Document Title	Requirements on Methodology
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	362

Document Status	Final
Part of AUTOSAR Standard	Foundation
Part of Standard Release	1.4.0

Document Change History			
Date	**Release**	**Changed by**	**Description**
2018-03-29	1.4.0	AUTOSAR Release Management	• No content changes
2017-12-08	1.3.0	AUTOSAR Release Management	• Migration of the Classic Platform requirements document to standard “Foundation” finalized • Enhanced quality of requirements • New requirement added which applies to both platforms • New requirement for Classic Platform only added
2017-10-27	1.2.0	AUTOSAR Release Management	• Enhanced formal quality of requirements and requirements tracing
2017-03-31	1.1.0	AUTOSAR Release Management	• – Migration of document to standard “Foundation” – • Only those requirements from Classic Platform incorporated which apply to Adaptive Platform as well • New requirements for Adaptive Platform added

Disclaimer

Table of Contents

Bibliography

[1] Standardization Template
AUTOSAR_TPS_StandardizationTemplate

[2] Main Requirements
AUTOSAR_RS_Main

1 Introduction

This document defines the requirements needed to specify the AUTOSAR methodology.
The document is structured into several sections with general requirements for the AUTOSAR methodology, see section 2.1, as well as dedicated requirements for the Adaptive Platform in section 2.3.

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]).

1.2 Abbreviations

Abbreviation	Description
AP	Adaptive Platform
AUTOSAR	Automotive Open System Architecture
CP	Classic Platform
ECU	Electronic Control Unit
OEM	Original Equipment Manufacture
RTE	Runtime Environment
SIL	Safety Integrity Level (IEC61508 definition)
SWC	Software Component
VFB	Virtual Functional Bus

1.3 Requirements Tracing

The following table references the requirements specified in [2] and links to the fulfillments of these.

Requirement	Description	Satisfied by
[RS_Main_00002]	AUTOSAR shall provide a software platform for high performance computing platforms	[RS_METH_00204] [RS_METH_00207]
[RS_Main_00030]	AUTOSAR shall support development processes for safety related systems	[RS_METH_00018] [RS_METH_00069]
[RS_Main_00060]	AUTOSAR shall provide a standardized software interface for communication between Applications	[RS_METH_00033] [RS_METH_00201]
[RS_Main_00080]	AUTOSAR shall provide means to describe a component model for Application Software	[RS_METH_00062] [RS_METH_00080] [RS_METH_00202]
[RS_Main_00130]	AUTOSAR shall provide an abstraction from hardware	[RS_METH_00032] [RS_METH_00033]
[RS_Main_00140]	AUTOSAR shall provide network independent communication mechanisms for applications	[RS_METH_00032] [RS_METH_00033]
[RS_Main_00150]	AUTOSAR shall support the deployment and reallocation of AUTOSAR Application Software	[RS_METH_00033] [RS_METH_00078] [RS_METH_00079] [RS_METH_00201] [RS_METH_00202] [RS_METH_00205] [RS_METH_00208]
[RS_Main_00161]	AUTOSAR shall provide a unified way to describe software systems deployed to Adaptive and / or Classic platforms.	[RS_METH_00200]
[RS_Main_00190]	AUTOSAR shall support standardized interoperability with non-AUTOSAR software	[RS_METH_00016] [RS_METH_00018]
[RS_Main_00220]	The functional interfaces of AUTOSAR shall be specified in standard C	[RS_METH_00015]
[RS_Main_00250]	AUTOSAR methodology shall provide a predefinition of typical roles and activities.	[RS_METH_00042] [RS_METH_00066]
[RS_Main_00290]	No description	[RS_METH_00069]
[RS_Main_00300]	AUTOSAR shall provide data exchange formats to support work-share in large inter and intra company development groups	[RS_METH_00006] [RS_METH_00018] [RS_METH_00020] [RS_METH_00033] [RS_METH_00069] [RS_METH_00077] [RS_METH_00078] [RS_METH_00079] [RS_METH_00080] [RS_METH_00208]
[RS_Main_00301]	AUTOSAR shall specify profiles for data exchange to support work-share in large inter- and intra-company development groups	[RS_METH_00083] [RS_METH_00084]
[RS_Main_00310]	AUTOSAR shall support hierarchical Application Software design methods	[RS_METH_00041]
[RS_Main_00320]	AUTOSAR shall provide formats to specify system development	[RS_METH_00206]
[RS_Main_00330]	No description	[RS_METH_00032]

