Document Title	Safety Requirements for AUTOSAR Adaptive Platform and AUTOSAR Classic Platform
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	986

Document Status	published
Part of AUTOSAR Standard	Foundation
Part of Standard Release	R25-11

Document Change History

Date	Release	Changed by	Description
2025-11-27	R25-11	AUTOSAR Release Management	• Fix editorial issues • Remove traces to FO_RS_MAIN (obsolete)
2024-11-27	R24-11	AUTOSAR Release Management	• Fix editorial issues • Add change history
2023-11-23	R23-11	AUTOSAR Release Management	• Fix editorial issues • Include linked requirements from AP_RS_OperatingSystemInterface
2022-11-24	R22-11	AUTOSAR Release Management	• Renamed safety goal to top level safety requirement • Added safety need for recovery upon failure • Added support to include extracted requirements from component requirement specification

△

2021-11-25	R21-11	AUTOSAR Release Management	• Add classic platform requirements chapter • Add requirements for CP WDG, CP OS, CP E2E • Rework top level safety requirements structure • Add TLSR RS_SAF_00006 • Update PHM, EM, SM requirements • Update functional safety requirements to be AUTOSAR Platform and Foundation
2020-11-30	R20-11	AUTOSAR Release Management	• Initial release • Functional safety requirements for the AUTOSAR Adaptive Platform • Technical safety requirements for PHM, EM, SM, OS, PER, CM and UCM

Disclaimer

Table of Contents

1 Scope of Document

This document specifies safety requirements on the AUTOSAR Platforms. It makes use of the intended functionality described in

- AUTOSAR_AP_EXP_PlatformDesign [1]
- AUTOSAR_CP_EXP_LayeredSoftwareArchitecture [2]
- AUTOSAR_AP_EXP_SWArchitecture[3].

The functional safety requirements are derived from the top level safety requirements and hazards mentioned in AUTOSAR_EXP_SafetyOverview[4].

This document contains a view on technical safety requirements of the AUTOSAR Functional Cluster, Services, and AUTOSAR Basic Software Modules. These requirements are extracted from the dedicated target requirement specifications and included in this document to give the reader the full overview in one document.

No ASIL Ratings

The AUTOSAR consortium, especially the AUTOSAR Adaptive Platform Working Groups are only providing an architecture definition, descriptions of the functional blocks and a *proof of concept* implementation, it is not possible to assign an ASIL rating to any requirement within this scope as described in ISO26262[5].

2 How to Read This Document

This document contains functional safety requirements which are generic and do not mention specific solutions/components of AUTOSAR. The technical safety requirements are then derived from functional safety requirements, which mention the specific responsibilities of AUTOSAR components. Each requirement has its unique identifier starting with the prefix "RS_SAF_" (for "Safety Requirement").

Technical Safety Requirements are partly extracted from the dedicated target component requirement specification and will therefore not have the prefix "RS_SAF_". Not all technical requirements have been consolidated yet. The goal is to have all technical requirements being part of the target requirements specification and only included here to provide a complete safety requirement catalog.

2.1 Document Conventions

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see [6, Standardization Template].

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see [6, Standardization Template].

2.2 Conventions used

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as follows.

Note that the requirement level of the document in which they are used modifies the force of these words.

- MUST: This word, or the adjective "LEGALLY REQUIRED", means that the definition is an absolute requirement of the specification due to legal issues.
- MUST NOT: This phrase, or the phrase "MUST NOT", means that the definition is an absolute prohibition of the specification due to legal issues.
- SHALL: This phrase, or the adjective "REQUIRED", means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective "RECOMMENDED", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

- SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED", means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item.

An implementation, which does not include a particular option, SHALL be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, SHALL be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides).

2.2.1 Requirement Identifier Coding

The unique identifier for safety requirements shall consist of

- a document identifier
- an identifier to distinguish functional safety requirements and technical safety requirements
- an identifier to identify a target component (either a Functional Cluster in the AUTOSAR Adaptive Platform or a Basic Software Component in the AUTOSAR Classic Platform)
- a requirement number

The coding pattern used in this requirements specification is `RS_SAF_<Z><YY><XX>`, where

Z is a single digit number, describing whether the requirement is a

0 safety goal or top level safety requirement functional safety requirement, where

YY *is reserved*

XX is a double digit number

1 functional safety requirement for the AUTOSAR Adaptive Platform, where

YY *is reserved*

XX is a double digit number

2 technical safety requirement for the AUTOSAR Adaptive Platform, where

YY is a double digit number, describing whether the requirement addresses

00 *reserved*

11 Platform Health Management (PHM)

12 Execution Management (EM)

13 State Management (SM)

14 Operating System (OS)

15 Persistency (PER)

16 Communication Management (CM)

17 Update and Configuration Management (UCM)

and

XX is a double digit number

3 technical safety requirement for the AUTOSAR Classic Platform, where

YY is a double digit number, describing whether the requirement addresses

00 *reserved*

11 Watchdog Manager (WDGM)

12 Operating System (OS)

13 E2E Protection (E2E)

and

XX is a double digit number

4-9 reserved for future use

3 Acronyms and abbreviations

The glossary below includes acronyms and abbreviations relevant to RS_Safety that are not included in the AUTOSAR Glossary [7].

Abbreviation / Acronym:	Description:
EM	Execution Management
SM	State Management
PER	Persistency
CM	Communication Management
UCM	Update and Configuration Management
S2S	Signal to Service
SG	Safety Goal
TLSR	Top Level Safety Requirement
WDG	Watchdog

Table 3.1: Acronyms and Abbreviations

4 Requirements Specification

This chapter contains top level safety requirements for AUTOSAR in 4.1. Functional safety requirements in 4.2 are derived from these requirements. The sub-chapter 4.3 contains technical safety requirements which are derived from the functional safety requirements.

