International Standard

ISO/IEC 33202

First edition
2024-07

# Software and systems engineering — Core agile practices

*Ingénierie du logiciel et des systèmes — Pratiques agiles essentielles*



Reference number
ISO/IEC 33202:2024(en)



ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

# Contents

Page

**Foreword** ..... **viii**
**Introduction** ..... **ix**
**1 Scope** ..... **1**
**2 Normative references** ..... **1**
**3 Terms and definitions** ..... **1**
**4 Conformance** ..... **4**
4.1 Intended usage ..... 4
4.2 Full conformance ..... 4
**5 Conceptual foundations** ..... **4**
5.1 Practices and processes ..... 4
5.2 Agile practices and concepts ..... 5
5.3 Agile practices ..... 5
5.4 Criteria for practices ..... 6
5.5 Description of practices ..... 6
5.6 Some agile concepts ..... 6
5.6.1 Cadence ..... 6
5.6.2 Gemba ..... 6
5.6.3 Limiting work in progress ..... 6
**6 Core agile practices** ..... **7**
6.1 Agreement practice ..... 7
6.1.1 Purpose ..... 7
6.1.2 Overview ..... 7
6.1.3 Outcomes ..... 7
6.2 Acquisition practice ..... 7
6.2.1 Purpose ..... 7
6.2.2 Overview ..... 7
6.2.3 Outcomes ..... 7
6.3 Supply practice ..... 8
6.3.1 Purpose ..... 8
6.3.2 Overview ..... 8
6.3.3 Outcomes ..... 8
6.4 Cross-functional team practice ..... 8
6.4.1 Purpose ..... 8
6.4.2 Overview ..... 8
6.4.3 Outcomes ..... 8
6.5 Feature team practice ..... 8
6.5.1 Purpose ..... 8
6.5.2 Overview ..... 8
6.5.3 Outcomes ..... 9
6.6 Component team practice ..... 9
6.6.1 Purpose ..... 9
6.6.2 Overview ..... 9
6.6.3 Outcomes ..... 9
6.7 Co-located team practice ..... 9
6.7.1 Purpose ..... 9
6.7.2 Overview ..... 10
6.7.3 Outcomes ..... 10
6.8 Distributed team practice ..... 10
6.8.1 Purpose ..... 10
6.8.2 Overview ..... 10
6.8.3 Outcomes ..... 10
6.9 Collective ownership practice ..... 10
6.9.1 Purpose ..... 10

6.9.2 Overview ..... 10
6.9.3 Outcomes ..... 11
6.10 Release planning practice ..... 11
6.10.1 Purpose ..... 11
6.10.2 Overview ..... 11
6.10.3 Outcomes ..... 11
6.11 Deployment planning practice ..... 11
6.11.1 Purpose ..... 11
6.11.2 Overview ..... 11
6.11.3 Outcomes ..... 12
6.12 Iteration planning practice ..... 12
6.12.1 Purpose ..... 12
6.12.2 Overview ..... 12
6.12.3 Outcomes ..... 12
6.13 Synchronized iteration practice ..... 12
6.13.1 Purpose ..... 12
6.13.2 Overview ..... 12
6.13.3 Outcomes ..... 12
6.14 Daily synchronization meeting practice ..... 13
6.14.1 Purpose ..... 13
6.14.2 Overview ..... 13
6.14.3 Outcomes ..... 13
6.15 Cross-team synchronization meeting practice ..... 13
6.15.1 Purpose ..... 13
6.15.2 Overview ..... 13
6.15.3 Outcomes ..... 13
6.16 Product owner synchronization meeting practice ..... 14
6.16.1 Purpose ..... 14
6.16.2 Overview ..... 14
6.16.3 Outcomes ..... 14
6.17 Value stream mapping practice ..... 14
6.17.1 Purpose ..... 14
6.17.2 Overview ..... 14
6.17.3 Outcomes ..... 14
6.18 Release review practice ..... 15
6.18.1 Purpose ..... 15
6.18.2 Overview ..... 15
6.18.3 Outcomes ..... 15
6.19 Iteration review practice ..... 15
6.19.1 Purpose ..... 15
6.19.2 Overview ..... 15
6.19.3 Outcomes ..... 15
6.20 Team retrospective practice ..... 16
6.20.1 Purpose ..... 16
6.20.2 Overview ..... 16
6.20.3 Outcomes ..... 16
6.21 Program retrospective review practice ..... 16
6.21.1 Purpose ..... 16
6.21.2 Overview ..... 16
6.21.3 Outcomes ..... 16
6.22 Program board governance practice ..... 17
6.22.1 Purpose ..... 17
6.22.2 Overview ..... 17
6.22.3 Outcomes ..... 17
6.23 Agile adoption improvement practice ..... 17
6.23.1 Purpose ..... 17
6.23.2 Overview ..... 17
6.23.3 Outcomes ..... 17
6.24 Visual management practice ..... 18

6.24.1 Purpose 18
6.24.2 Overview 18
6.24.3 Outcomes 18
6.25 Distributed agile governance practice 18
6.25.1 Purpose 18
6.25.2 Overview 18
6.25.3 Outcomes 18
6.26 User stories practice 19
6.26.1 Purpose 19
6.26.2 Overview 19
6.26.3 Outcomes 19
6.27 Product backlog elicitation practice 19
6.27.1 Purpose 19
6.27.2 Overview 19
6.27.3 Outcomes 19
6.28 Product backlog refinement practice 19
6.28.1 Purpose 19
6.28.2 Overview 19
6.28.3 Outcomes 20
6.29 User story mapping practice 20
6.29.1 Purpose 20
6.29.2 Overview 20
6.29.3 Outcomes 20
6.30 Relative estimation practice 20
6.30.1 Purpose 20
6.30.2 Overview 21
6.30.3 Outcomes 21
6.31 Active stakeholder participation practice 21
6.31.1 Purpose 21
6.31.2 Overview 21
6.31.3 Outcomes 21
6.32 Initial architecture practice 21
6.32.1 Purpose 21
6.32.2 Overview 22
6.32.3 Outcomes 22
6.33 Intentional architecture practice 22
6.33.1 Purpose 22
6.33.2 Overview 22
6.33.3 Outcomes 22
6.34 Architecture description practice 23
6.34.1 Purpose 23
6.34.2 Overview 23
6.34.3 Outcomes 23
6.35 Incremental architecture practice 23
6.35.1 Purpose 23
6.35.2 Overview 23
6.35.3 Outcomes 24
6.36 Simple design practice 24
6.36.1 Purpose 24
6.36.2 Overview 24
6.36.3 Outcomes 24
6.37 Test-driven development practice 24
6.37.1 Purpose 24
6.37.2 Overview 24
6.37.3 Outcomes 25
6.38 Acceptance test-driven development practice 25
6.38.1 Purpose 25
6.38.2 Overview 25
6.38.3 Outcomes 25

6.39 Behaviour-driven development practice ... 26
6.39.1 Purpose ... 26
6.39.2 Overview ... 26
6.39.3 Outcomes ... 26
6.40 Feature-driven development practice ... 26
6.40.1 Purpose ... 26
6.40.2 Overview ... 26
6.40.3 Outcomes ... 26
6.41 Agile testing pyramid practice ... 27
6.41.1 Purpose ... 27
6.41.2 Overview ... 27
6.41.3 Outcomes ... 27
6.42 Continuous integration practice ... 27
6.42.1 Purpose ... 27
6.42.2 Overview ... 27
6.42.3 Outcomes ... 27
6.43 Continuous deployment practice ... 28
6.43.1 Purpose ... 28
6.43.2 Overview ... 28
6.43.3 Outcomes ... 28
6.44 Continuous delivery practice ... 28
6.44.1 Purpose ... 28
6.44.2 Overview ... 28
6.44.3 Outcomes ... 28
6.45 Continuous documentation practice ... 29
6.45.1 Purpose ... 29
6.45.2 Overview ... 29
6.45.3 Outcomes ... 29
6.46 Lean documentation practice ... 29
6.46.1 Purpose ... 29
6.46.2 Overview ... 29
6.46.3 Outcomes ... 29
6.47 Feature toggles practice ... 30
6.47.1 Purpose ... 30
6.47.2 Overview ... 30
6.47.3 Outcomes ... 30
6.48 Refactoring practice ... 30
6.48.1 Purpose ... 30
6.48.2 Overview ... 30
6.48.3 Outcomes ... 30
6.49 Test strategy practice ... 31
6.49.1 Purpose ... 31
6.49.2 Overview ... 31
6.49.3 Outcomes ... 31
6.50 User story testing practice ... 31
6.50.1 Purpose ... 31
6.50.2 Overview ... 31
6.50.3 Outcomes ... 31
6.51 Feature testing practice ... 32
6.51.1 Purpose ... 32
6.51.2 Overview ... 32
6.51.3 Outcomes ... 32
6.52 User acceptance testing practice ... 32
6.52.1 Purpose ... 32
6.52.2 Overview ... 32
6.52.3 Outcomes ... 32
6.53 Regression testing practice ... 33
6.53.1 Purpose ... 33
6.53.2 Overview ... 33

