Document Title	Adaptive Platform Release Overview
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	782

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	Release Life Cycle Status: R19-11 is in Evolution, R19-11 supersedes R19-03

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

References

[1] Explanation of ara::com API
AUTOSAR_EXP_ARAComAPI

[2] Specification of Communication Management
AUTOSAR_SWS_CommunicationManagement

[3] SOME/IP Protocol Specification
AUTOSAR_PRS_SOMEIPProtocol

[4] Specification of Manifest
AUTOSAR_TPS_ManifestSpecification

[5] Requirements on Time Synchronization for Adaptive Platform
AUTOSAR_RS_TimeSync

[6] Specification of Crypto Interface
AUTOSAR_SWS_CryptoInterface

[7] Specification of RESTful communication
AUTOSAR_SWS_REST

1 Introduction

1.1 Scope of this document

This document provides an overview of the AUTOSAR standard Adaptive Platform release R19-11.

1.2 AUTOSAR standards

1.2.1 Introduction

AUTOSAR addresses a wide range of use cases in automotive software development with its standards. These use cases have different requirements and lead to different technical solutions.

Packaging its deliverables into different "standards"

- eases the access to AUTOSAR solutions for users and
- allows AUTOSAR to scale with market needs.

1.2.2 Definition

An AUTOSAR standard is a consistent set of AUTOSAR deliverables, which are released at the same time. AUTOSAR deliverables can, but are not limited to be of the following kinds:

- textual explanations
- textual specifications
- test specifications
- source code
- other formal or semi-formal textual formats (e.g. ARXML, UML models, XML Schemata)

At the time of release, AUTOSAR ensures that dependencies are fulfilled.

1.2.3 Overview on AUTOSAR's standards

AUTOSAR delivers the following standards:

Standard	Abbreviation
Adaptive Platform	AP
Classic Platform	CP
Foundation	FO

1.2.3.1 Adaptive Platform

The Adaptive Platform is AUTOSAR's solution for high-performance computing ECUs to build safety-related systems for use cases such as highly automated and autonomous driving.

1.2.3.2 Classic Platform

The Classic Platform is AUTOSAR's solution for embedded systems with hard real-time and safety constraints.

1.2.3.3 Foundation

The purpose of the Foundation standard is to enforce interoperability between the AUTOSAR platforms.

Foundation contains common requirements and technical specifications (e.g. protocols) shared between the AUTOSAR platforms.

1.2.4 Dependencies between Standards

Each release of Classic and Adaptive Platform relies on a dedicated version of Foundation. The specific dependency is documented in chapter 1.3.6.

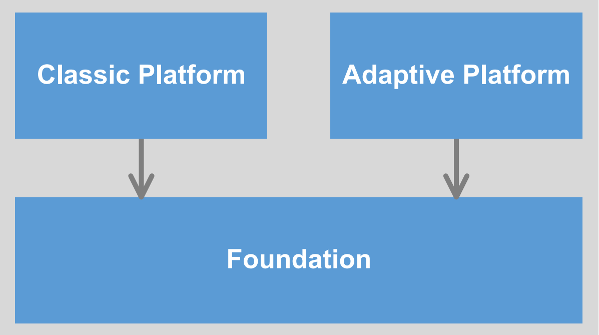


Figure 1.1: Dependencies of AUTOSAR Standards

1.2.5 Dependencies to other standards

This release of the Adaptive Platform depends on the standard Foundation in release R19-11, which

- defines protocols implemented by Adaptive Platform
- contains the project objectives and the common requirements from which the features of the Adaptive Platform are derived
- contains common specification parts which apply to both, the Adaptive Platform and the Classic Platform

These dependencies are refined in the trace information of the requirements in the respective specifications.

1.3 Release Numbering and Life Cycle

1.3.1 Platform release number

AUTOSAR applies a four-digit numbering scheme Ryy-mm to identify releases. The identifiers “yy” and “mm” depict the year and month of the release date, e.g. R19-11 for the November 2019 release.

1.3.2 Internal release number

AUTOSAR additionally maintains an internal release number for different purposes (e.g. usage in BSW modules in Classic Platform).
The internal release number is used for all platforms and follows up on the Classic Platform release number. In Adaptive Platform this is newly introduced. In Foundation this leads to a discontinuation of the former numbering pattern (e.g. R1.5.0).
A mapping list between Platform Releases and corresponding internal release numbers can be found in chapter 1.3.5. The internal release number uses a three-digit numbering scheme R<major>.<minor>.<revision> to identify releases. Its primary purpose is to identify a release as

- a major release: Valid and draft specification parts may be changed backward incompatibly.
- a minor release: Valid specification parts may only be changed backward compatibly. Draft specification parts may be changed backward incompatibly.
- a revision: Does not contain extensions but only backward compatible bugfixes.

1.3.3 Release life cycle of a major release

Each major release goes through four consecutive steps within its life cycle (examples based on the internal release numbering scheme):

1. Development: Between start of life cycle and the initial release (e.g. R4.0.1)
2. Evolution: Following the initial release with zero, one or several minor releases and/or revisions (e.g. R4.0.2, R4.1.1)
3. Maintenance: No new content is added to a major release but only maintenance of the existing content with zero, one or several revisions (e.g. R3.2.2) is provided
4. Issue Notice: No more revisions but zero, one or several issue notices, i.e. updates of the list of known issues until end of life cycle.

