Document Title	Requirements on Execution Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	720

Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R23-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2023-11-23	R23-11	AUTOSAR Release Management	• Requirements for deterministic execution are set to obsolete • The right to create child processes can be configured by integrator • Added support for standardized trace points
2022-11-24	R22-11	AUTOSAR Release Management	• Added: RS_EM_00151 • Changed uptraces to RS_SAF requirements
2021-11-25	R21-11	AUTOSAR Release Management	• Added: RS_EM_00015
2020-11-30	R20-11	AUTOSAR Release Management	• Added: RS_EM_00150
2019-11-28	R19-11	AUTOSAR Release Management	• Updated: RS_EM_00009 and RS_EM_00103 • Changed Document Status from Final to published
2019-03-29	19-03	AUTOSAR Release Management	• Updated: RS_EM_00008 and RS_EM_00010

▽

△

2018-10-31	18-10	AUTOSAR Release Management	• Removed: RS_EM_00003, RS_EM_00004, RS_EM_00110 and RS_EM_00111. • Added: [RS_EM_00014].
2018-03-29	18-03	AUTOSAR Release Management	• Removed: RS_EM_00006, RS_EM_00007 and RS_EM_00012 • Minor changes and document clean up
2017-10-27	17-10	AUTOSAR Release Management	• Minor changes, document clean up
2017-03-31	17-03	AUTOSAR Release Management	• Initial release

Disclaimer

Contents

1 Scope of this document

This document specifies requirements of the AUTOSAR Adaptive Platform on the Execution Management. The motivation is to provide a standardized way to start, stop and police applications platform wide.

2 Conventions to be used

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see Standardization Template [1], chapter Support for Traceability.

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see Standardization Template [1], chapter Support for Traceability.

2.1 Requirements Guidelines

2.1.1 Requirements quality

2.1.2 Requirements identification

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 market-

place 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.)

2.1.3 Requirements status

The following requirements are described within this document but not otherwise considered in this release:

- [RS_EM_00111] – Identification of Processes

The functionality described above is subject to modification and will be considered for inclusion in a future release of this document.

3 Acronyms and abbreviations

All technical terms used throughout this document – except the ones listed here – can be found in the official [2] AUTOSAR Glossary or [3] TPS Manifest Specification.

Term	Description
process	A process refers to the OS concept of a running process. **Attention:** process is **not equal** to Modelled Process (see below). Hence each Modelled Process has at some time a related (OS) process but a process may not always have a related Modelled Process.
Modelled Process	A Modelled Process is an instance of an Executable to be executed on a Machine.
Execution Dependency	Dependencies between Executable instances can be configured to define a sequence for starting and terminating them.
Execution Management	The element of the AUTOSAR Adaptive Platform responsible for the ordered startup and shutdown of the AUTOSAR Adaptive Platform and Adaptive Applications.
State Management	The element defining modes of operation for AUTOSAR Adaptive Platform. It allows flexible definition of functions which are active on the platform at any given time.
Identity and Access Management (IAM)	A Adaptive Platform Service within the AUTOSAR Adaptive Platform
Function Group	A Function Group is a set of coherent Modelled Processes, which need to be controlled consistently. Depending on the state of the Function Group, processes (related to the Modelled Processes) are started or terminated. processes can belong to more than one Function Group State (but at exactly one Function Group). "MachineFG" is a Function Group with a predefined name, which is mainly used to control Machine lifecycle and processes of platform level Applications. Other Function Groups are sort of general purpose tools used (for example) to control processes of user level Applications.
Function Group State	The element of State Management that characterizes the current status of a set of (functionally coherent) user-level Applications. The set of Function Groups and their Function Group States is machine specific and are configured in Machine Manifest.
Machine State	A state of Function Group "MachineFG" with some predefined states (Startup/Shutdown/Restart). This can term can refer to the current state ("The Machine State is ..."), to a specific state ("In Machine State Startup ..."), or to a set of states ("In Machine States Startup or Shutdown ...").
Time Determinism	The results of a calculation are guaranteed to be available before a given deadline.
Data Determinism	The results of a calculation only depend on the input data and are reproducible, assuming a given initial internal state.
Full Determinism	Combination of Time and Data Determinism.
Communication Management	A Functional Cluster within the Adaptive Platform Foundation

