

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

Document Status	Final
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	17-10

Document Change History			
Date	Release	Changed by	Description
2017-10-27	17-10	AUTOSAR Release Management	<ul style="list-style-type: none"> Minor changes, document clean up
2017-03-31	17-03	AUTOSAR Release Management	<ul style="list-style-type: none"> Initial release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1	Scope of this document	4
1.1	Document Conventions	5
2	Acronyms and abbreviations	6
3	Requirements Tracing	7
4	Functional overview	9
5	Requirements specification	10
5.1	Startup and Shutdown of Applications	10
5.2	Execution	14
5.3	State Management	15
5.4	Support for Diagnostics	16
6	References	17

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.

1.1 Document Conventions

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

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

2 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 is a loaded instance of an Executable to be executed on a Machine.
Application Dependency	Dependencies between Executable instances can be configured to define a sequence for starting and terminating them.
Execution Management	The element of the Adaptive Platform responsible for the ordered startup and shutdown of the Adaptive Platform and the Applications.
State Manager	The element of the Execution Management defining modes of operation for Adaptive Platform. It allows flexible definition of functions which are active on the platform at any given time. Architecture and functionality of State Management are still under discussion. A consolidated description will follow in a later release.
Machine State	The element of the State Management which characterize the current status of the machine. It defines a set of active Applications for any certain situation. The set of Machine States is machine specific and it will be deployed in the Machine Manifest. Machine States are mainly used to control machine lifecycle (startup/shut-down/restart) and platform-level processes.
Function Group State	The element of the State Management which characterize the current status of functionally coherent user-level Applications. The set of Function Groups and their Function Group States is machine specific and it will be deployed in the Machine Manifest.
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.

Table 2.1: Technical Terms

3 Requirements Tracing

The following tables reference the requirements specified in [4] 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.

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_00100]
[RS_Main_00010]	AUTOSAR shall support the development of safety related systems.	[RS_EM_00002] [RS_EM_00004] [RS_EM_00005] [RS_EM_00006] [RS_EM_00007] [RS_EM_00008] [RS_EM_00009] [RS_EM_00013] [RS_EM_00053]
[RS_Main_00011]	AUTOSAR shall support the development of reliable systems	[RS_EM_00006] [RS_EM_00009] [RS_EM_00012]
[RS_Main_00012]	AUTOSAR shall provide a software platform to support the development of highly available systems.	[RS_EM_00013]
[RS_Main_00049]	AUTOSAR shall provide an Execution Management for running multiple applications	[RS_EM_00002] [RS_EM_00003] [RS_EM_00009] [RS_EM_00010] [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]	AUTOSAR shall provide a standardized software interface for communication between Applications	[RS_EM_00051]
[RS_Main_00080]	AUTOSAR shall provide means to describe a component model for Application Software	[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_00006] [RS_EM_00007] [RS_EM_00008] [RS_EM_00010] [RS_EM_00012] [RS_EM_00103]
[RS_Main_00170]	AUTOSAR shall provide secure access to ECU	[RS_EM_00003] [RS_EM_00004]
[RS_Main_00200]	AUTOSAR specifications shall allow resource efficient implementations	[RS_EM_00010] [RS_EM_00011] [RS_EM_00052] [RS_EM_00101] [RS_EM_00103]
[RS_Main_00260]	AUTOSAR shall provide diagnostics means during runtime, for production and services purposes	[RS_EM_00110]

Requirement	Description	Satisfied by
[RS_Main_00320]	AUTOSAR shall provide formats to specify all aspects necessary to integrate Application Software on an ECU	[RS_EM_00002] [RS_EM_00007] [RS_EM_00008]
[RS_Main_00330]	AUTOSAR shall support the principle of information hiding	[RS_EM_00002] [RS_EM_00009]
[RS_Main_00340]	AUTOSAR shall support the observance of timing requirements	[RS_EM_00005] [RS_EM_00006] [RS_EM_00052] [RS_EM_00100]
[RS_Main_00460]	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	[RS_EM_00012] [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_Main_00514]	AUTOSAR shall support the development of secure systems	[RS_EM_00003] [RS_EM_00004]

4 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.