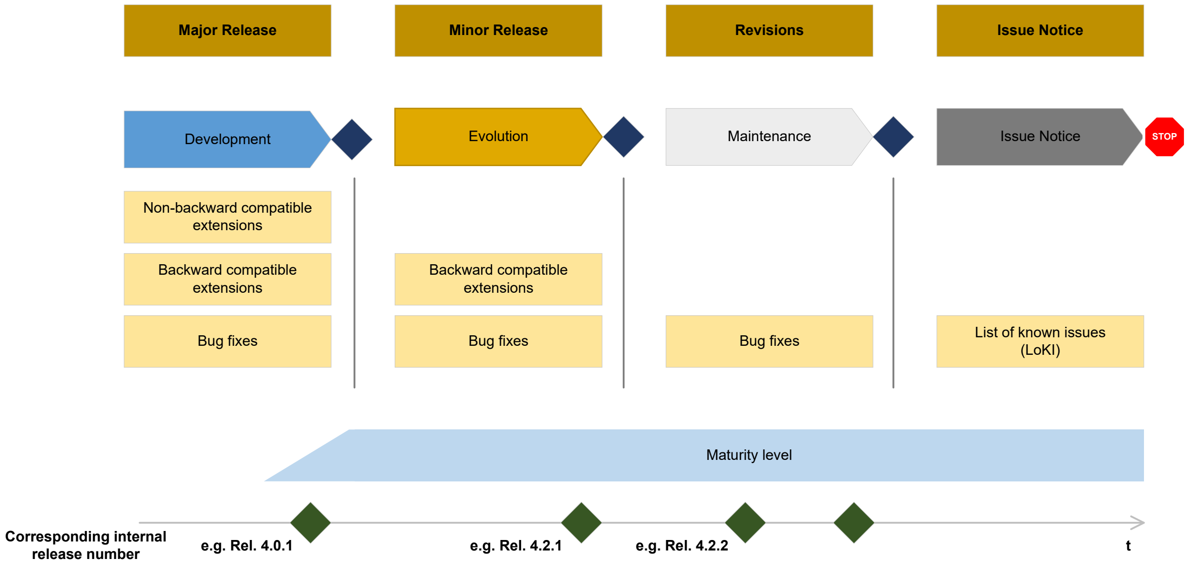


Figure 1.2: Life cycle model of AUTOSAR standards

1.3.4 Life cycle states of specification items and requirements

The life cycle state of a specification item is found after the specification item ID surrounded by curly brackets. The states are:

- {Valid}: This indicates that the related entity is a valid part of the document. This is the default and also applies if no dedicated life cycle status is annotated for the related entity.
- {Draft}: This indicates that the related entity is newly introduced but still experimental. This information is published but is subject to change without backward compatibility guarantee.

- {Obsolete}: This indicates that the related entity is subject to be removed in one of the following releases without further notice.

The life cycle state of a requirement is found in the attribute "type". The states are the same as the specification item states.

1.3.5 Overview of AUTOSAR schema versions and corresponding internal AUTOSAR releases

Schema Version	Platform release	Internal release number
AUTOSAR_00048	R19-11	R4.5.0

According to the release life cycle of AUTOSAR the release R19-11 is a minor release.

1.3.6 Overview of AUTOSAR schema versions and corresponding valid AUTOSAR releases

The AUTOSAR schema does not have an impact on the Foundation. The Foundation releases are mentioned for the sake of completeness.

Schema Version	Classic Platform release	Adaptive Platform release	Foundation release
AUTOSAR_00042	R4.3.0	R17-03	R1.1.0
AUTOSAR_00043	R4.3.0	R17-10	R1.2.0
AUTOSAR_00044	R4.3.1	R17-10	R1.3.0
AUTOSAR_00045	R4.3.1	R18-03	R1.4.0
AUTOSAR_00046	R4.4.0	R18-10	R1.5.0
AUTOSAR_00047	R4.4.0	R19-03	R1.5.1

Schema Version	AUTOSAR release
AUTOSAR_00048	R19-11

1.4 Introduction to the Adaptive Platform

The AUTOSAR Adaptive Platform is the standardized platform for microprocessor-based ECUs supporting use cases like highly automated driving as well as high speed on-board and off-board communication.

The Adaptive Platform differs in a number of aspects from the standardization approach of the Classic Platform:

- Parallel validation of specification via software implementation
- Specification of functional clusters instead of modules

1.4.1 Release strategy

The Adaptive Platform changes its life cycle state to "Evolution" according to AUTOSAR's life cycle model for its standards (as depicted in chapter 1.3.3). Starting with R19-11, AUTOSAR will release the Adaptive Platform together with the Classic Platform and Foundation in a yearly cycle. The life cycle state “Evolution” implies that users of the Adaptive Platform have a guarantee on backward compatibility for certain parts of the specifications. The differentiation is handled by the life cycle state of the requirements and specification items according to chapter 1.3.4.

1.4.2 Parallel validation of specification via implementation

The Adaptive Platform is planned to be partially validated through an AUTOSAR-internal implementation: the Adaptive Platform Demonstrator. This Demonstrator is available to all the partners and can provide further details to understand the underlying concepts of the Adaptive Platform. The Adaptive Platform Demonstrator is an exemplary implementation of the Adaptive Platform specifications. All further usage based on the Demonstrator (e.g. in series development) will become the responsibility of the respective partner. For legal constraints see the dedicated paragraphs in the Development Agreement.

For the current releases, the Demonstrator software implementation has undergone only informal reviews with no strict quality assurance. AUTOSAR is increasing the quality assurance significantly to ensure the quality criteria given by the project.

The Demonstrator comes with traceability up to the specifications to document the validation aspect.

Additionally AUTOSAR develops System Test specifications and implementation to support the test of the demonstrator implementation against the AUTOSAR requirements. These tests are also part of the release.

The main purpose of the demonstrator - the validation of the specifications with an underlying process and well-defined quality criteria - will be started in the upcoming release R20-11.

