Document Title	Requirements on Platform Health Management for Adaptive Platform
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	852

Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R19-11

Document Change History

Date	Release	Changed by	Description
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2019-03-29	19-03	AUTOSAR Release Management	• removed references to RS_Main_00330
2018-10-31	18-10	AUTOSAR Release Management	• minor corrections / clarifications / editorial changes
2018-03-29	18-03	AUTOSAR Release Management	• 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 Document

This document specifies requirements on Platform Health Management. Platform Health Management implements the Platform Health Monitoring on the AUTOSAR Adaptive Platform.

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.

3 Acronyms and abbreviations

The glossary below includes acronyms and abbreviations relevant to the specification or implementation of Health Monitoring that are not included in the [2, AUTOSAR glossary].

Abbreviation:	Description:
CM	AUTOSAR Adaptive Communication Management
DM	AUTOSAR Adaptive Diagnostic Management
PHM	Platform Health Management
SE	Supervised Entity

Acronym:	Description:
Alive Supervision	Mechanism to check the timing constraints of cyclic Supervised Entityes to be within the configured min and max limits.
Application	see [2] AUTOSAR Glossary
ara::com	Communication middleware for the AUTOSAR Adaptive Platform
AUTOSAR Adaptive Platform	see [2] AUTOSAR Glossary
Checkpoint	A point in the control flow of a Supervised Entity where the activity is reported.
Daisy chaining	Chaining multiple instances of Health Monitoring

Deadline End Checkpoint	A Checkpoint for which Deadline Supervision is configured and which is a ending point for a particular Transition. It is possible that a Checkpoint is both a Deadline Start Checkpoint and Deadline End Checkpoint - if Deadline Supervision is chained.
Deadline Start Checkpoint	A Checkpoint for which Deadline Supervision is configured and which is a starting point for a particular Transition.
Deadline Supervision	Mechanism to check that the timing constraints for execution of the transition from a Deadline Start Checkpoint to a corresponding Deadline End Checkpoint are within the configured min and max limits.
Executable	see [2] AUTOSAR Glossary
Execution Management	The element of the Adaptive Platform responsible for the ordered startup and shutdown of the Adaptive Platform and the Application.
Function Group	A Function Group is a set of coherent Processes, which need to be controlled consistently. Depending on the state of the Function Group, Processes are started or terminated.
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 deployed as part of the Machine Manifest.
Functional Cluster	see [2] AUTOSAR Glossary
Global Supervision Status	Status that summarizes the Local Supervision Status of all Supervised Entities of a software subsystem.
Health Channel	Channel providing information about the health status of a (sub)system. This might be the Global Supervision Status of an application, the result any test routine or the status reported by a (sub)system (e.g. voltage monitoring, OS kernel, ECU status, ...).
Health Monitoring	Supervision of the software behaviour for correct timing and sequence.
Health Status	A set of states that are relevant to the supervised software (e.g. the Global Supervision Status of an application, a Voltage State, an application state, the result of a RAM monitoring algorithm).
Logical Supervision	Kind of online supervision of software that checks if the software (Supervised Entity or set of Supervised Entities) is executed in the sequence defined by the programmer (by the developed code).

Machine Manifest	Manifest file to configure a Machine.
Machine	see [2] AUTOSAR Glossary
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.
Manifest	see [2] AUTOSAR Glossary
Platform Health Management	Health Monitoring for the Adaptive Platform
Process	A process is a loaded instance of an Executable to be executed on a Machine.
State Management	The element of the Execution Management defining modes of operation for AUTOSAR Adaptive Platform. It allows flexible definition of functions which are active on the platform at any given time.
Supervised Entity	A software entity which is included in the supervision. A Supervised Entity denotes a collection of Checkpoints within a software component. There may be zero, one or more Supervised Entities in a Software Component. A Supervised Entity may be instantiated multiple times, in which case each instance is independently supervised.
Supervision Mode	An overall state of a microcontroller or virtual machine. Modes are mutually exclusive and all Supervised Entities are in the same Supervision Mode. A mode can be e.g. Startup, Shutdown, Low power.

