INTERNATIONAL STANDARD

ISO 24089

First edition
2023-02

Road vehicles — Software update engineering

Véhicules routiers — Ingénierie de mise à jour du logiciel

Reference number
ISO 24089:2023(E)

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

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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 ISO documents 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).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

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.

This document was prepared by Technical Committee ISO/TC 22, *Road Vehicles*, Subcommittee SC 32, *Electrical and electronic components and general system aspects*.

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.

Introduction

Electronic control units and software of increasing complexity have become essential to the operation of road vehicles in recent years. This software is often updated to increase functionality and maintain the safety and cybersecurity of road vehicles.

Today, in-vehicle software is updated in a workshop by a skilled person or automatically over-the-air by the vehicle user. With the increased frequency of software update campaigns, it is important to have individual vehicle configuration information. Therefore, the establishment and application of software update engineering is important to ensure software quality, cybersecurity, and safety.

Software update engineering activities occur throughout the life cycle of vehicles.

This document provides terminology, objectives, requirements, and guidelines related to software update engineering as a foundation for common understanding throughout the supply chain. By applying requirements and recommendations in this document, the following benefits can be achieved for software update engineering:

— safety and cybersecurity are addressed in software update operations in road vehicles;

— establishment of processes, including goal setting, planning, auditing, process monitoring, process measurement, and process improvement;

— shared awareness of safety and cybersecurity among related parties.

Figure 1 shows the overview of this document.

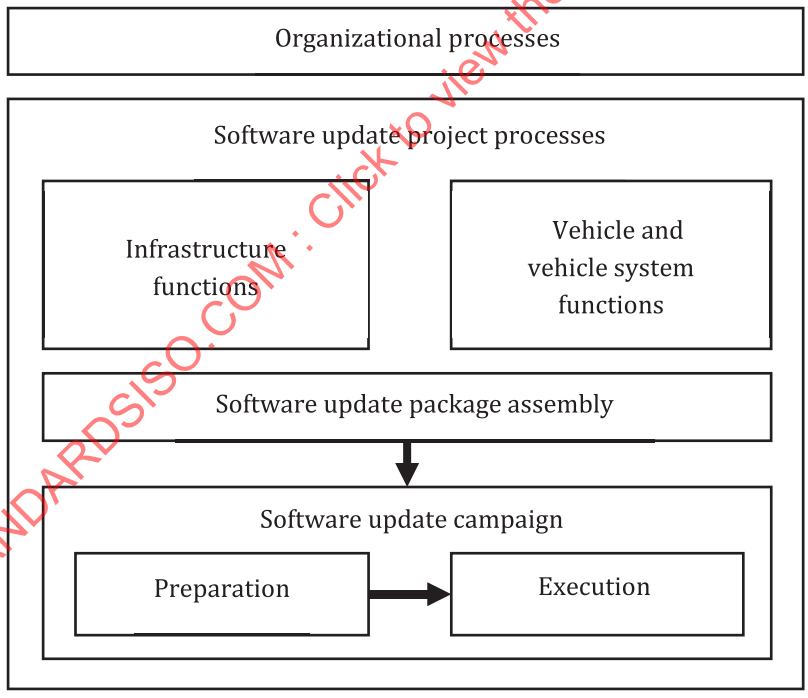

Figure 1 — Overview of this document

In this document, clauses are structured using the following approach:

— each process is defined and implemented before it is executed;

— each process is established, documented and maintained.

This document describes the following activities:

— implementation of organizational level processes for software update engineering;

— implementation of software update project level processes for each software update project;

— definitions of functions for the vehicle and infrastructure to support the activities and processes of this document;

— assembly of software update packages using functions in the infrastructure;

— preparation and execution of software update campaigns using functions in the vehicle and infrastructure.

Road vehicles — Software update engineering

1 Scope

This document specifies requirements and recommendations for software update engineering for road vehicles on both the organizational and the project level.

This document is applicable to road vehicles whose software can be updated.

The requirements and recommendations in this document apply to vehicles, vehicle systems, ECUs, infrastructure, and the assembly and deployment of software update packages after the initial development.

This document is applicable to organizations involved in software update engineering for road vehicles. Such organizations can include vehicle manufacturers, suppliers, and their subsidiaries or partners.

This document establishes a common understanding for communicating and managing activities and responsibilities among organizations and related parties.

The development of software for vehicle functions, except for software update engineering, is outside the scope of this document.

Finally, this document does not prescribe specific technologies or solutions for software update engineering.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 26262-6, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-8, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO/SAE 21434, *Road vehicles — Cybersecurity engineering*

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 General terminology

3.1.1
compatibility
capability of *software* (3.1.15) to be executable on *vehicle systems* (3.1.25) without conflicts

Note 1 to entry: Compatibility can be checked by *vehicle configuration information* (3.1.24).