[RS_Main_00350]	AUTOSAR specifications shall be analyzable and support according methods to demonstrate the achievement of safety related properties.	[RS_METH_00041]
[RS_Main_00360]	AUTOSAR shall support variant management	[RS_METH_00062] [RS_METH_00074] [RS_METH_00075] [RS_METH_00076]
[RS_Main_00400]	AUTOSAR shall provide a layered software architecture	[RS_METH_00032] [RS_METH_00033]
[RS_Main_00503]	AUTOSAR shall provide a Software Platform that supports adaptation of communication topology after production	[RS_METH_00203] [RS_METH_00204] [RS_METH_00205]
[RS_Main_00505]	No description	[RS_METH_00206]
[RS_Main_00507]	AUTOSAR shall reflect the stages of a software system development in a formal model description	[RS_METH_00056]

2 Methodology Requirements

This chapter provides the definition of the requirements.

2.1 General Requirements

This sections specifies the general requirements, which are mainly valid for both platforms.

2.1.1 Main Requirements

[RS_METH_00006] The methodology shall explain how to build an AUTOSAR system ⌈

Type:	valid
Description:	The methodology shall explain how to build an AUTOSAR system using the activities and work products. It should be like a user manual to help an organization efficiently apply AUTOSAR. In particular, the methodology shall explain how to build a system consisting of classic and adaptive platforms.
Rationale:	A strong methodology is necessary to effectively manage building a large system.
Dependencies:	–
Use Case:	An engineer would like to complete an activity and would like to know what inputs are needed, guidance should be used, etc. Typical use cases involved to build an AUTOSAR system include: • SWC implementation • ECU integration • System integration
Applies to:	CP,AP
Supporting Material:	–

⌋*(RS_Main_00300)*

[RS_METH_00041] The methodology shall support top-down and bottom-up approaches ⌈

Type:	valid
Description:	The methodology shall support the top-down and bottom-up approach. In the top-down approach, all constraints on the application software and their distribution on ECUs shall be considered. In the bottom-up approach, all constraints coming from the hardware (ECUs/sensors/actuators) should be taken into account.
Rationale:	To improve the integration phases, and to master the complexity in embedded RT distributed systems
Dependencies:	–

Use Case:	If in a given vehicle architecture, a new ECU is added or an existing ECU is replaced with a new one, all the new or modified resources from the ECU need to be included into the system configuration during integration.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00310*, *RS_Main_00350*)

[RS_METH_00016] The methodology shall support building a system of both AUTOSAR and Non-AUTOSAR ECUs ⌈

Type:	valid
Description:	The methodology shall support building a system of AUTOSAR compliant ECUs and non-AUTOSAR compliant ECUs.
Rationale:	The design of a complete vehicle system shall be supported.
Dependencies:	–
Use Case:	Legacy ECUs and LIN slaves need to interoperate with AUTOSAR ECUs.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00190*)

[RS_METH_00200] The methodology shall support building a system consisting of several AUTOSAR platforms ⌈

Type:	valid
Description:	The methodology shall support building a system consisting of several AUTOSAR platforms.
Rationale:	The design of a complete vehicle system shall be supported.
Dependencies:	–
Use Case:	The communication description between machines (or ECUs) based on classic and adaptive AUTOSAR platforms.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00161*)

[RS_METH_00018] The methodology shall be modular ⌈

Type:	valid
Description:	Utilize process components. Subprocesses shall be complete and testable on their own to allow the usage of certain portions of the methodology while still integrating with legacy tools and processes.