4.1 Top Level Safety Requirements

[RS_SAF_00001] Safe Execution

Status: DRAFT

⌈

Description:	AUTOSAR shall provide supporting mechanisms to monitor the control flow and manage the execution order of multiple applications with mixed safety criticality.
Rationale:	To ensure freedom from interference with respect to timing [5] and data processing.
AppliesTo:	FO
Supporting Material:	ISO26262 [5]

⌋

[RS_SAF_00002] Safe Configuration

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to support correct configuration during the entire driving cycle of the vehicle.
Rationale:	AUTOSAR needs to provide measures and mechanisms to keep the configuration consistent throughout the whole driving cycle of the vehicle.
AppliesTo:	FO
Supporting Material:	ISO 26262 [5]

⌋

[RS_SAF_00003] Safe Update or Safe Upgrade

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to support correct update and upgrade of multiple platform and non-platform applications with mixed criticality.
Rationale:	AUTOSAR supports updatability during the life cycle of the machine and therefore the platform is responsible to ensure that these updates are performed correctly and safely.
AppliesTo:	FO
Supporting Material:	ISO 26262 [5]

⌋

[RS_SAF_00004] Safe Exchange of Information

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to support safe exchange (transmission and reception) of information between safety relevant applications.
Rationale:	In a vehicle several ECUs with several software components are interrelating with each other to fulfill a goal or functionality. AUTOSAR provides standardized interfaces and mechanisms to achieve safe communication between these components. Safe communication with elements outside of the vehicle is also in scope.
AppliesTo:	FO
Supporting Material:	ISO 26262 [5]

⌋

[RS_SAF_00005] Detection of Data Corruption

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to detect faults while processing data, communicating with other systems or system elements.
Rationale:	Mechanisms to detect faults are required to achieve higher safety ratings and increase product quality. A list of potential faults is described in EXP_SafetyOverview [4] and ISO 26262 [5].
AppliesTo:	FO
Supporting Material:	ISO 26262 [5]

⌋

[RS_SAF_00006] Safe Storage

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to support safe storage for applications.
Rationale:	Many applications need to store and retrieve data from volatile or non-volatile/persistent memory. If the Application is safety relevant, the data shall be checked to ensure that it has not been altered.
AppliesTo:	FO
Supporting Material:	ISO 26262 [5]

⌋

[RS_SAF_00007] Recovery upon detected faults

Status: DRAFT

⌈

Description:	AUTOSAR shall monitor, detect and provide means to react on detected faults.
Rationale:	AUTOSAR is expected to be capable of • restarting applications in case of failures • restarting a machine/ECU in case of failures • recover last known configuration in case of update failures
AppliesTo:	FO
Supporting Material:	ISO 26262-6 [5]

⌋

4.2 Functional Safety Requirements

[RS_SAF_10001] AUTOSAR shall provide mechanisms to support safe initialization of application software.

Status: DRAFT
Upstream requirements: RS_SAF_00001, RS_SAF_00002

⌈

Description:	AUTOSAR shall provide mechanisms to support safe initialization of application software.
Rationale:	Safe initialization of the underlying hardware and the AUTOSAR Platforms and the application software is required to ensure the intended functionality.
Use Case:	SUC_02
AppliesTo:	FO
Dependencies:	–

▽

△

Supporting Material:	–

⌋

[RS_SAF_10002] AUTOSAR shall provide safety mechanisms for embedded middleware, application software and their respective configuration data.

Status: DRAFT
Upstream requirements: RS_SAF_00002, RS_SAF_00003

⌈

Description:	AUTOSAR shall provide safety mechanisms for embedded middleware, application software and their respective configuration data.
Rationale:	Due to the random hardware failures in the memory unit, the data integrity is required to be verified to ensure no loss of data has occurred over time during operation, stand-by or powered off and has not been tampered with. Note: Not with respect to cybersecurity.
Use Case:	SUC_02, SUC_06
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10005] AUTOSAR shall provide mechanisms to support safe shutdown and termination of application software and embedded middleware.

Status: DRAFT
Upstream requirements: RS_SAF_00001, RS_SAF_00003

⌈

Description:	AUTOSAR shall provide mechanisms to support safe shutdown and termination of application software and embedded middleware.
Rationale:	Before termination of application software and/or shut-down of the AUTOSAR Platforms or the whole ECU, the dependent applications have to be terminated properly in the right order to prevent conflicts or failures or unexpected behavior. Ensure safe degradation, fault evacuation and fault containment.
Use Case:	SUC_01, SUC_06
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10006] AUTOSAR shall provide mechanisms to support safe transition of states in embedded middleware or service life cycle.

Status: DRAFT

⌈

Description:	AUTOSAR shall provide mechanisms to support safe transition of states in embedded middleware or service life cycle.
Rationale:	AUTOSAR Platforms are responsible for managing and monitoring the internal states of the application.
Use Case:	SUC_01, SUC_06
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10008] AUTOSAR shall provide mechanisms to support safe resource management for application software and embedded middleware.

Status: DRAFT
Upstream requirements: RS_SAF_00001, RS_SAF_00002, RS_SAF_00004

⌈

Description:	AUTOSAR shall provide mechanisms to support safe resource management for application software and embedded middleware.
Rationale:	The application software and embedded middleware of the AUTOSAR Platforms shall be ensured with adequate resources and availability to that resource in the expected time with sufficient freedom from interference. No unexpected or unhandled exception shall prevent access or delay access to a required and properly managed and authorized resource. Resources are - among other - CPU, runtime, memory consumption, net bandwidth, peripherals (like ADC, DAC, Timer) . . .
Use Case:	SUC_01
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10014] AUTOSAR shall provide an interface to support safe communication for embedded middleware and application software.

Status: DRAFT
Upstream requirements: RS_SAF_00004

⌈

Description:	AUTOSAR shall provide an interface to support safe communication for embedded middleware and application software.
Rationale:	In a vehicle several ECUs with application software are interrelating with each other to fulfill a goal or functionality. AUTOSAR Platforms provides standardized interfaces and mechanisms to achieve safe communication between these components. Safe communication with elements outside of the vehicle is also in scope.
Use Case:	SUC_03, SUC_04, SUC_05
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10027] AUTOSAR shall provide mechanisms to prevent the loss of a valid configuration.