3.1.2
condition
criteria required for a *software update operation* (3.1.19) to be completed successfully

Note 1 to entry: Conditions can include *compatibility* (3.1.1), *safe vehicle state* (3.1.13), *in-vehicle resources* (3.1.11), and external resources.

EXAMPLE The presence of a *skilled person* (3.1.14) during a software update operation.

3.1.3
corrective action
action to eliminate or contain a problem or failure

3.1.4
cybersecurity
road vehicle cybersecurity
context in which assets are sufficiently protected against threat scenarios to *vehicle systems* (3.1.25) of road vehicles and *infrastructure* (3.1.10) required to support *software update engineering* (3.1.18)

Note 1 to entry: In this document, for the sake of brevity, the term cybersecurity is used instead of road vehicle cybersecurity.

[SOURCE: ISO/SAE 21434:2021, 3.1.9, modified — "to items of road vehicles, their functions and their electrical or electronic components" has been replaced by "to vehicle systems of road vehicles and infrastructure required to support software update engineering" and the Note 1 to entry has been modified.]

3.1.5
cybersecurity risk
effect of uncertainty on *cybersecurity* (3.1.4) expressed in terms of attack feasibility and impact

[SOURCE: ISO/SAE 21434:2021, 3.1.29]

3.1.6
dependency
effect of *software* (3.1.15) for one *vehicle system* (3.1.25) on the same or other *vehicle systems* (3.1.25)

Note 1 to entry: A dependency can generate a *condition* (3.1.2) in the metadata of a *software update package* (3.1.20).

EXAMPLE A communication interface between two *electronic control units (ECUs)* (3.1.7).

3.1.7
ECU
electronic control unit
embedded device in a vehicle whose *software* (3.1.15) can be updated

3.1.8
functional safety
absence of unreasonable risk due to hazards caused by malfunctioning behaviour of *vehicle systems* (3.1.25)

[SOURCE: ISO 26262-1:2018, 3.67, modified — "E/E" was replaced by "vehicle".]

3.1.9
functional safety risk
combination of the probability of occurrence of harm and the severity of that harm

[SOURCE: ISO 26262-1:2018, 3.128, modified — The term has been modified from "risk" to "functional safety risk" for the scope of this document.]

3.1.10
infrastructure
processes and information systems managing any combination of *software update operations* (3.1.19), *software update campaigns* (3.1.16), documentation, and *vehicle configuration information* (3.1.24), including both digital and manual activities

Note 1 to entry: Infrastructure can include any combination of servers, tools, and manual activities used in the software update operation.

3.1.11
in-vehicle resource
vehicle or *electronic control unit (ECU)* (3.1.7) available properties relevant for *software update engineering* (3.1.18)

EXAMPLE Available or remaining computational power, network capacity, RAM capacity, storage capacity, or battery capacity.

3.1.12
recipient
individual instance of a vehicle, *vehicle system* (3.1.25), or *electronic control unit (ECU)* (3.1.7) that receives a *software update package* (3.1.20) during a *software update campaign* (3.1.16)

3.1.13
safe vehicle state
vehicle operating mode based on *conditions* (3.1.2) for performing *software update operations* (3.1.19) without an unreasonable level of risk

Note 1 to entry: Safe vehicle state can be different depending on the *conditions* (3.1.2) required for the *software update package* (3.1.20).

Note 2 to entry: Safe vehicle state can vary based on the software update operation step being performed.

EXAMPLE The motor is off, the parking brake is applied.

3.1.14
skilled person
individual with relevant technical education, training or experience to execute *software update operations* (3.1.19)

Note 1 to entry: A skilled person can be a mechanic in a workshop.

Note 2 to entry: A skilled person can be authorized or certified for their specialized training or be a skilled *vehicle user* (3.1.26).

[SOURCE: ISO 10209:2022, 3.14.36, modified — The phrase "to enable them to perceive risks and avoid hazards occurring during use of a product" has been replaced by "to execute software update operations".]

3.1.15
software
computer programs and associated data intended for *installation* (3.2.2) on vehicles, *vehicle systems* (3.1.25), or *electronic control units (ECUs)* (3.1.7), that may be dynamically written or modified during execution

[SOURCE: NIST SP 800-53, modified — The phrase "intended for installation on vehicles, vehicle systems, or electronic control units (ECUs)" was added.]

3.1.16
software update campaign
sequence of identifying *targets* (3.1.23) and resolving *recipients* (3.1.12); distributing *software update packages* (3.1.20); and monitoring and documenting results of *software update operations* (3.1.19)

3.1.17
software update distribution method
mechanism for delivery of a *software update package* (3.1.20) during a *software update campaign* (3.1.16)

Note 1 to entry: The software update distribution method can be wired (e.g. tool, USB flash drive), wireless (e.g. cellular or Wi-Fi) or hardware replacement.

Note 2 to entry: Hardware replacement can be replacing an *electronic control unit (ECU)* (3.1.7) with the effect of *software* (3.1.15) version replacement.