Execution Manifest	Manifest file to configure execution of an Adaptive Application. An Execution Manifest is created at integration time and deployed onto a Machine together with the Executable to which it is attached. It supports the integration of the Executable code and describes the configuration properties (startup parameters, resource group assignment etc.) of each process, i.e. started instance of that Executable.
Machine Manifest	Manifest file to configure a Machine. The Machine Manifest holds all configuration information which cannot be assigned to a specific Executable or process.
Operating System	Software responsible for managing processes on a Machine and for providing an interface to hardware resources.
ResourceGroup	Configuration element to enable restrictions on resources uses by Adaptive Applications running in the group.
ExecutionClient	Adaptive Application interface to Execution Management.
DeterministicClient	Adaptive Application interface to Execution Management to support control of the process-internal cycle, a deterministic worker pool, activation time stamps and random numbers.
Platform Health Management	A Functional Cluster within the Adaptive Platform Foundation
Process State	Lifecycle state of a Modelled Process
Service Instance Manifest	Manifest file to configure Service usage of an Adaptive Application.
Trusted Platform	An execution platform supporting a continuous chain of trust from boot through to application supporting authentication (that all code executed is from the claimed source) and integrity validation (that prevents tampered code/data from being executed).

Table 3.1: Technical Terms

The following technical terms used throughout this document are defined in the official [2] AUTOSAR Glossary or [3] TPS Manifest Specification – they are repeated here for tracing purposes.

Term	Description
Adaptive Application	see [2] AUTOSAR Glossary
Application	see [2] AUTOSAR Glossary
AUTOSAR Adaptive Platform	see [2] AUTOSAR Glossary
Adaptive Platform Foundation	see [2] AUTOSAR Glossary
Manifest	see [2] AUTOSAR Glossary
Executable	see [2] AUTOSAR Glossary
Functional Cluster	see [2] AUTOSAR Glossary
Adaptive Platform Service	see [2] AUTOSAR Glossary
Machine	see [2] AUTOSAR Glossary
Service	see [2] AUTOSAR Glossary
Service Interface	see [2] AUTOSAR Glossary
Service Discovery	see [2] AUTOSAR Glossary

Table 3.2: Glossary-defined Technical Terms

4 Requirements Specification

4.1 Functional Overview

The AUTOSAR Adaptive Platform provides services to influence the lifecycle of Applications based on configuration. This document therefore includes requirements that determine the facilities provided by Execution Management to affect the machine-wide startup, shutdown and restart of an Application based on configuration.

Execution Management is responsible for all aspects of platform lifecycle management and application lifecycle management, including:

- Machine startup and shutdown.
  - Execution Management is the initial (“boot”) process of the operating system.
- Required process hierarchy of started services, e.g., init and its child process.
  - after booting. The boot process in this case corresponds to machine init process.
- Provision of process isolation with each instance of an Executable managed as a single process.
- Startup and shutdown of Applications.
  - Loading Executable based on a defined Execution Dependency.
  - Specific requirements until starting an Executable main function (i.e. entry point)
- Privileges and use of access control
  - description and semantics of access control in manifest files
- State management
  - Conditions for the execution of Applications

EM, PHM and SM are the main safety relevant functional clusters of the AUTOSAR Adaptive Platform. Consequently, their development may require certain processes to be followed - as recommended in ISO26262. A safety argumentation for the AUTOSAR Adaptive Platform, describing functional safety measures and use-cases is provided through Explanation of Safety Overview [4].

4.2 Functional Requirements

This section describes all requirements driving the work to define Execution Management functionality.

4.2.1 Startup and Shutdown of Applications

[RS_EM_00002]{DRAFT} **Execution Management shall set-up one process for the execution of each Modelled Process.** ⌈

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_Main_00010, RS_Main_00049, RS_Main_00080, RS_Main_00320, RS_Main_00150, RS_Main_00420, RS_SAF_10037)*

[RS_EM_00014]{DRAFT} **Execution Management shall support a Trusted Platform.** ⌈

Description:	Execution Management shall ensure that integrity and authenticity are checked for all Executables and their corresponding Execution Management meta-data (i.e. processed Machine and Execution Manifests), and shall only allow starting Executables that passed validation check.
Rationale:	Execution Management takes over the responsibility from Operating System and/or boot loader for AUTOSAR Adaptive Platform startup and hence for keeping the platform trusted. Execution Management is the only AUTOSAR Adaptive Platform entity allowed to start Executables and therefore responsible for the continuation of trust for the AUTOSAR Adaptive Platform.
Dependencies:	–
Use Case:	Verify the integrity and authenticity of software deployed on AUTOSAR Adaptive Platform.
Supporting Material:	–