Status: DRAFT
Upstream requirements: RS_SAF_00002, RS_SAF_00007

⌈

Description:	AUTOSAR shall provide mechanisms to prevent the loss of a valid configuration on either machine or vehicle level.
Rationale:	AUTOSAR Platforms should provide mechanisms to switch back to the latest working configuration.
Use Case:	SUC_02, SUC_06
AppliesTo:	CP,AP
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10028] AUTOSAR shall provide mechanisms to support dependable scheduling of application software and embedded middleware.

Status: DRAFT
Upstream requirements: RS_SAF_00001, RS_SAF_00002

⌈

Description:	AUTOSAR shall provide mechanisms to support dependable scheduling of application software and embedded middleware.
Rationale:	Dependable scheduling is required to ensure the proper time-allocation for all the available • functional-clusters, services and applications, • basic software modules and software components.
Use Case:	SUC_01
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10030] AUTOSAR shall provide mechanisms to support safe program execution.

Status: DRAFT
Upstream requirements: RS_SAF_00001

⌈

Description:	AUTOSAR shall provide mechanisms to support safe program execution.
Rationale:	The AUTOSAR Platforms shall offer flow monitoring mechanisms to detect and ensure that the intended program flow of functional-clusters and services as well as for user-applications and user-services is not violated.
Use Case:	SUC_01
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10031] AUTOSAR shall provide mechanisms to detect program execution time violation

Status: DRAFT
Upstream requirements: RS_SAF_00001

⌈

Description:	AUTOSAR shall provide mechanisms to detect program execution time violation
Rationale:	All the timing constraints of the basic software modules, functional-clusters, services and applications need to be supervised and monitored.

▽

△

Use Case:	SUC_01, SUC_06
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10037] AUTOSAR shall provide mechanisms to prevent unintended alteration of data.

Status: DRAFT
Upstream requirements: RS_SAF_00004, RS_SAF_00006

⌈

Description:	AUTOSAR shall provide mechanisms to prevent unintended alteration of data.
Rationale:	To achieve freedom from interference in systems running applications with mixed safety criticality, protection of data against unintended alteration is required.
Use Case:	SUC_06
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10038] AUTOSAR shall provide mechanisms to support that the safety relevant software is only updated/upgraded in a state that cannot cause a hazardous situation.

Status: DRAFT
Upstream requirements: RS_SAF_00003

⌈

Description:	AUTOSAR shall provide mechanisms to support that the safety relevant software is only updated/upgraded in a state that cannot cause a hazardous situation.
Rationale:	The update of safety critical application should be done when the car is stationary and at a safe location e.g. a parking garage.
Use Case:	SUC_02
AppliesTo:	FO
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10039] AUTOSAR shall support mechanisms to detect unintended alteration of data.

Status: DRAFT
Upstream requirements: RS_SAF_00002, RS_SAF_00003, RS_SAF_00004

⌈

Description:	There shall be a safety mechanism that detects communication errors. The mechanism shall be fully built-in in AUTOSAR (including AUTOSAR configuration and corresponding AUTOSAR basic software module). There shall be a support for all currently supported communication stacks (CAN, LIN, FlexRay, Ethernet).
Rationale:	To ensure safe data exchange between software components that fulfills ISO 26262-6:2018 D.2.4, while using a QM communication stack. D.2.4 defines following failure modes of the exchange of information: • repetition of information; • loss of information; • delay of information; • insertion of information; • masquerade or incorrect addressing of information; • incorrect sequence of information; • corruption of information; • asymmetric information sent from a sender to multiple receivers; • information from a sender received by only a subset of the receivers; • blocking access to a communication channel
Use Case:	SW-Cs or Adaptive Applications on different ECUs, Machines or Partitions exchange safety related data, using QM communication stack
AppliesTo:	CP, AP
Dependencies:	–
Supporting Material:	ISO 26262-6:2018 D.2.4[5]

⌋

[RS_SAF_10040] AUTOSAR shall support data recovery mechanisms.

Status: DRAFT
Upstream requirements: RS_SAF_00006

⌈

Description:	AUTOSAR shall support data recovery mechanisms
Rationale:	Applications want to recover altered data.
Use Case:	SUC_06
AppliesTo:	CP, AP
Dependencies:	–

▽

△

Supporting Material:	ISO 26262[5]

⌋

[RS_SAF_10041] AUTOSAR shall allow integrators to select and configure the set of safety mechanisms to detect communication faults.

Status: DRAFT
Upstream requirements: RS_SAF_00004, RS_SAF_00005

⌈

Description:	Based on individual safety concepts, AUTOSAR integrators need to individually configure the required mechanism to fulfill the safety requirements.
Rationale:	The AUTOSAR Platform is designed to be used in various applications. It is possible that for specific applications, a particular type of fault will not occur. Therefore, it is reasonable to have the configurability such that integrators may freely select the set of mechanisms to be deployed.
Use Case:	A high-level design change or new information requires a different communication protection mechanism. An integrator can select the proper protection by changing the manifest or description file.
AppliesTo:	CP, AP
Dependencies:	–
Supporting Material:	–

⌋

[RS_SAF_10042] AUTOSAR shall provide mechanisms to detect time synchronization violations.

Status: DRAFT
Upstream requirements: RS_SAF_00004

⌈

Description:	AUTOSAR shall provide mechanisms to detect time synchronization violations.
Rationale:	Time synchronization is a critical functionality for a distributed system where functions are deployed and data is acquired asynchronously in various machines within a network and have to work collaboratively.
Use Case:	A sender is adding a timestamp within a critical message and the receiver is 'running behind' and cannot detect the message delay if time-synchronization is not working properly. An important information from a sensor needs to be timestamped so that the information can be processed and fused with other data sources within the time domain.
AppliesTo:	CP, AP
Dependencies:	–
Supporting Material:	–

⌋

4.3 Technical Safety Requirements

Some of the following requirements are extracted from the dedicated requirements specifications, the extracted requirements are not named *RS_SAF*. The source document is mentioned within the chapter. The requirements named *RS_SAF* are to be consolidated with the corresponding target specification, so that this chapter does not contain any own requirements in the upcoming releases anymore.