3.1.18
software update engineering
application of a systematic and managed approach to the processes of planning, development, and deployment of *software update packages* (3.1.20)

[SOURCE: ISO/IEC/IEEE 24765:2017, 3.3810, modified — "disciplined, quantifiable" was replaced by "and managed", and "development, operation and maintenance of software" was replaced by "processes of development, planning, and deployment of software update packages".]

3.1.19
software update operation
steps involved in *receipt* (3.2.1), *installation* (3.2.2) and *activation* (3.2.3) of *software update packages* (3.1.20) in a vehicle, *vehicle systems* (3.1.25), or *electronic control units (ECUs)* (3.1.7)

3.1.20
software update package
set of *software* (3.1.15) and associated metadata that is intended to be deployed to one or more vehicles, *vehicle systems* (3.1.25), or *electronic control units (ECUs)* (3.1.7)

3.1.21
software update project
set of *software update engineering* (3.1.18) activities for one or more *targets* (3.1.23)

Note 1 to entry: Activities can include developing or adapting the *infrastructure* (3.1.10), vehicle capabilities, or processes described in this document.

Note 2 to entry: A software update project can encompass multiple *software update campaigns* (3.1.16).

3.1.22
tailor
to omit or perform an activity in a different manner compared to its description in this document

[SOURCE: ISO/SAE 21434:2021, 3.1.32]

3.1.23
target
one or more classes of vehicles, *vehicle systems* (3.1.25), or *electronic control units (ECUs)* (3.1.7) determined by *vehicle configuration information* (3.1.24)

3.1.24
vehicle configuration information
comprehensive accounting of hardware versions, *software* (3.1.15) versions and configuration parameters in a vehicle

3.1.25
vehicle system
functional group of one or more *electronic control units (ECUs)* (3.1.7) and attached hardware

Note 1 to entry: Attached hardware can be, for example, a sensor, actuator or light, that is not an ECU.

EXAMPLE Braking system or infotainment system.

3.1.26
vehicle user
person operating, driving, owning or managing a vehicle

Note 1 to entry: A vehicle user can be a *skilled person* (3.1.14).

3.2 Terms related to the software update operation

3.2.1
receipt
step in the *software update operation* (3.1.19) when a tool, vehicle, *vehicle system* (3.1.25), or *electronic control unit (ECU)* (3.1.7) receives a *software update package* (3.1.20)

EXAMPLE 1 Downloading a software update package.

EXAMPLE 2 Transferring a software update package using a tool.

3.2.2
installation
step in the *software update operation* (3.1.19) when the relevant parts of a *software update package* (3.1.20) are written to a vehicle, *vehicle system* (3.1.25), or *electronic control unit (ECU)* (3.1.7) but are not yet *activated* (3.2.3)

3.2.3
activation
step in the *software update operation* (3.1.19) when the relevant parts of an *installed* (3.2.2) *software update package* (3.1.20) become executable on a vehicle, *vehicle system* (3.1.25), or *electronic control unit (ECU)* (3.1.7)

EXAMPLE 1 A new automated driving function is *installed* (3.2.2) and ready for execution, but is only executed after the *vehicle user* (3.1.26) starts the function.

EXAMPLE 2 The relevant parts of a software update package for a vehicle, vehicle system, or *ECU* (3.1.7) are installed and executed immediately after activation without user interaction.

4 Organizational level

4.1 Objectives

The objectives of this clause are to ensure that the following are performed:

a) establishing organization-specific rules and processes for software update engineering;

b) adopting quality management, functional safety management and cybersecurity management for software update engineering;

c) instituting and maintaining a continuous improvement process for software update engineering;

d) establishing an information sharing policy for software update engineering; and

e) conducting an organizational audit for process compliance.

4.2 General

This clause covers the responsibility of the organization engaged in software update engineering to have governance in place so that the processes for software update engineering can conform to the requirements of this document. Governance includes compliance with required ISO standards as well as organizational activities such as continuous improvement, information sharing, and supporting processes. This clause also establishes auditing requirements for this document.

4.3 Requirements and recommendations

4.3.1 Governance

4.3.1.1 If the organization performs software update engineering activities, then this document applies.

4.3.1.2 The organization shall establish, document, and maintain rules and processes for software update engineering to:

— enable the implementation of the requirements of this document;

— support the execution of the corresponding activities, including the assignment of resources and responsibilities across all those involved in the software update engineering activities;

— confirm conformance with the requirements of this document.

NOTE 1 These rules and processes cover vehicle systems that are affected by software update engineering activities.

NOTE 2 These rules and processes cover the infrastructure used for software update engineering activities.

EXAMPLE Process definitions, technical rules, guidelines, methods, and templates.

4.3.1.3 The organization shall establish, implement and maintain software update engineering activities in accordance with applicable content of:

— ISO/SAE 21434;

— ISO 26262-6;

— ISO 26262-8.