6.53.3 Outcomes ..... 33
**Annex A** (Informative) **Summary of core agile practices** ..... **34**
**Annex B** (informative) **Relationship with other standards** ..... **35**
**Annex C** (informative) **Manifesto for agile software development** ..... **39**
**Annex D** (Informative) **Application of agile in ISO/IEC/IEEE 12207** ..... **41**
**Bibliography** ..... **42**

# Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

# Introduction

Many organizations in the software development industry are adopting agile life cycle to adapt to rapidly changing and evolving marketplaces. “Agile” is derived from the Agile Manifesto[10], which was designed to ‘uncover better ways of developing software by doing it and helping others do it’. It emerged as an alternative to documentation-driven, heavyweight software development processes. The Agile Manifesto[10] and related values and principles (Annex C) have served as reference for agile practices across many organizations.

Agile development is relative to how each organization interprets the agile values and principles (See Annex C). The agile practices discussed in this document offer a lightweight, adaptive and collaborative development approach based on empiricism with the focus on rapid business-value delivery.

Agile standardization refers to adopting agile practices and concepts across an organization or project for software suppliers, software acquirers, software service providers, large delivery and maintenance projects and system integration engagements having software components.

Standardization of agile practices is useful for the following reasons:

a) for suppliers: it provides a basis for delivering meaningful products and services;

b) for acquirers: it provides a basis to differentiate between different suppliers and provides guidance on their role in the development.

The purpose of this document is to catalogue the core agile practices (see Annex A) and concepts demonstrated by organizations or projects. These practices and concepts enable the application of agile in acquisition, supply, development, operation, maintenance and disposal of software systems, products and services.

This document can be used by acquirers, suppliers, developers, integrators, operators, maintainers, managers, quality assurers and users of software systems, products and services, for a range of situations and contexts.

# Software and systems engineering — Core agile practices

## 1 Scope

This document defines a set of core practices and concepts that have wide acceptance in organizations and industries using agile practices and concepts. This document defines a set of core practices that are present in agile methodologies.

The practices and concepts defined in this document are applicable to a single agile team, as well as to multiple agile teams. These practices and concepts are applicable throughout the life cycle of software systems, products and services.

## 2 Normative references

There are no normative references in this document.

## 3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

**3.1**
**acceptance criteria**
criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorized entity

[SOURCE: ISO/IEC/IEEE 24765:2017, 3.32, definition 1]

**3.2**
**agile**
approach to development, delivery and maintenance of products and services by enabling rapid response to feedback

Note 1 to entry: The need specification for the software under development is elaborated with specific requirements only when the work is started. This lean principle is meant to avoid waste of work and to provide *agile team* (3.6) with means of prioritizing their work.

**3.3**
**agile concept**
fundamental building blocks of *agile* (3.2)

**3.4**
**agile development**
software development approach based on iterative development, frequent inspection and adaptation, and incremental deliveries, in which requirements and solutions evolve through collaboration in cross-functional teams and through continuous stakeholder feedback

[SOURCE: ISO/IEC/IEEE 24765:2017, 3.119]

**3.5**
**agile environment**
organizational culture, infrastructure, and methodologies that support *agile development* (3.4)

[SOURCE: ISO/IEC/IEEE 26515:2018, 3.2]

**3.6**
**agile team**
organization or team using *agile development* (3.4) methods and approaches

[SOURCE: ISO/IEC/IEEE 26515:2018, 3.3, modified — Note 1 to entry has been removed.]

**3.7**
**architecture**
fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle *processes* (3.21)

[SOURCE: ISO/IEC/IEEE 42010:2022, 3.2]

**3.8**
**architecture description**
work product used to express an *architecture* (3.7)

[SOURCE: ISO/IEC/IEEE 42010:2022, 3.3, modified — The abbreviated term and notes to entry have been removed.]

**3.9**
**backlog**
collection of features or stories of both functional and non-functional requirements that are typically sorted in an order based on value priority

Note 1 to entry: Backlog traditionally means things that are not completed.

**3.10**
**continuous delivery**
software engineering *practices* (3.20) that allow for frequent releases of new systems (including software) to staging or various test environments through the use of automated tools

[SOURCE: ISO/IEC/IEEE 32675:2022, 3.1]

**3.11**
**continuous deployment**
automated *process* (3.21) of deploying changes to production by verifying intended features and validations to reduce risk

[SOURCE: ISO/IEC/IEEE 32675:2022, 3.1]

**3.12**
**continuous improvement**
ongoing cycle of identification, implementation and evaluation of changes to how *practices* (3.20) are used that strengthen the ability of an organization or project to meet its objectives

**3.13**
**definition of done**
**DoD**
statement on the required quality attributes that work meets before the work completes a specified life cycle activity or task and is ready for use

**3.14**
**definition of ready**
statement on the required quality attributes a *backlog* (3.9) item or other work product meets before it starts a specified activity or task

**3.15**
**gemba**
observation of work *processes* (3.21) actually in use

**3.16**
**improvement backlog**
intentional, emergent, ordered list of items of how to make improvements

**3.17**
**increment**
tested, deliverable version of a software product that provides new or modified capabilities

**3.18**
**iteration**
short time frame in which a set of software features is developed, leading to a working product that can be demonstrated to stakeholders

Note 1 to entry: Some *agile* (3.2) methodologies are not based on iterations.

[SOURCE: ISO/IEC/IEEE 26515:2018, 3.10, modified — The original note 1 to entry has been removed; note 2 to entry has been renumbered as note 1 to entry.]

**3.19**
**iteration backlog**
subset of a *backlog* (3.9) chosen for inclusion in the current *iteration* (3.18)

**3.20**
**practice**
way of consistently performing activities that contributes to achieving a specific purpose

**3.21**
**process**
set of interrelated or interacting activities that transforms inputs into outputs

[SOURCE: ISO/IEC/IEEE 12207:2017, 3.1.33]

**3.22**
**product backlog**
prioritized listing of product requirements

**3.23**
**product owner**
designated stakeholder accountable for defining and accepting outcomes of the work and managing *backlog* (3.9), while aligning with the stakeholder needs

**3.24**
**release backlog**
subset of *product backlog* (3.22) planned for the next release

**3.25**
**statement of work**
statement of the expected outcomes and outline of the work required to achieve the outcomes

Note 1 to entry: Statement of work is unique and distinct for each project and often provides a measurable *definition of done* (3.13). It is usually a contractual document prepared during project initiation and planning that describes what outcomes are expected and outlines the work required to achieve these outcomes.

**3.26**
**testing pyramid**
graduated series of tests which includes many simple and automated tests (unit tests) with less frequent integration tests and few lengthy end-to-end or manual tests

**3.27**
**user story**
brief description of desired functionality

Note 1 to entry: User story can describe the stakeholder roles, goals, benefits, and motivation

Note 2 to entry: Work in backlog can be referred in other ways like epics, features

**3.28**
**working agreement**
statement of the expected working relationship between two or more parties in an agile environment

EXAMPLE Between members of an *agile team* (3.6), between acquirer and supplier, or between multiple agile teams.