4.3.1 AUTOSAR Foundation

4.3.1.1 Health Monitoring (HM)

[RS_HM_09125] Health Monitoring shall provide an Alive Supervision
(State: DRAFT; Upstream Requirements: [RS_SAF_10031])

Description:	Health Monitoring shall check if the frequency of reaching a given Checkpoint in a Supervised Entity matches specified limits.
Rationale:	To detect if a periodic function is executed periodically according to specification/design.
AppliesTo:	AP, CP
Dependencies:	–
Use Case:	A safety critical application with alive supervision get stuck at some point in time during execution. HM detects that the supervised application is not alive.
Supporting Material:	–

[RS_HM_09222] Health Monitoring shall provide a Logical Supervision
(State: DRAFT; Upstream Requirements: [RS_SAF_10005], [RS_SAF_10006], [RS_SAF_10030])

Description:	Health Monitoring shall check if the sequence of Checkpoints in a Supervised Entity at runtime is the same as the one that is specified. This shall include: • start of if/else branch (decision node): exactly one of the code branches shall be entered, the choice is runtime-specific depending on logical condition • end of if/else branch (merge node): exactly one of the branches shall be reached so that the join is performed • fork of the flow into concurrent execution (fork node): all concurrent branches shall be entered • join of the flow of concurrent execution (join node): all concurrent branches shall be reached so that the join is performed.
Rationale:	To detect if the sequence in the execution is the same as specified/designed.
AppliesTo:	AP, CP
Dependencies:	–

▽

△

Use Case:	Supervision of any software components: application software components or platform components (e.g. execution manager, state manager).
Supporting Material:	–

[RS_HM_09235] Health Monitoring shall provide a Deadline Supervision
(State: DRAFT; Upstream Requirements: [RS_SAF_10031])

Description:	Health Monitoring shall check if the elapsed time between two Checkpoints is within the specified min and max limits, including the detection if the second Checkpoint never arrives.
Rationale:	To detect timeouts or loss of deadlines.
AppliesTo:	AP, CP
Dependencies:	–
Use Case:	A safety critical application is developed to reach specific checkpoints in a defined time window and is suddenly not behaving as intended. PHM detects the violation.
Supporting Material:	–

4.3.2 AUTOSAR Adaptive Platform

4.3.2.1 Functional Cluster: Platform Health Management (PHM)

[RS_PHM_00115] If supervision of State Management fails then Platform Health Management shall trigger a watchdog reset.
(State: DRAFT; Upstream Requirements: [RS_SAF_10006], [RS_SAF_10030], [RS_SAF_10005])

Description:	If supervision of State Management fails then Platform Health Management shall trigger a watchdog reset.
Rationale:	State Management is a fundamental functional cluster of the Adaptive AUTOSAR, if it fails then Platform Health Management (which controls the watchdog) shall trigger a reset which is the only reasonable safety measure
Dependencies:	SM
Use Case:	SM is managing a safety critical `Adaptive Application`. Supervision of SM fails and is detected by PHM. PHM shall trigger a watchdog reset.
Supporting Material:	–

[RS_PHM_00116] If supervision of Execution Management fails then Platform Health Management shall trigger a watchdog reset.
(State: DRAFT; Upstream Requirements: [RS_SAF_10006], [RS_SAF_10030], [RS_SAF_10005])

Description:	If supervision of Execution Management fails then Platform Health Management shall trigger a watchdog reset.
Rationale:	Execution Management is a fundamental functional cluster of the Adaptive AUTOSAR, if it fails then Platform Health Management (which controls the watchdog) shall trigger a reset which is the only reasonable safety measure
Dependencies:	EM
Use Case:	EM is managing safety critical `Adaptive Application`s and supervision of EM fails and is detected by PHM. PHM shall trigger a watchdog reset.
Supporting Material:	–

[RS_PHM_00117] Platform Health Management shall notify State Management in case an `AUTOSAR Adaptive Platform` functional cluster, `Adaptive Application` or service other than Execution Management and State Management fails.
(State: DRAFT; Upstream Requirements: [RS_SAF_10005], [RS_SAF_10006])

Description:	Platform Health Management shall notify State Management in case an `AUTOSAR Adaptive Platform` functional cluster, `Adaptive Application` or service other than Execution Management and State Management fails.
Rationale:	Recovery actions are coordinated in SM, the failures shall be reported to SM except if SM or EM themselves fail.
Dependencies:	–
Use Case:	PHM supervises a safety critical `Adaptive Application`. This application fails. PHM detects the issue and reports to SM.
Supporting Material:	–

[RS_PHM_00118] PHM shall only process a checkpoint reported from corresponding processes.
(State: DRAFT; Upstream Requirements: [RS_SAF_10030])

Description:	PHM shall only process a checkpoint reported from corresponding processes.
Rationale:	The checkpoint can only be considered valid if it was reported from the corresponding configured process.
AppliesTo:	AP
Dependencies:	RS_IAM_00002, RS_IAM_00010
Use Case:	–
Supporting Material:	–

[RS_PHM_00119] A security event shall be raised if a checkpoint is reported from a non-corresponding process.
(State: DRAFT; Upstream Requirements: [RS_SAF_10030])

Description:	A security event shall be raised if a checkpoint is reported from a non-corresponding process.
Rationale:	A malicious software might try to enforce a false positive or a false negative by reporting checkpoints corresponding to other processes.
AppliesTo:	AP
Dependencies:	RS_IAM_00002, RS_IAM_00010, RS_Ids_00810
Use Case:	–
Supporting Material:	–

4.3.2.2 Functional Cluster: Execution Management (EM)

[RS_EM_00002] Execution Management shall set-up one process for the execution of each Modelled Process.
(State: DRAFT; Upstream Requirements: [RS_SAF_10037])