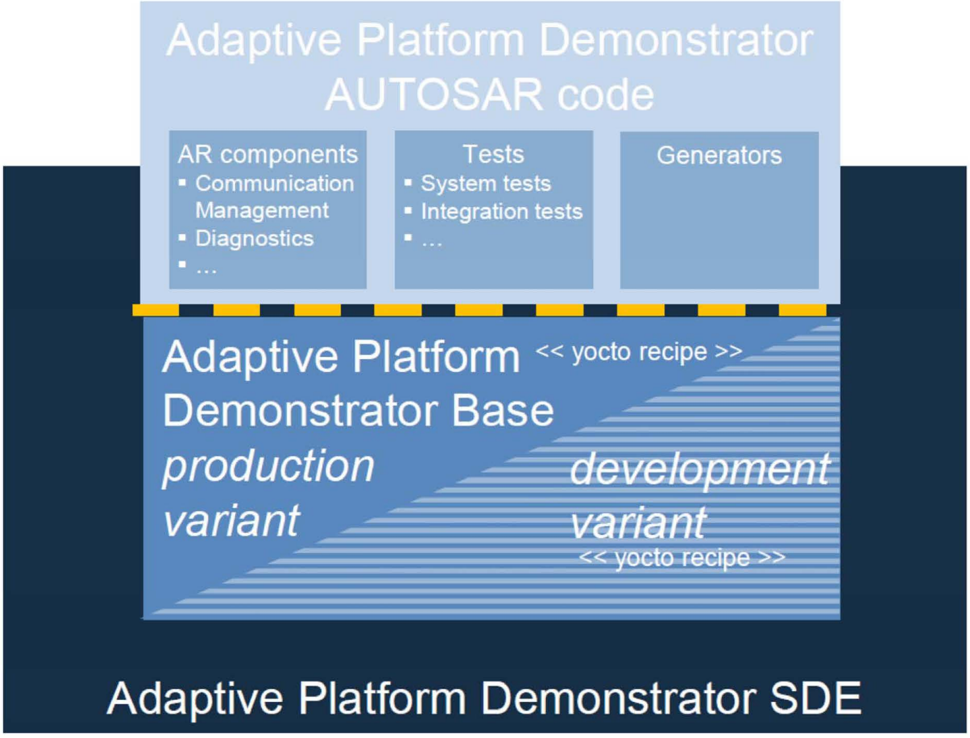


Figure 1.3: Overview of the AUTOSAR Adaptive Platform Demonstrator

1.4.3 Specification depth

Based on the development history of the Classic Platform, AUTOSAR has decided to specify functional clusters instead of a specific software architecture to provide the implementers with options to find efficient solutions for the standardized features.

1.5 Content of chapters

This document is structured as follows:

- Chapter 1 provides an introduction to AUTOSAR's release strategy, the Adaptive Platform and its standardization approach.
- Chapter 2 provides a summary of changes since the previous release of the Adaptive Platform.

- Chapter 3 contains the overview of specifications comprising the release R19-11. This chapter is structured according to the clusters of AUTOSAR release R19-11.
- Chapter 4 contains remarks about known technical deficiencies.
- Chapter 5 contains the detailed release history of all released specifications.

2 Summary of changes

This chapter contains a summary of changes which were implemented since the previous release R19-03.

2.1 Release R19-11

Release R19-11 is the first Adaptive Platform release to introduce topics with the AUTOSAR concept development process. Several concepts affecting solely the Adaptive Platform have been introduced thereby adding functionality to the platform (Socket API, Recovery Action via Application, UCM Master, Service Versioning ara::com).

Additionally some concepts target both the Classic and Adaptive Platform, strengthening the interaction between the two platforms.

Those concepts are related to security (IPSec Protocol), communication (Signal Service Translation) and diagnostics (DoIP Extension) and abstract from the platforms to enable an overall description of vehicle functionality (VFB++).

The AUTOSAR XML Schema requires the xml namespace definition file xml.xsd as xsd:import. This file is not released with the AUTOSAR specifications but can be downloaded from `https://www.w3.org/2001/03/xml.xsd`.

2.1.1 Concepts

2.1.1.1 Introduced Concepts

The following concepts in 2.1.1.1.1 - 2.1.1.1.8 have been introduced.

2.1.1.1.1 IPsec Protocol

The concept provides the ability to configure authenticated and/or encrypted communication between ECUs based on the the IETF IPsec standards. Since it works on the IP network layer 3, it can be used transparently. Applications need not be changed or even aware that their communication is secured.

2.1.1.1.2 ServiceVersioningARAcom

Support of contract service versioning. The service discovery can be configured to support version backwards-compatibility.

2.1.1.1.3 Signal Service Translation

The goal of this concept is to make Adaptive Machines interact with Classic ECUs. Adaptive Platform restricts communication to Service-oriented communication, whereas a major part of the vehicle's ECUs still uses Signal-based communication.

- A translation of these two communication approaches has to be performed:
  - Signal-to-service translation
  - Service-to-signal translation
  - To be implemented on Classic or Adaptive
  - Support for end-to-end safety and security

To support end-to-end safety (E2E) for signal service translation both directions - signal-to-service-translation and service-to-signal-translation - are covered. This includes use cases where both sides of communication use the same E2E profile as well as uses cases where different E2E profiles are in place. While performing signal-to-service translation the E2E status of the received payload is checked and forwarded to the targeted receiving communication part together with the translated payload.

2.1.1.1.4 DoIP Extension

The concept extends the DoIP specification with the possibility to allow diagnostic communication between DoIP nodes and internal testers located within the vehicle network.

2.1.1.1.5 Abstract Platform System Description (VFB++)

The first concept part of "Abstract Platform System Description (VFB++)" is released as draft. The concept will be further elaborated and extended in 2020.

The concept introduces a model of a software-platform-independent system description (VFB-level communications, requirement annotation, requirement tracing) on an abstract platform level using the AUTOSAR methodology and meta-model. The abstract platform description should allow an OEM to model the vehicle communications matrix without specifying deployment platform specifics which may be deferred to a later methodology design stage. Using AUTOSAR traceability, it shall be possible to bi-directionally trace between abstract platform and concrete level models.

The principal use cases are targeting AUTOSAR deployments in Classic or Adaptive AUTOSAR. However, deployments to "other" automotive or non-automotive domains should also be possible.

2.1.1.1.6 Socket Network Binding for ARA:com

"Socket Network Binding for ARA:com" is released as draft and will be validated in 2020.

The concept provides a static communication API (Raw Data Stream) that enables adaptive applications to read and write raw binary data streams to/from an external unit over Ethernet. The RawDataStreams API is provided as an alternative to SOME/IP, but without serialization. The main use case for the R19-11 release, is to transfer raw data from external sensors in ADAS applications, where the adaptive application acts as a client.

The concept will be further developed in the future releases, providing a more general approach with support for both client and server side of a Raw Data Stream connection.

2.1.1.1.7 Recovery Action via Application

The concept adds a new recovery action to Platform Health Management. This action notifies applications directly by call of a handler method. Recovery handling can be implemented in this handler.

2.1.1.1.8 UCM Master

This module enables the management and synchronization of "update campaigns" involving multiple ECUs/machines across a vehicle. This provides better means to update specific OEM applications (OTA client, vehicle safety manager, diagnostic applications, driver interaction).