NOTE Other parts of ISO 26262 series can provide guidance on how to identify applicable content and how to conform with ISO 26262-6 and ISO 26262-8.

EXAMPLE ISO 26262-3 can be used to show that ISO 26262-6 is not applicable if the software update operation is classification QM (quality management).

4.3.2 Continuous improvement

4.3.2.1 The organization shall establish, perform and maintain a continuous improvement process for software update engineering activities.

EXAMPLE 1 Evaluating, applying and communicating lessons learned.

EXAMPLE 2 Improvements from previous or similar software update projects, field monitoring and observations.

EXAMPLE 3 Key performance indicator (KPI) for continuous improvement process is the number of failures.

4.3.2.2 The organization shall establish, perform and maintain a process to verify that after any change to its software update engineering processes, the processes still meet the requirements of this document.

4.3.3 Information sharing

4.3.3.1 The organization shall establish, perform and maintain a policy for sharing information both inside and outside the organization concerning software update engineering activities.

NOTE The policy can include what information is shared, with whom the information is shared, when the information is shared, and how to permit sharing of information.

EXAMPLE Information being shared can include:

— schedule for the software update campaign;

— content description;

— possible implication of the software update campaign including safety or cybersecurity-relevant items;

— duration the vehicle or its functions are unavailable;

— reason for the software update campaign;

— treatment of sensitive or personal information;

— documentation about the software update campaign;

— license and intellectual property information.

4.3.4 Supporting processes

4.3.4.1 The organization shall establish, implement and maintain a document management process for software update engineering activities to handle the work products required by this document.

EXAMPLE IATF 16949 can be applied.

4.3.4.2 The organization shall establish, implement, and maintain a requirements management process for software update engineering activities.

EXAMPLE ISO/IEC 26551.

4.3.4.3 The organization should consider privacy implications of the activities required by this document.

NOTE Activities in this document can involve personal information.

EXAMPLE 1 Information on privacy can be found in ISO/IEC 27701 and ISO/IEC 29100.

EXAMPLE 2 Customer personally identifiable information included in software update campaigns.

4.3.4.4 The organization shall establish, implement and maintain a configuration management process.

NOTE Software update engineering activities involve configuration information for software update packages, vehicles and infrastructure.

EXAMPLE 1 ISO 10007 can be used for a configuration management process.

EXAMPLE 2 ISO/IEC/IEEE 15288 can be applied for configuration management on system life cycle management.

4.3.4.5 The organization shall establish, implement and maintain a quality management process for software update engineering activities.

EXAMPLE 1 IATF 16949, ISO 9001 and ISO/IEC 25000 can be used for quality management process.

EXAMPLE 2 Maintenance of the infrastructure.

4.3.4.6 The organization shall establish, implement and maintain a change management process for software update engineering activities.

EXAMPLE ISO 9001 can be used for change management process.

4.3.5 Auditing

4.3.5.1 An audit shall be performed to determine that the organizational processes for software update engineering achieve the objectives of this document.

NOTE 1 Such an audit can be included in, or combined with, an audit according to a quality management system standard.

NOTE 2 The audit can be performed by an internal or external organization.

NOTE 3 To ensure the organizational processes remain appropriate for software update engineering, an audit can be performed periodically.

NOTE 4 In a distributed development, right to audit can be included in the contract.

4.4 Work products

4.4.1 Organizational rules and processes resulting from the requirements of 4.3.1.1, 4.3.1.2, 4.3.4.1, 4.3.4.3, 4.3.4.4, 4.3.4.5, and 4.3.4.6.

4.4.2 Records of organizational management resulting from the requirements of 4.3.1.3, 4.3.4.2, and 4.3.4.5.

4.4.3 Documentation of continuous improvement resulting from the requirement of 4.3.2.1 and 4.3.2.2.

4.4.4 Information sharing policy resulting from the requirement of 4.3.3.1.

4.4.5 Audit report resulting from the requirement of 4.3.5.1.

5 Project level

5.1 Objectives

The objectives of this clause are to ensure that the following are performed:

a) planning for a software update project, including assigning roles and responsibilities;

b) managing and storing of information regarding a software update project;

c) providing justifications for any tailoring of a software update project;

d) confirming interoperability of the infrastructure and the vehicle functions for a software update project; and

e) preserving integrity of software, and either metadata or software update packages, or both.

5.2 General

This clause covers organizational requirements for software update projects including planning for software update projects, and managing information related to software update projects. In addition, this clause includes requirements on tailoring software update projects and interoperability between the parts of software update projects.

5.3 Requirements and recommendations

5.3.1 Project management

5.3.1.1 The organization shall develop, implement and maintain a plan for each software update project that covers all necessary activities.

NOTE 1 This plan can include activities involving the development and adaptation of vehicle or infrastructure functions, as well as any process described in this document.

NOTE 2 A software update project can encompass multiple software update campaigns.