Description:	For each instance of an `Executable`, `Execution Management` shall allocate one POSIX process. Furthermore process specific properties (like priority, scheduling policy and access rights) shall be assigned based on the `Execution Manifest`.
Rationale:	Isolation of `Executable` instances from each other.
Dependencies:	–
Use Case:	Safety and security related applications require isolation.
Supporting Material:	–

[RS_EM_00005] Execution Management shall support the configuration of OS resource budgets for process and groups of processes.
(State: DRAFT; Upstream Requirements: [RS_SAF_10008])

Description:	Based on the `Execution Manifest`, `Execution Management` shall allocate OS resources to the `Process`. The allocation shall be possible for single `Process` and groups of `Processes`.
Rationale:	Real-time guarantees shall be defined
Dependencies:	–
Use Case:	Like `cgroups` (based on containers which contain one or more processes) and `ulimit`.
Supporting Material:	–

[RS_EM_00008] Execution Management shall support the binding of all threads of a given process to a specified set of processor cores.
(State: DRAFT; Upstream Requirements: [RS_SAF_10008])

Description:	`Execution Management` shall allow the binding of threads to specific set of processor cores based on configuration in the `Execution Manifest`. The binding granularity shall be at process level.
Rationale:	Mechanism to influence load balancing, reaction times, and latencies.
Dependencies:	–
Use Case:	A `Process` can be assigned to designated cores to limit thread migration between cores available on the `Machine`.
Supporting Material:	–

[RS_EM_00009] Execution Management shall control the right to create child process for each process it starts.
(State: DRAFT; Upstream Requirements: [RS_SAF_10001], [RS_SAF_10008])

Description:	`Execution Management` is responsible for starting child `Processes` and shall prevent such child `Processes` from directly starting other `Processes`, unless configured otherwise.
Rationale:	`Execution Management` needs full control of starting applications to ensure required isolation of temporal and spatial properties. However, existing software may require rights to create child `Processes` and it can be unpractical to modify it for use with `AUTOSAR Adaptive Platform`. For this reason, `Execution Management` allows selected `Processes` to create child `Processes`, but this must be configured by integrator and is not a right that is granted by default.
Dependencies:	–
Use Case:	Segregation between applications with different safety and/or security properties.
Supporting Material:	–

[RS_EM_00151] Execution Management shall be implemented at least according to the highest safety integrity level from any process that is supported on the platform.
(State: DRAFT; Upstream Requirements: [RS_SAF_10001])

Description:	`Execution Management` shall be implemented at least according to the highest safety integrity level from any `Process` that is supported on the platform.
Rationale:	`Execution Management` manages `Process` instantiation and termination of all the `Processes` and therefore needs to be developed and executed according to the same safety standards as the highest rated safety application managed by `Execution Management` in the system.
Dependencies:	–

▽

△

Use Case:	An ASIL C, B and QM application is running on the AUTOSAR Adaptive Platform. Execution Management shall execute the ASIL C, B and the QM application, therefore Execution Management shall be implemented with an ASIL C.
Supporting Material:	–

4.3.2.3 Functional Cluster: State Management (SM)

[RS_SM_00600] State Management shall be implemented at least according to the highest safety integrity level from any process that is managed by State Management. *(State: DRAFT; Upstream Requirements: [RS_SAF_10001])*

Description:	`State Management` shall be implemented at least according to the highest safety integrity level from any process that is managed by `State Management`.
Rationale:	`State Management` manages state changes and recovery actions of all the processes and therefore needs to be developed and executed according to the same safety standards as the highest rated safety application managed by `State Management` in the system.
Dependencies:	–
Use Case:	An ASIL C, B and QM Application is running on the adaptive Platform. `State Management` shall manage the ASIL C, B and the QM application, therefore `State Management` shall be implemented with an ASIL C.
Supporting Material:	–

[RS_SM_00601] State Management shall coordinate recovery actions. *(State: DRAFT; Upstream Requirements: [RS_SAF_10005], [RS_SAF_10006])*

Description:	`State Management` shall coordinate recovery actions.
Rationale:	`State Management` is a central functional cluster to which Platform Health Management reports supervision failures and `State Management` decides which recovery action (e.g. functional group state change, notification to a safe application or even ECU reset) should be triggered.
Dependencies:	PHM
Use Case:	PHM supervises a safety critical `Adaptive Application`. This application fails. PHM detects the issue and reports to `State Management`. `State Management` coordinates the error recovery actions.
Supporting Material:	–

4.3.2.4 Operating System Interface (OSI)

[RS_OSI_00201] The Operating System shall provide mechanisms for system memory budgeting.
(Upstream Requirements: [RS_SAF_10008])

Description:	The `Operating System` shall provide mechanisms to configure memory budgeting for each `Process` or for groups of `Processes`.
Rationale:	In order to ensure resource availability in the context of a multi-`Process` system, the system integrator/architect may require a set of tools to configure memory budgeting for each `Process` or for groups of `Processes`.
Dependencies:	–
Use Case:	security - protection against DoS attacks - resource starvation types.
Supporting Material:	–

[RS_OSI_00202] The Operating System shall provide mechanisms for CPU time budgeting.
(Upstream Requirements: [RS_SAF_10008])

Description:	The `Operating System` shall provide mechanisms to configure resource budgeting in terms of CPU time for each `Process` or group of `Processes`.
Rationale:	In order to ensure schedulability in the context of a multi-`Process` system, the system integrator/architect may require a set of tools to configure CPU time allocated for each `Process` or for groups of `Processes`.
Dependencies:	–
Use Case:	security - protection against DoS attacks - resource starvation types.
Supporting Material:	–

[RS_OSI_00203] The Operating System should provide mechanisms for binding Processes to CPU cores.
(Upstream Requirements: [RS_SAF_10008])

