Document Title	Requirements on Software Component Template
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	212

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R24-11

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes.
2023-11-23	R23-11	AUTOSAR Release Management	• No content changes.
2022-11-24	R22-11	AUTOSAR Release Management	• No content changes.
2021-11-25	R21-11	AUTOSAR Release Management	• Uptrace to RS_Main_00050 corrected. • Editorial changes.
2020-11-30	R20-11	AUTOSAR Release Management	• No content changes.
2019-11-28	R19-11	AUTOSAR Release Management	• Changed status of RS_SWCT_03320 to valid. • Changed Document Status from Final to published.
2018-10-31	4.4.0	AUTOSAR Release Management	• Added requirement for definition of optional elements for communication.
2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes.

△

2016-11-30	4.3.0	AUTOSAR Release Management	• Added requirements for rapid prototyping support.
2015-07-31	4.2.2	AUTOSAR Release Management	• Layout update.
2014-10-31	4.2.1	AUTOSAR Release Management	• Added requirements for configuration of data transformation. • Added requirement for naming conventions.
2013-10-31	4.1.2	AUTOSAR Release Management	• Editorial changes for AR 4.1.2. • Formatting updated.
2013-03-15	4.1.1	AUTOSAR Administration	• Added requirements for extensions of the internal behavior and further concepts introduced in AR 4.1.1 • Adaptions due to the requirement on the representation of AUTOSAR requirements with respect to [TPS_STDT_00078] • Deleted superfluous requirements considered in other RS documents
2011-12-22	4.0.3	AUTOSAR Administration	Added requirements for: • Record Type subsetting • Partial networking
2009-12-18	4.0.1	AUTOSAR Administration	Added requirements for: • Variant handling • End-to-end communication protection • Documentation • Triggered events • Integrity and scaling at ports
2008-08-13	3.1.1	AUTOSAR Administration	• Legal disclaimer revised
2007-12-21	3.0.1	AUTOSAR Administration	• Document meta information extended • Small layout adaptations made
2007-01-24	2.1.15	AUTOSAR Administration	• "Advice for users" revised • "Revision Information" added

▽

△

2006-11-28	2.1.0	AUTOSAR Administration	• Legal disclaimer revised
2006-11-28	2.1.0	AUTOSAR Administration	• Initial release

Disclaimer

Contents

References

[1] Software Component Template
AUTOSAR_CP_TPS_SoftwareComponentTemplate

[2] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[3] Main Requirements
AUTOSAR_FO_RS_Main

[4] Requirements on AUTOSAR Features
AUTOSAR_CP_RS_Features

[5] Feature Definition
AUTOSAR_FeatureDefinition.pdf

[6] Specification of SW-C End-to-End Communication Protection Library
AUTOSAR_CP_SWS_E2ELibrary

1 Introduction

1.1 Scope of this document

This document collects the requirements on the `Software Component Template` (SWC-T).

The requirements collected in this document will be satisfied by the `Software Component Template` specification [1]. That document implements most of the requirements stated here.

1.2 Document Conventions

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see Standardization Template, chapter Support for Traceability ([2]).

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 ([2]).

1.3 Guidelines

Existing specifications shall be referenced (in form of a single requirement). Differences to these specifications are specified as additional requirements. All Requirements shall have the following properties:

- Redundancy
  Requirements shall not be repeated within one requirement or in other requirements.
- Clearness
  All requirements shall allow one possibility of interpretation only. Used technical terms that are not in the glossary must be defined.
- Atomicity
  Each Requirement shall only contain one requirement. A Requirement is atomic if it cannot be split up in further requirements.
- Testability
  Requirements shall be testable by analysis, review or test.
- Traceability
  The source and status of a requirement shall be visible at all times.

1.4 Requirements Tracing

The following table references the requirements specified in [3] and [4] and links to the fulfillments of these.

Requirement	Description	Satisfied by
[RS_Main_00050]	AUTOSAR shall provide an Execution Framework towards applications to implement concurrent application internal control flows	[RS_SWCT_00010]
[RS_Main_00060]	Standardized Application Communication Interface	[RS_SWCT_00020]
[RS_Main_00130]	Hardware Abstraction Layer	[RS_SWCT_00070]
[RS_Main_00180]	Intellectual Property Protection	[RS_SWCT_00120]
[RS_Main_00250]	AUTOSAR methodology shall provide a predefinition of typical roles and activities	[RS_SWCT_00160]
[RS_Main_00260]	Runtime Diagnostics Means	[RS_SWCT_00170]
[RS_Main_00280]	Standardized Automotive Communication Protocols	[RS_SWCT_03320]
[RS_Main_00310]	AUTOSAR shall support hierarchical Application Software design methods	[RS_SWCT_00190]
[RS_Main_00320]	AUTOSAR shall provide formats to specify system development	[RS_SWCT_00200]
[RS_Main_00360]	Variant Management Support	[RS_SWCT_00220]

Table 1.1: Requirements Tracing

2 Requirements

This chapter describes all requirements driving the work to define the `Software Component Template` specification [1]. The requirements trace back to `Main Requirements` [3] and `AUTOSAR Features` [4].

2.1 Category: AUTOSAR Main Requirements

This section re-defines requirements from relevant requirements defined in `Main Requirements` [3].

