Publicly Available Specification

ISO/PAS 8926

First edition
2024-01

Road vehicles — Functional safety — Use of pre-existing software architectural elements

Véhicules routiers — Sécurité fonctionnelle — Utilisation d'éléments d'architecture logicielle préexistants

Reference number
ISO/PAS 8926:2024(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

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 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).

ISO draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO takes 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 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. ISO 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.

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

This document addresses the use of pre-existing software architectural elements not originally developed in accordance with the ISO 26262:2018 series in the context of development aiming to achieve functional safety according to the ISO 26262:2018 series. It describes criteria for the integration of a pre-existing software architectural element to achieve functional safety.

The criteria establish confidence in a pre-existing software architectural element that enables its use in safety-related embedded software developed in accordance with the ISO 26262:2018 series when:

— it meets the needs of a target software architectural design because it provides required safety-related functionalities and properties (including safety mechanisms);

— it meets the needs of a target software architectural design because of its static and dynamic design, its interfaces and its resources are used.

The evidence supporting confidence is kept up to date as part of the safety case and is subject to confirmation measures.

Road vehicles — Functional safety — Use of pre-existing software architectural elements

1 Scope

This document describes a framework for functional safety to enable the use of pre-existing software architectural elements not originally developed in accordance with the ISO 26262:2018 series, but intended to be integrated into safety-related embedded software conformant with the ISO 26262:2018 series by:

— determining relevant criteria when using the pre-existing software architectural element as a safety-related element of safety-related embedded software;

— determining relevant criteria inherent to the pre-existing software architectural element, e.g. needs for external safety mechanisms to detect and control failures caused by the pre-existing software architectural element;

— providing suitable evidence and arguments for use of the pre-existing software architectural element that can include applicable procedures, techniques and safety measures;

— supporting the fulfilment of software safety requirements when using the pre-existing software architectural element as a safety-related element of safety-related embedded software;

— supporting the integration of the pre-existing software architectural element as a safety-related element of safety-related embedded software.

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 26262-1:2018, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2018, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-6:2018, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-8:2018, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2018, *Road vehicles — Functional safety — Part 9: Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 26262-1 and the following 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
complexity
degree to which a software or a software architectural element has a design, implementation and/or functionalities that are difficult to understand and verify

[SOURCE: ISO/IEC/IEEE 24765:2017, 3.694.3, modified — The phrase "system or component" was replaced by "software or software architectural element" and "and/or functionalities" was added.]

3.2
complexity measure
variable to which a value is assigned as a result of measurement concerning *complexity* (3.1)

[SOURCE: ISO/IEC 25000:2014, 4.18, modified — The original term was "measure", the phrase "concerning complexity" has been added and the Note 1 to entry has been deleted.]

3.3
pre-existing software architectural element
PSAE
already available commercial off-the-shelf or custom software element not specifically built-to-order, and not developed to conform with ISO 26262:2018 series

3.4
provenance
information regarding the origins, custody and ownership of a software and its associated data

[SOURCE: Reference [6], modified — The phrase "item or collection" has been replaced by "software and its associated data".]

3.5
target software architectural design
software architectural design, developed in accordance with ISO 26262:2018 series, into which the *pre-existing software architectural element (PSAE)* (3.3) is intended to be integrated

4 Use of pre-existing software architectural elements into safety-related embedded software conformant with the ISO 26262 series

4.1 Objectives

This clause applies to PSAE with the following objectives:

a) to provide evidence that functional safety is achieved for the target software architectural design after integration of the PSAE;

b) to provide evidence that the PSAE, once integrated, fulfils the requirements allocated to the PSAE in accordance with the target software architectural design;

c) to manage PSAE failure modes relevant to the integration of the PSAE in the target software architectural design;

d) to identify and apply appropriate safety measures required to support the achievement of functional safety when using the PSAE;

e) to identify foreseeable limitations and to confirm known limitations when using the PSAE.

4.2 General

A PSAE is safety-related if it is a safety element in the target software architectural design, i.e. if software safety requirements derived from the technical safety requirements are allocated to it or if errors of its software functions and/or properties can lead to a violation of the safety requirements.