Rationale:	It is easier to understand and verify all portions of the methodology. It is easier to manage modifications, encapsulates ripple effect due to changes to allow migration of current processes. It is easier to utilize both legacy and AUTOSAR activities. It should be possible to start from an intermediate activity and not necessarily from the beginning of the methodology. A modular methodology facilitates organizations to migrate from or merge with their current processes. A modular methodology allows organizations to insert intermediate activities such as quality gates, or other inspections, as well as collect metrics necessary to comply with CMMI processes and/or SIL-3.
Dependencies:	–
Use Case:	An organization is planning to introduce an AUTOSAR ECU into their existing architecture, but is not planning to use the System Template and their respective activities and work products. Rather they plan to begin directly at the ECU level.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00190*, *RS_Main_00300*, *RS_Main_00030*)

[RS_METH_00032] The methodology shall support different levels of abstractions ⌈

Type:	valid
Description:	The methodology shall support different views for the development of an AUTOSAR system. This corresponds to the typical domains and parties, which are involved in the system development.
Rationale:	To improve the integration phases and to master the complexity in embedded RT distributed systems.
Dependencies:	–
Use Case:	AUTOSAR is using several abstraction levels to describe the information exchanged between the different players. In an early phase for instance only the "Virtual Functional Bus" is used, whereas in later development phases we handle real implementations of the SWC deployed to several ECUs.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00130*, *RS_Main_00140*, *RS_Main_00330*, *RS_Main_00400*)

[RS_METH_00020] The methodology shall support round-trip engineering ⌈

Type:	valid
Description:	The methodology shall support round-trip engineering. This implies that several iteration loops might be necessary in order to finalize a task or work product.
Rationale:	Meet AUTOSAR Quality requirements.
Dependencies:	–
Use Case:	Automotive systems are typically developed in several sample phases (A, B, C, etc). A single Software Component is updated in a AUTOSAR System. The updated ECU Extract still matches the existing ECU Configuration (as long as no contradicting changes are made in the iteration).

Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00300*)

[RS_METH_00077] The methodology shall support different views on the SW-C structure by OEMs and suppliers ⌈

Type:	valid
Description:	The methodology shall support the interaction between OEM and supplier, where the OEM and the supplier have different views on the SW-C structure.
Rationale:	Possibility for the supplier to adapt SW-C structure.
Dependencies:	–
Use Case:	The OEM hands over the initial System Extract to the supplier as a formal requirements specification. The supplier extends and refactors this System Extract. In the next development cycle the OEM hands over an updated System Extract to the supplier. Thereafter the supplier has to update his System Extract structure based on the updates made by the OEM. The amount of changes on the supplier side shall be limited to the changes caused by OEM updates.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00300*)

[RS_METH_00078] The methodology shall explain the typical usage of different views on the system of the OEM ⌈

Type:	valid
Description:	The methodology shall support use cases of the OEM, where the OEM has different views on the system.
Rationale:	Methodology consistency in the software system development approach
Dependencies:	–
Use Case:	An OEM might structure the AUTOSAR software components from a functional point of view. However, for the concrete vehicle development project a topological view of structure of SW-Cs is needed. For better handling during the life-cycle, the SW-Cs from the functional decomposition are mapped to the topological view using appropriate mappings.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00300, RS_Main_00150*)

[RS_METH_00079] The methodology shall explain the typical usage of different views on the system of the supplier ⌈

Type:	valid
Description:	The methodology shall support use cases of the supplier where the supplier has different views on the system.
Rationale:	Methodology consistency in the software system development approach
Dependencies:	–

Use Case:	The supplier needs to map different views of the system, e.g. a) the supplier already has an existing software architecture. Via software sharing some of the components are substituted by the ones delivered by the OEM. b) The supplier needs to formally describe changes between system descriptions representing different releases. c) The supplier develops one ECU for different OEMs and therefore needs to map the requirement-views of the OEMs to his solution view. d) The supplier realizes the OEMs definition for one ECU by 2 ECUs and therefore needs to map the system descriptions.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00300*, *RS_Main_00150*)