⌋*(RS_Main_00170, RS_Main_00514, RS_Main_00180)*

[RS_EM_00015]{DRAFT} **Execution Management shall support integrity and authenticity monitoring.** ⌈

Description:	Execution Management shall support configurable integrity and authenticity monitoring for all Executables and their corresponding Execution Management meta-data (i.e. processed Machine and Execution Manifests).
Rationale:	Execution Management takes over the responsibility from Operating System and/or boot loader for AUTOSAR Adaptive Platform startup and hence for keeping the platform trusted. Execution Management is the only AUTOSAR Adaptive Platform entity allowed to start Executables and therefore responsible for the continuation of trust for the AUTOSAR Adaptive Platform. However unsigned SW (or incorrectly signed SW) may at times be used and to allow this, Execution Management should optionally support execution. However the presence of such deployments should be noted.
Dependencies:	–
Use Case:	Support deployment of prototype (unsigned) software during system development.
Supporting Material:	–

⌋*(RS_Main_00170, RS_Main_00514, RS_Main_00180)*

[RS_EM_00005]{DRAFT} **Execution Management shall support the configuration of OS resource budgets for process and groups of processes.** ⌈

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 processs.
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_Main_00002, RS_Main_00010, RS_Main_00106, RS_Main_00340, RS_Main_00150, RS_SAF_10008)*

[RS_EM_00008]{DRAFT} **Execution Management shall support the binding of all threads of a given process to a specified set of processor cores.** ⌈

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_Main_00010*, *RS_Main_00050*, *RS_Main_00106*, *RS_Main_00320*, *RS_Main_00501*, *RS_Main_00150*, *RS_SAF_10008*)

[RS_EM_00009]{DRAFT} **`Execution Management` shall control the right to create child `processes` for each `process` it starts.** ⌈

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_Main_00010*, *RS_Main_00011*, *RS_Main_00049*, *RS_Main_00150*, *RS_SAF_10001*, *RS_SAF_10008*)

[RS_EM_00010] `Execution Management` shall support multiple instances of `Executables`. ⌈

Description:	It shall be possible to start more than one `Modelled Process` from a single `Executable`. Instance specific information is described in `Modelled Process` startup configuration.
Rationale:	Avoid code duplication.
Dependencies:	–
Use Case:	Redundancy of an `Executable` by parallel execution of two instances.
Supporting Material:	–

⌋(*RS_Main_00002*, *RS_Main_00049*, *RS_Main_00106*, *RS_Main_00501*)

[RS_EM_00011] Execution Management shall support self-initiated graceful shutdown of processes. ⌈

Description:	Execution Management shall support self-initiated graceful shutdown of processes.
Rationale:	Self-initiated graceful shutdown enables a process to free allocated dedicated resources and inform other interacting entities about its shutdown (e.g. de-registering a service) to create a consistent state within the Machine/vehicle. Self-initiated process shutdown is, by definition, only be initiated by the process itself.
Dependencies:	–
Use Case:	The process of an Executable instance is finished and shuts down itself.
Supporting Material:	–

⌋*(RS_Main_00002, RS_Main_00049)*

[RS_EM_00100] Execution Management shall support the ordered startup and shutdown of processes. ⌈

Description:	Execution Management shall support the ordered startup and shutdown of Executable instances.
Rationale:	Ensure that startup and shutdown dependencies between Executable instances are respected, if an execution dependency is specified in the Execution Manifest of an Executable instance. If no execution dependency is specified between Executable instances, they can be started and stopped in an arbitrary order.
Dependencies:	–
Use Case:	An Executable needs a specific functional cluster to be up and running before it can be started.
Supporting Material:	–

⌋*(RS_Main_00002, RS_Main_00049, RS_Main_00340, RS_Main_00460)*

4.2.2 Execution

[RS_EM_00050]{DRAFT} **Execution Management shall perform Machine-wide coordination of processes.** ⌈

Description:	Execution Management shall provide an API for a process to register its activities for being able to coordinate their execution.
Rationale:	Coordinated scheduling of activities across Executables.
Dependencies:	–