EXAMPLE 1 An operating system (OS) that is used to host safety-related software applications can have safety-related properties for the correct execution with partitioning to achieve freedom from interference and a strategy for fault handling.

EXAMPLE 2 A safety-related device driver can include hardware diagnostics, a client software interface that enables freedom from interference and a strategy for fault handling.

An examination of PSAE used in the target software architectural design is performed to assess the functional safety implications, including:

— the functionalities and properties of the PSAE including identifying those mechanisms that conform to the allocated safety requirements;

— the implementation and interfaces of the PSAE that conform to the static and dynamic design aspects of the target software architectural design;

— determining that the target environment has sufficient hardware and software resources to meet the software safety requirements of the target software architectural design after the integration of PSAE;

— determining that unused functionalities and properties of the PSAE do not interfere with the achievement of functional safety or can be excluded from integration (e.g. selected configuration settings during build-process);

— determining that either any unintended behaviours are absent or the risk introduced due to the unintended behaviours is sufficiently low.

Annex A provides examples of PSAE including the implications of its use on functional safety.

Classification of a PSAE is defined to determine whether software qualification is applicable (in accordance with ISO 26262-8:2018, Clause 12) or whether specific safety activities are to be tailored (in accordance with ISO 26262-2:2018, 6.4.5.1 and 6.4.5.2) and planned (in accordance with ISO 26262-2:2018, 6.4.6.7).

NOTE 1 The specific safety activities are described in 4.4.4 and 4.4.5.

NOTE 2 The confirmation measures defined in ISO 26262-2:2018, 6.4.9 can apply to prevent any anomalies resulting from 4.4.2 and 4.4.3.

For this purpose, the classification (see 4.4.2) is used to justify the tailoring of specific safety activities to mitigate the risk of integrating the PSAE in the target software architectural design.

The classification is based on criteria that considers:

— the possibility that the uncertainty related to the process applied to PSAE development may increase the likelihood of systematic faults;

— the possibility that the complexity of the PSAE can make finding systematic faults more difficult.

The complexity of PSAE is evaluated for suitability, applying a set of selected complexity measures . Reasoning for its acceptance can be documented as part of the impact analysis report.

NOTE 3 Complexity can depend on the use case and the reasoning to justify complexity can vary as well. In some cases, numerical methods, such as cyclomatic complexity or number of lines, can be used while in some other cases qualitative methods can be used to evaluate complexity.

NOTE 4 Criteria for the use of these complexity measures can be established to determine whether the activities in ISO 26262-8:2018, Clause 12 provide a suitable risk reduction and improve the detection of systematic faults in the PSAE.

NOTE 5 Criteria for the use of these complexity measures can be established to define the organizational (or project) upper bound for the application of additional safety activities, where the application of the PSAE becomes excessively unmanageable and thus not recommended.

Figure 1 illustrates the role of the classification and the dependencies with the target software architectural design.