2.1.1 Support of ECU-communication

[RS_SWCT_00010] The Software Component Template shall support inter- and intra-ECU-communication mechanisms with high reliability

Upstream requirements: RS_Main_00050

⌈

Description:	See requirement [RS_Main_00050].
Rationale:	See requirement [RS_Main_00050].
Use Case:	See requirement [RS_Main_00050].
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_00020] The Software Component Template shall provide open and standardized software interfaces for intra-ECU and inter-ECU communication

Upstream requirements: RS_Main_00060

⌈

Description:	See requirement [RS_Main_00060].
Rationale:	See requirement [RS_Main_00060].
Use Case:	See requirement [RS_Main_00060].
Dependencies:	–
Supporting Material:	–

⌋

2.1.2 Interfaces to application software and basic software modules

[RS_SWCT_00030] The Software Component Template shall provide complete interfaces to application software and basic software modules ⌈

Description:	See requirement [Main70].
Rationale:	See requirement [Main70].
Use Case:	See requirement [Main70].
Dependencies:	–
Supporting Material:	–

⌋

2.1.3 Abstraction and Independence of the application software

[RS_SWCT_00070] The Software Component Template shall provide an abstraction of the application software from hardware

Upstream requirements: RS_Main_00130

⌈

Description:	See requirement [RS_Main_00130].
Rationale:	See requirement [RS_Main_00130].
Use Case:	See requirement [RS_Main_00130].
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_00080] The Software Component Template shall provide an independence of application software from in-vehicle communication technologies ⌈

Description:	See requirement [RS_Main_00140].
Rationale:	See requirement [RS_Main_00140].
Use Case:	See requirement [RS_Main_00140].
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_00090] The Software Component Template should provide an independence of application software from operating systems ⌈

Description:	See requirement [RS_Main_00141].
Rationale:	See requirement [RS_Main_00141].
Use Case:	See requirement [RS_Main_00141].
Dependencies:	–
Supporting Material:	–

⌋

2.1.4 Functional interface view

[RS_SWCT_00110] The Software Component Template shall provide a functional interface view of the entire system ⌈

Description:	See requirement [RS_Main_00160].
Rationale:	See requirement [RS_Main_00160].
Use Case:	See requirement [RS_Main_00160].
Dependencies:	–
Supporting Material:	–

⌋

2.1.5 Protection/unlock mechanisms for software

[RS_SWCT_00120] The Software Component Template shall provide protection/unlock mechanisms for software through appropriate services in the infrastructure

Upstream requirements: RS_Main_00180

⌈

Description:	See requirement [RS_Main_00180].
Rationale:	See requirement [RS_Main_00180].
Use Case:	See requirement [RS_Main_00180].
Dependencies:	–

▽

△

Supporting Material:	–

⌋

2.1.6 Protection of SW-Components from malicious SW-Components

[RS_SWCT_00150] The Software Component Template shall provide means to protect SW-Components from malicious SW-Components ⌈

Description:	See requirement [RS_Main_00240].
Rationale:	See requirement [RS_Main_00240].
Use Case:	See requirement [RS_Main_00240].
Dependencies:	–
Supporting Material:	–

⌋

2.1.7 Compositionality

[RS_SWCT_00160] The Software Component Template shall provide means to achieve compositionality

Upstream requirements: RS_Main_00250

⌈

Description:	See requirement [RS_Main_00250].
Rationale:	See requirement [RS_Main_00250].
Use Case:	See requirement [RS_Main_00250].
Dependencies:	–
Supporting Material:	–

⌋

2.1.8 Diagnostics during runtime, for production and services purposes

[RS_SWCT_00170] The Software Component Template shall provide diagnostics means during runtime, for production and services purposes

Upstream requirements: RS_Main_00260

⌈

Description:	See requirement [RS_Main_00260].
Rationale:	See requirement [RS_Main_00260].
Use Case:	See requirement [RS_Main_00260].
Dependencies:	–
Supporting Material:	–

⌋

2.1.9 Hierarchical design methods

[RS_SWCT_00190] The Software Component Template shall support hierarchical design methods

Upstream requirements: RS_Main_00310

⌈

Description:	See requirement [RS_Main_00310].
Rationale:	See requirement [RS_Main_00310].
Use Case:	See requirement [RS_Main_00310].
Dependencies:	–
Supporting Material:	–

⌋

2.1.10 Relations between SW components

[RS_SWCT_00200] Definitions of relations between SW components are exhaustive and formal

Upstream requirements: RS_Main_00320

⌈

Description:	See requirement [RS_Main_00320].
Rationale:	See requirement [RS_Main_00320].
Use Case:	See requirement [RS_Main_00320].
Dependencies:	–
Supporting Material:	–

⌋

2.1.11 Protection from illegal access

[RS_SWCT_00210] SW components are protected from illegal access ⌈

Description:	See requirement [RS_Main_00330].
Rationale:	See requirement [RS_Main_00330].
Use Case:	See requirement [RS_Main_00330].
Dependencies:	–
Supporting Material:	–

⌋

2.1.12 Management of vehicle diversity

[RS_SWCT_00220] The Software Component Template shall support management of vehicle diversity

Upstream requirements: RS_Main_00360

⌈

Description:	See requirement [RS_Main_00360].
Rationale:	See requirement [RS_Main_00360].
Use Case:	See requirement [RS_Main_00360].
Dependencies:	–
Supporting Material:	–

⌋

2.1.13 Naming conventions

[RS_SWCT_00230] The Software Component Template shall provide the ability to define naming conventions for public symbols ⌈

Description:	The Software Component Template shall provide the ability to define naming conventions for public symbols. This especially includes requirement ids, module abbreviations, meta data and configuration symbols used in the document of a release
Rationale:	Avoid ambiguities and name clashes inside the specification. Provide a consistent uniform presentation of meta data to the reader of the specification. Allow automatic processing of specification elements
Use Case:	
Dependencies:	–
Supporting Material:	–

⌋

The Software Component Template does not define specific naming conventions. Such naming conventions are actually defined in the AUTOSAR AISpecification [2].

2.2 Category: AUTOSAR Feature Definition Requirements

This section defines the requirements from use cases in chapter 3.2 in `Feature Definition` [5]. The referenced document is obsolete from release 4.0 of AUTOSAR, but it exists in previous releases.