2.1.2 Programming Language

[RS_METH_00015] The methodology shall be independent of programming languages ⌈

Type:	valid
Description:	The methodology shall be independent of programming languages by providing generic solutions. For portions that are necessarily dependent on the programming language, these sections shall be explicitly noted and be modular such that the overall methodology can be tailored to accommodate other programming languages.
Rationale:	By appropriately structuring the methodology to support existing and emerging programming languages, the methodology can be consistently and successfully applied across an entire vehicle.
Dependencies:	–
Use Case:	An ECU built for a particular microcontroller is explicitly optimized for programming language ABC. The methodology explains when and how to specify and to select the implementation of the software components compatible with the required programming language.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00220*)

2.1.3 Activities

[RS_METH_00066] The methodology shall allow activities that reference tools ⌈

Type:	valid
Description:	Activities may reference tools that help to complete the activity. The methodology shall describe these types of tools and when they are used.

Rationale:	By defining which tools are needed, the performers of the activity can ensure that all tools have been sourced and installed prior to the beginning of the activity. As well, the implementers of tools that are AUTOSAR specific, have a clear understanding of what activities their tool should support and know what the input and output work products are available. This will help to ensure interoperability of AUTOSAR tools.
Dependencies:	–
Use Case:	For the Classic Platform, the activity "Generate RTE" requires an RTE generator tool and a compiler.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00250*)

[RS_METH_00042] The methodology shall incorporate the usage of industry standard tools ⌈

Type:	valid
Description:	Where industry standard tools, such as compilers and linkers exist, the methodology shall incorporate them.
Rationale:	AUTOSAR should not require the use of particular tools when industry standard tools already exist.
Dependencies:	–
Use Case:	Compilers are industry standard tools.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00250*)

2.1.4 Process Requirements

[RS_METH_00056] The AUTOSAR methodology shall not be bound to a particular life-cycle model ⌈

Type:	valid
Description:	The AUTOSAR methodology shall not be bound to a particular life-cycle model. Activities must be independent with respect to the time and phase of the development process they are executed.
Rationale:	Connection to company specific life-cycle model: The methodology shall enable the use of different life-cycle models such as e.g. V-Model, Rational Unified Process.
Dependencies:	–
Use Case:	If e.g. extreme programming is used, the test cases are created prior to the implementation. For most other development processes, the implementation is generated prior to the creation of test cases.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00507*)

[RS_METH_00069] It shall be possible to add precise and human readable documentation to each work product ⌈

Type:	valid
Description:	The methodology shall allow that precise and human readable documentation be added to each work product. This documentation shall be either part of the work product or uniquely referred.
Rationale:	This is necessary in order to document design decisions or restrictions, which cannot obviously be deduced from the formal content, e.g. from names. Such documentation will increase the traceability which is demanded by quality or safety standards.
Dependencies:	–
Use Case:	Choosing a redundancy mechanism, e.g. in the configuration for a NVRAM data block, may be related to a safety requirement. This may need verbal explanation.
Applies to:	CP,AP
Supporting Material:	–

⌋(*RS_Main_00030*, *RS_Main_00290*, *RS_Main_00300*)

2.2 Requirements for the Classic Platform

2.2.1 General Requirements

[RS_METH_00033] The methodology should support the VFB concept ⌈

Type:	valid
Description:	The Virtual Functional Bus concept allows early checks between SW-C with a complete abstraction of the hardware. The methodology should include this concept.
Rationale:	To improve the integration phases and the concurrent development.
Dependencies:	–
Use Case:	In AUTOSAR, an application is modeled as a composition of interconnected components. The VFB is the communication mechanism that allows these components to interact. Even if all the resources used by these components are not available yet during the development (HW/Network) some basic checks can be done and early problems can be solved that will ease the integration phase later.
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00060*, *RS_Main_00130*, *RS_Main_00150*, *RS_Main_00300*, *RS_Main_00400*)