Table 3.1: Acronyms

4 Requirements Specification

This chapter describes all requirements driving the work to define the Platform Health Management.

4.1 Functional Overview

See RS Health Monitoring [3] for the overview of the functionality.

This document specifies the requirements regarding the realization of the Health Monitoring on Adaptive Platform. This includes:

- Standardized interfaces
- Mapping of abstract functionalities/concepts defined in Foundation to entities in Adaptive Platform.

4.2 Constraints and assumptions

4.2.1 Limitations

No known limitation.

4.2.2 Applicability to car domains

No restriction.

4.3 Functional Requirements

4.3.1 Supervision functions

[RS_PHM_00101]{DRAFT} **Platform Health Management shall provide a standardized C++ interface for the reporting of Checkpoints.** ⌈

Type:	draft
Description:	Platform Health Management shall provide a standardized C++ interface for the reporting of Checkpoints.
Rationale:	Checkpoints are locations inside the code of Supervised Entitys. Platform Health Management checks that these locations are reached in correct time and order. Therefore Platform Health Management needs to be informed when a Checkpoint is reached.
Dependencies:	
Use Case:	Reporting of reached code locations for Alive Supervision, Deadline Supervision and Logical Supervision.
Supporting Material:	

⌋*(RS_Main_00011, RS_HM_09254)*

[RS_PHM_00102]{DRAFT} **Platform Health Management shall provide a standardized C++ interface for the reporting of Health Channel.** ⌈

Type:	draft
Description:	Platform Health Management shall provide a standardized C++ interface for the reporting of Health Channel.
Rationale:	A Health Channel is a channel for passing external supervision results (e.g. from RAM test, voltage monitoring, ...) to Platform Health Management. Therefore Platform Health Management needs to be informed the status of Health Channels.
Dependencies:	
Use Case:	Reporting of Global Supervision Status, results of test routines or status of (sub)systems (e.g. voltage monitoring, OS kernel, ECU status).
Supporting Material:	

⌋*(RS_Main_00011, RS_HM_09257)*

[RS_PHM_00103]{DRAFT} **Platform Health Management functionality shall be available within the same process and as a separate one.** ⌈

Type:	draft
Description:	PHM functionality shall be able to be available, with respect to the monitored process, as: • library component executed in the context of the monitored process • a separate process in the same OS or in the same machine
Rationale:	Provide optimized functionallity for process local usage
Dependencies:	
Use Case:	Local monitoring is necessary within the same process for efficiency reasons. Monitoring is also needed in another process for achieving independence. This means the reporting of checkpoints or reporting of health channel status do not cross the boundaries of the OS/VM.
Supporting Material:	

⌋*(RS_Main_00410)*

4.3.2 Mapping of Supervised Entitys to threads and processes

[RS_PHM_00104]{DRAFT} **Platform Health Management shall realize the Supervision Mode as a tuple of Execution Management states.** ⌈

Type:	draft
Description:	Platform Health Management shall realize the Supervision Mode as a tuple <machine state, function group state, application state>.
Rationale:	There is no need to specify the abstract Supervision Mode in the configuration or in the standardized interface. Supervision Mode is an abstract concept and it is realized by those three states, so they need to be used.
Dependencies:	
Use Case:	Depending on those three states, the behavior of processes is different, so the supervision functions need to perform differently.
Supporting Material:	

⌋*(RS_Main_00049, RS_HM_09253)*

[RS_PHM_00105]{DRAFT} **Platform Health Management shall support different allocations/distributions of a Supervised Entity through threads and processes.** ⌈