2.2.1 Top-down hierarchical design

[RS_SWCT_02000] The Software Component Template shall support a top-down hierarchical design ⌈

Description:	See use case in [5], chapter 3.2.1
Rationale:	See use case in [5], chapter 3.2.1
Use Case:	See use case in [5], chapter 3.2.1
Dependencies:	[RS_SWCT_00190]
Supporting Material:	–

⌋

2.2.2 Interfaces of atomic software-components

[RS_SWCT_02010] Interfaces of atomic software-components shall be supported ⌈

Description:	See use case in [5], chapter 3.2.2
Rationale:	See use case in [5], chapter 3.2.2
Use Case:	See use case in [5], chapter 3.2.2
Dependencies:	[RS_SWCT_00020]
Supporting Material:	–

⌋

2.2.3 Bottom-up design of CompositionTypes

[RS_SWCT_02020] Bottom-up design of CompositionTypes shall be supported ⌈

Description:	See use case in [5], chapter 3.2.3
Rationale:	See use case in [5], chapter 3.2.3
Use Case:	See use case in [5], chapter 3.2.3
Dependencies:	[RS_SWCT_00160], [RS_SWCT_00200]
Supporting Material:	–

⌋

2.2.4 Specification of Communications

[RS_SWCT_02030] Specification of Communications shall be supported ⌈

Description:	See use case in [5], chapter 3.2.5
Rationale:	See use case in [5], chapter 3.2.5
Use Case:	See use case in [5], chapter 3.2.5
Dependencies:	[RS_SWCT_00010], [RS_SWCT_00020]
Supporting Material:	–

⌋

2.2.5 Interaction with basic software

[RS_SWCT_02060] Interaction with basic software shall be considered ⌈

Description:	See use case in [5], chapter 3.2.9
Rationale:	See use case in [5], chapter 3.2.9
Use Case:	See use case in [5], chapter 3.2.9
Dependencies:	[RS_SWCT_00030]
Supporting Material:	–

⌋

2.2.6 Sensor Actuator Components

[RS_SWCT_02080] Designing a Sensor Actuator Component shall be supported ⌈

Description:	See use case in [5], chapter 3.2.14
Rationale:	See use case in [5], chapter 3.2.14
Use Case:	See use case in [5], chapter 3.2.14
Dependencies:	–
Supporting Material:	–

⌋

2.2.7 Data-consistency for communication among RunnableEntities

[RS_SWCT_02090] Data-consistency for communication among RunnableEntities shall be supported ⌈

Description:	See use case in [5], chapter 3.2.20
Rationale:	See use case in [5], chapter 3.2.20
Use Case:	See use case in [5], chapter 3.2.20
Dependencies:	[RS_SWCT_00010]
Supporting Material:	–

⌋

2.2.8 Physical units

[RS_SWCT_02100] Definition of physical units shall be supported ⌈

Description:	See use case in [5], chapter 3.2.21
Rationale:	See use case in [5], chapter 3.2.21
Use Case:	See use case in [5], chapter 3.2.21
Dependencies:	–
Supporting Material:	–

⌋

2.2.9 Comments

[RS_SWCT_02110] Definition of comments shall be supported ⌈

Description:	See use case in [5], chapter 3.2.22
Rationale:	See use case in [5], chapter 3.2.22
Use Case:	See use case in [5], chapter 3.2.22
Dependencies:	–
Supporting Material:	–

⌋

2.3 Category: Software Component Template Requirements

This section defines requirements from various AUTOSAR Work Packages, e.g. WP Methodology and Configuration etc.

2.3.1 Compositions

[RS_SWCT_03000] The SW-Component template shall support compositions ⌈

Description:	The SW-Component template must allow the aggregation of existing SW-Component(s) as a component, which may also be aggregated.
Rationale:	–
Use Case:	
Dependencies:	[RS_SWCT_02020], [RS_SWCT_00160]
Supporting Material:	–

⌋

2.3.2 Interfaces

[RS_SWCT_03010] The SW-Component template shall support interfaces ⌈

Description:	The SW-Component template must allow specifying interfaces independently from the SW Components. (A definition might exist even if no component uses it.)
Rationale:	–
Use Case:	
Dependencies:	[RS_SWCT_02010], [RS_SWCT_00010], [RS_SWCT_00020]
Supporting Material:	–

⌋

2.3.3 Behavior

[RS_SWCT_03040] The SW-Component template shall support description of the behavior ⌈

Description:	The SW-Component Template must allow linking of the SW-Component to a formal description of its behavior.
Rationale:	"...components are in theory exchangeable as long as they implement the same logic and provide the same public communication..." Thus the functionality / logic of a SW-Component should be describable..
Use Case:	Exchange / compatibility of SW-components
Dependencies:	–
Supporting Material:	–

⌋

2.3.4 RTE Events

[RS_SWCT_03045] The SW-Component template shall allow enabling of RTE-Feature to get the activating RTE-Event of Runnable Entity ⌈

Description:	The Software Component Template shall provide means to request, that the RTE activates the feature to pass the activating RTE-Event to the called Runnable Entity. The request shall be available per Runnable Entity.

▽

△

Rationale:	In case the RTE-Api is not required by the Runnable Entity code it shall not be available and the generated RTE shall not keep track of the activating RTE-Events for this Runnable Entity.
Use Case:	A Runnable Entity is defined to be activated by a "TimingEvent" as well as by a "DataReceivedEvent". During the execution of the Runnable Entity the code needs to distinguish which activation source actually triggered the execution.
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03046] The SW-Component template shall support instance specific RTE-Events ⌈

Description:	The Software Component Template shall provide means to describe instance specific RTE-Events.
Rationale:	s. Use Case
Use Case:	Reuse of components in different cycles, i.e. the execution of a timing event is determined by the time base of the controller individually for each instance of a software component.
Dependencies:	–
Supporting Material:	–

⌋

2.3.5 Schedulability

[RS_SWCT_03050] The SW-Component template shall support the definition of schedulability ⌈

Description:	The SW-Component Template must contain enough information, as far as the component can provide, for generating the runtime environment and integrating multiple SW-Component(s) on one ECU or networked ECUs.
Rationale:	The information needs to be sufficient to enable/support scheduling of components
Use Case:	
Dependencies:	[RS_SWCT_00090]
Supporting Material:	–

⌋

2.3.6 Runnable Entities

[RS_SWCT_03055] The SW-Component template shall support optional configuration of ExclusiveArea usage within RunnableEntities ⌈