2.1.1.2 Impact of Concepts

The introduced concepts had impact on several specifications. The following table provides a detailed overview.

Please note that some of the specifications are marked by special text formatting:

- Specifications in **bold** font are completely new specifications originating from the particular concept.
- Specifications in *italic* font are affected indirectly as they provide artefacts for the actually impacted specifications.

Concept Name	Specification Long Name	Standard
IPsecProtocol	Requirements on IPsec Protocol	Foundation
	Requirements on AUTOSAR Features	Classic Platform
	System Template	
	Requirements on Ethernet Support in AUTOSAR	
	Specification of TCP/IP Stack	
	Specification of Manifest	Adaptive Platform
ServiceVersioningARAcom	Specification of Service Discovery	Classic Platform
	System Template	
	Specification of ECU Configuration Parameters	
	Requirements on Communication Management	Adaptive Platform
	Specification of Communication Management	
	Specification of Manifest	
	Requirements on Manifest Specification	
Signal Service Translation	Explanation of Foundation Diagram Source	Foundation
	E2E Protocol Specification	
	Main Requirements	
	Glossary	
	General Specification on Transformers	Classic Platform
	Basic Software UML Model	
	Specification of SW-C End-to-End Communication Protection Library	
	Specification of Module E2E Transformer	
	Specification of RTE Software	
	Software Component Template	
	System Template	
	Specification of SOME/IP Transformer	
	Specification of COM Based Transformer	
	Layered Software Architecture	
	Requirements on System Template	
	General Specification on Transformers	Adaptive Platform
	Requirements on Manifest Specification	
	Specification of Manifest	
	Requirements on Communication Management	
	Specification of Communication Management	
DoIPExtension	Requirements on Diagnostics	Foundation

▽

△

Concept Name	Specification Long Name	Standard
	Specification of Diagnostic over IP	Classic Platform
	System Template	
	Specification of ECU Configuration Parameters (XML)	
	Basic Software UML Model	
	Specification of Diagnostics	Adaptive Platform
	Specification of Manifest	
Abstract Platform System Description (VFB++)	Glossary	Foundation
	Main Requirements	
	Specification of Abstract Platform	Adaptive Platform
Socket Network Binding for ARA:com	Glossary	Foundation
	Specification of Manifest	Adaptive Platform
	Specification of Communication Management	
	Requirements on Communication Management	
	Explanation of ara::com API	
	Explanation of Sensor Interfaces	
	Explanation of Adaptive Platform Design	
Recovery Action via Application	Specification of Platform Health Management for Adaptive Platform	Adaptive Platform
	Specification of Manifest	
UCM Master	Requirements on Update and Configuration Management	Adaptive Platform
	Specification of Update and Configuration Management	
	Manifest Specification	
	Explanation of Adaptive Platform Design	

Table 2.1: Impact of Concepts

2.1.2 Specifications

2.1.2.1 New Specifications

The following new specifications were introduced via concepts:

- Specification of Abstract Platform (UID 947, TPS)

2.1.2.2 Migrated Specifications

With this release, the following specifications were moved from AUTOSAR Classic Platform to the AUTOSAR Foundation standard:

- none

2.1.2.3 Obsolete Specifications

The following specification is set to status "obsolete" in this release:

- Guidelines for the use of the C++14 language in critical and safety-related systems (UID 839, RS)

  This specification will be released by MISRA in their future release but will still be used by AUTOSAR.

2.1.2.4 Removed Specifications

The following specification is set to status "removed" in this release:

- Requirement on Time Synchronization for Adaptive Platform (UID 879, RS)

2.1.2.5 Reworked Specifications

The following documents have been changed fundamentally in R19-11

- none

2.1.2.6 Mapping Specifications

The following documents have been mapped in R19-11

- none

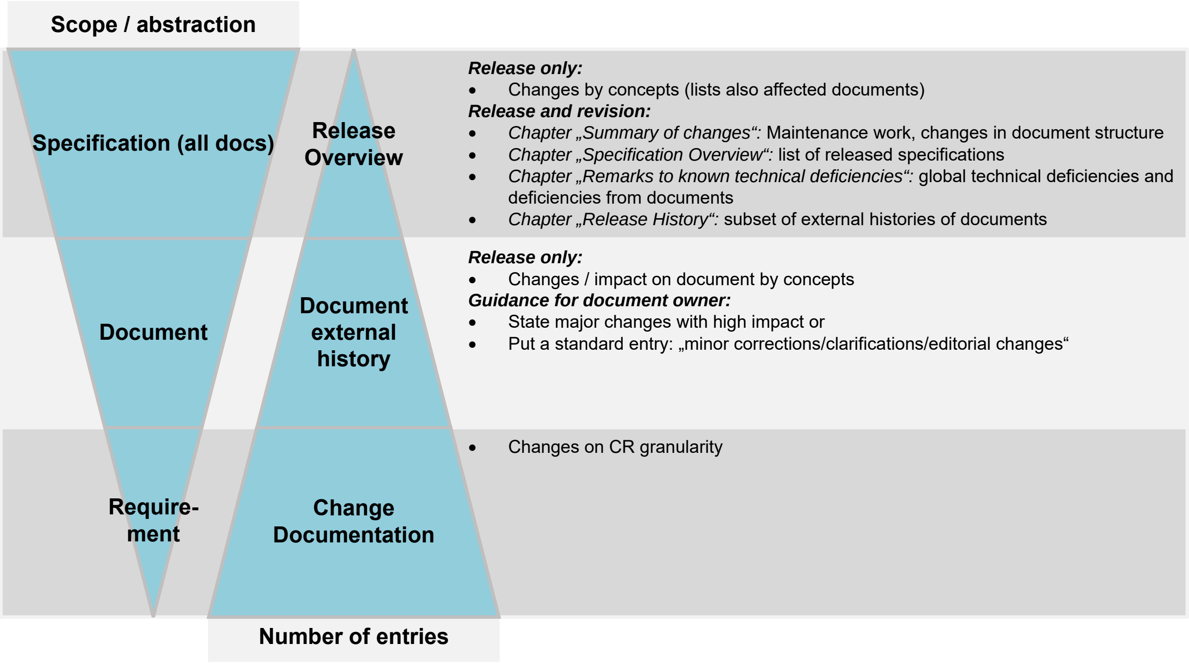
2.1.3 Release Documentation

There were no major changes regarding the Release Documentation.

