapping ECSS-E-ST-40 to Agile activities. Software Detailed Design, Coding and testing, and Integration.....	83
Table 8-4: Mapping ECSS-E-ST-40 to Agile activities. Software validation.....	85
Table 8-5: Mapping ECSS-E-ST-40 to Agile activities. Software Delivery and Acceptance	87
Table 8-6 Mapping ECSS-E-ST-40 to Agile activities. Software Verification	89
Table 8-7: Mapping ECSS-E-ST-40 to Agile activities. Software Maintenance	94
Table 9-1: Mapping ECSS-Q-ST-80 to Agile activities. Software product assurance	100
Table 9-2: Mapping ECSS-M-ST-40 to Agile activities. Software Configuration Management.....	105

<https://standards.cen.eu/catalog/standards/sist/1410c268-a465-4615-94aa-8651b672c57d/sist-tp-cen-tr-17603-40-01-2022>

European Foreword

This document (CEN/TR 17603-40-01:2022) has been prepared by Technical Committee CEN/CLC/JTC 5 “Space”, the secretariat of which is held by DIN.

It is highlighted that this technical report does not contain any requirement but only collection of data or descriptions and guidelines about how to organize and perform the work in support of EN 16603-40.

This Technical report (CEN/TR 17603-40-01:2022) originates from ECSS-E-HB-40-01A.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN [and/or CENELEC] shall not be held responsible for identifying any or all such patent rights.

This document has been prepared under a mandate given to CEN by the European Commission and the European Free Trade Association.

This document has been developed to cover specifically space systems and has therefore precedence over any TR covering the same scope but with a wider domain of applicability (e.g.: aerospace).

SIST-TP CEN/TR 17603-40-01:2022

<https://standards.iteh.ai/catalog/standards/sist/f4f0c2b8-a485-46f3-94aa-865fb672c57d/sist-tp-cen-tr-17603-40-01-2022>

Introduction

EN 16603-40 (ECSS-E-ST-40) Space Engineering Software Standard defines the principles and requirements applicable to space software engineering. ECSS-E-ST-40 is always complemented by the EN 16602-80 (ECSS-Q-ST-80) Space Product Assurance Standard, which specifies the product assurance aspects. This ECSS-E-HB-40-01 handbook provides more detailed guidelines and advice for adopting an Agile software development approach in space projects where ECSS-E-ST-40 and ECSS-Q-ST-80 are applicable.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

[SIST-TP CEN/TR 17603-40-01:2022](https://standards.iteh.ai/catalog/standards/sist/f4f0c2b8-a485-46f3-94aa-865fb672c57d/sist-tp-cen-tr-17603-40-01-2022)

<https://standards.iteh.ai/catalog/standards/sist/f4f0c2b8-a485-46f3-94aa-865fb672c57d/sist-tp-cen-tr-17603-40-01-2022>

1

Scope

This Handbook provides recommendations for the implementation of an Agile approach in space software projects complying with EN 16603-40 (ECSS-E-ST-40) and EN 16602-80 (ECSS-Q-ST-80).

This handbook is not an Agile development book, though it provides an Agile reference model based on Scrum and also covers other major Agile methods and techniques. Scrum has been selected as reference because of its widespread application in industry and its flexibility as a development framework to introduce or merge with other Agile methods and techniques. In relation to the ECSS-E-ST-40 and ECSS-Q-ST-80, this handbook does not provide any tailoring of their requirements due to the use of the Agile approach, but demonstrates how compliance towards ECSS can be achieved. This handbook does not cover contractual aspects for this particular engineering approach, although it recognises that considering the approach of fixing cost and schedule and making the scope of functionalities variable, the customer and supplier need to establish specific contractual arrangements. Furthermore, it does not impose a particular finality for the use of Agile, either as a set of team values, project management process, specific techniques or supporting exploration by prototypes.

This handbook, covers, in particular, the following:

- In clause 4, the fundamentals and principles of Agile. It also describes major Agile methods and general issues of implementing an Agile approach.
- In clause 5, the criteria for selecting an Agile lifecycle.
- In clause 6, a reference process model based on Scrum to be used to map its elements to relevant clauses of ECSS-E-ST-40.
- In clause 7, guidelines for software project management, providing advice for ECSS-E-ST-40 clause 5.3 considering the reference process model based on Scrum.
- In clause 8, guidelines for software engineering processes, providing advice for ECSS-E-ST-40 clauses 5.2, and 5.4 to 5.10, considering the reference process model based on Scrum.
- In clause 9, guidelines for software product assurance and software configuration management, providing general advice for the implementation of ECSS-Q-ST-80 and ECSS-M-ST-40 with an Agile approach.

Individual agile practices, introduced in this HB, can also be taken on-board in other software development life-cycles.