[RS_METH_00208] The methodology shall explain the high-level usage of the AUTOSAR templates ⌈

Type:	valid
Description:	The AUTOSAR templates include numerous features for the system design, the software component, the ECU configuration etc. The methodology shall clearly describe the activities to use/modify these and which activities require them to be completed before commencing.
Rationale:	Usage of AUTOSAR templates in the development process
Dependencies:	–
Use Case:	Usage of the SW-C template: An existing software component is implemented and is to be integrated into a suppliers ECU. A description of the component is needed in order to correctly integrate that component into the ECU. Usage of the system template: OEM as integrator is fixing the ECUs, the topology and the list of the SW-C for one vehicle for the suppliers. Usage of the BSW module template: An existing BSW module is implemented and is to be integrated into an ECU. A description of the BSW Module is needed in order to correctly integrate that module into the ECU. The ECU configuration template is used to describe the configuration when integrating a BSW module into an ECU. The Safety Extensions allow a standardized exchange of safety information and provide the basis for consistent management among different vendors and tools as required by ISO 26262. The Diagnostic Extract Template that represents a standardized exchange format on diagnostic functionality allows the decentralized configuration of diagnostic aspects.
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00300*, *RS_Main_00150*)

[RS_METH_00080] The AUTOSAR methodology shall support the concept of implicit communication behavior ⌈

Type:	valid
Description:	The AUTOSAR methodology shall support the exchange of information to configure the Implicit Communication Behavior of the RTE according to the requirements of the Software Components. The information can be defined first time at the design of an Atomic Software Component but can be added as well if compositions are created. The design of an Atomic Software Component with respect to implicit communication behavior may be guided by blueprints of the Implicit Communication Behavior descriptions.
Rationale:	Define Implicit Communication Behavior requirements in a top down design approach
Dependencies:	–
Use Case:	–
Applies to:	CP
Supporting Material:	[RS_SWCT_03065], [RS_STDT_00034]

⌋(*RS_Main_00080*, *RS_Main_00300*)

[RS_METH_00083] The AUTOSAR methodology shall explain the description and handling of Data Exchange Points ⌈

Type:	valid

Description:	The methodology shall explain workflows for the development and the use of Data Exchange Points. E.g., it shall describe which artifacts are provided by AUTOSAR that support the development of profiles of Data Exchange Points that can be used to analyze potential tool interoperability issues or to configure validation engines of AUTOSAR tools according to the described data exchange point.
Rationale:	Tool interoperability
Dependencies:	[RS_METH_00084]
Use Case:	• AUTOSAR specifies the contents of artifacts for different steps in the methodology. • A contract is established between producing and consuming AUTOSAR tools with respect to exchanged artifacts. The producing tool assures its adherence to a an agreed profile and the consuming tool specifies its expectations using this profile.
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00301*)

[RS_METH_00084] The AUTOSAR methodology shall relate templates to a distributed development process ⌈

Type:	valid
Description:	The AUTOSAR templates specify the language for describing an AUTOSAR-based software or system. The methodology shall support the specification of a subset of the templates, which is used for a specific work product in a distributed development process.
Rationale:	Exchange of AUTOSAR artifacts in distributed development
Dependencies:	[RS_METH_00083]
Use Case:	A (VFB) system description shall only contain relevant information for the development of SW-Cs without the deployment to an ECU network yet.
Applies to:	CP
Supporting Material:	

⌋(*RS_Main_00301*)

2.2.2 Variant Handling Requirements

[RS_METH_00062] The methodology shall support configuration of parameters with different binding time. ⌈

Type:	valid
Description:	The AUTOSAR methodology shall support system development with different combinations of configuration classes.
Rationale:	Configuration of parameters can be performed in different process steps: pre-compile, link time, and post-build.