Description:	The software component configuration shall support specifying optional configuration information for each implemented RunnableEntitiy to describe • which ExclusiveAreas are used in a nested way by the entity and • which other RunnableEntities are invoked from within an ExclusiveArea or nested ExclusiveArea.
Rationale:	The additional configuration information can be checked by a tool at configuration time. The goal is to prevent deadlocks by providing warnings to the implementer in case of possible conflicts when resources are shared between different RunnableEntities.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03065] The Software Component Template shall support the definition of implicit communication behavior ⌈

Description:	The Software Component Template shall support the exchange of supporting information to configure the implicit communication behavior of the RTE according the requirements of the Software Components. The information can be defined first time at the design of a Atomic Software Component but can be added as well if compositions are created. For example, stable data are typically expected for Runnable Entitys of several Atomic Software Components.
Rationale:	Ensure correct environmental behavior for Software Components.
Use Case:	This can conceptually be condensed to two basic use cases: • stable data during the execution of a group of Runnable Entities • coherent data consumption and propagation for a group of Data Prototypes
Dependencies:	–
Supporting Material:	–

⌋

2.3.7 Needed and usable sensors and actuators

[RS_SWCT_03090] The SW-Component template shall support the definition of needed and usable sensors and actuators ⌈

Description:	The SW-Component Template must allow: • listing the required sensors / actuators • enumerating the sensors / actuators that can be used • referencing external descriptions of sensors / actuators for a SW-Component
Rationale:	–
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

2.3.8 Variants

[RS_SWCT_03100] The SW-Component template shall support variant handling ⌈

Description:	The SW-Component template must allow specifying the different variants of a SW-Component e.g. SW Configuration.
Rationale:	–
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

2.3.9 Modes

[RS_SWCT_03110] The SW-Component template shall support modes ⌈

Description:	The SW-Component template must provide some simple means to define modes. The name of the mode is the most important attribute that has to be provided

▽

△

Rationale:	The assumption from the SW-Component point of view is that State Managers are using a Standardized AUTOSAR Interface to influence the SW-Component and also provide an interface to get requests and confirmations from the SW-Component.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03115] The SW-Component template shall support mapping of mode declarations ⌈

Description:	The Software Component Template shall support the mapping of Mode Declarations.
Rationale:	s. Use Case
Use Case:	A receiving Software Component has to be connected to a Mode Manager providing a ModeDeclarationGroup with different ModeDeclarations than the user.
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03120] The SW-Component template shall support dependency on modes ⌈

Description:	SW-Component must provide a matrix where the runnable entities and ports are enabled or disabled depending on the modes coming from the State Managers.
Rationale:	A SW-Component can change its active interface due to the different modes provided by the State Managers. There could be a different communication behavior in the diagnostic mode than in the normal operational mode.
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03202] The SW-Component template shall support enabling SWCs to request dedicated modes ⌈

Description:	If configured, on each ECU one or several ECUs SWCs can request a mode. The mode request is propagated to a specific functionality, which is responsible to control the affected BSW according to the mode requests received.
Rationale:	See "Vehicle and Application Mode Management Concept".
Use Case:	Use cases which require a control of the basic software and runnable execution depending on mode information.
Dependencies:	[RS_SWCT_03200]
Supporting Material:	–

⌋

[RS_SWCT_03203] The SW-Component template shall support propagation of mode information ⌈

Description:	A mode can affect SWCs located on several ECUs. Therefore by configuration the mode information must be propagated to several ECUs.
Rationale:	See "Vehicle and Application Mode Management Concept".
Use Case:	All use cases which require controlling the basic software and runnable control.
Dependencies:	[RS_SWCT_03200]
Supporting Material:	–

⌋

2.3.10 Connections between PortInterfaces

[RS_SWCT_03130] The SW-Component template shall support connections between PortInterfaces ⌈

Description:	The Software Component Template shall support means to define connections between compatible PortInterfaces irrespective from the names of the PortInterface Elements.
Rationale:	Support work-share in large inter-company development groups. (Short) Names are not necessarily coordinated defined in distributed development. But Short Names are relevant for SWC implementation and shouldn't be change just for the need of defining a connection.
Use Case:	
Dependencies:	[RS_SWCT_00200]

▽

△

Supporting Material:	–

⌋

[RS_SWCT_03135] The SW-Component template shall support record type subsetting ⌈

Description:	The Software Component Template shall support that ports are connected which are typed by different interfaces and where elements of the provide port are typed by composite data types, which composite elements are mapped to elements of the require port. Hereby the require port might contain only a subset of the elements contained in the provide port.
Rationale:	Since the handling of data in a consistent manner requires using a record type, it shall be allowed at the receiver of a RecordType to only receive a subset of the sent record data elements. Since different receivers do require a different subset of the provided data.
Use Case:	4 wheel speed signals and the movement direction signal are provided in one record. If a receiver is only interested in the movement direction information all of the other information from this record does not have to be considered at this specific receiver.
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03136] The SW-Component template shall support record type subsetting with primitive types ⌈

Description:	The Software Component Template shall support that ports are connected which are typed by different interfaces and where elements of the provide port are typed by composite data types, while elements of the require port are typed by primitive types.
Rationale:	see [RS_SWCT_03135]
Use Case:	see [RS_SWCT_03135]
Dependencies:	–
Supporting Material:	–

⌋

2.3.11 Conditional existence of Prototypes due to variant handling

[RS_SWCT_03140] The SW-Component template shall support conditional existence of PortPrototypes ⌈

Description:	The AUTOSAR Meta Model shall support the description of conditional existent PortPrototypes.
Rationale:	Conditional data flow depended from system configuration.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

[RS_SWCT_03141] The SW-Component template shall support the conditional existence of data element prototypes, operation prototypes, parameter prototypes in an interface ⌈

Description:	The SWCT shall support the description of conditional existence of data element prototypes, operation prototype and parameter prototypes in interfaces.
Rationale:	Conditional data, services and calibration characteristics differ between system configurations.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

[RS_SWCT_03142] The SW-Component template shall support the conditional existence of ComponentPrototypes ⌈

Description:	The SWCT shall support the description of conditional existence ComponentPrototypes.
Rationale:	Conditional use of ComponentPrototypes depended from system configuration.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