Description:	The `Operating System` should provide mechanisms for binding individual `Process` or groups of `Processes` to CPU cores.
Rationale:	In order to ensure correct `Task` schedulability, the system integrator may require a set of tools to configure the CPU affinity of `Processes`. In a multi-core system, it may be relevant to ensure some `Processes` can only run on some CPU cores, to allow other less- or differently-restricted `Processes` to concurrently progress.
Dependencies:	–
Use Case:	–
Supporting Material:	–

[RS_OSI_00206] The Operating System shall provide multi-Process support for isolation of applications.
(Upstream Requirements: [RS_SAF_10008], [RS_SAF_10037])

Description:	The `Operating System` shall provide mechanisms to let multiple `Processes` run isolated from each other.
Rationale:	Each `Process` may have a different robustness, safety and security level. As a consequence, an incorrect memory access from one `Process` execution shall not result in a corruption of memory in another `Process`, unless the data area is explicitly shared. In addition, a `Process` may not access or read data from another `Process` without explicit data sharing.
Dependencies:	–
Use Case:	–
Supporting Material:	–

4.3.2.5 Functional Cluster: Persistency (PER)

[RS_PER_00008] Detection of Data Corruption
(Upstream Requirements: [RS_SAF_10037])

Description:	Persistency shall be able to detect data corruption in persistent memory. **Additional Information:** The corruption may be caused by systematic or random failures. To be able to detect corrupted data, some redundancy is needed, which can be anything from a checksum to a full copy. Additionally, an adaptive application can register for receiving information about data corruption. The actual mechanisms used for ensuring data consistency are subject to configuration.

[RS_PER_00009] Recovery of Corrupted Data
(Upstream Requirements: [RS_SAF_10040])

Description:	Persistency shall be able to recover data that was corrupted. **Additional Information:** To be able to recover corrupted data, a redundant copy of the data is needed. Additionally, an adaptive application can register for receiving information about data recovery. The actual mechanisms and the granularity of redundancy are subject to configuration.

4.3.2.6 Functional Cluster: Communication Management (CM)

[RS_CM_00223] The Communication Management shall protect the transmission of events using E2E protocol. The E2E Protection has to be executed behind the event

API.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039])

Description:	Application developers shall be able to have an E2E-protected event-based communication, regardless of the bus used.
Rationale:	It shall be ensured that communication failure modes introduced by the communication bus (on the E2E-protected serialized data) which are detectable by the E2E protocol are detected by Communication Management. Note: It depends on the used communication type (periodic/ non-periodic) and the application which failure modes are to be detected.
Dependencies:	–
Use Case:	Application "A" receives an E2E-protected speed (as a part of an event). In case of a corruption or a loss, this is detected by a periodic polling by application and by E2E checks (CRC and a stuck-at counter), reported by Communication Management by E2E result. As a result, the application could enforce the safe state of its function, e.g. refusing to open tail gate.
Supporting Material:	[8]

[RS_CM_00224] The communication management shall provide the E2E information of the received event to the application.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039], [RS_SAF_10014])

Description:	The communication management shall provide the E2E information of the received event to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to perform an appropriate error handling. The access to the event data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application "A" polls gets invalid E2E check result and as a result it switches to a safe state.
Supporting Material:	The provided E2E information shall be, for each event in the queue: E2E status, E2E state, and the sample. Note that in case applications are triggered, there may be a need of an application-level detection of timeouts. This is because in case of delay or loss, the event will not arrive and E2E check will not be performed.

[RS_CM_00400] Communication Management shall protect the transmission of methods using E2E protocol.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039])

Description:	Communication Management shall, transparent to the application, protect the transmission of methods using E2E protocol.
Rationale:	It shall be ensured that communication failure modes introduced by the communication bus (on the E2E-protected serialized request or response data) which are detectable by the E2E protocol are detected at the client side by Communication Management. Note: It depends on the used communication type (periodic/ non-periodic) and the application which failure modes are to be detected.
Dependencies:	–
Use Case:	E2E protected method calls in client-server based communication
Supporting Material:	[8]

[RS_CM_00401] The communication management shall provide the E2E information of the received method call to the application.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039], [RS_SAF_10014])

Description:	The communication management shall provide the E2E information of the received method call to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to propagate detected E2E failure modes to the response data provided to the client. The access to the request data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application "B" provides a method and this method is called by application "A" and receives with the request invalid E2E check result and as a result the same invalid E2E data are added to the response data
Supporting Material:	The provided E2E information shall be E2E status, E2E state and object data.

[RS_CM_00403] Communication management shall provide an interface to detect delay of E2E protected service responses at the client side by supervision of a predefined response deadline.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039], [RS_SAF_10014])

Description:	Communication management shall provide an interface to detect delayed service responses at the client side by supervision of a predefined response deadline.
Rationale:	A delayed response shall be detected and the application can apply a safety related error reaction.
Dependencies:	–
Use Case:	Client is sending a method call. Client is awaiting the response within 300ms. After reaching the deadline the fault is detected at client side.
Supporting Material:	–

[RS_CM_00404] The communication management shall provide the E2E information of the method response to the application.
(State: DRAFT; Upstream Requirements: [RS_SAF_10039], [RS_SAF_10014])

Description:	The communication management shall provide the E2E information of the method response to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to perform an appropriate error handling. The access to the response data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application "A" requests a method call and receives with the response an invalid E2E check result and as a result it switches to a safe state.
Supporting Material:	Note, there may be a need of an application-level monitoring of a deadline to stop waiting for a response.