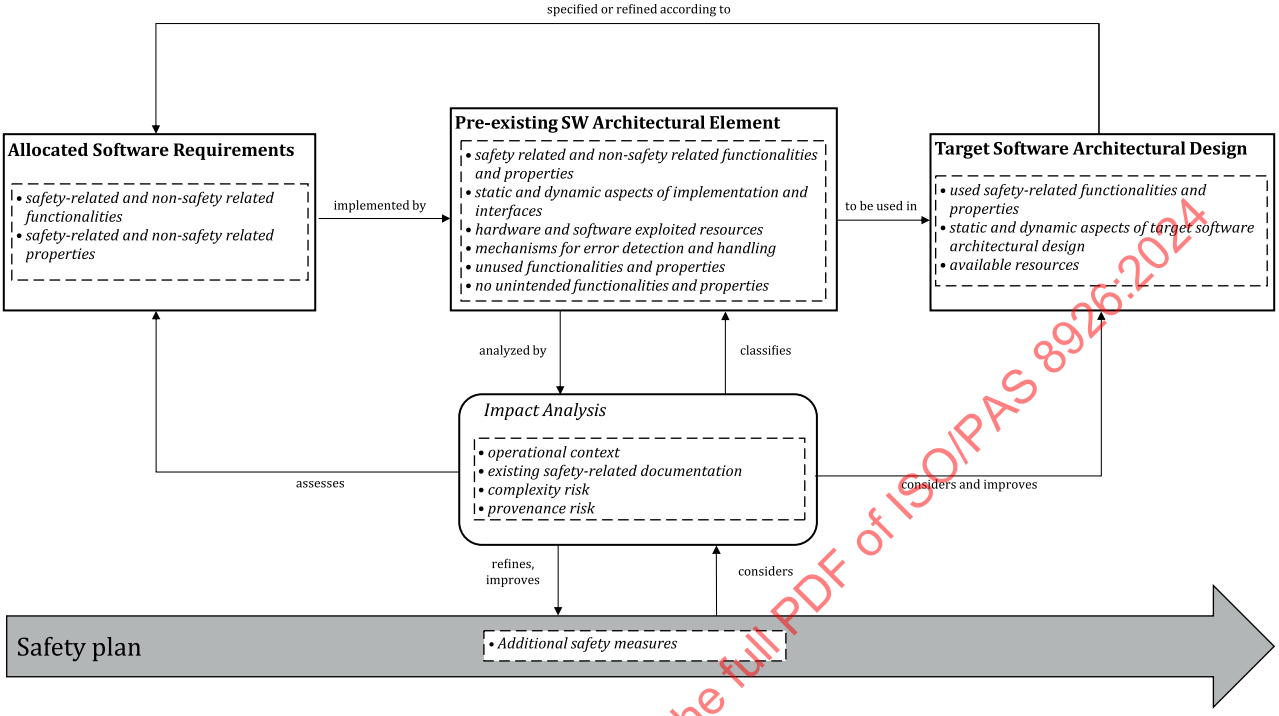

Figure 1 — Overview of impact analysis extended by classification

4.3 Input to this clause

4.3.1 Prerequisites

The following information shall be available:

— software safety requirements specification for the target software architectural design in accordance with ISO 26262-6:2018, 6.5.1;

— safety analysis report for the target software architectural design in accordance with ISO 26262-6:2018, 7.5.2;

— documentation of the software development environment related to the target software architectural design in accordance with ISO 26262-6:2018, 5.5.1;

— organization-specific rules and processes for functional safety in accordance with ISO 26262-2:2018, 5.5.1.

4.3.2 Further supporting information

The following information can be considered:

— technical safety requirements specification in accordance with ISO 26262-4:2018, 6.5.1;

— technical safety concept in accordance with ISO 26262-4:2018, 6.5.2;

— system architectural design specification in accordance with ISO 26262-4:2018, 6.5.3;

— hardware-software interface (HSI) specification in accordance with ISO 26262-4:2018, 6.5.4;

— rules and processes applied for the design, the implementation and the verification of the PSAE (from an external source);

— design specification of the PSAE (from an external source);

— specification of non-safety-related functions and properties of the PSAE (from an external source);

— PSAE implementation (from an external source);

— previous verification report of the PSAE (from an external source);

— requirements of the PSAE (from an external source);

— specification of functionalities and properties of the PSAE (from an external source);

— other existing information useful for conducting an impact analysis (from an external source);

— configuration data of the PSAE (from an external source).

4.4 Requirements and recommendations

4.4.1 General

This subclause is applicable to the specific version and configuration, if applicable, of the PSAE that is proposed for its integration into the target software architectural design.

4.4.2 Classification of a PSAE

4.4.2.1 A PSAE shall be classified based on provenance and complexity of the PSAE.

4.4.2.2 A set of complexity measures shall be determined to evaluate the likelihood of PSAE systematic faults occurring, including the rationale for its appropriateness.

NOTE Annex B provides examples of complexity measures.