# 4 Conformance

## 4.1 Intended usage

The requirements in this document are contained in Clause 6. This document provides requirements for agile practices suitable for usage by an organization or project. Particular projects or organizations may need only some of the practices provided by this document. Therefore, implementation of this document typically involves selecting and declaring a set of practices suitable to the organization or project. Claiming full conformance asserts that all of the required outcomes of the declared set of practices are achieved.

NOTE An organization (e.g. national or industrial association or company) imposing this document as a condition of trade can specify and make public the minimum set of required practices, and outcomes which constitute suppliers' compliance with the conditions of trade.

## 4.2 Full conformance

A claim of full conformance declares the set of practices for which conformance is claimed. Full conformance to outcomes is achieved by demonstrating that all of the outcomes of the declared set of practices have been achieved.

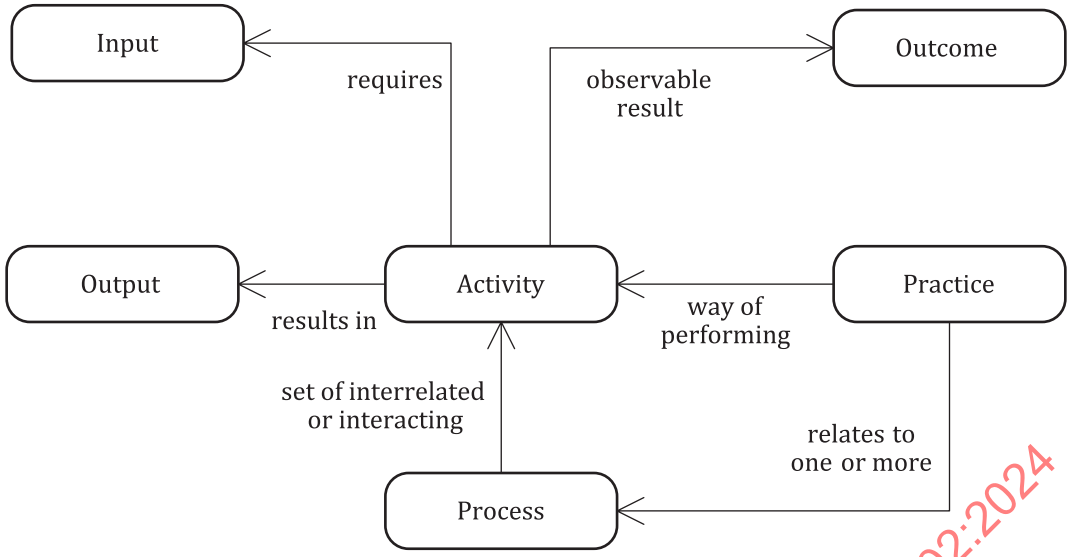
# 5 Conceptual foundations

## 5.1 Practices and processes

Processes are often considered as a high-level view of the organization or project's activities. These activities require the allocation of resources such as people, tools and materials. Every process has a clearly identified input and output. Such inputs and intended outputs can be tangible or intangible.

Practices are pragmatic actions that are repeatable, regular and recognizable in a context. They describe what really happens rather than what is expected to happen. A practice can be considered as a sound way of working on a problem so that a certain specified quality results in the solution. A practice is a way of consistently performing activities that contributes to achieving a specific purpose. The specifics of the practice are left to the individual or team who make the practice work based on their knowledge and skills.

Figure 1 depicts key concepts pertaining to practices and processes.




NOTE This figure uses an informal entity-relationship diagram notation to facilitate comprehension by readers of this document. In this figure, rounded rectangles represent information objects; and arrows represent relationships between objects with the annotation read in the arrow direction.

**Figure 1 — Relationship between process, practice and activity**

Over a period of time, many alternate practices can coalesce to become best practices, for some given situations. If all the practices coalesce into a single best practice, then it becomes the only way to accomplish a task in the organization or project. Such a best practice evolves and can be encoded as a standard for the organization and the industry.

While processes are rigid with inherent dependencies, practices are flexible and adaptable ways of working that serve as guidelines that empower individuals and teams. While the purpose of a process is to produce an acceptable or satisfactory deliverable with desired quality characteristics, the purpose of a practice is to apply the idea, belief or method for different situations. Consequently, in this document, the term practice is utilized to encode the agile way of working.

## 5.2 Agile practices and concepts

This document is intended to be applied in organizations and software projects using agile practices and concepts. In a complex project environment with many dependencies, vague requirements, high market dynamics and unpredictable changes, agile development approaches help to minimize the risk of failure and to constantly deliver value to the customer. For this purpose, the software product is developed in small steps (product increments) and the stakeholders are closely involved and have the responsibility to provide regular feedback. In large software development efforts as well as in smaller projects, many agile practices can be used with various life cycle models, and different practices can be used at different points in the life cycle (see Annex B).

NOTE See Annex D for more information about the application of agile.

## 5.3 Agile practices

This document provides agile practices that can be utilized while adopting agile in organizations or projects. Each of the practices is described in terms of its purpose and desired outcomes. The practices described in this document are not intended to preclude or discourage the use of additional practices that organizations may find useful.

The practices in this document can be used by any organization fully or partially while using, creating, or supplying a software product or service. The sequence that the practices are presented in, does not imply any prescriptive order in their use. Concurrent use of practices may exist in a project and between projects.

NOTE This document lists practices and outcomes that are specific to agile. This list of outcomes and practices is not comprehensive; other valid practices and outcomes that are not specific to agile are also normally applied.

## 5.4 Criteria for practices

The determination of agile practices is based upon three basic principles:

a) each agile practice has strong relationships to its outcomes;

b) the dependencies between the practices are minimal;

c) a practice can be adopted by a single organization or project.

## 5.5 Description of practices

Each practice is described in terms of the following attributes:

a) title: the title summarizes the practice; it identifies the principal concern of the practice and distinguishes the practice from other practices;

b) purpose: the purpose describes the overall goal for performing the practice at a high level;

c) overview: the overview describes some basic facts of the practice;

d) outcomes: the outcomes describe the observable results that are expected on successful execution of the practice.

## 5.6 Some agile concepts

### 5.6.1 Cadence

Cadence establishes a predictable schedule around events and activities. One such capability is the length of the development cycle and related activities and events leading to the capacity of predicting project outcomes. Cadence establishes a sequence of repeatable time-bound activities and events which leads to a predictable, synchronized rhythm of product development with appropriate overlaps. All the other down-stream activities depend on this rhythm, thereby enabling the agile project team to know what they are doing and when will the work get done. This enables the team to make reliable commitments on their deliverables as the team obtains a fair idea of their capability and capacity from the cadence. There are many factors that impact the selection of the cadence: criticality, risk, release frequency, success criteria, priorities, type of project, and so on. Cadence is used in iteration-based development approaches like iteration planning, iteration review, iteration retrospective, and so on.

### 5.6.2 Gemba

The gemba provides ways and means to observe the actual work processes, identify the bottlenecks and explore opportunities for continuous improvement. This includes leaders being in touch with the agile team, observing their work and participating in agile events, monitoring progress of work and providing feedback. The gemba provides for observation, identification, analysis and improvement of work processes for different agile programs and projects.

### 5.6.3 Limiting work in progress

The limiting work in progress provides ways and means to identify and eliminate inefficiencies, remove bottlenecks, reduce risks, optimize work capacity, establish optimal pace of work, thereby improving the ability of the team to meet the release goals, focus the team's effort and improve throughput.

# 6 Core agile practices

## 6.1 Agreement practice

### 6.1.1 Purpose

The purpose of the agreement practice is to establish the working agreement between parties involved in an agile endeavour.

NOTE The primary parties are the acquirer and the supplier. There can also be an agreement between the product owner and the development team. There can be different kinds of agreement items.

### 6.1.2 Overview

The agreement practice values customer collaboration over contract negotiation. This practice encourages customers and the development team to collaborate to chart the best way forward together, rather than to view each other as adversaries. The agreement practice provides for establishing a working agreement based on specific terms, conditions, capacity and deliverables.

### 6.1.3 Outcomes

As a result of the successful implementation of the agreement practice:

a) a statement of work shall be established;

b) a contract agreement shall be established;

c) terms and conditions of engagement shall be established;

d) service-level agreements should be established.