2 References

EN Reference	Reference in text	Title
EN 16601-00-01	ECSS-S-ST-00-01	ECSS system - Glossary of terms
EN 16603-40	ECSS-E-ST-40	Space engineering - Software
EN 17603-40	ECSS-E-HB-40	Space engineering - Software engineering handbook
EN 16601-10	ECSS-M-ST-10	Space project management - Project planning and implementation
EN 16601-40	ECSS-M-ST-40	Space project management - Configuration and information management
EN 16601-80	ECSS-M-ST-80	Space project management - Risk management
EN 16602-80	ECSS-Q-ST-80	Space product assurance - Software product assurance
EN 16601-80-04	ECSS-Q-HB-80-04	Space product assurance - Software metrication programme definition and implementation handbook
https://standards.iteh.ai/catalog/standards/sist-17603-40-01-2022	Agile Manifesto	Beck, K., et al.: Agile Manifesto and Twelve Principles of Agile Software (2001). http://agilemanifesto.org
	ISO/IEC 26515:2011	Systems and software engineering - Developing user documentation in an Agile environment
	LEAN-PRIMER	Craig Larman and Bas Vodde. 2009. Lean Primer. Available at: http://www.leanprimer.com/downloads/lean_primer.pdf
	Agilealliance	https://www.agilealliance.org
	SCRUM	https://www.scrum.org

Terms, definitions and abbreviated terms

3.1 Terms from other documents

- a. For the purpose of this document, the terms and definitions from ECSS-S-ST-00-01 apply, in particular the following terms:
 1. **process**
 2. **product**
- b. For the purpose of this document, the terms and definitions from ECSS-E-ST-40 apply, in particular the following term:
 1. **critical software**

3.2 Terms specific to the present document

3.2.1 Burndown chart

A chart that records project status, usually showing tasks completed versus time and against total number of tasks

[ISO/IEC 26515:2011]

3.2.2 Capacity

Total number of available hours for a sprint. The capacity is the available hours calculated based on resources planned holiday and company holiday if any.

3.2.3 Continuous integration

development practice according to which the source code in development is uploaded into a shared repository regularly to allow automated build, tests and quality control tools to detect and raise issues early to the development team

3.2.4 Cycle time

time elapsed between the start of the work on a particular task and its completion.

3.2.5 Daily stand-up

short daily team meeting where the team members share their project activities, synchronize themselves and identify and solve impediments that hamper them from being productive

NOTE 1 It is also known as daily meeting, daily Scrum or roll-call.

NOTE 2 Duration of a "Daily stand-up" is about 15 minutes.

3.2.6 Definition of done (DoD)

list of activities and criteria to be achieved to declare an element of the backlog as complete

NOTE 1 This element can be defined at user story, epic, feature, sprint and product release levels

NOTE 2 Elements of the DoD can be, for example: writing source code, update specification, design and user documentation, execution of unit testing, achieving a certain level of test coverage, establish compliance to a certain level of coding rules.

[adapted from the Agile Alliance]

3.2.7 Definition of ready

list of criteria that a user story has to meet to be accepted into the upcoming sprint or iteration

[adapted from the Agile Alliance]

3.2.8 Epic

big user story that is too big to be estimated. It is usually too big for an iteration and will be broken down into smaller user stories.

NOTE It is usually too big for an iteration and will be broken down into smaller user stories.

3.2.9 Feature

functional or non-functional distinguishing characteristic of a system, usually an enhancement to an existing system

[ISO/IEC 26515:2011]

3.2.10 Increment

sum of all the Product Backlog items completed during a sprint and the value of the increments of all previous sprints

3.2.11 Iteration

See “sprint”[3.2.22].

3.2.12 Lead time

time elapsed between the customer request for an increment and its delivery

3.2.13 Load factor

KPI measuring how many person days it takes to complete a story point

NOTE A story point is a relative measurement unit used to compare relatively user stories, epics (e.g. one ideal working day).

3.2.14 Pair programming

practice where two developers share the same development environment, where the first developer writes the code and the second reviews the code simultaneously and thinks ahead

NOTE Both change roles often so that the strong points of both developers can be utilised and knowledge is shared in the team.

CEN/TR 17603-40-01:2022 (E)**3.2.15 Peer review**

practice where code written by a developer is reviewed by another developer of the same team

3.2.16 Product backlog

ordered list of everything that is known to be needed in the product

- NOTE 1 It is the single source of requirements for any changes to be made to the product.
- NOTE 2 A Product Backlog is never complete. The earliest development of it lays out the initially known and best-understood requirements. The Product Backlog evolves as the product and the environment in which it will be used evolves. The Product Backlog is dynamic; it constantly changes to identify what the product needs to be appropriate, competitive, and useful. If a product exists, its Product Backlog also exists.
- NOTE 3 The items can be, for examples, features, requirements, software problem reports or technical tasks.
- NOTE 4 The backlog is the primary point of entry for knowledge about requirements, and the single authoritative source defining the work to be done.

[adapted from the Agile Alliance]

3.2.17 Product Owner

entity as close as possible to the end customer and users and as knowledgeable as possible to the business and solution context

- NOTE See also clause 6.2.3.

3.2.18 Product Backlog Refinement

formal or informal meeting or activity to refine the backlog

- NOTE 1 Examples of refinement are:
- removing user stories that no longer appear relevant,
 - creating new user stories in response to newly discovered needs,
 - re-assessing the relative priority of user stories, assigning estimates to user stories which have yet to receive one,
 - correcting estimates in light of newly discovered information, splitting user stories which are high priority but too coarse grained to fit in an upcoming iteration.
- NOTE 2 For detailed information about “Product backlog refinement” see clause 6.3.5.

[adapted from the Agile Alliance]

3.2.19 Sprint backlog

set of product backlog items selected for the sprint, plus a plan for delivering the product increment and realizing the sprint goal