4.4.2.3 The values of the complexity measures shall be determined for the PSAE.

4.4.2.4 The PSAE with its available supporting information shall be analysed and evaluated to determine:

a) the provenance-related uncertainty that the PSAE can contain systematic faults impacting the target software architectural design. This is expressed as follows:

   — P1 shall be selected when there is evidence that the software development process applied to the PSAE is based on an appropriate national or international standard (e.g. ISO/IEC/IEEE 12207) or a different functional safety standard (e.g. IEC 61508, RTCA DO-178C[7]);

   — P2 shall be selected when P1 cannot be fully claimed, but the gaps in the software development process are evaluated as acceptable, i.e. the risk of systematic faults due to these gaps is sufficiently low in the context of target software architectural design or manageable by mitigating the gaps; or

   — P3 in all other cases;

b) the complexity-related uncertainty that the PSAE can contain systematic faults which are difficult to find. This is expressed as follows:

   — C1 shall be selected when none of the determined complexity measures indicate high complexity;

— C2 shall be selected when the determined complexity measures indicate high complexity, but they are evaluated as acceptable, i.e. the risk of systematic faults due to identified complexity is sufficiently low in the context of the target software architectural design or high, but manageable; or

— C3 in all other cases.

NOTE 1 P3 can be assigned to a PSAE with an architectural design and development process that is neither sufficiently established nor sufficiently robust to justify its use for a safety-related application.

NOTE 2 A conservative judgement can be considered as a starting point if the provenance-related or complexity-related uncertainties cannot be clearly determined.

NOTE 3 Examples for the expression of C1, C2 and C3 are provided in Annex B.

4.4.2.5 A rationale for the determination of provenance-related and complexity-related uncertainties in 4.4.2.4 shall be provided and documented.

4.4.2.6 Based on the provenance-related and complexity-related uncertainties in 4.4.2.4, the classification of the PSAE shall be determined according to Table 1.

Table 1 — Classification of the PSAE

Complexity	Provenance		
	P1	P2	P3
C1	Class I	Class I[a]	Class II
C2	Class II[b]	Class II	Class II
C3	Class II	Class II	NR[c]

[a] See 4.4.2.8.

[b] See 4.4.2.9.

[c] Usage not recommended for functional safety.

NOTE The classification determination in Table 1 reflects the balance between the likelihood that complexity results in the PSAE containing systematic faults and the ability of the applied software development process mitigating those faults, versus the ability of the Class I and Class II subsequent activities identifying the presence of such faults.

4.4.2.7 A plan justifying the suitability of a PSAE or its sub-element shall be provided in accordance with the following classes:

— Class I: by directly following the qualification of software components in accordance with ISO 26262-8:2018, Clause 12;

— Class II: by defining the necessary safety activities in accordance with 4.4.4 and 4.4.5.

4.4.2.8 If measures addressing provenance-related uncertainty require additional safety activities to be planned in accordance with ISO 26262-2:2018, 6.4.4 c) and 6.4.6.7 c), then Class II shall be applied for the [P2, C1] combination.

NOTE Insufficient verification rigour and high software complexity increase the uncertainty that unused and/or unintended functionality or properties are present in the PSAE. Minimizing the verification insufficiencies in unused or unintended functionality or properties, reduces the uncertainty to an acceptable level.

4.4.2.9 If the PSAE development evidence shows conformity with national or international standards, suggesting that the risk of systematic failures due to complexity-related uncertainty is sufficiently low in the context of the target software architectural design, then Class I may be chosen for the [P1, C2] combination.

4.4.2.10 If software safety requirements with different ASILs are allocated to the PSAE and the entire PSAE is not assigned the highest ASIL, then the PSAE shall not be classified as Class I.

NOTE For a PSAE classified as Class II, an ASIL decomposition within the PSAE in accordance with ISO 26262-9:2018, Clause 5 can be applicable.

4.4.3 Impact analysis

4.4.3.1 Operational context

For the use of a PSAE within the target software architectural design, the impact analysis at the element level in accordance with ISO 26262-2:2018, 6.4.4 shall evaluate the operational context considering at least:

— ASIL of the safety requirement(s) allocated to the PSAE;

— external interfaces of the PSAE;

— target environment;

— scheduling, timing and concurrency aspects (e.g. data race conditions and the behaviour of OS synchronization mechanisms);

— calibration and configuration data.

NOTE 1 If unused functionality cannot be deselected using a configuration, then the freedom from interference can be verified, e.g. by call flow analysis or simulations can be applied.

EXAMPLE Adapting calibration data to a new context.

NOTE 2 If the PSAE has dependencies on the hardware, the hardware-software interface specification can be refined, if required, according to ISO 26262-6:2018, Clause 6.