Dependencies:	–
Use Case:	OEM configuration of post-build data after a release from a Tier1 supplier. Handling information related to different configuration classes as separate configuration items (units for version control).
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00080*, *RS_Main_00360*)

[RS_METH_00074] The methodology shall specify binding times ⌈

Type:	valid
Description:	The AUTOSAR Methodology shall specify particular points in the workflow on which variation can be resolved.
Rationale:	Need for a stable reference on Binding times.
Dependencies:	–
Use Case:	During the development of an AUTOSAR System and ECU, specific variants need to be created, and eventual chosen, e.g pre-compile, or post-build.
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00360*)

[RS_METH_00075] The methodology shall specify the tasks of resolving variant ⌈

Type:	valid
Description:	The AUTOSAR Methodology shall specify particular tasks/activities in which variation will be resolved.
Rationale:	Need for clarification of methodology of variants.
Dependencies:	–
Use Case:	If two software components provide the same interface in different variants of the system, a task is needed to select the one provider to resolve that system variant.
Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00360*)

[RS_METH_00076] The methodology shall specify a work product for values of variant selectors ⌈

Type:	valid
Description:	AUTOSAR Methodology shall specify particular work products to maintain the values of variant selectors.
Rationale:	This makes it clear where the values for variant selectors are stored and maintained.
Dependencies:	–
Use Case:	The possible variants are known up front: they are created at a certain time and owned as a work product, and finally consumed when the variant is selected.

Applies to:	CP
Supporting Material:	–

⌋(*RS_Main_00360*)

2.3 Requirements for the Adaptive Platform

This section specifies requirements, which are valid for the Adaptive Platform only.

2.3.1 Main Requirements

[RS_METH_00201] The methodology shall explain how to design the services of a system ⌈

Type:	valid
Description:	The methodology shall explain how to describe services for service-oriented communication used in an Adaptive AUTOSAR system. The service interfaces consist of methods, events and fields, which need to be specified.
Rationale:	Consistent description of the information that is exchanged between applications.
Dependencies:	–
Use Case:	Specify a service interface, which consists of three events and one method.
Applies to:	AP
Supporting Material:	–

⌋(*RS_Main_00150*, *RS_Main_00060*)

[RS_METH_00206] The methodology shall explain how to configure the instances of services of a system ⌈

Type:	valid
Description:	The methodology shall explain the necessary steps for the deployment of services. This starts with the configuration of the deployment of service interfaces for the chosen network binding. The methodology shall further describe how service instances are defined and configured for a specific machine.
Rationale:	Complete description of service instances within a system.
Dependencies:	–
Use Case:	Define if service instances are required or provided as well as their search or offer criteria for service-oriented communication.
Applies to:	AP
Supporting Material:	–

⌋(*RS_Main_00505*, *RS_Main_00320*)

[RS_METH_00202] The methodology shall explain how to develop an Adaptive Application ⌈

Type:	valid
Description:	An Adaptive Application is developed based on the service interfaces. The methodology shall describe the necessary activities for first designing and then implementing the Adaptive Application.
Rationale:	Clear navigation with a description of possible development approaches for the application developer.
Dependencies:	–
Use Case:	Design a model of the software component with all necessary ports in order to use the service interfaces.
Applies to:	AP
Supporting Material:	–

⌋*(RS_Main_00080, RS_Main_00150)*

[RS_METH_00203] The methodology shall explain the high-level usage of the Manifest Specification ⌈

Type:	valid
Description:	The manifest contains all necessary information that is needed in order to integrate applications onto the Adaptive Platform. The methodology shall explain how this information will be collected, for the machine, the service instances as well as for the application itself, and later on how the manifest will be used for configuration purposes.
Rationale:	Methodology consistency using the Manifest Specification
Dependencies:	–
Use Case:	The Application Manifest is used for describing all process related aspects of an executable.
Applies to:	AP
Supporting Material:	–

⌋*(RS_Main_00503)*

[RS_METH_00207] The methodology shall explain how to develop Platform Software for the Adaptive Platform ⌈

Type:	valid
Description:	The methodology shall explain how to develop the functional clusters for an Adaptive Platform.
Rationale:	Efficient development of Adaptive Platform.
Dependencies:	–
Use Case:	Development of the Execution and Communication Management of an Adaptive Platform.
Applies to:	AP
Supporting Material:	–