[RS_SWCT_03143] The SW-Component template shall support the conditional existence of ConnectorPrototypes ⌈

Description:	The SWCT shall support the description of conditional existence ConnectorPrototypes.
Rationale:	Conditional data flow depended from system configuration.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.12 Configurable size of Arrays

[RS_SWCT_03144] The SW-Component template shall support a configurable size of arrays ⌈

Description:	The SWCT shall support configurable Size of Arrays. (maxNumberOfElements).
Rationale:	Smart adaptation of interfaces to the number of existing system parts.
Use Case:	Adjustment of interfaces to configurable number of cylinders or number of cylinder banks.
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.13 Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by an System Constant Definition

[RS_SWCT_03148] Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by an System Constant Definition ⌈

Description:	The Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by a System Constant Definition.
Rationale:	Adjustment of calibration axis to configurable number of cylinders or number of cylinder banks.
Use Case:	

▽

△

Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.14 Conditional existence of RunnableEntitys

[RS_SWCT_03149] The SW-Component template shall support the conditional existence of RunnableEntitys ⌈

Description:	The SWCT shall support the description of conditional existence RunnableEntitys.
Rationale:	Adaptation of algorithms depended from System Configuration
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.15 Conditional existence of RTEEvents

[RS_SWCT_03150] The SW-Component template shall support the conditional existence of RTEEvents ⌈

Description:	The SWCT shall support the description of conditional existence of RTEEvent.
Rationale:	Adaptation of algorithms depended from System Configuration.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.16 Conditional existence of InterRunnableVariables

[RS_SWCT_03151] The SW-Component template shall support the conditional existence of InterRunnableVariables ⌈

Description:	The SWCT shall support the description of conditional existent InterRunnableVariables.
Rationale:	Adaptation of algorithms depended from System Configuration.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.17 Conditional accessibility for measurement

[RS_SWCT_03152] The SW-Component template shall support the conditional accessibility for measurement ⌈

Description:	The SWCT shall support the description of conditional accessibility for external measurement tools.
Rationale:	Switch off ability of measurability of elements in later development steps for resource optimization.
Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.18 Conditional existence of parameter prototypes

[RS_SWCT_03153] The SW-Component template shall support the conditional existence of parameter prototypes ⌈

Description:	The SWCT shall support the description of conditional existent parameter prototypes.
Rationale:	Adaptation of algorithms depended from System Configuration.

▽

△

Use Case:	
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.19 Support of conditional ports for SW-C

[RS_SWCT_03154] The SW-Component template shall support conditional ports for software components ⌈

Description:	If a particular SWC is available, then the interface must be implemented, i.e. is a core interface; the presence of such elements depend only on the vehicle configuration • A Port associated to a SW-Component is said as being a Provider “Conditional” Port if a certain condition is fulfilled that justifies that the corresponding information needs to be sent by this SW - Component. Such conditions can be the presence of a certain SW –Component on board the vehicle, the presence of a certain function within a SW – Component that is present, a particular safety concept, etc ... In that case, the condition has to be indicated precisely. • A Port associated to a SW-Component is said as being a Receiver “Conditional” Port if a certain condition is fulfilled that justifies that the corresponding information needs to be received by this SW-Component. Such conditions can be the presence of a certain function within this SW-Component, etc ... In that case, the condition has to be indicated precisely.
Rationale:	
Use Case:	[VH30] Support of connection of ports with different names, [VH31] Support of core ports for SW-C.
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.20 Support of Interfaces with different resolutions

[RS_SWCT_03155] The SW-Component template shall support interfaces with different resolutions ⌈

Description:	For some signals in the MasterTable the resolution seems too high, especially in case these signals would eventually be exchanged by 2 ECU's, because then it can be very sizing for the communication network bus.
Rationale:	
Use Case:	• some Torque signals sent by a 10.2 SW-C to a 10.3 SW-C are 22 bits. It is too much for 10.3, but at the same type, this signal is probably used internally between 2 SW-C's in 10.2, which justifies their definition of 22 bits. • some 10.3 internal signals (only exchanged by 2 SW-C's within chassis domain) have a high resolution for premium cars, whereas for more "standard" cars, a lower resolution (for instance, 8 bytes instead of 16) would be "enough". As this high resolution would induce some additional costs, we cannot be satisfied with using this high resolution for these "standard" cars.
Dependencies:	[RS_SWCT_03100]
Supporting Material:	–

⌋

2.3.21 Fixed data exchange

[RS_SWCT_03170] The SW-Component template shall support fixed data exchange ⌈

Description:	Fixed data (macros) and calibration are handled by a unique parameter interfaces. Compatibility rules allow to interconnect data, fixed data, constant, calibration and NV data. Current situation: Fixed data that do not change frequently are implemented as macros. Fixed data that can be changed by tuning engineer are implemented as const calibration.
Rationale:	–
Use Case:	• UC1: the integrator wants to fix the value of an input data of a SWC ([RS_BRF_00157]) • UC2: a SWC produces a fixed data that is set in the SWC template • UC3: the integrator wants to definitively set the value of a calibration parameter
Dependencies:	–
Supporting Material:	–

⌋

2.3.22 M2 support for definition of calibration datasets

[RS_SWCT_03175] The SW-Component template shall support the definition of calibration datasets ⌈

Description:	Need to augment the common approach to specify initial values. The specification of initial values might not be limited to Calibration Parameters.It is also possible to consider initial values of DataElementPrototypes.
Rationale:	–
Use Case:	• UC1: Provide initial values for calibration parameter types. Predefine start values for the calibration process. • UC2: Provide initial values for calibration parameter instances. Utilizes the result of the calibration process. • UC3: Provide multiple sets of initial values. The initial values are often taken from previous projects. • UC4: Provide initial values in multiple domains (physical and coded). Allow to transfer values between different projects. Maintain highest accuracy within a project. • UC5: Support "'Variant coding"'. "'Variant coding"' provides multiple values for a particular calibration parameter • UC6: Provide initial values related to SWCT names. It appears that there are various name domains. To support a comprehensive methodology, initial values shall be related to SWCT names.
Dependencies:	–
Supporting Material:	–

⌋

2.3.23 Support of SAE J1939 Protocol Features

[RS_SWCT_03180] The SW-Component template shall support SAE J1939 Protocol Features ⌈

Description:	Subset of selected SAE J1939 protocol features to allow proper communication with SAE J1939 components.
Rationale:	The requested subset of SAE J1939 functionality will allow integration and communication with J1939 CAs. Many Truck OEMs needs to maintain systems with multiple networks, including SAE J1939.