2.2 History information in AUTOSAR

The following diagram shows the location of documentation of changes.

The Change Documentation will be available for Adaptive Platform starting with R20-11.



Release only:

- Changes by concepts (lists also affected documents)

Release and revision:

- *Chapter „Summary of changes“:* Maintenance work, changes in document structure
- *Chapter „Specification Overview“:* list of released specifications
- *Chapter „Remarks to known technical deficiencies“:* global technical deficiencies and deficiencies from documents
- *Chapter „Release History“:* subset of external histories of documents

Release only:

- Changes / impact on document by concepts

Guidance for document owner:

- State major changes with high impact or
- Put a standard entry: „minor corrections/clarifications/editorial changes“

- Changes on CR granularity

Figure 2.1: History information in AUTOSAR

3 Specification overview

The published specifications are divided into the clusters

- Release Documentation
- General
- Methodology and Manifests
- Adaptive Foundation
- Adaptive Services

The assignment of the specifications to these clusters is shown below.

Long Name	File Name	Life cycle changes
Release Documentation		
Adaptive Platform Release Overview	AUTOSAR_TR_AdaptivePlatform ReleaseOverview	
AUTOSAR Adaptive Platform Specification Hashes	AUTOSAR_TR_AdaptivePlatform SpecificationHashes	
General		
Design guidelines for using parallel processing technologies on Adaptive Platform	AUTOSAR_EXP_ParallelProcessing Guidelines	
Explanation of Adaptive Platform Design	AUTOSAR_EXP_PlatformDesign	
Explanation of Safety Overview	AUTOSAR_EXP_SafetyOverview	
Functional Cluster Shortnames	AUTOSAR_TR_FunctionalCluster Shortnames	
General Requirements specific to Adaptive Platform	AUTOSAR_RS_General	
General Specification of Adaptive Platform	AUTOSAR_SWS_General	
Guidelines for the use of the C++14 language in critical and safety-related systems	AUTOSAR_RS_CPP14Guidelines	obsolete (R19-03 version)
Guidelines for using Adaptive Platform interfaces	AUTOSAR_EXP_AdaptivePlatform InterfacesGuidelines	
System Tests of Adaptive Platform	AUTOSAR_TR_AdaptivePlatform SystemTests	
Methodology and Manifests		
Collection of blueprints for AUTOSAR Adaptive Platform models	AUTOSAR_MOD_AdaptivePlatform GeneralBlueprints	
Meta Model	AUTOSAR_MMOD_MetaModel	
Meta Model-generated XML Schema	AUTOSAR_MMOD_XMLSchema	
Methodology for Adaptive Platform	AUTOSAR_TR_AdaptiveMethodology	
Requirements on Manifest Specification	AUTOSAR_RS_ManifestSpecification	
Specification of Abstract Platform	AUTOSAR_TPS_AbstractPlatform Specification	Initial release

▽

△

Long Name	File Name	Life cycle changes
Specification of Manifest	AUTOSAR_TPS_ManifestSpecification	
Specification of Platform Types for Adaptive Platform	AUTOSAR_SWS_AdaptivePlatform Types	
Supplementary material of the AUTOSAR XML Schema	AUTOSAR_TR_XMLSchema Supplement	
Adaptive Foundation		
Specification of Core Types for Adaptive Platform	AUTOSAR_SWS_CoreTypes	
Explanation of ara::com API	AUTOSAR_EXP_ARAComAPI	
Explanation of IPsec Implementation Guidelines	AUTOSAR_EXP_IPsecImplementation Guidelines	
Requirements on Communication Management	AUTOSAR_RS_Communication Management	
Requirements on Cryptography	AUTOSAR_RS_Cryptography	R19-03 version
Requirements on Execution Management	AUTOSAR_RS_Execution Management	
Requirements on Identity and Access Management	AUTOSAR_RS_IdentityAndAccess Management	
Requirements on Operating System Interface	AUTOSAR_RS_OperatingSystem Interface	
Requirements on Persistency	AUTOSAR_RS_Persistency	
Requirements on Platform Health Management for Adaptive Platform	AUTOSAR_RS_PlatformHealth Management	
Requirements on Security Management for Adaptive Platform	AUTOSAR_RS_SecurityManagement	
Specification of Communication Management	AUTOSAR_SWS_Communication Management	
Specification of Cryptography for Adaptive Platform	AUTOSAR_SWS_Cryptography	R19-03 version
Specification of Diagnostics	AUTOSAR_SWS_Diagnostics	
Specification of Execution Management	AUTOSAR_SWS_Execution Management	
Specification of Identity and Access Management	AUTOSAR_SWS_IdentityAndAccess Management	
Specification of Log and Trace	AUTOSAR_SWS_LogAndTrace	
Specification of Operating System Interface	AUTOSAR_SWS_OperatingSystem Interface	
Specification of Persistency	AUTOSAR_SWS_Persistency	
Specification of Platform Health Management for Adaptive Platform	AUTOSAR_SWS_PlatformHealth Management	
Specification of RESTful communication	AUTOSAR_SWS_REST	
Specification of Time Synchronization for Adaptive Platform	AUTOSAR_SWS_Time Synchronization	
Adaptive Services		
Explanation of Sensor Interfaces	AUTOSAR_EXP_SensorInterfaces	
Requirements of State Management	AUTOSAR_RS_StateManagement	
Requirements on Update and Configuration Management	AUTOSAR_RS_UpdateAndConfig Management	

▽

△

Long Name	File Name	Life cycle changes
Specification for Network Management	AUTOSAR_SWS_Network Management	
Specification of State Management	AUTOSAR_SWS_StateManagement	
Specification of Update and Configuration Management	AUTOSAR_SWS_UpdateAndConfig Management	

Table 3.1: Specification Overview

4 Remarks to known technical deficiencies

The technical deficiencies per specification are - if applicable - mentioned inside the respective specification in a chapter "Known Limitations" located after the table of contents.