## 6.2 Acquisition practice

### 6.2.1 Purpose

The purpose of the acquisition practice is to obtain a product or service in accordance with the acquirer's requirements, where speed of value delivery and/or responsiveness to change are key expectations.

### 6.2.2 Overview

The acquirer should define a strategy for how the acquisition should be conducted and prepare a request for the supply of the product or service in accordance with agile requirement management practices.

### 6.2.3 Outcomes

As a result of the successful implementation of the acquisition practice:

a) one or more suppliers capable of meeting the speed/responsiveness objectives of the acquirer shall be identified;

b) an agreement that mutually supports the speed/responsiveness objectives of the acquirer and supplier should be established;

c) an agreement that limits the financial obligations of the acquirer, consistent with the expected uncertainty of the effort (e.g. technical scope) should be established.

## 6.3 Supply practice

### 6.3.1 Purpose

The purpose of the supply practice is to provide an acquirer with a product or service that meets agreed requirements, where either or both of the following circumstances apply:

a) the key expectations of the acquirer include speed of value delivery and/or responsiveness to change;

b) the supplier gains an important business or mission advantage through an approach that emphasizes value delivery and/or responsiveness to change (e.g. faster return on investment).

### 6.3.2 Overview

The supplier should define a supply strategy towards identifying potential acquirers for the product or service and responding to requests for supply of the product or service from acquirers.

### 6.3.3 Outcomes

As a result of the successful implementation of the supply practice, an agreement that mutually supports the speed/responsiveness objectives of the acquirer and supplier shall be established.

## 6.4 Cross-functional team practice

### 6.4.1 Purpose

The purpose of the cross-functional team practice is to ensure the agile team has all the required skills to deliver and not restrict the application of skills to certain individuals.

### 6.4.2 Overview

The cross-functional team practice provides for establishing a friendly and collaborative atmosphere which can deal effectively with cross-functional dependencies. All the necessary skills for delivering a successful incremental product are available within the team. This allows for a less structured, more interactive, highly performant team that can leverage the experience and variety in the team to make better decisions.

### 6.4.3 Outcomes

As a result of the successful implementation of the cross-functional team practice:

a) joint ability of the team to deliver shall be established;

b) cross-functional dependencies shall be identified and managed;

c) an agile team comprising resources from different functional areas shall be established.

## 6.5 Feature team practice

### 6.5.1 Purpose

The purpose of the feature team practice is to define a set of end-to-end features and the order in which they need to be delivered for a cross-functional, cross-component, long-lived team that leads to a potentially shippable increment. Feature teams are used where end-to-end functionalities are required.

### 6.5.2 Overview

The feature team practice provides for establishing an end-to-end feature ownership and delivery mechanism which can deal effectively with shipping a potentially shippable increment on schedule. All the

skills that are necessary to deliver a successful incremental product are available within the team. Further, the team should focus on delivering one end-to-end customer centric complete feature at a time. This allows for a focused, less structured, more interactive, highly performant team that are self-organized and can leverage the expertise in the team to deliver high quality products.

### 6.5.3 Outcomes

As a result of the successful implementation of the feature team practice:

a) dependencies between the teams shall be minimized;

b) a set of end-to-end features and the order of delivery shall be established;

c) common working goals, working criteria and objectives shall be established;

d) an agile team that works on many features shall be established;

e) a working agreement, required resources and logistics should be established;

f) customer-centricity of the product should be increased.

## 6.6 Component team practice

### 6.6.1 Purpose

The purpose of the component team practice is to define a set of components that are owned, developed, maintained, and delivered by a focused component development team which contributes to solutions. Component teams are used where special competencies or architectures are required.

### 6.6.2 Overview

The component team practice provides for establishing an end-to-end component ownership and delivery mechanism which can deal effectively with shipping a customer-valuable solution. All the skills that are necessary to develop, maintain and deliver these components are available within the team. This allows for a focused, interactive, tightly coupled team that are self-organized and can leverage the expertise in the team to deliver high quality reusable components. Typical things that are detailed out and agreed upon are: set of components, roles, responsibilities, architecture robustness, goals and objectives, resources and logistics, and so on.

### 6.6.3 Outcomes

As a result of the successful implementation of the component team practice:

a) common working goals, criteria and objectives shall be established;

b) component solutions shall be delivered;

c) an agile team that focuses on specific set of components shall be established;

d) a working agreement, necessary resources and logistics should be established.

## 6.7 Co-located team practice

### 6.7.1 Purpose

The purpose of the co-located team practice is to define strategies, resources and goals to be achieved in order to achieve high productivity, collaboration, efficient and effective communication.

### 6.7.2 Overview

The co-located team practice provides for establishing a physically co-located team-based delivery mechanism. The goals and objectives of the team are well understood and all the skills that are necessary to achieve the goals and objectives are co-located together. This allows for a focused, interactive, productive, tightly coupled team that are self-organized and can deliver at high rates.

### 6.7.3 Outcomes

As a result of the successful implementation of the co-located team practice:

a) an agile team with open communication based out of a common location shall be established;

b) a working agreement, necessary resources, location, and logistics should be established.

## 6.8 Distributed team practice

### 6.8.1 Purpose

The purpose of the distributed team practice is to define strategies, resources, and goals to be achieved to deal with organizational scale-up, global reach, harnessing global talent, quicker time to market, and increasing production costs.

### 6.8.2 Overview

The distributed team practice provides for establishing a physically geographically distributed team-based delivery mechanism that scales up with less investment, deals with technical and domain complexity, deals with customers spread across different geographies, and reaches geographically distributed talents with requisite skills. Communication and collaboration occur using virtual spaces which allows a team located geographically at different locations to come together for delivering a solution.

### 6.8.3 Outcomes

As a result of the successful implementation of the distributed team practice:

a) ability to leverage global talent shall be established;

b) an agile team that works out of different geographical locations shall be established;

c) the workflow, delivery schedules, decision guidelines and working agreement should be established;

d) virtual communication, collaboration and cooperation should be established.

## 6.9 Collective ownership practice

### 6.9.1 Purpose

The purpose of the collective ownership practice is to define strategies, resources and goals to own the deliverable in its entirety and share responsibility resulting in improved quality.

### 6.9.2 Overview

The collective ownership practice provides for cooperative ownership and collective responsibility of the deliverables by all the members of the team, encourages contribution of new ideas to all segments of the deliverables, focuses on creative problem solving, and establishes an explicit convention that every team member can own, create and update the deliverables.

### 6.9.3 Outcomes

As a result of the successful implementation of the collective ownership practice:

a) an agile team with responsibility and authority of all the decisions shall be established;

b) working agreements should be established;

c) communication channels, decision and design guidelines should be established;

d) dependency on individuals or between teams should be reduced.

## 6.10 Release planning practice

### 6.10.1 Purpose

The purpose of the release planning practice is to establish the right business and functional goals for the subsequent release and to assert the ability and feasibility of meeting the goals.

### 6.10.2 Overview

The release planning practice provides for establishing the delivery goals, priorities, delivery ownership while considering the uncertainty, intangibility, mission debt and flexible nature of software development.

### 6.10.3 Outcomes

As a result of the successful implementation of the release planning practice:

a) the release vision, release plan and release schedule shall be established;

b) the product backlog and impediment register shall be updated;

c) the release shall be aligned to the planned business value realization;

d) a release communication plan shall be established for the relevant stakeholders;

e) prioritised business features shall be planned for the subsequent release;

f) technical debt [11] and functional dependencies should be made visible;

g) dependencies and mitigation plans should be established;

h) team composition and configuration should be established;

i) team should be committed to the release plan;

j) delivery goals, ownership and priorities should be established.

## 6.11 Deployment planning practice

### 6.11.1 Purpose

The purpose of the deployment planning practice is to establish the right systematic, repeatable approach for transitioning the developed software into the target production environment.

### 6.11.2 Overview

The deployment planning practice provides for defining and governing specific activities for release management and artefacts management as part of deployment.