Type:	draft
Description:	Platform Health Management shall support the following Supervised Entities: • A Supervised Entity belonging to one thread • A Supervised Entity spread across several threads of the same process • A Supervised Entity spread accross different processes
Rationale:	Algorithms can be executed in one thread, multiple threads or processes. It must be possible to supervise a whole algorithm.
Dependencies:	
Use Case:	Supervision of the global flow of algorithms distributed to multiple threads or processes.
Supporting Material:	

⌋*(RS_Main_00410, RS_Main_00460, RS_HM_09242)*

[RS_PHM_00106]{DRAFT} **Platform Health Management shall support allocating of multiple Supervised Entitys to the same process or thread.** ⌈

Type:	draft
Description:	Platform Health Management shall support allocating of multiple Supervised Entitys to the same process or thread

▽

△

Rationale:	It shall be possible to define separate Supervised Entitys for different supervision functionalities or for subfunctions within the same process or thread
Dependencies:	
Use Case:	Separate Supervised Entitys for Alive Supervision and Logical Supervision of the same thread.
Supporting Material:	

⌋*(RS_Main_00501, RS_Main_00460)*

[RS_PHM_00107]{DRAFT} **Platform Health Management shall support multiple instantiation.** ⌈

Type:	draft
Description:	Platform Health Management shall support: • multiple instantiation of the same executable (resulting with several processes) • multiple instantiation of threads (performing the same action) in an executable • static and dynamic libraries executed in different context • services/servers that can be concurrently invoked by different clients.
Rationale:	The Health Status shall be collected and passed between multiple instances by daisy chaining.
Dependencies:	
Use Case:	Collect and validate the Health Status reported by the instance(s) on one or multiple microcontroller(s)/cores by another instance running on a separate controller for safety supervisions.
Supporting Material:	

⌋*(RS_Main_00460, RS_HM_09240)*

4.3.3 Daisy chaining

[RS_PHM_00108]{DRAFT} **Platform Health Management shall provide a standardized interface between Platform Health Management components used in a daisy chain.** ⌈

Type:	draft
Description:	Platform Health Management shall provide a standardized interface between Platform Health Management components used in a daisy chain.
Rationale:	Provide the possibility to use the output of one PHM instance as input to another PHM instance
Dependencies:	
Use Case:	The components are possibly provided by different vendors, working on different microcontrollers or virtual machines. On each controller or (virtual) machine a separate instance of Platform Health Management might be used and it should be possible to operate these instances in a daisy chain.
Supporting Material:	

⌋*(RS_Main_00511, RS_Main_00190)*

[RS_PHM_00109]{DRAFT} **Platform Health Management shall provide the Daisy chaining interface over ara::com.** ⌈

Type:	draft
Description:	Platform Health Management shall provide the Daisy chaining interface over at least ara::com.
Rationale:	PHM instance shall be able to communicate across microcontrollers or virtual machines
Dependencies:	
Use Case:	The Platform Health Management is possibly provided by different vendors, working on different microcontrollers or virtual machines. On each controller or (virtual) machine a separate instance of Platform Health Management might be used and it should be possible to operate these instances in a daisy chain. Note: Providing the ara::com is mandatory for each implementation Platform Health Management, but it is also possible to add more efficient implementations locally.
Supporting Material:	

⌋*(RS_Main_00511, RS_Main_00190)*

4.3.4 Interaction with other functional clusters

[RS_PHM_00110]{DRAFT} **Platform Health Management shall invoke the interfaces of Execution Management and State Management.** ⌈

Type:	draft
Description:	Platform Health Management shall invoke the interfaces of Execution Management and State Management in order to request error reactions: • request to restart a process • request to go to another function group state • request to enter a safe state.
Rationale:	Error reactions on failures detected by Platform Health Management might change the state of an application or require the restart of a process.
Dependencies:	
Use Case:	Restart of a process after Platform Health Management detected an unrecoverable error.
Supporting Material:	

⌋*(RS_Main_00011, RS_Main_00049, RS_HM_09251)*

4.4 Non-Functional Requirements (Qualities)

[RS_PHM_00001]{DRAFT} **The Platform Health Management shall provide a standardized header file structure for each service.** ⌈