4.3.2.7 Functional Cluster: Update and Configuration Management (UCM)

[RS_UCM_00008] `UCM` shall support a recovery mechanism in case of failed activation
(State: DRAFT; Upstream Requirements: [RS_SAF_10027])

Description:	`UCM` shall assure that, in case of failed update process, the system will recover to the state it was before the update process started.
Rationale:	A failed update shall not result in a loss of desired functionality of the `AUTOSAR Adaptive Platform`.
Dependencies:	[RS_UCM_00021]
Use Case:	After a failed remote update the `AUTOSAR Adaptive Platform` recovers to the previous system state.
Supporting Material:	–

[RS_UCM_00012] `UCM` shall check the consistency of transferred `Software Package`
(State: DRAFT; Upstream Requirements: [RS_SAF_10039])

Description:	`UCM` shall check the consistency of the received `Software Package`.
Rationale:	`AUTOSAR Adaptive Platform` shall make sure that the `Software Package` can be installed safely.
Dependencies:	–
Use Case:	To detect possible errors which might have occurred during creation of the `Software Package`, `UCM` shall check that provided `Software Package` meta-data and content match.
Supporting Material:	–

[RS_UCM_00027] `UCM` shall be able to safely recover from unexpected interruption.
(State: DRAFT; Upstream Requirements: [RS_SAF_10027])

Description:	At startup, UCM shall be able to identify if some action was interrupted and exited in an uncontrolled way and needs to be reverted or finished to return the software into the previous state.
Rationale:	UCM shall make sure that software should not be started up into inconsistent and not updatable state.
Dependencies:	–
Use Case:	After unexpected reset or crash UCM shall identify that there was an interruption while an action was on going and UCM shall handle this by reverting or by finishing the unfinished action.
Supporting Material:	–

[RS_UCM_00030] UCM shall be able to verify the updated software during activation *(State: DRAFT; Upstream Requirements: [RS_SAF_10002])*

Description:	UCM shall require the updated software to be executed and verified before declaring that SW was successfully activated.
Rationale:	UCM shall declare activation to be successful only after it detects that Execution Manager can execute the software successfully.
Dependencies:	–
Use Case:	Ensuring that safety-critical application can be executed and thus monitored by the `Platform Health Manager`.
Supporting Material:	–

4.3.3 AUTOSAR ClassicPlatform

4.3.3.1 Basic Software: Watchdog Manager (WDGM)

[RS_SAF_31101] Watchdog Manager inherits highest safety integrity level from Software Component.

Status: DRAFT
Upstream requirements: RS_SAF_10001

⌈

Description:	Watchdog Manager shall inherit at least the highest safety integrity level from any Software Component that is running on the platform.
Rationale:	Watchdog Manager is responsible for ensuring part of the safe execution of safety relevant software components/applications, it should at least be developed with the highest ASIL as the software component/application that is being supervised.

⌋

△

Use Case:	An ASIL C, B and QM Application is running on the Classic Platform. WdgM shall supervise the ASIL C and B application, therefore WdgM shall be implemented with an ASIL C.
AppliesTo:	CP
Dependencies:	WdgIf, Wdg Drv
Supporting Material:	–

⌋

[RS_SAF_31102] Watchdog Manager monitors aliveness.

Status: DRAFT
Upstream requirements: RS_SAF_10031

⌈

Description:	Watchdog Manager shall monitor the aliveness of safety relevant software components/applications and modules.
Rationale:	Alive Supervision is one of the mechanisms of Watchdog Manager by which it monitors safety relevant software components/applications and modules.
Use Case:	A safety critical functionality with alive supervision gets stuck at some point in time during execution. WdgM detects that the supervised application is not alive.
AppliesTo:	CP
Dependencies:	WdgIf, Wdg Drv
Supporting Material:	–

⌋

[RS_SAF_31103] Watchdog Manager monitors control flow.

Status: DRAFT
Upstream requirements: RS_SAF_10005, RS_SAF_10006, RS_SAF_10030

⌈

Description:	Watchdog Manager shall monitor the control flow of safety relevant software components/applications and modules.
Rationale:	Logical Supervision is one of the mechanisms of Watchdog Manager by which it monitors safety relevant software components/applications and modules.
Use Case:	A safety relevant functionality is developed to follow a specific control flow and is suddenly not following the intended sequence. Watchdog Manager detects the control flow violation.
AppliesTo:	CP
Dependencies:	WdgIf, Wdg Drv
Supporting Material:	–

⌋

[RS_SAF_31104] Watchdog Manager monitors deadline.

Status: DRAFT
Upstream requirements: RS_SAF_10031

⌈

Description:	Watchdog Manager shall monitor that the duration between the checkpoints of safety relevant software components/applications and modules are within the minimum and maximum configured time limits.
Rationale:	Deadline Supervision is one of the mechanisms of Watchdog Manager by which it monitors safety relevant functionalities.
Use Case:	A safety critical application is developed to reach specific checkpoints in a defined time window and is suddenly not behaving as intended. WdgM detects the violation.
AppliesTo:	CP
Dependencies:	WdgIf, Wdg Drv
Supporting Material:	–

⌋

4.3.3.2 Basic Software: Operating System (OS)

[RS_SAF_31201] Memory Protection of Applications

Status: DRAFT
Upstream requirements: RS_SAF_10037

⌈

Description:	The Operating System shall prevent applications from performing write accesses outside their assigned memory regions
Rationale:	To achieve freedom from interference it is necessary to prevent applications from adversely affecting other applications. Access to private memory which is reserved for applications shall be protected against un-allowed write accesses from other applications.
Use Case:	A QM application and a ASIL application are executed on the same machine. OS prevents the QM application from changing the memory assigned to the safety critical application.
AppliesTo:	CP
Dependencies:	OSEK OS
Supporting Material:	–

⌋

[RS_SAF_31202] Timing Protection of Applications

Status: DRAFT
Upstream requirements: RS_SAF_10031

⌈

Description:	The Operating System shall not allow a timing fault in any application to propagate. A timing fault may be caused by • exceeding a statically/pre-runtime specified execution time budget • exceeding a statically/pre-runtime specified blocking time budget • exceeding a statically/pre-runtime specified arrival rate
Rationale:	A timing fault in one application might trigger a chain of timing faults and this shall be prevented.
Use Case:	A QM application and a ASIL application are executed on the same machine. OS prevents the QM application from propagating its delay to the safety critical application.
AppliesTo:	CP
Dependencies:	OSEK OS
Supporting Material:	–

⌋

4.3.3.3 E2E Protection

[RS_SAF_31301] E2E Protection with E2E Transformer and E2E Library

Status: DRAFT
Upstream requirements: RS_SAF_10014

⌈

Description:	Communication Service, E2E Transformer and E2E Library shall provide mechanisms for detection of errors during the exchange of information among software components, by considering all faults listed in the ISO standard (ISO 26262:6-2018 D.2.4). **The Result of the E2E check needs to be published to the application.** If E2E Transformer is used RTE Interfaces need to be developed according to the same ASIL Level as the Application and data being transformed.
Rationale:	This requirement is created initially to fulfill the goal of AUTOSAR in supporting the development of safety-related systems by offering safety measures and mechanisms. As users may build project-specific applications, it is only possible for AUTOSAR to provide the safe exchange of information. ISO 26262 is mentioned and to be followed, as it is the international standard for functional safety of E/E systems for automotive.