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

- Machine startup, and Machine shutdown.
- 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.
- The Execution Management is the initial (“boot”) process of the operating system and is responsible for Machine startup.
 - bounded time and time variance of the boot process
- The Execution Management enforces process isolation with each instance of an Executable managed as a single process.
- Startup and shutdown of Applications, including platform-level Applications and Adaptive Applications.
 - Loading Executable based on a defined precedence
 - 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
- Machine state management
 - conditions for the execution of Applications

5 Requirements specification

This chapter describes all requirements driving the work to define the execution manager's functionality.

5.1 Startup and Shutdown of Applications

[RS_EM_00002] The Execution Management shall set-up one process for the execution of each Executable instance [

Type:	draft
Description:	For each instance of an Executable, the Execution Management shall allocate one POSIX process. Furthermore process specific properties (like priority, scheduling policy and access rights) shall be assigned based on the Application 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_00330](#))

[RS_EM_00003] The Execution Management shall support the checking of the integrity of Executables at startup of Executable. [

Type:	draft
Description:	Before executing the Executable the Execution Management shall check whether the Executable are corrupted, were accidentally changed or intentionally tampered.
Rationale:	Executable of an Application could get changed after installation.
Dependencies:	–
Use Case:	Security
Supporting Material:	Note: It is still to be decided if external stored data (outside the Executable of the Application) is to be used by the Executable.

]([RS_Main_00049](#), [RS_Main_00170](#), [RS_Main_00514](#))

[RS_EM_00004] The Execution Management shall support the authentication and authorization of Executables at startup of Executable [

Type:	draft
Description:	Before executing the Executable the Execution Management shall validate the authenticity of the Executable and check whether the Executable is given access and user rights to required resources.
Rationale:	Different access rights for different Executables.
Dependencies:	–
Use Case:	Security
Supporting Material:	–

]([RS_Main_00010](#), [RS_Main_00170](#), [RS_Main_00514](#))

[RS_EM_00005] The Execution Management shall support the configuration of OS resource budgets for Executable and groups of Executables [

Type:	draft
Description:	Based on the Application Manifest the Execution Management shall allocate OS resources to the Executable. The allocation shall be possible for single Executable and groups of Executables.
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_EM_00006] The Execution Management shall support the supervision of available and required OS resource budgets for Executables and groups of Executables during installation [

Type:	draft
Description:	Based on the configuration of OS resource budgets the Execution Management shall observe violations of assigned OS resources used by the Executable.
Rationale:	Real-time guarantees shall be kept.
Dependencies:	–
Use Case:	–
Supporting Material:	

]([RS_Main_00010](#), [RS_Main_00011](#), [RS_Main_00106](#), [RS_Main_00340](#))

[RS_EM_00007] The Execution Management shall support allocation of dedicated resources for the Executable (e.g GPU) [

Type:	draft
Description:	Not applicable for this release.
Rationale:	–
Dependencies:	–
Use Case:	–
Supporting Material:	–

]([RS_Main_00010](#), [RS_Main_00106](#), [RS_Main_00320](#))

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

Type:	draft
Description:	The Execution Management shall allow to bind threads to specific set of processor cores based on configuration in the Application Manifest.
Rationale:	Mechanism to influence load balancing, reaction times, and latencies.

Dependencies:	–
Use Case:	Assign two parallel threads to two processor cores to achieve true parallelism.
Supporting Material:	–

]([RS_Main_00010](#), [RS_Main_00050](#), [RS_Main_00106](#), [RS_Main_00320](#),
[RS_Main_00501](#))

[RS_EM_00009] Only Execution Management shall start Executables [

Type:	draft
Description:	Only Execution Management shall start Executable instances. The Execution Management shall prevent Executables from directly starting other Executables .
Rationale:	Execution Management needs full control of starting applications to ensure required isolation of temporal and spatial properties.
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_00330](#))

[RS_EM_00010] The Execution Management shall support multiple instances of Executables [

Type:	draft
Description:	It shall be possible to start more than one process from a single Application Executable .
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_00200](#),
[RS_Main_00501](#))

[RS_EM_00011] Execution Management shall support self-initiated graceful shutdown of Executable instances [

Type:	draft
Description:	When an Executable instance finished its functionality, the Execution Management shall support self-initiated graceful shutdown of Executable instances. This contains freeing of 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 . Executable instance shutdown shall only be initiated by the Executable itself or by Execution Management .
Rationale:	The process of an Executable instance is finished and shuts down itself.
Dependencies:	–
Use Case:	–