Type:	draft
Description:	The Platform Health Management shall provide a standardized header file structure for each service. The application uses the standardized header files which are independent of the underlying implementation.
Rationale:	The application code shall be reusable for different AUTOSAR Adaptive platform implementations.
Dependencies:	–
Use Case:	The application developers implement their code against the standardized header files.
Supporting Material:	–

⌋*(RS_Main_00060)*

[RS_PHM_00002]{DRAFT} **The service header files shall define the namespace for the respective service.** ⌈

Type:	draft
Description:	The service header files shall define the namespace for the respective service to uniquely identify each service instance.

▽

△

Rationale:	The application code shall be reusable for different AUTOSAR Adaptive platform implementations and for different vehicle lines.
Dependencies:	–
Use Case:	To avoid conflicts with other applications and other services, each service shall have its own namespace.
Supporting Material:	–

⌋*(RS_Main_00060)*

[RS_PHM_00003]{DRAFT} **The Platform Health Management shall define how language specific data types are derived from modeled data types.** ⌈

Type:	draft
Description:	The Platform Health Management shall define how language specific data types, e.g. C++ data types, are derived from modeled data types.
Rationale:	The Platform Health Management shall support different language bindings.
Dependencies:	–
Use Case:	The Health Management supports C++ language binding and therefore has to define the modeled data types in C++.
Supporting Material:	–

⌋*(RS_Main_00060)*

5 Requirements Tracing

The following table references the requirements specified in [3] and links to the fulfillments of these.

Feature	Description	Satisfied by
[RS_HM_09240]	Health Monitoring shall support multiple occurrences of the same Supervised Entity.	[RS_PHM_00107]
[RS_HM_09242]	Health Monitoring shall support the supervision within and across Supervised Entities.	[RS_PHM_00105]
[RS_HM_09251]	Health Monitoring shall be able to request a restart a Supervised entity.	[RS_PHM_00110]
[RS_HM_09253]	Health Monitoring shall support mode-dependent behavior of Supervised Entities and it shall support the supervision on the transitions between Checkpoints belonging different Supervision Modes.	[RS_PHM_00104]
[RS_HM_09254]	Health Monitoring shall provide an interface to Supervised Entities to report the currently reached Checkpoint.	[RS_PHM_00101]
[RS_HM_09257]	Health Monitoring shall provide an interface to Supervised Entities to report their health status.	[RS_PHM_00102]
[RS_Main_00011]	AUTOSAR shall support the development of reliable systems	[RS_PHM_00101] [RS_PHM_00102] [RS_PHM_00110]
[RS_Main_00049]	AUTOSAR shall provide an Execution Management for running multiple applications	[RS_PHM_00104] [RS_PHM_00110]
[RS_Main_00060]	AUTOSAR shall provide a standardized software interface for communication between Applications	[RS_PHM_00001] [RS_PHM_00002] [RS_PHM_00003]
[RS_Main_00190]	AUTOSAR shall support standardized interoperability with non-AUTOSAR software	[RS_PHM_00108] [RS_PHM_00109]
[RS_Main_00410]	AUTOSAR shall provide specifications for routines commonly used by Application Software to support sharing and optimization	[RS_PHM_00103] [RS_PHM_00105]
[RS_Main_00460]	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	[RS_PHM_00105] [RS_PHM_00106] [RS_PHM_00107]
[RS_Main_00501]	AUTOSAR shall support redundancy concepts	[RS_PHM_00106]
[RS_Main_00511]	AUTOSAR shall support virtualization	[RS_PHM_00108] [RS_PHM_00109]

6 References

[1] Standardization Template
AUTOSAR_TPS_StandardizationTemplate

[2] Glossary
AUTOSAR_TR_Glossary

[3] Requirements on Health Monitoring

AUTOSAR_RS_HealthMonitoring