The following technical deficiencies are to be mentioned:

- **Specification of Operating System Interface (UID 719, SWS)**
  There is currently no sufficient API providing periodic time-based processing to fulfill requirement RS_OSI_00102. This will be defined in a future release.
- **Specification of Communication Management (UID 717, SWS)**
  The current version of this document is missing some functionality which is not standardized and specified within the SWS Communication Management document but described in Explanation of ara::com API [1] and implemented in the demonstrator code:
  - Local Buffer Overruns: Currently it is not specified what happens if local buffers are full because the application accesses data slower than they are received over the network. The limitations regarding E2E protection and the detectable failure modes are described in chapter 7.3.1 of [2].
  - The optionality introduced with the Tag-Length-Value serialization principle described in [3] and [4] does not support the existence of optional method arguments.
- **Specification of Execution Management (UID 721, SWS)**
  The sections “Resource Limitation” and “Fault Tolerance” are not complete. In particular the contents will be expanded with more properties and formal requirements in the next release.
- **System Tests of Adaptive Platform (UID 890, TR)**
  - Test cases might not cover the whole RS as specified against test cases
  - Test Setup and configurations are exemplary only and may cover broader scope than represented by test cases in corresponding sections
  - Test cases might not be fully covered by corresponding system test implementations
  - System test cases are just examples, since there could be many ways to define and implement use case scenarios
  - Diagnostics traceability is obsolete as SRS is changed to RS
  - Log and Trace does not have any RS traceability. Traceability will be added in the upcoming release
  - In the E2E test case, the common parts of the E2E profiles are checked

  - Time Base (TB) of Time Synchronization has five TB types. (Synchronized Master TB, Offset Master TB, Synchronized Slave TB, Offset Slave TB, Pure Local TB.) [5] describes multiple TB types as scope, but system test cases may not cover whole TB types.
  - In Cryptography test cases STS_CRYPTO_00002 and STS_CRYPTO_00004, public and private keys are used only by the test application to simplify the test case (i.e. not corresponding to practical use of asymmetric keys)
  - Even if the behavior is different, same application and/or service numbers are used across different test cases

- **Specification of Persistency (UID 858, SWS)**
  The configuration of encryption for Persistency is not defined in [4].

- **Specification of Crypto Interface (UID 883 ,SWS)**
  The released version of [6] is identical to the released version in R19-03. The specification of the API will undergo a rework for the upcoming release.
  Additionally the current version of this document is missing some functionality that is available in the AUTOSAR Classic Platform:

  - Secure Counter: There is currently no API available to access secure counter primitives that an implementation may provide.

  The following functional domains and descriptions are still missing in the current version of Crypto API specification:

  - Asynchronous interfaces: Currently there is only a synchronous API specification and asynchronous behavior (if required) should be implemented on the consumer application level. It can be done via utilization of dedicated execution threads for long-time operations.
  - X.509 certificates support: Crypto API doesn't provide complete specification of the X.509 certificates management on the client (ECU) side yet. Current version of Crypto API specifies only minimal subset of interfaces responsible for basic X.509 functionality and related on utilization of cryptographic algorithms. Current API supports extraction and parsing of only basic attributes of X.509 certificates and certification requests. An extension of the API specification by additional interfaces dedicated for complete support of X.509 extensions is planned for the next release of this specification. Note: Generally current specification of the X.509 Provider API is preliminary and subject for extensions and changes.
  - Memory management: In the current version of the specification Crypto Provider supports the safety-aligned memory management concept suitable for real-time applications. Up to the next release this concept will be extended for X.509 Provider too. Application of any memory management mechanisms specific for support of asynchronous calls (like std::future) is in scope a developer responsibility.

  - Formats of cryptographic objects: Current version of Crypto API has minimal support of well-known cryptographic formats encoding/decoding: support of only DER and PEM encoding for X.509 certificates and certificate signing requests is required from any implementation of Crypto API. For other cryptographic objects an implementation can support only "raw" formats. Following extension of the Crypto API by unified interfaces for encoding/decoding of complex objects to standard formats is planned for the next release of this specification.
  - Key slots modeling: Now Crypto API defines some structures that should be produced as a result of the key slots modeling process. But the whole concept of the key slots modeling is not finished yet. Therefore Key Storage API can be updated slightly for next release in order to extend support of the ara::core::InstanceSpecifier type as one of mechanisms for the Logical Key Slot identification.
  - Functional specification: Detailed functional specification (chapter 7) is not available yet and will be elaborated for next Autosar AP release.
  - Depth of inheritance: The performance of the inheritance tree design applied for the Crypto Provider interfaces is still subject to further investigation. Therefore a redesign of APIs defined in namespace ara::crypto::cryp may be executed for next Autosar release, in order to achieve a very limited inheritance depth (or completely "flattened" API design).

- **Specification of Core Types for Adaptive Platform (UID 903, SWS)**

  - The specification of some data types (Array, Map, Optional, String, StringView, Variant) mentions "supporting constructs", but lacks a precise scope definition of this term.
  - The specification of some data types (Map, Vector, String) is lacking a comprehensive definition of memory allocation behavior; it currently only describes it as "implementation-defined".
  - Chapter 7 describes some behavior informally that should rather be given as specification items.

- **Specification of Identity and Access Management (UID 900, SWS)**

  - A detailed API will be added probably in Release R19-11.
  - Currently limited to ara::com
  - For other Functional Clusters, implementation on Policy Enforcement Points are envisaged for the next release (R19-11).

- **Specification of Log and Trace (UID 853, SWS)**
  The provided Logging framework API is designed to be independent from the

underlying Logging back-end implementation and as such doesn't impose limitations.

- **Specification for Network Management (UID 898, SWS)**
  - The Adaptive Network Management is actually only supporting UdpNM.
  - The Adaptive Network Management does not allow node detection (Repeat Message State) but only handles incoming requests.
  - The Adaptive Network Management cannot be configured as the master network coordinator.
  - The Adaptive Network Management does not support coordinated shutdown using the information in CBV.
  - The Adaptive Network Management does not support passive mode and passive start-up. Passive start-up would mean that a node has started (i.e. goes to Normal mode), but the network has been woken up by another node.
  - Modeling part for mapping the logical networks to the BitVector positions is not available in the manifest.
  - Update and access of User Data was removed as the service interface to Applications has been removed. State Management will control the network request/release and it must be clarified if user data changes/indications shall be done via State Management or directly by applications.
- **Specification of Operating System Interface (UID 719, SWS)**
  The following functionality is mentioned within this document but is not fully specified in this release:
  - There is currently no sufficient API providing periodic time-based processing to fulfill RS_OSI_00102. This will be defined in a future release.
- **Specification of Platform Health Management for Adaptive Platform (UID 851, SWS)**
  - Daisy chaining (i.e. forwarding Supervision Status, Checkpoint or Health channel information to an entity external to PHM or another PHM instance) is currently not supported in this document release.
  - Platform Health Management configuration related to Supervision Modes is not fully supported in this document release.
  - An API to inform Supervised Entities about the Supervision states is available only in polling mode. No API using notification mode is available in this release.
  - Interface with the Diagnostic Manager is not specified in this release.