### 6.11.3 Outcomes

As a result of the successful implementation of the deployment planning practice:

a) a deployment plan shall be established for the relevant stakeholders;

b) a plan for rollback and/or fix-forward shall be in place;

c) all released functions, services and capabilities should be available;

d) the team should be informed and engaged.

## 6.12 Iteration planning practice

### 6.12.1 Purpose

The purpose of the iteration planning practice is to establish the delivery goals and related commitments for the iteration.

### 6.12.2 Overview

The iteration planning practice helps in establishing a sensible commitment at the beginning of each iteration by determining the value that can be created by delivering high priority features.

### 6.12.3 Outcomes

As a result of the successful implementation of the iteration planning practice:

a) iteration goals and iteration backlog shall be established;

b) acceptance criteria, estimates, risk register and product backlog shall be updated;

c) shared understanding between the development team and product owner shall be established;

d) predictability of project outcomes shall be enhanced;

e) the iteration backlog should incorporate items from the improvement backlog;

f) team commitment towards the iteration goal should be established.

## 6.13 Synchronized iteration practice

### 6.13.1 Purpose

The purpose of the synchronized iteration practice is to establish a common synchronization at which all the iterations are initiated, performed or terminated in a project across multiple agile teams.

### 6.13.2 Overview

The synchronized iteration practice provides for establishing a common synchronization across multiple agile teams in a project. The iterations can start or stop at around the same schedule or the respective durations can also be similar.

### 6.13.3 Outcomes

As a result of the successful implementation of the synchronized iteration practice:

a) the solution shall be integrated periodically across all the teams;

b) the iteration schedule and duration should be established;

c) dependency management between agile teams should be improved;

d) collaboration and alignment between agile teams should be improved.

## 6.14 Daily synchronization meeting practice

### 6.14.1 Purpose

The purpose of the daily synchronization meeting practice is to facilitate the progress of the agile team towards achieving the iteration goals and commitments by effectively dealing with risks, dependencies, impediments, related information and alignment.

### 6.14.2 Overview

The daily synchronization meeting practice offers a platform for communication between the team members.

NOTE Daily synchronization meeting is commonly referred to as the "daily scrum" or "daily standup" when using the scrum framework.

### 6.14.3 Outcomes

As a result of the successful implementation of the daily synchronization meeting practice:

a) the sprint backlog, impediments and risks shall be updated;

b) sprint goals shall be aligned with the Iteration goals;

c) the team task board and iteration goals should be updated;

d) dependencies and alignment issues should be made visible.

## 6.15 Cross-team synchronization meeting practice

### 6.15.1 Purpose

The purpose of the cross-team synchronization meeting practice is to facilitate the progress of multiple agile teams towards achieving the collective release goal, team iteration goals and commitments by effectively dealing with risks, dependencies, impediments, related information and alignment.

### 6.15.2 Overview

The cross-team synchronization meeting practice provides ways and means for establishing a fundamental change in how responsibilities are met by different teams; improving the alignment between the different teams and providing requisite information pertaining to the progress towards achieving the release goals.

### 6.15.3 Outcomes

As a result of the successful implementation of the cross-team synchronization meeting practice:

a) the product backlog, iteration goals and release goals shall be updated;

b) impediments and risk register shall be updated;

c) dependencies between multiple agile teams should be reduced;

d) individual team plans and iteration goals shall be updated;

e) alignment towards the release goals should be improved;

f) cross-team dependencies and impediments should be made visible and addressed;

g) cross-team alignment issues should be made visible.

## 6.16 Product owner synchronization meeting practice

### 6.16.1 Purpose

The purpose of the product-owner synchronization meeting practice is to optimize the value throughput of the organization.

### 6.16.2 Overview

The product owner synchronization meeting practice provides ways and means for establishing a fundamental change in how responsibilities are met by providing visibility into progress and impediments.

### 6.16.3 Outcomes

As a result of the successful implementation of the product owner synchronization meeting practice:

a) program backlogs, program goals and release goals shall be updated;

b) all backlogs and goals shall be aligned to product vision;

c) program impediments shall be made visible and resolved;

d) business priorities shall be aligned, and status updated;

e) program alignment issues should be made visible;

f) program dependencies should be addressed.

## 6.17 Value stream mapping practice

### 6.17.1 Purpose

The purpose of the value stream mapping practice is to establish ways and means for analysing, improving the flow of information, reducing and eliminating waste by discovering the root causes and sources of waste thereby increasing efficiency, throughput and effectiveness.

### 6.17.2 Overview

The value stream mapping practice continuously optimizes the value stream in pursuit of maximum flow and quality. It provides ways and means to outline the flow of information to determine where value is added for the different stakeholders. Additionally, bottlenecks, obstacles and things that do not add value are determined.

### 6.17.3 Outcomes

As a result of the successful implementation of the value stream mapping practice:

a) product backlogs shall be updated;

b) the current state value stream map shall be established;

c) information flows should be established;

d) key performance indicators should be established and monitored;

e) waste, inefficiencies, and opportunities for error should be identified and impact quantified;

f) based on the quantified impact, process optimizations for value flow should be identified, implemented, or streamlined;

g) information flows should be updated based on the process optimizations.

## 6.18 Release review practice

### 6.18.1 Purpose

The purpose of the release review practice is to validate the potentially shippable increment and verify if the potentially shippable increment is ready for release.

### 6.18.2 Overview

The release review practice provides for verification and validation of the potentially shippable increment. A review does not perform verification, which is usually done by testing. The review can confirm that verification has been performed and that the increment is ready for release. Also, a review of one release does not align business needs and future releases, unless features are withheld for rework and inclusion in future releases.

### 6.18.3 Outcomes

As a result of the successful implementation of the release review practice:

a) business needs and future releases shall be aligned;

b) the improvement backlog should be updated;

c) the decision to release should be established;

d) continuous improvement areas should be established;

e) strategies for future releases should be established;

f) effectiveness and efficiency of the product should be improved.

## 6.19 Iteration review practice

### 6.19.1 Purpose

The purpose of the iteration review practice is to inspect the outcome and determine future adaptations.

### 6.19.2 Overview

The iteration review practice provides for verification and validation of the iteration progress.

NOTE The iteration review practice is also referred to as the "sprint review", when using the scrum framework

### 6.19.3 Outcomes

As a result of the successful implementation of the iteration review practice:

a) the definition of done and the definition of ready shall be updated;

b) the product backlog should be updated;

c) the continuous feedback loop should be initiated;

d) iteration alignment towards iteration goals should be assessed and improved;

e) continuous improvement areas should be established;

f) effectiveness and efficiency of the product should be improved.

## 6.20 Team retrospective practice

### 6.20.1 Purpose

The purpose of the team retrospective practice is to inspect the current way of working with respect to people, processes, resources, tools, methods, and technologies and establish actions for improvement.

### 6.20.2 Overview

The team retrospective practice provides for self-reflection and causal analysis on a completed iteration and establishes improvements and future adaptations on the way forward.

NOTE The team retrospective practice is also referred to as the "sprint retrospective", when using the scrum framework.

### 6.20.3 Outcomes

As a result of the successful implementation of the team retrospective practice:

a) the improvement backlog shall be updated;

b) future adaptations and improvement areas shall be established;

c) a collaborative and communicative environment should be created;

d) effectiveness and efficiency of the team should be improved.

## 6.21 Program retrospective review practice

### 6.21.1 Purpose

The purpose of the program retrospective practice is to evaluate the extent of value delivered over multiple releases and establish the way forward for the program.

### 6.21.2 Overview

The program retrospective practice provides for self-reflection and causal analysis “in the way of working of the team” on a completed release and establishes improvements and future adaptations on the way forward.

### 6.21.3 Outcomes

As a result of the successful implementation of the program retrospective review practice:

a) the program backlog, the product backlog, the release backlog and the improvement backlog shall be updated;

b) future adaptations and improvements shall be established;

c) plans to improve delivery efficiency should be established;

d) continuous improvement areas should be established;

e) significant impediments affecting the program should be addressed.

## 6.22 Program board governance practice

### 6.22.1 Purpose

The purpose of the program board governance is to manage the dependency of product backlog items across multiple agile teams.