▽

△

Use Case:	Usage of computation resources within the running `processes` shall be managed in the `Machine` to ensure that activities can be coordinated across `processes`. Registration enables `Execution Management` to form the necessary `Machine`-wide view for the coordination.
Supporting Material:	–

⌋(*RS_Main_00460*, *RS_SAF_10008*)

[RS_EM_00051]{DRAFT} **`Execution Management` shall provide APIs to the `process` for configuring external trigger conditions for its activities.** ⌈

Description:	`Execution Management` shall provide an API for configuring the trigger conditions of registered activities.
Rationale:	`Execution Management` shall have the information when to schedule the activities.
Dependencies:	–
Use Case:	Execution on data receipt, sequencing of activity execution.
Supporting Material:	–

⌋(*RS_Main_00050*, *RS_Main_00060*)

[RS_EM_00052]{DRAFT} **`Execution Management` shall provide APIs to the `process` for configuring cyclic triggering of its activities.** ⌈

Description:	`Execution Management` shall provide an API for configuring the cyclic triggering of registered activities.
Rationale:	`Execution Management` shall have the information when to schedule the activities.
Dependencies:	–
Use Case:	Cyclic execution of activities
Supporting Material:	–

⌋(*RS_Main_00050*, *RS_Main_00340*)

[RS_EM_00053]{OBSOLETE} **`Execution Management` shall provide APIs to the `process` to support deterministic redundant execution of `processes`.** ⌈

Description:	`Execution Management` shall provide APIs to support deterministic redundant execution of `processes`.
Rationale:	High ASIL systems require safety mechanism like software lockstep to be implemented on non-automotive grade microprocessors. The redundant execution shall guarantee deterministic, i.e. reproducible results.
Dependencies:	–
Use Case:	Redundant execution of activities to implement software lockstep

▽

△

Supporting Material:	–

⌋(*RS_Main_00010*, *RS_Main_00501*, *RS_SAF_10028*)

[RS_EM_00113]{DRAFT} **`Execution Management` shall support time-triggered execution.** ⌈

Description:	`Execution Management` shall facilitate time-triggered periodic execution.
Rationale:	Algorithms in processes can be time-triggered. The OS needs to provide mechanisms to allow the time-triggered execution of applications. The triggers need to contain at least external timers, but are not limited to.
Dependencies:	–
Use Case:	Redundant execution of activities to implement software lockstep
Supporting Material:	–

⌋(*RS_Main_00010*, *RS_Main_00501*, *RS_SAF_10028*)

[RS_EM_00111]{DRAFT} **`Execution Management` shall assist identification of `processes` during `Machine` runtime.** ⌈

Description:	`Adaptive Applications` shall be identifiable, for example by `Identity and Access Management`, during runtime so that access restrictions can be enforced. `Execution Management` spawns runtime processes based on `Execution Manifest`. `Execution Management` is qualified to assist `AUTOSAR Adaptive Platform` software, such as `Identity and Access Management`, by providing information about the link between runtime representation and `Modelled Process`.
Rationale:	`Adaptive Applications` shall be identifiable by `Identity and Access Management` on the basis of their runtime representation as spawned by `Execution Management`.
Dependencies:	–
Use Case:	App A requests access on `Service Interface`. `Identity and Access Management` is able to retrieve runtime information of App A, e.g. POSIX pid or cryptographic token. `Execution Management` assists `Identity and Access Management` by resolving this runtime information to the `Adaptive Application`.
Supporting Material:	–

⌋(*RS_Main_00170*, *RS_Main_00514*, *RS_Main_00420*)

[RS_EM_00152]{DRAFT} **Execution Management shall support standardized trace points throughout the state transitions.** ⌈

Description:	Execution Management shall support standardized trace points throughout the state transitions.
Rationale:	Providing standardized trace points in a Functional Cluster allows comparison of timing in different implementations upon state changes, such as upon application process creation, initialization, and termination.
Dependencies:	ara::log
Use Case:	Tracing and timing analysis of different Applications and Functional Clusters behavior.
Supporting Material:	–

⌋*(RS_Main_01026)*

4.2.3 State Management

[RS_EM_00101]{DRAFT} **Execution Management shall support State Management functionality.** ⌈