4.4.3.2 Allocation of software safety requirements to the PSAE functionalities and properties

4.4.3.2.1 Software safety requirements shall be allocated to the functionalities and properties of the PSAE in accordance with ISO 26262-6:2018 Clause 6.

4.4.3.2.2 The PSAE shall be evaluated, based on a rationale, to determine whether the PSAE conforms to the allocated software safety requirements.

NOTE If the software safety requirements allocated to the PSAE have different ASILs, this includes the evaluation of the capability of the PSAE to conform to the highest ASIL allocated to it.

4.4.3.3 Review of PSAE existing documentation

The existing documentation for the PSAE shall be evaluated to determine if it provides sufficient evidence to support the achievement of functional safety for the PSAE integrated into the target software architectural design.

EXAMPLE Design guidelines relevant for the integration of the PSAE.

4.4.3.4 Planning of the safety activities

4.4.3.4.1 The PSAE shall be classified in accordance with 4.4.2 to plan the safety activities resulting from 4.4.3.1, 4.4.3.2 and 4.4.3.3 needed to identify and mitigate potential systematic faults of the PSAE and to ensure the implementation of necessary safety measures (ISO 26262-1:2018, 3.1.41).

NOTE If the PSAE does not meet the allocated software safety requirements, the activities to address conformity issues are defined in the safety plan.

4.4.3.4.2 The safety activities including the rationales resulting from the classification of the PSAE in accordance with 4.4.2 shall be documented.

NOTE The determination of provenance-related and complexity-related uncertainties can be subject to confirmation measures.

4.4.4 Suitability evaluation for Class II PSAE

4.4.4.1 Refine architectural design

4.4.4.1.1 If the available PSAE design information is insufficient to evaluate implications of the PSAE's use in the target software architectural design, the PSAE architectural design shall be retrospectively analysed and the sub-elements of the PSAE shall be identified.

NOTE 1 Figure 2 provides an example.

NOTE 2 A sub-element can be safety-related or not safety-related.

NOTE 3 An unused or unintended functionality cannot be easily identified without having a suitable understanding of the PSAE (e.g. internal structure). An unused functionality cannot be identified or distinguished from unintended functionality based on ISO 26262-8:2018, Clause 12, which is primarily requirement-based testing.

EXAMPLE Design information can include: conditions and compatibility with the target software architectural design; set of input data combinations; sequences of software safety functions; timing relations within sequences of execution of the software function(s), response time; resource management and timing interactions (e.g. hardware control, parallel processing, interrupt handling); internal states observed externally.