EXAMPLE A software update project for a vehicle model; a software update project for a vehicle system; a software update project for a single type of ECU.

5.3.1.2 The organization shall manage and store documentation for each software update project.

NOTE Relevant processes are defined in 4.3.4.1 and 4.3.4.2.

5.3.1.3 The organization shall establish, assign and maintain the roles and responsibilities for each software update project.

NOTE Documentation of roles and responsibilities can be included in the plan in 5.3.1.1.

EXAMPLE A software update engineering responsibility is assigned to a department.

5.3.2 Tailoring and rationale

5.3.2.1 A software update project may be tailored.

EXAMPLE 1 A tailored software update project identifies applicable content of ISO 26262-6, ISO 26262-8 and ISO/SAE 21434 in the context of software update engineering activities.

EXAMPLE 2 A body builder tailors a software update project to conform with functional safety standards such as either ISO 13849 or the IEC 61508 series, or both.

5.3.2.2 If a software update project is tailored, then a rationale shall be provided as to how the tailored activities achieve the applicable objectives of this document.

NOTE 1 An activity is tailored if it is omitted or performed in a different manner compared to its description in this document.

NOTE 2 Software update engineering activities in this document that are performed by another entity in the supply chain are considered distributed activities rather than tailored activities.

EXAMPLE 1 Distributed cybersecurity activities under ISO/SAE 21434.

NOTE 3 The organization can consult with their suppliers on which clauses of this document are applicable to the supplier's work.

EXAMPLE 2 A supplier creates a new version of software for an organization to distribute to a vehicle.

5.3.3 Interoperability

5.3.3.1 The organization shall establish, implement and maintain a process to confirm the interoperability of the functions developed in accordance with the requirements from Clause 6 and Clause 7.

NOTE Since the infrastructure and the vehicle system can be implemented separately, it is important to confirm the interoperability between them to achieve a successful software update campaign.

EXAMPLE Maintenance of the infrastructure for preserving interoperability.

5.3.4 Integrity

5.3.4.1 The organization shall establish, implement and maintain processes to preserve the integrity of software, and either metadata or software update packages, or both, during distribution in the context of interoperability:

— within the infrastructure of organizations;

— between organizations within the supply chain;

— from organizations to vehicles;

— within the vehicle and between vehicle systems.

NOTE 1 The organization distributing software to vehicles can be an OEM, a supplier or other authorized entity.

NOTE 2 The risk-based approaches in ISO/SAE 21434 and the ISO 26262 series can be used to select measures to preserve integrity.

5.4 Work products

5.4.1 Software update project plan resulting from the requirements of 5.3.1.1 and 5.3.1.3.

5.4.2 Documentation of software update project resulting from the requirement of 5.3.1.2.

5.4.3 Rationale for tailored activities, if applicable, resulting from the requirement of 5.3.2.2.

5.4.4 Documentation of confirmation of interoperability resulting from 5.3.3.1.

5.4.5 Documentation of processes to preserve integrity from 5.3.4.1.

6 Infrastructure level

6.1 Objectives

The objectives of this clause are to ensure that the following are developed:

a) management of cybersecurity risks for the infrastructure;

b) functionality for collecting and managing vehicle configuration information for the infrastructure;

c) functionality for collecting and distributing information about software update campaigns; and

d) functionality for creating, managing, and distributing software update packages.

6.2 General

This clause includes the requirements for the development of infrastructure that is used for software update campaigns. The requirements cover the functions that are assigned to the infrastructure for the software update campaigns, such as distribution, communication, information storage, and cybersecurity. Functions described in this clause support the software update campaigns in the infrastructure. Such functions can be on or off the vehicle depending on the architectural decisions of the organization.

6.3 Requirements and recommendations

6.3.1 Managing risk

6.3.1.1 The organization shall manage the cybersecurity risks of the infrastructure.

EXAMPLE 1 Application of the ISO/IEC 27000 series.

EXAMPLE 2 Application of ISO/SAE 21434 for cybersecurity risks in the vehicle.

6.3.2 Managing vehicle configuration information

6.3.2.1 The infrastructure shall have one or more functions for receiving, storing and processing of vehicle configuration information.

6.3.2.2 The infrastructure shall have one or more functions to maintain the integrity of the collected vehicle configuration information.

6.3.2.3 The infrastructure shall have one or more functions to distribute vehicle configuration information to related parties.

NOTE 1 Distribution functions can be manual (e.g. paper-based).

NOTE 2 Vehicle configuration information can be distributed before, during or after a software update campaign.

EXAMPLE Vehicle configuration information is distributed to regulatory entities or suppliers.

6.3.2.4 The infrastructure shall have one or more functions to support the identification of dependencies for software update packages.

NOTE 1 This can be done on the vehicle or in the infrastructure or a combination of both.

NOTE 2 This function is used in Clause 8.