Description:	Execution Management shall provide an interface to State Management to request a change in Function Group State.
Rationale:	To support the starting and stopping of processes based on declared Function Group State dependencies, Execution Management provides an interface to request Function Group State (including Machine State) changes by the State Management fuctional cluster. In response to state change requests, Execution Management ensures that only the required set of Application processes are running in any given operation conditions and therefore platform resources are saved for relevant processes.
Dependencies:	–
Use Case:	Provide a mechanism to define modes of operation of the Machine.
Supporting Material:	–

⌋*(RS_Main_00460)*

[RS_EM_00103] `Execution Management` shall support `process` lifecycle management. ⌈

Description:	The lifecycle of a `process` consists of its initialization, running and terminating (shutdown) phases. As well as supporting transitions between these phases of the `process` lifecycle, `Execution Management` should ensure that phases, e.g. the startup and shutdown, of `processes` can be coordinated between groups of `processes` which shall run in the same `Machine State` or `Function Group State`. Coordination and tracking of lifecycle phases enables `Execution Management` to ensure that `Executable`'s `processes` are fully established and running before other `processes` which depend on their functionality can be started.
Rationale:	Coordination and tracking of lifecycle phases enables `Execution Management` to ensure that Executable `processes` are fully established and running before other executable `processes` which depend on their functionality can be started.
Dependencies:	–
Use Case:	
Supporting Material:	–

⌋*(RS_Main_00049, RS_Main_00050, RS_Main_00106, RS_Main_00460)*

4.2.4 Error Handling

[RS_EM_00150]{DRAFT} **Error Handling.** ⌈

Description:	`Execution Management` shall support error handling including unrecoverable errors.
Rationale:	`Execution Management` may face conditions where it has no mechanism to recover the system. These situations are typically expected to result from a misconfigured system and therefore a suitable response might be to halt startup so that the misconfiguration can be resolved.
Dependencies:	–
Use Case:	`Execution Management` can not start PHM or `State Management` and hence the platform as a whole cannot be started, it is not possible to recover from this situation hence `Execution Management` must halt startup.
Supporting Material:	–

⌋*(RS_Main_00011)*

4.2.5 Support for Diagnostics

Support for Diagnostics is handled by `State Management` and therefore the requirements are replaced by the ones from [5].

4.3 Non-Functional Requirements

[RS_EM_00151]{DRAFT} **Execution Management shall be implemented at least according the highest safety integrity level from any process that is supported on the platform.** ⌈

Description:	`Execution Management` shall be implemented at least according 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.
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.
AppliesTo:	AP
Dependencies:	EM
Supporting Material:	–

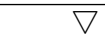
⌋*(RS_SAF_10001)*

5 Requirements Tracing

The following tables reference the requirements specified in [6] and links to the fulfillment of these.

Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document. Likewise, an entry of [RS_EM_NA] indicates that the source requirement has been evaluated as “not applicable” to `Execution Management`.

Requirement	Description	Satisfied by
[RS_Main_00002]	AUTOSAR shall provide a software platform for high performance computing platforms	[RS_EM_00005] [RS_EM_00010] [RS_EM_00011] [RS_EM_00100]
[RS_Main_00010]	Safety Mechanisms	[RS_EM_00002] [RS_EM_00005] [RS_EM_00008] [RS_EM_00009] [RS_EM_00053] [RS_EM_00113]
[RS_Main_00011]	Mechanisms for Reliable Systems	[RS_EM_00009] [RS_EM_00150]
[RS_Main_00049]	AUTOSAR shall provide an Execution Management for running multiple applications	[RS_EM_00002] [RS_EM_00009] [RS_EM_00010] [RS_EM_00011] [RS_EM_00100] [RS_EM_00103]
[RS_Main_00050]	AUTOSAR shall provide an Execution Framework towards applications to implement concurrent application internal control flows	[RS_EM_00008] [RS_EM_00051] [RS_EM_00052] [RS_EM_00103]
[RS_Main_00060]	Standardized Application Communication Interface	[RS_EM_00051]
[RS_Main_00080]	Formal Description Language	[RS_EM_00002]
[RS_Main_00106]	AUTOSAR shall provide the possibility to extend the software with new SWCs without recompiling the platform foundation	[RS_EM_00005] [RS_EM_00008] [RS_EM_00010] [RS_EM_00103]
[RS_Main_00150]	AUTOSAR shall support the deployment and reallocation of AUTOSAR Application Software	[RS_EM_00002] [RS_EM_00005] [RS_EM_00008] [RS_EM_00009]
[RS_Main_00170]	AUTOSAR shall provide secure access to ECU data and services	[RS_EM_00014] [RS_EM_00015] [RS_EM_00111]
[RS_Main_00180]	Intellectual Property Protection	[RS_EM_00014] [RS_EM_00015]
[RS_Main_00320]	AUTOSAR shall provide formats to specify system development	[RS_EM_00002] [RS_EM_00008]
[RS_Main_00340]	AUTOSAR shall support the continuous timing requirement analysis	[RS_EM_00005] [RS_EM_00052] [RS_EM_00100]
[RS_Main_00420]	AUTOSAR shall use established software standards and consolidate de-facto standards for basic software functionality	[RS_EM_00002] [RS_EM_00111]
[RS_Main_00460]	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	[RS_EM_00050] [RS_EM_00100] [RS_EM_00101] [RS_EM_00103]
[RS_Main_00501]	AUTOSAR shall support redundancy concepts	[RS_EM_00008] [RS_EM_00010] [RS_EM_00053] [RS_EM_00113]
[RS_Main_00514]	System Security Support	[RS_EM_00014] [RS_EM_00015] [RS_EM_00111]
[RS_Main_01026]	AUTOSAR shall support tracing and profiling on the target and onboard	[RS_EM_00152]