⌋*(RS_Main_00002)*

[RS_METH_00204] The methodology shall describe how to configure a machine for the Adaptive Platform ⌈

Type:	valid
Description:	The methodology shall describe the different steps for defining and configuring the machine so that software can be deployed on it. This step shall be independent of other development steps in order to ensure that software can be easily uploaded later without a new configuration of the machine.
Rationale:	Deployment or updating of software without adapting machine configuration.
Dependencies:	–
Applies to:	AP
Use Case:	Configuration of all ports and IP addresses on the machine for service-oriented communication.
Supporting Material:	–

⌋(*RS_Main_00503*, *RS_Main_00002*)

[RS_METH_00205] The methodology shall describe how to deploy software on the Adaptive Platform ⌈

Type:	valid
Description:	A SW package is the smallest unit for deployment onto an Adaptive AUTOSAR Platform instance. The methodology shall describe the content of a SW package and how it is deployed on the Adaptive Platform.
Rationale:	Complete description of application development workflow until software is deployed.
Dependencies:	–
Applies to:	AP
Use Case:	Downloading and deploying a software update.
Supporting Material:	–

⌋(*RS_Main_00503*, *RS_Main_00150*)

3 Change History

3.1 Change History FO 1.4.0

3.1.1 Added Requirements in FO 1.4.0

N/A

3.1.2 Changed Requirements in FO 1.4.0

N/A

3.1.3 Deleted Requirements in FO 1.4.0

N/A

3.2 Change History FO 1.3.0

3.2.1 Added Requirements in FO 1.3.0

Id	Heading
[RS_METH_00200]	The methodology shall support building a system consisting of several AUTOSAR platforms
[RS_METH_00208]	The methodology shall explain the high-level usage of the AUTOSAR templates

Table 3.1: Added Requirements in FO 1.3.0

3.2.2 Changed Requirements in FO 1.3.0

Id	Heading
[RS_METH_00006]	The methodology shall explain how to build an AUTOSAR system
[RS_METH_00041]	The methodology shall support top-down and bottom-up approaches
[RS_METH_00016]	The methodology shall support building a system of both AUTOSAR and Non-AUTOSAR ECUs
[RS_METH_00032]	The methodology shall support different levels of abstractions
[RS_METH_00020]	The methodology shall support round-trip engineering
[RS_METH_00077]	The methodology shall support different views on the SW-C structure by OEMs and suppliers
[RS_METH_00078]	The methodology shall explain the typical usage of different views on the system of the OEM
[RS_METH_00079]	The methodology shall explain the typical usage of different views on the system of the supplier
[RS_METH_00066]	The methodology shall allow activities that reference tools

[RS_METH_00042]	The methodology shall incorporate the usage of industry standard tools
[RS_METH_00056]	The AUTOSAR methodology shall not be bound to a particular life-cycle model
[RS_METH_00069]	It shall be possible to add precise and human readable documentation to each work product
[RS_METH_00033]	The methodology should support the VFB concept
[RS_METH_00015]	The methodology shall be independent of programming languages
[RS_METH_00084]	The AUTOSAR methodology shall relate templates to a distributed development process
[RS_METH_00201]	The methodology shall explain how to design the services of a system
[RS_METH_00202]	The methodology shall explain how to develop an Adaptive Application
[RS_METH_00203]	The methodology shall explain the high-level usage of the Manifest Specification
[RS_METH_00204]	The methodology shall describe how to configure a machine for the Adaptive Platform
[RS_METH_00205]	The methodology shall describe how to deploy software on the Adaptive Platform
[RS_METH_00206]	The methodology shall explain how to configure the instances of services of a system
[RS_METH_00207]	The methodology shall explain how to develop Platform Software for the Adaptive Platform