▽

△

Use Case:	• Use Case A: Existing SAE J1939 off-the-shelf components can be integrated or reused. • Use Case B: SAE J1939 protocol is an industry standard in many markets. Therefore support of J1939 is mandatory for Truck OEMs.
Dependencies:	–
Supporting Material:	–

⌋

2.3.24 Need data type and access support for arrays of variable number of elements within the maximum size

[RS_SWCT_03181] The SW-Component template shall support arrays of variable number of elements within the maximum size ⌈

Description:	Add support for array of complex type, with variable number of elements (0 .. maxNumberOfElements). Arrays containing a variable number of data sets to access COM signal groups where only the first few signals are really available.
Rationale:	See Support of SAE J1939 Protocol Features
Use Case:	Use case: Signal modeling in DM1.
Dependencies:	[RS_SWCT_03180]
Supporting Material:	–

⌋

2.3.25 Need data type and access support for byte arrays of variable number of elements

[RS_SWCT_03182] The SW-Component template shall support byte arrays of variable number of elements ⌈

Description:	Add support for a primitive type byte arrays of variable size with no termination or a configurable termination. The handling shall be comparable to strings but without the zero termination semantics.
Rationale:	See Support of SAE J1939 Protocol Features
Use Case:	Signal modeling in e.g. ECUID.

▽

△

Dependencies:	[RS_SWCT_03180]
Supporting Material:	–

⌋

2.3.26 Ability to publish/specify the diagnostic capabilities and its resources of an SWC

[RS_SWCT_03190] The SW-Component template shall support the ability to publish/specify the diagnostic capabilities and its resources of an SWC ⌈

Description:	The SWC designer need to be able to publish and specify the relationship of diagnostic capabilities and its resources (e.g. ports, data elements, service needs, etc.) These diagnostic capabilities cover the following aspects: • Current value of a signal (which is represented as a VariableDataPrototype in AUTOSAR) Read only access for diagnostics testers • IOcontrol of a IO-signal (which is ... e.g. in a Sensor/Actuator SWC) Read and control access for diagnostic testers • Parameter (which is represented as a ParameterDataPrototype e.g. for CalibrationParameter) Read and Write access for diagnostic testers • DiagnosticRoutine (which is represented ...) RoutineControl access for diagnostic tester • DiagnosticMonitor (represents the detection of a DTC/event)
Rationale:	–
Use Case:	SWC designer should be able to publish/specify diagnostic capabilities and its resources (e.g. ports, data elements, service needs, etc.)
Dependencies:	–
Supporting Material:	–

⌋

2.3.27 Add support for Vehicle and Application Mode Management Concept

[RS_SWCT_03200] The SW-Component template shall support vehicle and application mode management ⌈

Description:	The SWC Template is affected by the following features: • PortGroups shall represent the concept of users of the communication management in the VFB and shall be used to group ports of a SWC that are required by the same functionality. • Enable SWCs to request dedicated Modes • Propagation of Mode Information One sub-requirement per feature is created.
Rationale:	See "Vehicle and Application Mode Management Concept".
Use Case:	See "Vehicle and Application Mode Management Concept".
Dependencies:	–
Supporting Material:	–

⌋

2.3.28 Add support for Portgroups

[RS_SWCT_03201] The SW-Component template shall support Portgroups ⌈

Description:	PortGroups shall represent the concept of users of the communication management in the VFB (internal behavior) and shall be used to group ports of a SWC that are required by the same functionality. These ports will logically be one COMM-user. A PortGroup shall communicate with the COMM as one user.
Rationale:	See "Vehicle and Application Mode Management Concept".
Use Case:	It is common use cases that a software component has some ports that have to be highly available and some ports that are required only for some special functionality. This could be realized by grouping these ports into two PortGroups
Dependencies:	[RS_SWCT_03200]
Supporting Material:	–

⌋

2.3.29 Integrity and Scaling at Ports

[RS_SWCT_03210] The SW-Component template shall support integrity and scaling at ports ⌈

Description:	Specify means to connect ports with incompatible interfaces and to use lower resolution data types for low bandwidth inter-ECU communication.
Rationale:	Be able to ensure that the port interface conversions between SWC are consistent.
Use Case:	The engineer shall be able to connect ports with incompatible interfaces without any need for explicitly specifying a conversion formula. The re-scaling of the data shall be done automatically. The engineer shall be able to select an alternative data type if data is communicated inter-ECU instead of intra-ECU. The conversion between the internal data type and the data type sent over the network shall be done automatically. Maintain integrity of re-scaled/converted data.
Dependencies:	–
Supporting Material:	–

⌋

2.3.30 Need to add application data type on top of implementation data type

[RS_SWCT_03215] The SW-Component template shall define the need to add application data type on top of implementation data type ⌈

Description:	Define two different breeds of AUTOSAR data types; ApplicationDataType and ImplementationDataType.
Use Case:	• ApplicationDataType should be used whenever something "physical" is at stake. ApplicationDataType is not limited to physical attributes, no bit-size, endianess, etc. • ImplementationDataType formalize implementation aspects of data types e.g. pointer. This is needed for e.g. debugging concepts. Can also be used for specifying data mapping. Software-Components may use a mixture of ApplicationDataTypes and ImplementationDataTypes where applicable.
Rationale:	Need to redefine the data type concept in AUTOSAR to support the usage of types on a "physical/application level" apart from the usage of types on an "implementation level".
Dependencies:	[RS_SWCT_00070], [RS_SWCT_00080]
Supporting Material:	–

