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**

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

RAPPORT TECHNIQUE

TECHNISCHER BERICHT

CEN/TR 17603-40-01

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.

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

Ref. No. CEN/TR 17603-40-01:2022 E

Table of contents

Figures

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).

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.

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
	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

3 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.

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