EXAMPLE 1 Dependencies that affect intelligent traffic systems or communication with consumer devices.

EXAMPLE 2 Dependencies that affect electric vehicle charging or map data processing.

6.3.2.5 The infrastructure shall have one or more functions to check compatibility of a software update package.

6.3.3 Communicating software update campaign information

6.3.3.1 The infrastructure shall have one or more functions to provide notifications as required by this document.

NOTE 1 This infrastructure notification function can be used to notify vehicle users instead of an in-vehicle notification function.

NOTE 2 These functions support notification requirements in Clause 9.

6.3.3.2 The infrastructure shall have one or more functions to receive, store, process and distribute results of software update campaigns.

NOTE See requirements concerning results of software update campaigns in Clause 9.

EXAMPLE The infrastructure receives success or failure from vehicles during a software update campaign.

6.3.4 Processing software update packages

6.3.4.1 The infrastructure shall have one or more functions to create, process, receive and store software update packages.

EXAMPLE The infrastructure receives a software update package from another organization in the supply chain.

6.3.4.2 The infrastructure shall have one or more functions to associate software update packages with targets.

6.3.4.3 The infrastructure shall have one or more functions to resolve targets into recipients for software update campaigns.

6.3.4.4 The infrastructure shall have one or more functions to support software update distribution methods.

6.3.4.5 The infrastructure should have one or more functions to determine whether there are sufficient in-vehicle resources to apply the software update package to the recipients.

6.3.4.6 The infrastructure shall have one or more functions to maintain the integrity of software update packages and their contents:

— within the infrastructure; and

— from the infrastructure to vehicles.

6.3.4.7 The infrastructure should have one or more functions to initiate actions if the infrastructure is notified of the failure of a software update operation.

NOTE A software update operation failure can be mitigated either by functions in the vehicle or by a skilled person, or both.

EXAMPLE Infrastructure sends notice of failure to dealership or local mechanic to pick up the vehicle.

6.4 Work products

6.4.1 Documentation of managing cybersecurity risk resulting from 6.3.1.1.

6.4.2 Documentation of functions for managing vehicle configuration information resulting from 6.3.2.1 to 6.3.2.5.

6.4.3 Documentation of functions for performing software update campaigns resulting from 6.3.3.1 and 6.3.3.2.

6.4.4 Documentation of functions for processing software update packages resulting from 6.3.4.1 to 6.3.4.6.

6.4.5 Documentation of functions for performing actions in the event of software update operation failure resulting from 6.3.4.7.

7 Vehicle and vehicle systems level

7.1 Objectives

The objectives of this clause are to establish the following functionalities in and for vehicles or vehicle systems:

a) managing safety and cybersecurity risks for software update operations;

b) managing vehicle configuration information;

c) communicating software update campaign information; and

d) enabling software update operations, verifying software update packages and managing failures during software update campaigns.

7.2 General

This clause contains the requirements for the functions needed for vehicles and vehicle systems to support software update campaigns. These functions include communications, generating necessary vehicle configuration information and enabling software update operations in vehicles.

Functions described in this clause support the software update operation in the vehicle. Such functions can be implemented in one or both of the vehicle and the infrastructure depending on the architectural decisions of the organization.

7.3 Requirements and recommendations

7.3.1 Managing risks

7.3.1.1 Functional safety risks of software update operations in the vehicle shall be managed.

NOTE 1 Management includes identification, analysis, evaluation and treatment of risks.

NOTE 2 The ISO 26262 series provides guidance on achieving functional safety through appropriate requirements and processes.

EXAMPLE 1 An OEM performs an assessment of potential safety impacts of a software update package for a brake system and decides, based on that assessment, whether a skilled person is necessary for the software update operation.

EXAMPLE 2 The architecture of the vehicle and the body builder equipment are defined such that safety-related functions are not impacted by software update operations.

EXAMPLE 3 ISO 26262-3 is used to identify functional safety risks of software update operations.

7.3.1.2 Safety risks due to reasonable and foreseeable misuse of software update operations in the vehicle shall be managed.

NOTE 1 Management includes identification, analysis, evaluation and treatment of risks.

NOTE 2 ISO 21448 provides guidance on achieving safety of the intended functionality through appropriate requirements and processes.

EXAMPLE A function is put in place to prevent either unintentional installation or unintentional activation, or both, of software by a vehicle user while driving.

7.3.1.3 Cybersecurity risks of software update operations in the vehicle shall be managed.

NOTE 1 ISO/SAE 21434 provides guidance on implementing threat analysis and risk assessment methods to manage risks of software update operations.

NOTE 2 Cybersecurity risks include the risk that vehicle configuration information might be modified without authorization.

7.3.2 Managing vehicle configuration information

7.3.2.1 There shall be one or more functions to collect vehicle configuration information.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.2.2 There shall be one or more functions to maintain the integrity of collected vehicle configuration information.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.2.3 There shall be one or more functions to identify the ECUs to which a software update package applies.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.3 Communicating software update campaign information

7.3.3.1 There shall be one or more functions to provide information to related parties as required by this document.

NOTE 1 These functions support notification requirements in Clause 9.