Table 3.2: Changed Requirements in FO 1.3.0

3.2.3 Deleted Requirements in FO 1.3.0

Id	Heading
RS_METH_00017	Methodology shall clearly define what is standardized and what is not standardized
RS_METH_00002	Methodology shall explain the typical usage of SW-C template
RS_METH_00003	Methodology shall explain the typical usage of BSW Module Template
RS_METH_00004	Methodology shall explain the typical usage of the ECU Configuration template
RS_METH_00005	Methodology shall explain the typical usage of the System Template
RS_METH_00081	Methodology shall explain the typical usage of Safety Extensions
RS_METH_00082	Methodology shall explain the typical usage of Diagnostic Extract Template
RS_METH_00038	Methodology shall support the C programming language
RS_METH_00021	Methodology shall define Activities
RS_METH_00043	Activities shall have a purpose
RS_METH_00046	Activities shall have input work products
RS_METH_00047	Activities shall have output work products
RS_METH_00048	Activities shall include roles
RS_METH_00025	Methodology shall define Work products
RS_METH_00050	Work products shall have a description
RS_METH_00051	Work products shall have a reference(s) to metaclass(es) in the Autosar Metamodel
RS_METH_00052	It must be possible to avoid duplication of data in Work Products
RS_METH_00054	Work Products shall not have circular references with other work products
RS_METH_00061	Methodology shall describe the change of existing work products
RS_METH_00027	Methodology shall define unambiguous guidance terminology
RS_METH_00028	Methodology shall define Roles
RS_METH_00064	Roles shall have a description
RS_METH_00009	Methodology should be modeled

RS_METH_00010	Methodology should define rules to translate methodology model into a document
RS_METH_00057	AUTOSAR methodology shall support traceability to external artifacts
RS_METH_00067	Methodology document shall include hyperlinks between Activities, Roles, Work Products, and Guidance

Table 3.3: Deleted Requirements in FO 1.3.0

3.3 Change History FO 1.2.0

3.3.1 Added Requirements in FO 1.2.0

N/A

3.3.2 Changed Requirements in FO 1.2.0

Id	Heading
[RS_METH_00006]	Methodology shall explain how Autosar system is built
[RS_METH_00041]	Methodology shall support Bottom/Up Approach
[RS_METH_00018]	Methodology shall be modular
[RS_METH_00032]	The methodology shall respect the different levels of abstractions
[RS_METH_00020]	Methodology shall support iterations
[RS_METH_00077]	Methodology shall explain the typical interaction between OEMs and suppliers
[RS_METH_00078]	Methodology shall explain the typical usage of different views on the system of the OEM
[RS_METH_00079]	Methodology shall explain the typical usage of different views on the system of the Supplier
[RS_METH_00084]	AUTOSAR methodology shall relate templates to a distributed development process
[RS_METH_00015]	Methodology shall be independent of programming language
[RS_METH_00066]	Methodology shall support activities that reference tools
[RS_METH_00042]	Methodology shall incorporate the usage of industry standard tools
[RS_METH_00056]	AUTOSAR methodology shall not be bound to a particular lifecycle model

Table 3.4: Changed Requirements in FO 1.2.0

3.3.3 Deleted Requirements in FO 1.2.0

N/A

3.4 Change History FO 1.1.0

3.4.1 Added Requirements in FO 1.1.0

Id	Heading

[RS_METH_00201]	Methodology shall explain how to design the services of a system
[RS_METH_00202]	Methodology shall explain how to develop an Adaptive Application
[RS_METH_00203]	Methodology shall explain the high-level usage of the Manifest Specification
[RS_METH_00204]	Methodology shall describe how to configure a machine for the Adaptive Platform
[RS_METH_00205]	Methodology shall describe how to deploy software on the Adaptive Platform
[RS_METH_00206]	Methodology shall explain how to configure the instances of services of a system
[RS_METH_00207]	Methodology shall explain how to develop Platform Software for the Adaptive Platform

Table 3.5: Added Requirements in FO 1.1.0

3.4.2 Changed Requirements in FO 1.1.0

N/A

3.4.3 Deleted Requirements in FO 1.1.0

N/A