- **Specification of RESTful communication (UID 876, SWS)**
  The interfaces are only specified to the point to make semantics clear. To be precise this document does not yet fully specify the qualification C++ functions noexcept, overloading of functions to provide move semantics for optimization purposes nor does it claim to be const-correct. Move semantics in particular are specified where required for semantic correctness only. Also only HTTP network binding aspects of the AUTOSAR meta model are currently supported by the SWS_REST. No modeling of the RestServiceInterface internal structure is possible with the current SWS_REST. The error handling for RESTful communication is currently limited due to the fact that errors are not reported in the context of a request transmission.

- **Specification of State Management (UID 908, SWS)**
  This section lists known limitations of State Management and their relation to this release of the AUTOSAR Adaptive Platform with the intent to provide an indication how State Management within the context of the AUTOSAR Adaptive Platform will evolve in future releases.

  The following functionality is mentioned within this document but is not (fully) specified in this release:

  - This document will show the basic principles of the intended functionality of State Management. To enable State Management to be portable, in future versions of this document standardized fields and values shall be introduced.

  - Communication Control for Diagnostic reasons this is not yet discussed with Adaptive Diagnostics.

  - RequestRestart for Diagnostic reasons this is discussed with Adaptive Diagnostics, but some interface details are not yet finalized.

- **Specification of Time Synchronization for Adaptive Platform (UID 880, SWS)**
  The Time Synchronization module is bound to Adaptive Platform Systems.
  For the TS, it is necessary that at least there is one TBR in the system, otherwise no functionality can be provided to the Adaptive Applications (i.e. the Adaptive Applications should not get any handle for Time Base Resources).
  The current concept on TimeSync is not in line with the port prototype approach. The topic on InstanceSpecifier is not yet finalized. Further changes to be expected in R19-11.
  API design is not fully compliant to Adaptive Platform Design Rules which request the usage of UpperCamelCase.

  - Configuration: Please refer to the corresponding model elements.

  - Time Gateway: Time Gateway functionality is currently not in scope of the Time Synchronization module for the Adaptive Platform.

  - Out of Scope: Errors, which occurred during Global Time establishment and which are not caused by the module itself (i.e. loss of PTP global time is not an issue of the TS but of the TSP modules) are out of the scope of this module.

- **Specification of Update and Configuration Management (UID 888, SWS)**
  UCM is not responsible to initiate the update process. UCM realizes a service interface to achieve this operation. The user of this service interface is responsible to verify that the vehicle is in a safe state before executing a software update procedure on demand. It is also in the responsibility of the user to communicate with other AUTOSAR Adaptive Platforms or AUTOSAR Classic Platforms within the vehicle. Therefore management of software dependencies between different physical or virtual ECU software platforms is currently out of UCM's scope but will be managed by the UCM Master which will be introduced in the next release.

  The UCM receives a locally available software package for processing. The software package is usually downloaded from the OEM backend. The download of the software packages has to be done by another application, i.e.UCM does not manage the connection to the OEM backend. Prior to triggering their processing, the software packages have to be transferred to UCM by using the provided ara::com interface.

  The UCM update process is designed to cover updates on use case with single AUTOSAR Adaptive Platform. UCM can update Adaptive Applications, the AUTOSAR Adaptive Platform itself, including all functional clusters and the underlying OS. Distinction between different types of updates, such as safety critical updates vs infotainment updates, isn't addressed in this release. Currently such distinction shall be included into vendor specific meta-data.

  The UCM is not responsible for enforcing authentication and access control to the provided interfaces. The document currently does not provide any mechanism for the confidentiality protection as well as measures against denial of service attacks. The assumption is that the platform preserves the integrity of parameters exchanged between UCM and its user.

  The UCM do not support update of ECUs not supporting ARA::COM or UDS with aligned diagnostic flash sequence support.

  This UCM Master specification release scope is limited to update, install or remove of Adaptive platform Software Clusters. It is planned to specify any modification of Classic platform (being FOTA or non FOTA compatible) and non-Autosar platform from release 20-11.

- **Specification of Manifest (UID 713, TPS)**
  The AUTOSAR SWS REST [7] defines a low-level API for REST-based communication. The content of section 12, on the other hand, applies for the configuration of a not-yet standardized API on top of the ara::rest API.

5 Release history

5.1 Release R19-11

Release R19-11 was originally released on 28 November 2019.

Name	Specification history entry
Adaptive Platform Release Overview	• Release Life Cycle Status: R19-11 is in Evolution, R19-11 supersedes R19-03
Design guidelines for using parallel processing technologies on Adaptive Platform	• No content changes • Changed Document Status from Final to published
Explanation of Adaptive Platform Design	• Updated the architecture logical view • Updates in Execution Management, Communication Management, Security, Diagnostics, Persistency, State Management, Network Management, Update and Configuration Management, Platform Health Management, Core Types chapters updated due to changes in SWS • Various minor updates for clarification • Changed Document Status from Final to published
Explanation of ara::com API	• Added access to current field value from Get/SetHandler • Changed Document Status from Final to published
Explanation of IPsec Implementation Guidelines	• No content changes • Changed Document Status from Final to published
Explanation of Safety Overview	• No content changes • Changed Document Status from Final to published
Explanation of Sensor Interfaces	• Added reference to Communication Management in chapter 4 • Spelling updates in chapters 3 and 5 • Changed Document Status from Final to published
Functional Cluster Shortnames	• Functional cluster shortname s2s for Signal to Service removed • Changed Document Status from Final to published
General Requirements specific to Adaptive Platform	• More design guidelines added • Changed Document Status from Final to published
General Specification of Adaptive Platform	• No content changes. • Changed Document Status from Final to published.
Guidelines for the use of the C++14 language in critical and safety-related systems	• Added the obsolete statement
Guidelines for using Adaptive Platform interfaces	• Persistency and Platform Health Management chapters added • Changed Document Status from Final to published
Methodology for Adaptive Platform	• disentangle service interface handling • remove machine state • Changed Document Status from Final to published • editorial changes