NOTE 2 These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE The vehicle informs the vehicle user about a successful software update operation.

7.3.3.2 There should be one or more functions to obtain the confirmation of the vehicle user for a software update operation.

NOTE 1 Confirmation can be obtained for each single instance of software update campaign or a general confirmation can be obtained at the beginning of the relationship between the vehicle user and the organization initiating a software update campaign.

NOTE 2 These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE Methods to obtain vehicle user confirmation:

— in-vehicle display;

— mobile application;

— website;

— contractual agreement.

7.3.4 Processing software update packages

7.3.4.1 There shall be one or more functions to support software update distribution methods.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE 1 The software update distribution method is wired (e.g. tool, USB flash drive), wireless (e.g. cellular or Wi-Fi) or hardware replacement.

EXAMPLE 2 A connector on the vehicle for a wired tool.

7.3.4.2 There shall be one or more functions to support software update operations.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE Ability for receipt, installation, and activation of a software update package.

7.3.4.3 There shall be one or more functions to determine that necessary conditions are met to perform software update operations.

NOTE 1 These functions can be implemented either in the vehicle or in the infrastructure, or both.

NOTE 2 Necessary conditions can differ for each step of a software update operation.

EXAMPLE A function to check if battery state of charge can support completion of the software update operation.

7.3.4.4 There shall be one or more functions to arbitrate simultaneous access requests to the vehicle to maintain a safe vehicle state.

NOTE 1 These functions can be implemented either in the vehicle or in the infrastructure, or both.

NOTE 2 Arbitration can be limitation, acceptance or rejection of simultaneous access requests.

EXAMPLE 1 Requests are received simultaneously from a wired tool and a wireless tool.

EXAMPLE 2 Multiple simultaneous wireless requests.

EXAMPLE 3 Simultaneous software update operations for different software update packages.

7.3.4.5 There shall be one or more functions to handle interruptions in communications while receiving a software update package.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.4.6 There shall be one or more functions to verify the integrity and authenticity of the received software update package any time before the activation.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE Signature verification is used for integrity and authenticity check.

7.3.4.7 There shall be one or more functions to maintain the integrity of software update packages and their contents:

— from the infrastructure to vehicles; and

— within the vehicle and between vehicle systems.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.4.8 There shall be one or more functions to check the compatibility of a software update package before activation.

NOTE These functions can be implemented either in the vehicle or in the infrastructure, or both.

7.3.4.9 There shall be one or more functions to ensure a safe vehicle state in each step of a software update operation.

NOTE 1 The safe vehicle state for a software update package is identified under Clause 8.

NOTE 2 These functions can include disabling or restricting vehicle features and functions to allow a software update operation to proceed safely.

NOTE 3 These functions can be implemented either in the vehicle or in the infrastructure, or both.

EXAMPLE 1 Safe vehicle state is ensured by a skilled person in a workshop.

EXAMPLE 2 The software update operation is paused or aborted because a safe vehicle state cannot be maintained.

EXAMPLE 3 Receipt occurred while the vehicle was in motion, but installation and activation occur at a later time.

EXAMPLE 4 Preventing the vehicle user from starting the vehicle during a software update operation.

7.3.4.10 There shall be one or more functions to ensure vehicle safety if a software update operation fails.

NOTE 1 These functions can be implemented either in the vehicle or in the infrastructure, or both.

NOTE 2 These functions can be the responsibility of a skilled person.

NOTE 3 These functions can be developed as a result of the implementation of the requirements in 7.3.1.

EXAMPLE Safety measures:

— parking the vehicle;

— reducing functionality of the vehicle;

— reducing performance of the vehicle.

7.4 Work products

7.4.1 Documentation of managing risks resulting from 7.3.1.1 to 7.3.1.3.

7.4.2 Documentation of functions for managing vehicle configuration information resulting from 7.3.2.1 to 7.3.2.3.

7.4.3 Documentation of functions for communications related to software update campaigns resulting from 7.3.3.1 and 7.3.3.2.

7.4.4 Documentation of functions for processing of software update packages resulting from 7.3.4.1 to 7.3.4.10.

8 Software update package

8.1 Objectives

The objectives of this clause are to ensure that the following are performed:

a) identifying the target(s) and contents of the software update package;

b) assembling the software update package containing the necessary software and metadata for the target(s);

c) verifying and validating the software update package; and

d) approving release of the software update package.

8.2 General

This clause includes requirements for assembling the software update package, and verifying and validating the software update package's contents, as well as identifying the classes of vehicles and vehicle systems to receive the software update package. Software update package development is the process of putting all necessary elements into a form for the software update operation at the vehicle level. The software update package is approved for release based on the performed verification and validation.