⌋

2.3.31 Application data type

[RS_SWCT_03216] The SW-Component template shall support application data type ⌈

Description:	• It shall be possible to model a VFB system (without Services) completely with application types. • Application types are used to check interface compatibility. • Application types are used to describe semantics (unit, compuMethod, name) as well as data structure from the application perspective.
Rationale:	See [RS_SWCT_03215]
Use Case:	See [RS_SWCT_03215]
Dependencies:	[RS_SWCT_03215]
Supporting Material:	See [RS_SWCT_03215]

⌋

2.3.32 Implementation data type

[RS_SWCT_03217] The SW-Component template shall support implementation data type ⌈

Description:	• Implementation types represent the data types communicated between SWCs and RTE. • It shall be possible to model all Service Interfaces with implementation types only. • It shall be possible to model AR debug variables with implementation types. • It shall be possible to use implementation types in port interfaces without needing application types. • It shall be possible to model all BSW Interfaces with implementation types. • An implementation type can be specified independently of any application types.
Rationale:	See [RS_SWCT_03215]
Use Case:	See [RS_SWCT_03215]
Dependencies:	[RS_SWCT_03215]
Supporting Material:	See [RS_SWCT_03215]

⌋

2.3.33 Data Types for Primitive Data Mapping

[RS_SWCT_03218] The SW-Component template shall support data types for primitive data mapping ⌈

Description:	The Software Component Template shall provide means to define the criteria for Data Types, which can be used for the Primitive Data Mapping. The criteria shall be defined for ApplicationDataTypes and ImplementationDataTypes.
Rationale:	The System Template allows for a Primitive Data Mapping of arrays if they fulfill the UINT8 array criteria. Unfortunately there are no definitions what the criteria are which make an Application- or Implementation Data Type compatible to a UINT8 array.
Use Case:	s. Rationale
Dependencies:	–
Supporting Material:	–

⌋

2.3.34 Allow Communication Attributes on Compositions

[RS_SWCT_03220] The SW-Component template shall allow communication attributes on compositions ⌈

Description:	Specify means to connect ports with incompatible interfaces and to use lower resolution data types for low bandwidth inter-ECU communication.
Rationale:	Allow the exchange of Software Component descriptions which contain empty compositions and still describe the communication attributes.
Use Case:	Today (AR rel. 3.1) it is forbidden to attach ComSpecs to a PortPrototype belonging to a CompositionType. This is a strong restriction in use-cases where not the complete information of a system shall be exchanged (e.g. the OEM provides a description of parts of the Software (especially the communication related parts) and the Supplier does introduce his AtomicSoftwareComponents in the predefined containers provided by the OEM). It should be possible to transport the requirements on the ComSpec even when no AtomicSoftwareComponentType is available yet. When the actual AtomicSoftwareComponentType is added the ComSpec shall be attached (manually or tool supported) to the AtomicSoftwareComponentType's ▽

▽

△

	△ PortPrototypes. The RTE Generation should only consider the ComSpec at the AtomicSoftwareComponentType's PortPrototypes.
Dependencies:	[RS_SWCT_00160]
Supporting Material:	–

⌋

2.3.35 Allow Port Specific Configuration of Data Transformation Properties

[RS_SWCT_03221] The SW-Component template shall allow port specific configuration of data transformation properties ⌈

Description:	The SW-Component template shall allow port specific configuration of data transformation properties which will specify properties of the transformers used for inter-ECU communication.
Rationale:	Some properties of transformers used for inter-ECU communication are not signal but port specific.
Use Case:	The same safety related signal is received by two SW-components inside an ECU. The two SW-Components have different safety requirements with different acceptance criteria. The signal might be unsafe for one SW-C but still safe for the other SW-C.
Dependencies:	–
Supporting Material:	–

⌋

2.3.36 Error notification of data transformation

[RS_SWCT_03222] The SW-Component template shall support error notification for transformed data communication ⌈

Description:	The Software Component Template shall provide means to configure the error notification of data transformation.
Rationale:	SWCs might want to react on transformation errors.
Use Case:	Safety related SWCs react on errors during safety-checks in the transformation.

▽

△

Dependencies:	–
Supporting Material:	–

⌋

2.3.37 Enhancing the Non-Volatile (NV) memory interface

[RS_SWCT_03225] The SW-Component template shall support an enhanced non-volatile (NV) memory interface ⌈

Description:	Specify means to define and configure one NvBlockComponent that all SW Components in one ECU can use to access NvData.
Rationale:	Ensure data consistency by defining an NvBlockComponent with sender-receiver interfaces that will be provided to the SW-Components. By using one common NvBlockComponent the amount of RAM needed for keeping RAM copies of the NvData in each SW Component decreases.
Use Case:	One use case is to provide a really efficient (both memory and calculation) for Nv memory accesses from only one SW Component in the ECU. The other use case concerns providing a NvBlockComponent that can be used as Nv memory interface from many SW Components. This common Nv memory interface can ensure consistency and configuration means for all SW components in the ECU.
Dependencies:	[RS_SWCT_00030], [RS_SWCT_00070]
Supporting Material:	–

⌋

2.3.38 Documentation of M1 artifacts

[RS_SWCT_03230] The SW-Component template shall support documentation of M1 artifacts ⌈

Description:	Specify means to be able to provide documentation of M1 artifacts that supports: • structuring (e.g. paragraph, list, cross-references) • content focus • validation and identification of document parts

▽

△

Rationale:	Adding this will allow: • more efficient documentation information exchange among stakeholders • more precise and clear documentation of M1 artifacts • easier and faster integration of new team member working on artifacts • more efficient maintenance of the M1 artifacts' documentation
Use Case:	The OEM or supplier need to be able to provide a more efficient, precise and clearer documentation of M1 artifacts.
Dependencies:	[RS_SWCT_02110], [RS_SWCT_02060]
Supporting Material:	–

⌋

2.3.39 Support end-to-end communication protection

[RS_SWCT_03240] The SW-Component template shall support end-to-end communication protection ⌈