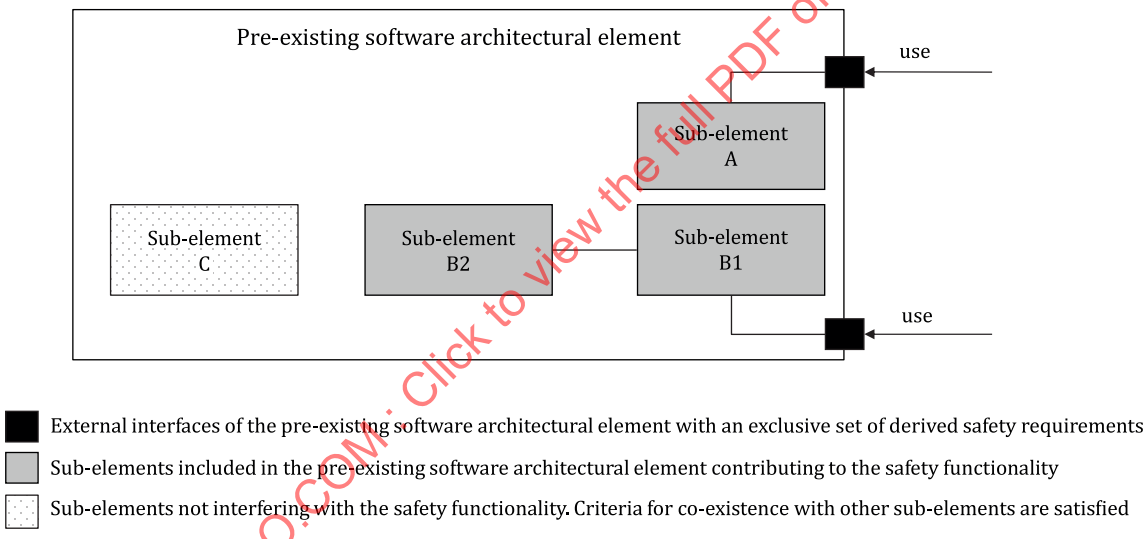

Figure 2 — A PSAE composed of sub-elements

4.4.4.1.2 The identification of safety-related sub-elements in accordance with ISO 26262-9:2018, Clause 6 shall be supported by PSAE design information that includes:

— the functionalities and properties of sub-elements and their impact on the achievement of functional safety;

— the static and dynamic interactions between the sub-elements.

NOTE 1 The design information includes how the PSAE and its sub-elements implement the allocated software safety requirements.

NOTE 2 If 4.4.2.10 is applied, different ASILs are assigned to the sub-elements accordingly.

NOTE 3 The design information identifies how the interfaces of the PSAE and its sub-elements are reachable and under which conditions, including actions and expected reactions to and/or from the hardware-software interface.

EXAMPLE Interaction diagrams showing safety-related and non-safety-related sub-elements and their dependencies.

4.4.4.2 Assign and refine software safety requirements

4.4.4.2.1 The software safety requirements allocated to the PSAE in accordance with 4.4.3.2 shall be refined based on the architectural design resulting from 4.4.4.1.

NOTE Safety-oriented analysis in 4.4.4.3 can also lead to the refinement of software safety requirements.

4.4.4.2.2 Additional software safety requirements for the sub-elements of PSAE shall be derived according to ISO 26262-6:2018, 7.4.6 by identifying the safety-related functionalities and properties of one or more sub-elements of the PSAE, the failures of which could lead to a violation of the software safety requirements.

NOTE 1 This can include to derive requirements for freedom from interference for the safety-related and non-safety-related sub-elements identified in accordance with 4.4.4.1.2, and for safety-related sub-elements identified in accordance with 4.4.4.1.2 with different ASILs (allocated/assigned).

NOTE 2 This can include to derive requirements for independence of a set of sub-elements from others required by the target software architectural design or derived for the sub-elements identified in accordance with 4.4.4.1.2.

4.4.4.2.3 The software safety requirements for the sub-elements of the PSAE shall be managed in accordance with ISO 26262-8:2018, Clause 6.

NOTE As described in ISO 26262-6:2018, 7.4.2 and 8.4.3, the traceability between software safety requirements and the PSAE design are established down to the lowest level of software sub-elements.

4.4.4.3 Safety-oriented analysis

4.4.4.3.1 The safety-oriented analysis, in accordance with ISO 26262-6:2018, 7.4.10 and based on the refined architectural design (4.4.4.1) and the refined software safety requirements (4.4.4.2), shall verify that the risk of violation of software safety requirements allocated to the PSAE is sufficiently low.

NOTE 1 The selection and application of safety-oriented analysis methods can be supported by considering ISO 26262-6:2018, Annex E.

NOTE 2 Detailed knowledge of the PSAE design and implementation can support effective safety-oriented analysis.

EXAMPLE 1 “Late timing” in the PSAE modelled as a fault leading to a deadlock situation recognized as a failure mode of the target software architectural design.

NOTE 3 If potential sources of failure relate to hardware functionalities or resources, then the information can be fed back into the system architectural design to enable the safety-oriented analysis at that level.

EXAMPLE 2 Assumptions for resource or hardware usage, such as interrupt handling.

NOTE 4 If existing mechanisms in PSAE are intended to be used to achieve functional safety, the safety-oriented analysis can verify the effectiveness of those existing mechanisms for detecting or handling faults.

EXAMPLE 3 Verification of sufficient independence between a monitoring function already implemented in PSAE and its monitored PSAE functionality.

4.4.4.3.2 The safety-oriented analysis shall verify that the required freedom from interference is sufficiently achieved within the PSAE in accordance with ISO 26262-6:2018, 7.4.11, if applicable.