8.3 Requirements and recommendations

8.3.1 Identification of targets and the contents for the software update package

8.3.1.1 The target(s) for a software update package shall be determined.

NOTE For suppliers, the target can be one or more ECUs. For OEMs, the target can be the vehicles, vehicle systems, or ECUs.

8.3.1.2 The software and associated metadata released for the identified target(s) shall be selected for the software update package.

NOTE For suppliers, the software and associated metadata can be for a single ECU. For OEMs, the software and associated metadata can cover the vehicle, multiple vehicle systems, or ECUs.

EXAMPLE 1 Metadata include:

— safe vehicle state;

— conditions;

— compatibility information;

— either version information or release information, or both;

— necessary in-vehicle resources.

EXAMPLE 2 Conditions are parked, engine off, or availability of vehicle functions, etc.

EXAMPLE 3 Associated metadata for engine systems or motor control units of electric vehicles include different conditions if performed in the workshop or by the vehicle user.

8.3.1.3 Compatibility of the software update package with the existing software and hardware of the target shall be identified.

8.3.1.4 Dependencies for the software update package with the target shall be determined.

8.3.1.5 Necessary in-vehicle resources and conditions in the target shall be identified.

NOTE 1 Conditions can include compatibility.

NOTE 2 Compatibility can be derived from dependencies.

NOTE 3 In-vehicle resources can be necessary to complete the software update operation or run the new software.

EXAMPLE Conditions are parked, engine off, or availability of vehicle functions, etc.

8.3.1.6 Constraints on the software update distribution methods of the software update package shall be identified.

EXAMPLE A software update package is unable to be sent via wireless software update distribution method.

8.3.1.7 Necessary cybersecurity actions for the software update package shall be identified.

8.3.1.8 Necessary actions by the vehicle user or a skilled person for the software update package shall be determined.

NOTE These actions can be cybersecurity or safety related.

8.3.2 Assembly of the software update package

8.3.2.1 A software update package shall be created.

8.3.2.2 Only the selected software and associated metadata shall be included in the created software update package.

NOTE Software and associated metadata are selected in 8.3.1.2.

8.3.2.3 The organization shall assign a unique identifier to the software update package.

8.3.3 Verification and validation of the software update package

8.3.3.1 The required verification and validation for the software update package shall be determined and performed before the release of the software update package.

8.3.3.2 Compatibility of the software update package with the existing software and hardware of the target shall be validated.

8.3.3.3 Dependencies for the software update package with the target shall be validated.

8.3.3.4 Necessary in-vehicle resources in the target shall be validated.

8.3.3.5 The organization shall determine the implications of failure during software update operations for the target(s) for each software update package.

EXAMPLE The failure of a software update operation impacts functional safety.

8.3.3.6 A software update package shall be verified and validated to contain only the selected software and metadata.

EXAMPLE Confirming the correct software version is in the software update package.

8.3.3.7 Inclusion of necessary cybersecurity actions of the software update package shall be verified and validated.

NOTE These activities are determined in 8.3.1.7.

8.3.3.8 Inclusion of necessary actions in the software update package shall be validated and verified.

NOTE These actions are determined in 8.3.1.8.

8.3.4 Approval for release of the software update package

8.3.4.1 The software update package shall be approved for release based on verification and validation.

EXAMPLE Confirming validation and verification have been successfully completed.

8.4 Work products

8.4.1 Documentation of targets, contents, compatibility, dependencies, conditions, actions and necessary in-vehicle resources for the software update package resulting from 8.3.1.1 to 8.3.1.8.

8.4.2 Software update package with only intended and necessary contents resulting from 8.3.2.1 to 8.3.2.3.

8.4.3 Documentation of verification and validation resulting from 8.3.3.1 to 8.3.3.8.

8.4.4 Documentation of approval for release resulting from 8.3.4.1.

9 Software update campaign

9.1 Objectives

The objectives of this clause are to ensure that the following activities are performed:

a) preparing software update campaigns;

b) executing software update campaigns; and

c) completing software update campaigns.

9.2 General

This clause includes requirements for identifying the targets of a software update campaign, obtaining vehicle configuration information, resolving targets into recipients, distributing the software update package and providing relevant communication about a software update campaign.

9.3 Requirements and recommendations

9.3.1 Software update campaign preparation

9.3.1.1 The organization shall determine the purpose(s) of each software update campaign.

EXAMPLE Purposes include:

— new features provided by software update packages;

— cybersecurity or safety issues fixed by software update packages;

— altering existing features by software update packages.

9.3.1.2 The organization shall assign roles and responsibilities for each software update campaign.

9.3.1.3 The organization shall select the software update packages for each software update campaign.

9.3.1.4 The organization shall confirm that each selected software update package for each software update campaign has been approved for release (see 8.4.4).

9.3.1.5 The organization shall determine what vehicle configuration information is affected by each software update campaign.