### 6.22.2 Overview

Program governance boards are established to represent the organization's interests in the agile project which would involve making organization specific decisions related to the progress of the various agile projects. The program board governance practice provides for governance of the technical, domain and functional dependencies items across multiple agile teams and enables agile teams to address these.

### 6.22.3 Outcomes

As a result of the successful implementation of the program board governance practice:

a) future adaptations to resolve dependencies shall be established;

b) the program backlog, the iteration plan and milestones shall be updated;

c) the dependency matrix should be established and updated;

d) risks and dependencies should be captured and mitigated;

e) domain, technical, functional and resource dependencies should be addressed.

## 6.23 Agile adoption improvement practice

### 6.23.1 Purpose

The purpose of the agile adoption improvement practice is to manage the maturity of agile practices and their adoption by agile teams and enable continuous improvement.

### 6.23.2 Overview

The agile adoption improvement practice provides for evaluation, management and improvement of the maturity of agile practices and their adoption by agile teams.

### 6.23.3 Outcomes

As a result of the successful implementation of the agile adoption improvement practice:

a) unplanned changes shall be identified and coped with;

b) planned changes shall be monitored and managed;

c) improvement plans should be updated;

d) predictability should be increased;

e) stakeholders' satisfaction should be improved;

f) agile practices should be successfully implemented.

## 6.24 Visual management practice

### 6.24.1 Purpose

The purpose of the visual management practice is to provide ways and means to build new, flexible and better ways of working in relevant areas.

### 6.24.2 Overview

The visual management practice provides for organization, management, and communication of large sets of information that are relevant for the different agile programs and projects.

### 6.24.3 Outcomes

As a result of the successful implementation of the visual management practice:

a) progress towards achieving DoD shall be visualized using one or more visualization techniques;

b) flow of information and materials between various resources shall be visualized using one or more visualization techniques;

c) value-adding activities should be visualized using one or more visualization techniques;

d) waste should be visualized using one or more visualization techniques;

e) work organization should be improved;

f) transparency should be improved;

g) quick communication should be achieved.

## 6.25 Distributed agile governance practice

### 6.25.1 Purpose

The purpose of the distributed agile governance practice is to enable agile teams to self-organize, collaborate, and adapt to geographically distributed environments and deliver the desired value.

### 6.25.2 Overview

The distributed agile governance practice provides ways and means for self-organization, collaboration, and adaptation to different environments to improve the efficiency of the agile teams.

### 6.25.3 Outcomes

As a result of the successful implementation of the distributed agile governance practice:

a) a shared vision and understanding shall be established;

b) clear directives and policies shall be established;

c) goals and milestones shall be clearly defined;

d) a sense of ownership should be established;

e) the definition of done shall be clearly articulated;

f) business efficiency should be improved;

g) transparency should be improved;

h) trust between distributed teams should be improved;

i) right skills and roles should be utilized appropriately.

## 6.26 User stories practice

### 6.26.1 Purpose

The purpose of the user stories practice is to provide ways and means to gather stakeholder needs and translate them into specific requirements, features, and capabilities from an end-user point of view.

### 6.26.2 Overview

The user stories practice provides for identification and elicitation of needs as user stories from an end-user perspective.

### 6.26.3 Outcomes

As a result of the successful implementation of the user stories practice:

a) needs shall be clearly stated;

b) stated requirements shall be well understood by adequate acceptance criteria;

c) a shared understanding of the needs should be promoted.

## 6.27 Product backlog elicitation practice

### 6.27.1 Purpose

The purpose of the product backlog elicitation practice is to provide ways and means to gather and prioritize high level requirements, constraints, boundaries, high level objectives and work items.

### 6.27.2 Overview

The product backlog elicitation practice is a continuous practice to arrive at the product backlog which is a living and evolving document.

### 6.27.3 Outcomes

As a result of the successful implementation of the product backlog elicitation practice:

a) the product backlog shall be established;

b) priorities of business and other stakeholders should be established;

c) conflicting priorities should be identified and reconciled.

## 6.28 Product backlog refinement practice

### 6.28.1 Purpose

The purpose of the product backlog refinement practice is to refine product backlog items for the current scope of work.

### 6.28.2 Overview

The product backlog refinement practice is a continuous practice that provides for refining the backlog items to the suitable levels of granularity. Not all the backlog items are of the same size or level of detail, the

items that are planned to be worked soon are very detailed, while those that will be taken up later are less detailed.

#### 6.28.3 Outcomes

As a result of the successful implementation of the product backlog refinement practice:

a) the product backlog shall be refined;

b) the product backlog shall be tracked and prioritized;

c) estimates for user stories, features and iteration goals should be updated;

d) multiple aspects of business domain and immediate priorities should be understood;

e) dependencies across agile teams should be understood;

f) priorities of business and other stakeholders should be well understood and any possible conflicts should be resolved.

### 6.29 User story mapping practice

#### 6.29.1 Purpose

The purpose of the user story mapping practice is to facilitate the gathering, analysis, understanding and organizing requirements and managing their subsequent changes within a particular context and in prioritizing product delivery.

#### 6.29.2 Overview

The user story mapping practice provides the ways and means to develop views, identify dependencies, establish desired functionality, identify gaps, plan iterations and releases, and deliver value to stakeholders.

#### 6.29.3 Outcomes

As a result of the successful implementation of the user story mapping practice:

a) the product backlog shall be refined;

b) the release backlog shall be refined;

c) target outcomes and the corresponding impact should be established;

d) relationships between user stories should be established;

e) dependencies across agile teams should be understood;

f) potential risks should be identified.

### 6.30 Relative estimation practice

#### 6.30.1 Purpose

The purpose of the relative estimation practice is to determine the estimates or measures for different entities relative to a point of reference.

### 6.30.2 Overview

The relative estimation practice provides the ways and means of estimating or measuring different kinds of entities by determining the relative difference or scale from a point of reference. The relative estimation practice provides credible estimates to determine cost and effort, establish priorities and forecast a schedule.

### 6.30.3 Outcomes

As a result of the successful implementation of the relative estimation practice:

a) a realistic estimate of release effort should be established;

b) a realistic estimate of iteration effort should be established;

c) user stories and features should be refactored into achievable increments for estimating purposes;

d) accuracy of estimation should be improved.

## 6.31 Active stakeholder participation practice

### 6.31.1 Purpose

The purpose of the active stakeholder participation practice is to establish the access, interaction and communication channels between stakeholders and agile teams.

### 6.31.2 Overview

The active stakeholder participation practice provides the ways and means to ensure active involvement of stakeholders in the development and delivery of the product. This involves establishing the access to the stakeholders, establishing communication channels, and having continuous interactions with the stakeholders.

### 6.31.3 Outcomes

As a result of the successful implementation of the active stakeholder participation practice:

a) openness should be increased;

b) accountability should be established;

c) the degree of information exchange between team and stakeholders should be established;

d) clarity on stakeholder needs and priorities should be established;

e) effectiveness and impact of the release should be increased;

f) stakeholders should be involved in decision making;

g) stakeholder needs should be clearly stated;

h) stakeholder engagement and interactions should be ensured.

## 6.32 Initial architecture practice

### 6.32.1 Purpose

The purpose of the initial architecture practice is to lay the essential architectural foundations for development so that implementation can begin early. This means that the basic architectural decisions that

bear the highest risks are made first, details and not so risky decisions can be postponed to later project phases (last responsible moment). See the incremental architecture practice (6.35).

NOTE The last responsible moment (LRM) is a strategy of delaying a decision until the moment when the cost of not making the decision is greater than the cost of making it.

### 6.32.2 Overview

The initial architecture practice provides ways and means to establish the initial technical architecture to establish a technical strategy within the agile team and the various stakeholders.

### 6.32.3 Outcomes

As a result of the successful implementation of the initial architecture practice:

a) the product backlog shall be updated;

b) the architecture vision and roadmap shall be established;

NOTE 1 The architecture vision describes how the new products or services will meet the goals, objectives, requirements and address the stakeholder concerns.

c) the architecture strategy should be established;

NOTE 2 The architecture strategy translates business strategy into objectives for building, enhancing or replacing products or services.

d) technical challenges should be identified;

e) technical risk should be reduced;

f) a prioritized list of the 3 to 5 most important quality goals for the project should be identified;

NOTE 3 A model for product quality is defined in ISO/IEC 25010.

g) list of the required technical and organizational constraints should be established;

h) delimitation of the system from the environment (system-context) should be established;

i) communication should be improved.