▽

△

Use Case:	Two ASIL rated applications on different control devices shall exchange information through a component (HW or SW) with a lower rated ASIL. E2E Transformer and E2E Library shall support safety mechanisms like a counter, a checksum and a timestamp to allow the ASIL applications or the E2E Transformer and E2E Library implementations to detect and ensure that the information has been transmitted correctly, in time and in-order.
AppliesTo:	CP
Dependencies:	E2E Library, E2E Transformer, RTE, SWC
Supporting Material:	–

⌋

[RS_SAF_31302] Allow integrators to configure safety mechanisms to detect communication faults

Status: DRAFT
Upstream requirements: RS_SAF_10001

⌈

Description:	Communication Service, E2E Transformer and E2E Library shall, based on individual safety concepts, allow integrators to select and configure the set of safety mechanisms to detect communication faults.
Rationale:	Different communication buses and data information may have different needs to be protected by the E2E. Therefore, it is reasonable to have the configurability (pre-deployment) such that integrators may freely select the set of mechanisms to be deployed.
Use Case:	A high-level design change or new information requires a different communication protection mechanism. An integrator can select the proper protection by changing the Manifest.
AppliesTo:	CP
Dependencies:	E2E Library, E2E Transformer, RTE, SWC
Supporting Material:	–

⌋

5 References

[1] Explanation of Adaptive Platform Design
AUTOSAR_AP_EXP_PlatformDesign

[2] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture

[3] Explanation of Adaptive Platform Software Architecture
AUTOSAR_AP_EXP_SWArchitecture

[4] Explanation of Safety Overview
AUTOSAR_FO_EXP_SafetyOverview

[5] ISO 26262:2018 Road vehicles -– Functional Safety
https://www.iso.org

[6] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[7] Glossary
AUTOSAR_FO_TR_Glossary

[8] Requirements on E2E
AUTOSAR_FO_RS_E2E

A Change history of AUTOSAR traceable items and requirements

A.1 AUTOSAR Release R22-11

A.1.1 Added Requirements in R22-11

[RS_SAF_00007] [RS_SAF_10039] [RS_SAF_10040] [RS_SAF_10041] [RS_SAF_-10042]

A.1.2 Changed Requirements in R22-11

[RS_SAF_10001] [RS_SAF_10002] [RS_SAF_10005] [RS_SAF_10006] [RS_SAF_-10008] [RS_SAF_10014] [RS_SAF_10027] [RS_SAF_10028] [RS_SAF_10030] [RS_SAF_10031] [RS_SAF_10037] [RS_SAF_10038] [RS_SAF_21401] [RS_SAF_31202]

A.1.3 Deleted Requirements in R22-11

[RS_SAF_21101] [RS_SAF_21102] [RS_SAF_21103] [RS_SAF_21104] [RS_SAF_-21105] [RS_SAF_21106] [RS_SAF_21107] [RS_SAF_21201] [RS_SAF_21202] [RS_SAF_21301] [RS_SAF_21302] [RS_SAF_21501] [RS_SAF_21502] [RS_SAF_21601] [RS_SAF_21602] [RS_SAF_21701] [RS_SAF_21702] [RS_SAF_21703] [RS_SAF_-21704]

A.2 AUTOSAR Release R23-11

A.2.1 Added Requirements in R23-11

none

A.2.2 Changed Requirements in R23-11

[RS_SAF_00005] [RS_SAF_00006] [RS_SAF_00007] [RS_SAF_10001] [RS_SAF_-10005] [RS_SAF_10006] [RS_SAF_10008] [RS_SAF_10014] [RS_SAF_10027] [RS_SAF_10028] [RS_SAF_10030] [RS_SAF_10031] [RS_SAF_10039] [RS_SAF_10041]

A.2.3 Deleted Requirements in R23-11

[RS_SAF_21401] [RS_SAF_21402] [RS_SAF_21403]

A.3 AUTOSAR Release R24-11

A.3.1 Added Requirements in R24-11

none

A.3.2 Changed Requirements in R24-11

none

A.3.3 Deleted Requirements in R24-11

none

A.4 AUTOSAR Release R25-11

A.4.1 Added Requirements in R25-11

none

A.4.2 Changed Requirements in R25-11

[RS_SAF_00001] [RS_SAF_00002] [RS_SAF_00003] [RS_SAF_00004] [RS_SAF_-00005] [RS_SAF_00006] [RS_SAF_00007] [RS_SAF_10001] [RS_SAF_10002] [RS_SAF_10005] [RS_SAF_10006] [RS_SAF_10008] [RS_SAF_10014] [RS_SAF_10027] [RS_SAF_10028] [RS_SAF_10030] [RS_SAF_10031] [RS_SAF_10037] [RS_SAF_-10038] [RS_SAF_10039] [RS_SAF_10040] [RS_SAF_10041] [RS_SAF_10042] [RS_SAF_31101] [RS_SAF_31102] [RS_SAF_31103] [RS_SAF_31104] [RS_SAF_31201] [RS_SAF_31202] [RS_SAF_31301] [RS_SAF_31302]

A.4.3 Deleted Requirements in R25-11

none