NOTE Different design abstraction levels can be considered. For example, between the PSAE and other components of the target software architectural design and/or within the PSAE itself.

EXAMPLE Demonstration of freedom from interference for unused or untested sub-elements contained in the PSAE.

4.4.4.3.3 The results of the safety-oriented analysis, applied in accordance with ISO 26262-9:2018, Clause 8, shall identify whether the PSAE does or does not conform to the software safety requirements.

NOTE 1 The safety-oriented analysis applied to the PSAE can lead to further refinement of the software safety requirements (4.4.4.1) and of the PSAE architectural design (4.4.4.2) including the modification of architectural design, and application of existing or additional safety mechanisms in accordance with ISO 26262-6:2018, 7.4.12. Furthermore, the results of the safety-oriented analyses can identify additional verifications required at unit or integration level (4.4.5), and eventually to a modification of the PSAE in accordance with 4.4.6.

EXAMPLE A safety-oriented analysis can be performed on a complex middleware consisting of several software elements to ensure the suitability for its use and to define the verification activities for its integration into the target software architectural design.

NOTE 2 The safety-oriented analysis applied to the PSAE can lead to further refinement of the target software architectural design. Any modifications resulting from such refinement can be addressed by ISO 26262-8:2018, Clause 8.

NOTE 3 If the PSAE is configurable, the safety-oriented analysis can be refined by considering ISO 26262-6:2018, Annex C to evaluate the impact of configuration data to potentially violate the assigned software safety requirements. Similarly, calibration data errors can be considered in the safety-oriented analysis.

4.4.4.4 Verification of the PSAE suitability evaluation

4.4.4.4.1 The results of the suitability evaluation of the PSAE in accordance with 4.4.4.1 to 4.4.4.3, including the identified safety-related sub-elements, shall be verified.

4.4.4.4.2 The safety-related sub-elements of the PSAE may be classified in accordance with 4.4.2 to plan the safety activities needed to provide evidence for their suitability.

NOTE 1 The safety-related sub-elements can be classified in the early phase of the suitability evaluation (4.4.4.1, 4.4.4.2 or 4.4.4.3).

NOTE 2 Hierarchical classification of PSAE sub-elements results in an architectural design of the PSAE consisting of sub-elements of Class I, Class II or a combination of both.

4.4.4.4.3 A sub-element of the PSAE coexisting with safety-related sub-elements (in accordance with ISO 26262-9:2018, 6.4.3 and 6.4.4) for which evidence of freedom from interference is not available shall be classified in accordance with 4.4.2.

4.4.5 Verification of the Class II PSAE use

The PSAE and its integration in the target software architectural design shall be verified in accordance with ISO 26262-6:2018, Clauses 9 and 10 considering the ASIL(s) of the safety requirements allocated to the PSAE to provide evidence that:

a) the safety-related functionalities and properties implemented into the PSAE fulfil the requirements resulting from the impact analysis (4.4.3) and the suitability evaluation (4.4.4);

b) each safety measure resulting from safety-oriented analysis is effective.

NOTE 1 This includes the verification and integration of the sub-elements of the Class II PSAE considering the ASIL(s) of safety requirements allocated to them.

NOTE 2 The existing verification result of the PSAE can be evaluated to support verification for use in the target architectural design.

EXAMPLE Identification of additional test cases due to difference in the target environment and previously used test environment for the PSAE.

NOTE 3 Safety-oriented analysis can support verification to provide evidence and argument as to why the presence of untested interfaces, untested accesses to hardware and software resources and untested code do not impact the achievement of functional safety.

4.4.6 Changes to the PSAE design

4.4.6.1 Initiating change management for PSAE

4.4.6.1.1 Modifications to a PSAE design shall be kept to a minimum.

NOTE 1 Modifications are not applicable for Class I PSAE or Class I sub-elements in accordance with ISO 26262-6:2018, 7.4.7.

NOTE 2 Modifications are applicable for Class II PSAE or Class II sub-elements in accordance with ISO 26262:2018 series. The impact analysis (4.4.3) and the suitability evaluation (4.4.4) can be basis of tailoring in accordance with ISO 26262-2:2018, 6.4.5.1 and 6.4.5.2 by confirming the use of PSAE is highly compatible with the target software architectural design.