## 6.33 Intentional architecture practice

### 6.33.1 Purpose

The purpose of the intentional architecture practice is to define a set of purposeful architectural strategies and principles to enhance the design, performance and usability of the solution.

### 6.33.2 Overview

The intentional architecture practice provides ways and means to define a set of architectural strategies, principles, and directives to enhance solution design, performance, extensibility and usability by leveraging common architectural patterns, design constraints and implementation technologies.

### 6.33.3 Outcomes

As a result of the successful implementation of the intentional architecture practice:

a) the product backlog shall be updated;

b) room for flexibility and extensibility shall be provided;

c) higher confidence about product feasibility shall be established;

d) the architecture vision and strategy should be updated;

e) the problem being addressed should be clearly understood;

f) the potential solutions should be clearly understood;

g) the architecture's key concepts, properties and principles should be clearly defined;

h) the architecture should be clearly conceived and key trade-offs understood.

## 6.34 Architecture description practice

### 6.34.1 Purpose

The purpose of the architecture description practice is to define the representation of the solution in a form that provides the ability to portray, understand or predict the solution characteristics under certain situations of interest.

NOTE 1 The concept of architecture description is addressed in ISO/IEC/IEEE 42010.

NOTE 2 The requirements on architecture description are stated in ISO/IEC/IEEE 42010.

### 6.34.2 Overview

The architecture description practice provides ways and means to establish a representation of an architecture to illustrate a specific set of information items that are relevant to the agile teams and stakeholders.

### 6.34.3 Outcomes

As a result of the successful implementation of the architecture description practice:

a) the system structure and behaviour shall be visualized;

b) architecture decisions shall be documented;

c) system dependencies should be exposed;

d) the architecture should be amenable for analysis;

e) system complexity should be understood.

## 6.35 Incremental architecture practice

### 6.35.1 Purpose

The purpose of the incremental architecture practice is to make outstanding decisions from initial architectural practice (see 6.32), to refine architectural designs, to address cross-cutting concerns, and to set up the necessary technical infrastructure to implement new features and capabilities without extensive redesign and delay, thus ensuring overall speed.

### 6.35.2 Overview

The incremental architecture practice provides ways and means to establish a technical infrastructure for implementing new features and capabilities with minimal or no change in the existing architecture or agile environment. The objective is to extend the architecture as and when necessary.

### 6.35.3 Outcomes

As a result of the successful implementation of the incremental architecture practice:

a) the product backlog shall be updated;

b) a continuous delivery pipeline shall be established;

c) architecture artefacts and fundamental concepts shall be established;

d) the ability to respond to business challenges should be increased;

e) technical debt [11] should be reduced;

f) the velocity should be sustained;

g) economic outcomes should be improved.

## 6.36 Simple design practice

### 6.36.1 Purpose

The purpose of the simple design practice is to keep things as simple as possible, designing a system that is easy to understand and change, while addressing current requirements, moving forward with least resistance, and taking into account required system characteristics ('ilities') to avoid or eliminate the creation of unintended technical debt.

### 6.36.2 Overview

The simple design practice provides ways and means to keep the design of the solution as simple and as relevant as possible to the current requirements so that the iteration and release can move forward with least resistance. The objective is to extend the design as and when necessary.

### 6.36.3 Outcomes

As a result of the successful implementation of the simple design practice:

a) code should be modularized, flexible and extensible;

b) risk due to overdesign should be mitigated;

c) design flaws should be identified and fixed;

d) the cost should be reduced;

e) the system design should be flexible;

f) design decisions should be agnostic to changes.

## 6.37 Test-driven development practice

### 6.37.1 Purpose

The purpose of the test-driven development practice is to identify defects at an early stage of development before increments are integrated.

### 6.37.2 Overview

The test-driven development practice is a test-first practice that involves developers writing a test before implementing the part of the system under test. This is done in so-called micro-increments: developers first write a test that fails, then write just enough code to pass the currently failing test. The test-driven

development practice establishes a fast and immediate feedback loop, increases the quality and leads to a better software design.

### 6.37.3 Outcomes

As a result of the successful implementation of the test-driven development practice:

a) new code shall make a failing unit test pass;

b) the unit test shall be developed only enough to fail;

c) code shall be testable;

d) only the function that is needed should be developed.

## 6.38 Acceptance test-driven development practice

### 6.38.1 Purpose

The purpose of the acceptance test-driven development (ATDD) practice is to improve software quality by specifying and verifying desired behaviour. Practitioners who utilize ATDD avoid ambiguity in requirements by means of a collaborative effort amongst different stakeholders to develop the software smoothly while meeting the acceptance criteria.

### 6.38.2 Overview

The ATDD is more specific and detailed, concentrating on the creation of concrete acceptance tests for individual features or user stories. It involves a collaborative effort among team members from various backgrounds (customer, development, testing) to create acceptance tests before implementing the corresponding functionality. This brings together different perspectives to define the problem, propose solutions, and consider potential issues. The acceptance tests are written from the user's perspective and serve as a set of requirements detailing the system's expected behaviour. Additionally, these tests ensure that the system operates as intended, avoids last-minute changes or modifications, with some teams opting to automate the acceptance tests.

### 6.38.3 Outcomes

As a result of the successful implementation of the acceptance test-driven development practice:

a) comprehension of the needs shall be improved;

b) acceptance tests shall be written from the user's perspective;

c) debugging effort should be reduced;

d) test configurations, frameworks and strategy should be updated;

e) automated tests should be performed and test results gathered;

f) defects should be reduced;

g) stakeholders should own the acceptance tests;

h) precise requirements should be captured;

i) customer engagement should be enhanced.

## 6.39 Behaviour-driven development practice

### 6.39.1 Purpose

The purpose of the behaviour-driven development (BDD) practice is to improve software quality by specifying and verifying desired behaviour. Practitioners who utilize BDD focus on collaboration and shared understanding among stakeholders, using natural language descriptions for identifying the desired systems behaviour.

### 6.39.2 Overview

The BDD is a practice that emphasizes showcasing the behaviour of a system. It promotes the use of conversations and practical examples in simple language to ensure that all stakeholders involved in the development process have a clear understanding of the system's behaviour. BDD draws techniques from different approaches for developing features based on the system's behaviour.

### 6.39.3 Outcomes

As a result of the successful implementation of the behaviour-driven development practice:

a) a shared understanding of the problem shall be developed;

b) behaviour shall be the focus of development and testing;

c) behaviour shall be the primary source of requirements;

d) automated tests should be performed, and test results should be gathered;

e) the test configuration should be updated;

f) debugging effort should be reduced;

g) the testing strategy for specific behaviours should be developed;

h) defects should be reduced;

i) customer engagement should be enhanced.

## 6.40 Feature-driven development practice

### 6.40.1 Purpose

The purpose of the feature-driven development (FDD) practice is to provide for establishing, managing, and developing feature sets comprising features that deliver business value with minimum effort. FDD is customer-centric, iterative, and incremental, with the goal of delivering tangible software results often and efficiently.

### 6.40.2 Overview

The FDD practice provides for establishing, managing and developing feature sets comprising features that deliver business value with minimum effort.

### 6.40.3 Outcomes

As a result of the successful implementation of the feature driven development practice:

a) the definition of ready and the definition of done shall be updated;

b) features shall be the focus of development and testing;

c) features shall be the primary source of requirements;

d) time to market should be improved;

e) customer engagement should be enhanced.

## 6.41 Agile testing pyramid practice

### 6.41.1 Purpose

The purpose of the agile testing pyramid practice is to establish the testing and evaluation strategy for agile development. It establishes a graduated series of tests which includes many simple and automated tests (unit tests) with less frequent integration tests and few lengthy end-to-end or manual tests.

### 6.41.2 Overview

Agile testing involves all members of the project team, special experts and testers and is performed along with development phases that include requirements, design, coding, and generation of test cases. The agile testing pyramid practice provides for establishing, managing and developing test strategies and test automation strategies for agile development with minimum effort.