▽

△

Name	Specification history entry
Requirements of State Management	• No content changes • Changed Document Status from Final to published
Requirements on Communication Management	• Added new chapter Requirements Guidelines • Introduced – Signal2Service Translation – Service Versioning – Raw Data Streaming • Communication • Changed Document Status from Final to published • Additional E2E support
Requirements on Cryptography	• Editorial changes and rephrasing • Improved requirements description and rationale (Updated: – RS_CRYPTO_02001 – RS_CRYPTO_02002 – RS_CRYPTO_02003 – RS_CRYPTO_02004 – RS_CRYPTO_02005 – RS_CRYPTO_02007 – RS_CRYPTO_02009 – RS_CRYPTO_02109 – RS_CRYPTO_02116 – RS_CRYPTO_02202 – RS_CRYPTO_02206)
Requirements on Execution Management	• Updated: RS_EM_00009 and RS_EM_00103 • Changed Document Status from Final to published
Requirements on Identity and Access Management	• No content changes • editorial changes • Changed Document Status from Final to published
Requirements on Manifest Specification	• Added requirements for – Service Versioning – Signal-To-Service Translation • Changed Document Status from Final to published
Requirements on Operating System Interface	• Updated document structure • Changed Document Status from Final to published
Requirements on Persistency	• Statistic data • Initialization/De-initialization • Changed Document Status from Final to published
Requirements on Platform Health Management for Adaptive Platform	• No content changes. • Changed Document Status from Final to published.

▽

△

Name	Specification history entry
Requirements on Security Management for Adaptive Platform	• Reworded secure channel requirements RS_SEC_04001, RS_SEC_04003 and RS_SEC_04003 • Changed Document Status from Final to published
Requirements on Update and Configuration Management	• Added UCM Master concept requirements • Changed Document Status from Final to published
Specification for Network Management	• Added Functional Cluster life cycle chapter • Several quality improvments • Improved linking to Manifest • Changed Document Status from Final to published
Specification of Abstract Platform	• Initial release
Specification of Communication Management	• Introduced – Signal2Service TranslationBinding – Support for Invalid Values – Additional E2E support – Service Versioning – Raw Data Streaming Interface • Minor changes and bugfixes • Changed Document Status from Final to published
Specification of Core Types for Adaptive Platform	• Rework error handling definitions • Add specifications of BasicString and Byte, and add overloads and template specializations for ErrorCode, Result, Future, and Promise • Add bits about validity of InstanceSpecifier arguments, and rework the specification of its construction mechanism • Rework ErrorCode to get rid of "User Message" and make "SupportDataType" implementation-defined • Replace PosixErrorDomain with CoreErrorDomain • Rename FutureErrorDomain accessor function • Changed Document Status from Final to published
Specification of Cryptography for Adaptive Platform	• "Direct" prefix of Crypto API is removed, because now it is single • All bugs found after R18-03 are fixed • Crypto API is converted for usage of basic ara::core types • Crypto API is converted for support of the "Exception-less" approach • Detalization of Crypto API specification is extended
Specification of Diagnostics	• Document quality improvement and fixing bugs • Incorporated Quality Scope Review Findings • Partly removed obsolete requirements • Removed obsolete service interfaces • Changed Document Status from Final to published

▽

△

Name	Specification history entry
Specification of Execution Management	• Further refinement of State Management API and semantics • Introduced support for trusted platform • Added support for non-reporting Processes • Execution Management API uses Core types • Changed Document Status from Final to published
Specification of Identity and Access Management	• No content changes. • Changed Document Status from Final to published.
Specification of Log and Trace	• Removed Class LogManager. Moved remoteClientState() to chapter 8.2 Function definitions (logging.h) • Added Functional Cluster shutdown behavior. Added Funtional Cluster initialization via ara::core::Initialize() • Removed TSYNC related spec items from chapter 7.4 • Refactoring and editorial changes • Changed Document Status from Final to published
Specification of Manifest	• Overhaul of Signal-to-Service Translation • Support for Raw Data Streams • Support for Vehicle Package • Support for Service Versioning • Changed Document Status from Final to published
Specification of Operating System Interface	• Added description of startup and shutdown of OSI. • Clarified that Operating System must allow calling getenv() from C++ constructors. • Document template upgrade. • Changed Document Status from Final to published.
Specification of Persistency	• Introduced reset and restore of storages • Introduced storage statistics • Improved compliance with general AUTOSAR concepts • Improved naming and consistency of classes / methods / functions /constants • Changed Document Status from Final to published
Specification of Platform Health Management for Adaptive Platform	• Added recovery action via application • Usage of ara::core types in PHM APIs • Set data types to uint32_t by default • Editorial rework of chapters 7 and 8 • Changed Document Status from Final to published
Specification of Platform Types for Adaptive Platform	• No content changes. • Changed Document Status from Final to published.
Specification of RESTful communication	• No content changes • Changed Document Status from Final to published

▽

△

Name	Specification history entry
Specification of State Management	• Interface with ExecutionManagement changed to StateClient • RequestState and ReleaseRequest kept deprecated • Changed Document Status from Final to published
Specification of Time Synchronization for Adaptive Platform	• Requirements traceability changed to Foundation RS TimeSync specification • Add Time Validation
Specification of Update and Configuration Management	• Introduced UCM Master concept • Software Package state machine updated for processing while streaming • Reviewed UCM State Machine • Added new security analysis appendix • Changed Document Status from Final to published
System Tests of Adaptive Platform	• Changed format for Actors (App, Events, Services etc.) • Added new sections and test cases for Security Management, NetworkManagement and Cryptography • Added more test cases for CM, EMO, TS, and E2E • Changed Document Status from Final to published

Table 5.1: Release History