△

Requirement	Description	Satisfied by
[RS_SAF_10001]	AUTOSAR shall provide mechanisms to support safe initialization of software components.	[RS_EM_00009] [RS_EM_00151]
[RS_SAF_10008]	AUTOSAR shall provide mechanisms to support safe resource management for the AUTOSAR Adaptive Platform functional-clusters, applications and services and AUTOSAR Classic Platform basic software modules and software components.	[RS_EM_00005] [RS_EM_00008] [RS_EM_00009] [RS_EM_00050]
[RS_SAF_10028]	AUTOSAR shall provide mechanisms to support dependable scheduling of AUTOSAR Adaptive Platform functional-clusters, applications and services and AUTOSAR Classic Platform basic software modules and software components.	[RS_EM_00053] [RS_EM_00113]
[RS_SAF_10037]	AUTOSAR shall provide mechanisms to prevent unintended alteration of data.	[RS_EM_00002]

Table 5.1: RequirementsTracing

5.1 Not applicable requirements

[RS_EM_NA]{DRAFT} ⌈These requirements are not applicable as they are not within the scope of this release.⌋ *(RS_Main_01025, RS_Main_00650, RS_Main_00026, RS_Main_00030, RS_Main_00190, RS_Main_00230, RS_Main_00250, RS_Main_00260, RS_Main_00261, RS_Main_00270, RS_Main_00280, RS_Main_00285, RS_Main_00300, RS_Main_00301, RS_Main_00310, RS_Main_00350, RS_Main_00360, RS_Main_00410, RS_Main_00440, RS_Main_00445, RS_Main_00490, RS_Main_00491, RS_Main_00500, RS_Main_00503, RS_Main_00507, RS_Main_00510, RS_Main_00511, RS_Main_00512, RS_Main_00653, RS_Main_01001, RS_Main_01002, RS_Main_01003, RS_Main_01004, RS_Main_01005, RS_Main_01007, RS_Main_01008)*

6 References

[1] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[2] Glossary
AUTOSAR_FO_TR_Glossary

[3] Specification of Manifest
AUTOSAR_AP_TPS_ManifestSpecification

[4] Explanation of Safety Overview
AUTOSAR_FO_EXP_SafetyOverview

[5] Requirements of State Management
AUTOSAR_AP_RS_StateManagement

[6] Main Requirements
AUTOSAR_FO_RS_Main

7 History of Constraints and Specification Items

Please note that the lists in this chapter also include constraints and specification items that have been removed from the specification in a later version. These constraints and specification items do not appear as hyperlinks in the document.