### 6.41.3 Outcomes

As a result of the successful implementation of the agile testing pyramid practice:

a) a balanced portfolio of automated tests and manual tests shall be established;

b) test levels, entry and exit criteria for each level shall be established;

c) the testing strategy shall be established;

d) the testing process should be streamlined;

e) defects should be reduced;

f) customer engagement should be enhanced.

## 6.42 Continuous integration practice

### 6.42.1 Purpose

The purpose of the continuous integration practice is to get fast feedback, identify defects and ease debugging.

### 6.42.2 Overview

The continuous integration practice provides for establishing, managing and developing system integration strategies for agile development.

### 6.42.3 Outcomes

As a result of the successful implementation of the continuous integration practice:

a) the build shall be automated;

b) test results shall be transparent;

c) code shall be committed frequently;

d) code versioning shall be improved;

e) every commit shall be built;

f) integration effort should be reduced.

## 6.43 Continuous deployment practice

### 6.43.1 Purpose

The purpose of the continuous deployment practice is to automate deployment tasks and create a frictionless deployment process.

### 6.43.2 Overview

The continuous deployment practice provides for establishing, managing and developing system deployment strategies for agile development. It provides automated processes for deploying changes to production by verifying implemented features and validating them to reduce risk.

### 6.43.3 Outcomes

As a result of the successful implementation of the continuous deployment practice:

a) test configurations shall be updated;

b) testing strategies shall be updated;

c) code changes shall be released automatically to production environment;

d) tests shall be automated;

e) Deployment processes shall be automated.

## 6.44 Continuous delivery practice

### 6.44.1 Purpose

The purpose of the continuous delivery practice is to automate delivery tasks and create a frictionless delivery process.

### 6.44.2 Overview

The continuous delivery practice provides for establishing, managing and developing system release strategies for agile development. It allows for frequent releases of new systems (including software) to staging or various test environments using automated tools.

### 6.44.3 Outcomes

As a result of the successful implementation of the continuous delivery practice:

a) code changes shall be deployed to a testing environment;

b) code changes shall be released to production after successful test;

c) a tested deployment-ready built shall be available;

d) software delivery should be more efficient and rapid;

e) product quality should be improved;

f) customer satisfaction should be improved;

g) time to deliver should be faster and more frequent.

## 6.45 Continuous documentation practice

### 6.45.1 Purpose

The purpose of the continuous documentation practice is to establish the documentation strategy for agile development by creating and maintaining documents during the agile development workflow.

### 6.45.2 Overview

The continuous documentation practice provides for establishing, managing and developing documentation strategies for agile development.

### 6.45.3 Outcomes

As a result of the successful implementation of the continuous documentation practice:

a) the communication strategy shall be established;

b) the documentation shall be updated incrementally;

c) the documentation shall be in sync with code base;

d) the drift between code base and code-specific knowledge should be reduced;

e) the documentation should be relevant and valid.

## 6.46 Lean documentation practice

### 6.46.1 Purpose

The purpose of the lean documentation practice is to eliminate waste due to documentation by creating and revising documents that provide exactly the information that is needed, at the right time, to the right audience and nothing more.

### 6.46.2 Overview

The lean documentation practice provides for establishing, managing and developing documentation for relevant stakeholders.

### 6.46.3 Outcomes

As a result of the successful implementation of the lean documentation practice:

a) the documentation strategy shall be updated;

b) the documentation shall provide the right information at the right time to the right audience;

c) the documentation shall be easy to create and update;

d) documentation activities should be updated;

e) the documentation should be small and simple;

f) communication should be improved;

g) relevant documents should be developed;

h) waste due to documentation should be reduced.

## 6.47 Feature toggles practice

### 6.47.1 Purpose

The purpose of the feature toggle practice is to establish ways and means of reducing feature branches.

### 6.47.2 Overview

The features toggle practice provides for establishing, managing and developing strategies for dealing with feature branches.

NOTE Feature toggles are also referred as feature flags.

### 6.47.3 Outcomes

As a result of the successful implementation of the feature toggles practice:

a) the code repository and test configurations shall be updated;

b) feature branches should be reduced;

c) branching risk should be mitigated;

d) merging changes across branches should be removed;

e) code should be modularized, flexible and extensible;

f) dead code should be reduced;

g) dynamic extensibility should be provisioned.

## 6.48 Refactoring practice

### 6.48.1 Purpose

The purpose of the refactoring practice is to restructure software code without altering its behaviour for the purpose of improving quality attributes, easing future extension or adaptation, or adhering to an architectural style.

### 6.48.2 Overview

The refactoring practice provides ways and means for dealing with the degradation in software and improving the ability to maintain and extend the software.

### 6.48.3 Outcomes

As a result of the successful implementation of the refactoring practice:

a) code shall be refactored;

b) code quality should be improved;

c) technical debt [11] should be reduced;

d) delivery outcomes should be improved;

e) code should be clean and simple;

f) code should be consistent and coherent;

g) duplicate code and other design issues should be reduced;

h) the maintenance cost should be reduced.

## 6.49 Test strategy practice

### 6.49.1 Purpose

The purpose of the test strategy practice is to create an understanding of the overall targets, approach, tools and timing of test activities to be done.

### 6.49.2 Overview

The test strategy practice clarifies the major challenges and tasks of the testing in project.

### 6.49.3 Outcomes

As a result of successful implementation of the test strategy practice:

a) the test strategy shall be established;

b) verification coverage should be enhanced;

c) test configurations, frameworks and timing should be updated;

d) cost efficiency of verification should be improved;

e) time to market should be improved;

f) performance should be enhanced.

## 6.50 User story testing practice

### 6.50.1 Purpose

The purpose of the user story testing practice is to define a content, coverage, approach, tools and timing of test activities to be done.

### 6.50.2 Overview

The user story testing practice clarifies how user stories are validated and provides improved quality content for the feature testing.

### 6.50.3 Outcomes

As a result of successful implementation of the user story testing practice:

a) the user story test plan shall be updated;

b) coverage of tests should be improved;

c) defects should be reduced;

d) test configurations should be updated;

e) debugging effort should be reduced;

f) time to market should be improved;

g) transparency around user story testing should be increased.

## 6.51 Feature testing practice

### 6.51.1 Purpose

The purpose of the feature testing practice is to define content, coverage, approach, tools and timing of test activities to be done.

### 6.51.2 Overview

The feature testing practice clarifies how features are tested and provides high quality content for user acceptance testing.

### 6.51.3 Outcomes

As a result of successful implementation of the feature testing practice:

a) the feature test plan shall be updated;

b) coverage of tests should be improved;

c) defects should be reduced;

d) test configurations should be updated;

e) debugging effort should be reduced;

f) time to market should be improved;

g) transparency around feature testing should be increased.

## 6.52 User acceptance testing practice

### 6.52.1 Purpose

The purpose of the user acceptance testing (UAT) practice is to create a common understanding of the overall acceptance criteria, approval approach, responsibilities, tools and timing of test activities to be done.

### 6.52.2 Overview

The UAT practice clarifies how user acceptance testing is done.

### 6.52.3 Outcomes

As a result of successful implementation of the user acceptance testing practice:

a) the UAT test plan shall be established;

b) acceptance test cases shall be agreed;

c) test configurations should be agreed;

d) debugging effort should be reduced;

e) time needed for acceptance should be decreased;

f) transparency around UAT should be increased.

## 6.53 Regression testing practice

### 6.53.1 Purpose

The purpose of the regression testing practice is to create an understanding of the overall regression testing targets, approach, tools and timing of test activities to be done.

### 6.53.2 Overview

The regression testing practice clarifies how regression testing is done during the project.

### 6.53.3 Outcomes

As a result of successful implementation of the regression testing practice:

a) the regression test plan shall be updated;

b) coverage of tests should be improved;

c) the test automation plan should be updated;

d) defects should be reduced;

e) code quality should be enhanced;

f) test configurations should be updated;

g) debugging effort should be reduced;

h) time to market should be improved;

i) transparency around regression testing should be increased.