4.4.6.1.2 During the implementation of the change, evidence related to the PSAE and the target software architectural design shall be subject to configuration management in accordance with ISO 26262-8:2018, Clause 7.

4.4.6.2 Change request analysis

Design change requests derived from the suitability evaluation or from the verification execution shall be analysed in accordance with ISO 26262-8:2018, Clause 8 to identify the impact and feasibility of proposed change on the PSAE and other interacting software elements.

EXAMPLE Modifications resulting from 4.4.4.4.1.

NOTE The impact of a modification related to the PSAE or its sub-element can be analysed for the need to consider:

— replacement with a more suitable element, or

— replacement with a different element developed in accordance with ISO 26262-6:2018.

4.4.6.3 Implementing the change

The change shall be planned and implemented in accordance with ISO 26262-8:2018, Clause 8.

EXAMPLE Reconfiguration of the PSAE or replacement of sub-elements.

4.5 Work products

4.5.1 Applicable for all PSAE Classes (see 4.4.2.7)

4.5.1.1 Impact analysis resulting from 4.4.1 and 4.4.2.

4.5.1.2 Safety plan (refined) resulting from 4.4.3.4.

4.5.2 Applicable for PSAE Class II (see 4.4.2.7)

4.5.2.1 Software safety requirements (refined) resulting from 4.4.4.2.

4.5.2.2 Hardware-software interface (HSI) specification (refined) resulting from 4.4.4.1.

4.5.2.3 Software architectural design specification (refined) resulting from 4.4.4.1.

4.5.2.4 Safety-oriented analysis report resulting from 4.4.4.3.

4.5.2.5 Software verification specification (refined) resulting from 4.4.5.

4.5.2.6 Software verification report (refined) resulting from 4.4.4.4 and 4.4.5.

4.5.2.7 Change report resulting from 4.4.6.

Annex A
(informative)

PSAE examples

The following examples are considered within the scope of this document.

a) Company's internal and previously used software elements (legacy code)

The element of this example has not been developed in accordance with the ISO 26262:2018 series and it has already been in use in road vehicles, potentially in a different ECU, e.g. steering control, braking control or engine control ECUs.

The reason for this element re-use can be driven by the necessity to change hardware (due to obsolescence) to run the same element and to avoid the re-development of software functionalities already developed.

Potential difficulties for the software integrator can be:

— the documentation fulfilling ISO 26262-6 is unavailable or incomplete;

— the field data are not detailed enough to adopt proven in use argumentation in line with ISO 26262-8:2018, Clause 14;

— porting the software to the new target architecture.

b) Open source operating system

The element of this example has been developed and maintained by well-established open-source communities and based on their software best practices, vetted and documented by the supplier or integrator.

The reason for this element re-use can be driven by:

— the necessity to avoid the re-development of software functionalities already available, widely utilized in other domains and subject to thorough software maintenance practices;

— the possibility of examining the sources and documentation or the availability of support from the community (e.g. history, management, maintenance).

Since the element can be a significantly large and complex monolithic OS not developed in accordance with the ISO 26262:2018 series, potential difficulties for the integrator can be:

— to perform the necessary activities retrospectively;

— to provide sufficient evidence of conformity with the software best practices;

— to cover missing freedom from interference mechanisms;

— to provide suitable reaction to hardware diagnostic safety mechanisms (e.g. invalid instructions, memory faults);

— to ascertain the test coverage of unused code in the deployed application.

c) Software libraries (e.g. libgcc) or runtimes

The element of this example is a software library not developed in accordance with the ISO 26262:2018 series or other standards.

Potential difficulties for the software integrator can be:

— complexity (e.g. due to the use of preprocessor directive and compiler switches for supporting multiple microprocessor platforms);

— lack of detailed documentation;

— insufficient test coverage.

d) Software elements from other safety-related domains (e.g. aviation)

The element of this example has been developed under a safety standard other than the ISO 26262:2018 series.

Using a software element that is already used in a safety-related system can be advantageous. However, even if some of the available artefacts can be applicable for its use in automotive safety-related systems, the potential difficulty for the integrator can be to identify and address the gaps in order to conform with the ISO 26262:2018 series.