7.1 Constraint and Specification Item History of this document according to AUTOSAR Release 17-03

7.1.1 Added Requirements in 17-03

Number	Heading
[RS_EM_00001]	The Execution Management shall load Executables
[RS_EM_00002]	The Execution Management shall set-up one process for the execution of each Executable instance
[RS_EM_00003]	The Execution Management shall support the checking of the integrity of Executables at startup of Executable
[RS_EM_00004]	The Execution Management shall support the authentication and authorization of Executables at startup of Executable
[RS_EM_00005]	The Execution Management shall support the configuration of OS resource budgets for Executable and groups of Executables
[RS_EM_00006]	The Execution Management shall support the analysis of available and required OS resource budgets for Executables and groups of Executables during installation and run-time
[RS_EM_00007]	The Execution Management shall support of the allocation of dedicated resources for the Executable (e.g GPU)
[RS_EM_00008]	The Execution Management shall support the binding of Executable threads to a specified set of processor cores.
[RS_EM_00009]	Only Execution Management shall start Executables
[RS_EM_00010]	The Execution Management shall support multiple instantiation of Executables
[RS_EM_00011]	Execution Management shall support self-initiated graceful shutdown of Executable instances
[RS_EM_00012]	Application Manifest shall support unambiguous identification of Executable instances
[RS_EM_00013]	Execution Management shall support configurable recovery actions
[RS_EM_00100]	The Execution Management shall support the ordered startup and shutdown of Executables
[RS_EM_00050]	The Execution Management shall do a system-wide coordination of activities.
[RS_EM_00051]	The Execution Management shall provide functions to the Executable for configuring external trigger conditions for its activities

△

Number	Heading
[RS_EM_00052]	The Execution Management shall provide functions to the Executable for configuring cyclic triggering of its activities
[RS_EM_00101]	The Execution Management shall provide Machine State Management functionality
[RS_EM_00103]	Execution Management shall support application lifecycle management

Table 7.1: Added Requirements in 17-03

7.1.2 Changed Requirements in 17-03

none

7.1.3 Deleted Requirements in 17-03

none

7.2 Constraint and Specification Item History of this document according to AUTOSAR Release 17-10

7.2.1 Added Requirements in 17-10

Number	Heading
[RS_EM_00053]	The Execution Management shall provide functions to support redundant execution of Executables
[RS_EM_00110]	Execution Management shall support diagnostic reset cause

Table 7.2: Added Requirements in 17-10

7.2.2 Changed Requirements in 17-10

none

7.2.3 Deleted Requirements in 17-10

Number	Heading
[RS_EM_00001]	The Execution Management shall load Executables
[RS_EM_00103]	Execution Management shall support application lifecycle management

Table 7.3: Deleted Requirements in 17-10

7.3 Constraint and Specification Item History of this document according to AUTOSAR Release 18-03

7.3.1 Added Requirements in 18-03

Number	Heading
[RS_EM_00151]	Execution Management shall be implemented at least according the highest safety integrity level from any process that is supported on the platform.

Table 7.4: Added Requirements in 18-03

7.3.2 Changed Requirements in 18-03

Number	Heading
[RS_EM_00009]	Only Execution Management shall start Executables
[RS_EM_00101]	The Execution Management shall provide Machine State Management functionality
[RS_EM_00103]	Execution Management shall support application lifecycle management
[RS_EM_00110]	Execution Management shall support diagnostic reset cause

Table 7.5: Changed Requirements in 18-03

7.3.3 Deleted Requirements in 18-03

Number	Heading
[RS_EM_00006]	The Execution Management shall support the analysis of available and required OS resource budgets for Executables and groups of Executables during installation and run-time

▽

△

Number	Heading
[RS_EM_00007]	The Execution Management shall support of the allocation of dedicated resources for the Executable (e.g GPU)
[RS_EM_00012]	Application Manifest shall support unambiguous identification of Executable instances

Table 7.6: Deleted Requirements in 18-03

7.4 Constraint and Specification Item History of this document according to AUTOSAR Release 18-10

7.4.1 Added Requirements in 18-10

Number	Heading
[RS_EM_00014]	Execution Management shall support a Trusted Platform
[RS_EM_00111]	Execution Management shall assist identification of Processes during Machine runtime

Table 7.7: Added Requirements in 18-10

7.4.2 Changed Requirements in 18-10

Number	Heading
[RS_EM_00002]	The Execution Management shall set-up one process for the execution of each Executable instance
[RS_EM_00005]	The Execution Management shall support the configuration of OS resource budgets for Executable and groups of Executables
[RS_EM_00011]	Execution Management shall support self-initiated graceful shutdown of Executable instances
[RS_EM_00013]	Execution Management shall support configurable recovery actions
[RS_EM_00101]	The Execution Management shall provide Machine State Management functionality