Supporting Material:	–
-----------------------------	---

]([RS_Main_00200](#))

[RS_EM_00012] Application Manifests shall support unique identification of Executable instances [

Type:	draft
Description:	The Application Manifest shall include a Machine-wide unique identifier and version information for each Executable instance (i.e. process). The Execution Management shall consider this identifier and version information to determine the dependencies between instances of Executables.
Dependencies:	–
Use Case:	–
Supporting Material:	–

]([RS_Main_00011](#), [RS_Main_00106](#), [RS_Main_00460](#))

[RS_EM_00013] Execution Management shall support configurable recovery actions [

Type:	draft
Description:	The Execution Management shall support recovery actions (e.g. Application, Executable or Machine restart, degradation) in case an Executable deviates from normal behavior. The recovery action shall be configurable in the Application Manifest.
Dependencies:	–
Use Case:	–
Supporting Material:	–

]([RS_Main_00010](#), [RS_Main_00012](#))

[RS_EM_00100] The Execution Management shall support the ordered startup and shutdown of Executables [

Type:	draft
Description:	The 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 Application 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](#))

5.2 Execution

[RS_EM_00050] The Execution Management shall do a system-wide coordination of activities [

Type:	draft
Description:	The Execution Management shall provide an API for an Executable 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 Executables must be managed in the system to ensure that activities can be coordinated across Executables . Registration enables the execution manager to form the necessary system-wide view for the coordination.
Supporting Material:	–

] ([RS_Main_00460](#))

[RS_EM_00051] The Execution Management shall provide functions to the Executable for configuring external trigger conditions for its activities [

Type:	draft
Description:	The Execution Management shall provide an API for configuring the trigger conditions of registered activities.
Rationale:	The Execution Management must 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] The Execution Management shall provide functions to the Executable for configuring cyclic triggering of its activities [

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

] ([RS_Main_00050](#), [RS_Main_00200](#), [RS_Main_00340](#))

[RS_EM_00053] The Execution Management shall provide functions to support redundant execution of Executables [

Type:	draft
--------------	-------

Description:	The Execution Management shall provide APIs to support deterministic redundant execution of high performance Executables
Rationale:	High ASIL systems require safety mechanism like software lockstep to be implemented on non-automotive grade microprocessors. The redundant execution must 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](#))

5.3 State Management

[RS_EM_00101] The Execution Management shall provide State Management functionality [

Type:	draft
Description:	The Execution Management starts and stops Applications in response to changes in the Machine State or Function Group State . It must therefore include an interface to manage Machine State and Function Group State changes.
Rationale:	Ensure that only the required set of Applications (and hereby Executable instances) is running in any given operation conditions and therefore platform resources are saved for relevant Executables .
Dependencies:	–
Use Case:	Provide mechanism to define modes of operation of the Machine .
Supporting Material:	–

]([RS_Main_00200](#), [RS_Main_00460](#))

[RS_EM_00103] Execution Management shall support application lifecycle management [

Type:	draft
Description:	Execution Management shall support Application lifecycle management to control their startup and shutdown.
Rationale:	Ensure that startup and shutdown of Applications can be coordinated between groups of Applications which shall run in the same Machine State or Function Group State .
Dependencies:	–
Use Case:	Executable processes need to be fully up and running before other Executable processes which depend on their functionality can be started.
Supporting Material:	–

]([RS_Main_00049](#), [RS_Main_00050](#), [RS_Main_00106](#), [RS_Main_00200](#), [RS_Main_00460](#))

5.4 Support for Diagnostics

[RS_EM_00110] Execution Management shall support diagnostic reset cause [

Type:	draft
Description:	Execution Management shall support diagnostics in term of last reset cause.
Rationale:	(Diagnostic) Applications have to get the last reset cause.
Dependencies:	–
Use Case:	–
Supporting Material:	–

]([RS_Main_00260](#))

6 References

- [1] Standardization Template
AUTOSAR_TPS_StandardizationTemplate
- [2] Glossary
AUTOSAR_TR_Glossary
- [3] Specification of Manifest
AUTOSAR_TPS_ManifestSpecification
- [4] Main Requirements
AUTOSAR_RS_Main