Table 7.8: Changed Requirements in 18-10

7.4.3 Deleted Requirements in 18-10

Number	Heading
[RS_EM_00003]	The Execution Management shall support the checking of the integrity of Executables at startup of Executable
[RS_EM_00004]	The Execution Management shall support the authentication and authorization of Executables at startup of Executable

Table 7.9: Deleted Requirements in 18-10

7.5 Constraint and Specification Item History of this document according to AUTOSAR Release 19-03

7.5.1 Added Requirements in 19-03

none

7.5.2 Changed Requirements in 19-03

Number	Heading
[RS_EM_00008]	The Execution Management shall support the binding of Executable threads to a specified set of processor cores.

Table 7.10: Changed Requirements in 19-03

7.5.3 Deleted Requirements in 19-03

none

7.6 Constraint and Specification Item History of this document according to AUTOSAR Release R19-11

7.6.1 Added Requirements in 19-11

none

7.6.2 Changed Requirements in 19-11

Number	Heading
[RS_EM_00009]	Only Execution Management shall start Executables

Table 7.11: Changed Requirements in 19-11

7.6.3 Deleted Requirements in 19-11

none

7.7 Constraint and Specification Item History of this document according to AUTOSAR Release R20-11

7.7.1 Added Requirements in R20-11

Number	Heading
[RS_EM_00113]	Execution Management shall support time-triggered execution
[RS_EM_00150]	Error Handling

Table 7.12: Added Requirements in R20-11

7.7.2 Changed Requirements in R20-11

none

7.7.3 Deleted Requirements in R20-11

Number	Heading
[RS_EM_00013]	Execution Management shall support configurable recovery actions

Table 7.13: Deleted Requirements in R20-11

7.8 Constraint and Specification Item History of this document according to AUTOSAR Release R21-11

7.8.1 Added Requirements in R21-11

Number	Heading
[RS_EM_00015]	Execution Management shall support integrity and authenticity monitoring

Table 7.14: Added Requirements in R21-11

7.8.2 Changed Requirements in R21-11

none

7.8.3 Deleted Requirements in R21-11

none

7.9 Constraint and Specification Item History of this document according to AUTOSAR Release R22-11

7.9.1 Added Requirements in R22-11

Number	Heading
[RS_EM_00151]	Execution Management shall be implemented at least according the highest safety integrity level from any process that is supported on the platform.

Table 7.15: Added Requirements in R22-11

7.9.2 Changed Requirements in R22-11

Number	Heading
[RS_EM_00002]	`Execution Management` shall set-up one process for the execution of each `Modelled Process`.
[RS_EM_00005]	`Execution Management` shall support the configuration of OS resource budgets for `process` and groups of `processes`.

▽

△

Number	Heading
[RS_EM_00008]	Execution Management shall support the binding of all threads of a given process to a specified set of processor cores.
[RS_EM_00009]	Execution Management shall ensure it is the sole entity starting processes.
[RS_EM_00050]	Execution Management shall perform Machine-wide coordination of processes.
[RS_EM_00053]	Execution Management shall provide APIs to the process to support deterministic redundant execution of processes.
[RS_EM_00103]	Execution Management shall support process lifecycle management.
[RS_EM_00113]	Execution Management shall support time-triggered execution.
[RS_EM_NA]	

Table 7.16: Changed Requirements in R22-11

7.9.3 Deleted Requirements in R22-11

none

7.10 Constraint and Specification Item History of this document according to AUTOSAR Release R23-11

7.10.1 Added Requirements in R23-11

Number	Heading
[RS_EM_00152]	Execution Management shall support standardized trace points throughout the state transitions.

Table 7.17: Added Requirements in R23-11

7.10.2 Changed Requirements in R23-11

Number	Heading
[RS_EM_00008]	Execution Management shall support the binding of all threads of a given process to a specified set of processor cores.
[RS_EM_00009]	Execution Management shall control the right to create child processes for each process it starts.

▽

△

Number	Heading
[RS_EM_00053]	`Execution Management` shall provide APIs to the `process` to support deterministic redundant execution of `processes`.
[RS_EM_00151]	Execution Management shall be implemented at least according the highest safety integrity level from any process that is supported on the platform.

Table 7.18: Changed Requirements in R23-11

7.10.3 Deleted Requirements in R23-11

none