Description:	It shall be possible to assign a unique data ID to a specific communication path (in the context of an ECU) and also the selection of a E2E Profile ID shall be supported.
Rationale:	Adding these properties will allow protected end-to-end communication between SW components.
Use Case:	The OEM or supplier need to be able to specify protected end-to-end communication between SW components.
Dependencies:	[RS_SWCT_00010]
Supporting Material:	The details of the end-to-end protection in AUTOSAR are described in [6]

⌋

2.3.40 Partial Networking

[RS_SWCT_03241] The SW-Component template shall support partial networking ⌈

Description:	Software Component Template shall support the definition of Virtual Function Clusters (VFC).
Rationale:	Partial networking is supported by means of a Virtual Function Cluster (VFC) on VFB Level.
Use Case:	A VFC defines all ports, which participate in the communication of a partial network. Some ports control the VFC-behavior and other ports read out and react on the current status of the VFC.
Dependencies:	–
Supporting Material:	–

⌋

2.3.41 Bidirectional communication

[RS_SWCT_03250] The SW-Component template shall support bidirectional communication ⌈

Description:	The Software Component Template shall support ports which are expressing the combined semantic of require and provide ports. This kind of port shall support that the own software component • can access the same data element for reading and writing, • can invoke own server runnables, • can trigger own runnables incase of mode switch and • can trigger own runnables. Further on a bidirectional data flow shall be possible if such require and provide port is connected to other ports with provide, require or provide and require semantic.
Rationale:	Software algorithms requiring produced data as well as an input. NvBlockComponents do typically provide read and write access to the same data.
Use Case:	A software component produces a data which is as well input for the own algorithm for the next iteration.
Dependencies:	–
Supporting Material:	–

⌋

2.3.42 Initialization of Data

[RS_SWCT_03260] The SW-Component template shall support rule-based initialization of arrays ⌈

Description:	The Software Component Template shall provide means to define rule-based initialization of `DataPrototype`s, whose data type is of an array-nature.
Rationale:	`DataPrototype`s may require a high volume of initialization data (in terms of AUTOSAR XML). This can be improved by a rule-based initialization of arrays.
Use Case:	For example, an `ApplicationArrayDataType` that has 100 elements would need to be initialized such that for each element a dedicated initial value is provided. In the most prominent cases the majority of these elements are initialized with an identical value (e.g. 0) and only the first few elements differ in terms of initialization values.
Dependencies:	–
Supporting Material:	–

⌋

2.3.43 Instance specific timing

[RS_SWCT_03270] The SW-Component template shall support overriding the activation period time on instance level ⌈

Description:	The Software Component Template shall support the ability to override the applicable period of a time periodic runnable entity activation for particular instances of a software component.
Rationale:	Support closed loop controllers with different time base.
Use Case:	Reuse of components in different time rasters. As an example there might be a component providing a closed loop controller. This component can be applied to slow and fast control paths. As the time base of the controller is derived from the scheduling it should be possible to specify this on an instance level.
Dependencies:	–
Supporting Material:	–

⌋

2.3.44 Rapid Prototyping support

[RS_SWCT_03280] The SW-Component template shall support the description of bypass points and bypass scenarios ⌈

Description:	The Software Component Template shall support the description of bypass points and bypass scenarios at software component level. A bypass consists in reading/modifying/writing a data for testing or rapid prototyping purposes. A bypass point is a place in the logical data flow where the bypass can be implemented. A bypass scenario is composed of bypass points and represents the bypass data flow between the bypass points.
Rationale:	To support the exchange of bypass requirements between the OEM and the ECU supplier. To support re-use of bypass configuration over different projects and with different bypass tools.
Use Case:	An OEM provides to an ECU supplier the description of the bypass points that the ECU shall support. The ECU supplier configures the ECU to support the bypass points and delivers the ECU and a bypass tool to the OEM. The OEM provides to the bypass tool the description of a bypass scenario based on the bypass points available in the ECU. The bypass tool interacts with the ECU to implement the bypass scenario.
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03281] The SW-Component template shall support post-build hooking tools for rapid prototyping ⌈

Description:	The Software Component Template shall the support post-build hooking tools for rapid prototyping through the specification of the required rapid prototyping memory interface. The rapid prototyping memory interface mandates a code generation strategy that includes a write-read cycle which provides an unambiguous point at which rapid prototyping tools can modify the value subsequently used.
Rationale:	To support the use of post-build hooking the SWC must include use of the rapid prototyping memory interface.
Use Case:	.
Dependencies:	–
Supporting Material:	–

⌋

[RS_SWCT_03282] The SW-Component template shall support the description of service points and rapid prototyping scenarios ⌈

Description:	The Software Component Template shall support the description of service points and associated rapid prototyping scenarios at RTE Event level within a software component. A service point is a location where one or more rapid prototyping service functions provided by an rapid prototyping service component are invoked. An rapid prototyping service function is an invocation of a function provided by a rapid prototyping service component where data is sampled and/or stimulated. An rapid prototyping service component is an AUTOSAR or vendor specific BSW module providing an rapid prototyping service. An rapid prototyping scenario is composed of service points and represents the data flow between the service points.
Rationale:	To support the use of service-based bypass the SWC must include manually assigned service points which must be documented in the SWCT.
Use Case:	An OEM provides to an ECU supplier the description of the service points that the ECU shall support for a SWC. The ECU supplier configures the ECU to support the service points and delivers the ECU and a service component to the OEM.
Dependencies:	–
Supporting Material:	–

⌋

2.3.45 Initialization of Runnables

[RS_SWCT_03290] The SW-Component template shall support the initialization of runnables without usage of mode management ⌈

Description:	The Software Component Template shall provide means to define a mechanism for initialization of runnables without using the AUTOSAR Mode Management feature.
Rationale:	This requirement has been realized by tool vendors without being standardized by AUTOSAR.
Use Case:	A function developer delivers a set of runnables, where some of them are created for initialization purposes. This shall be described without using the AUTOSAR Mode Management feature, since this would introduce a new level of complexity for this simple circumstance.
Dependencies:	–
Supporting Material:	–

⌋

2.3.46 Diagnostics over IP

[RS_SWCT_03310] The SW-Component template shall support Diagnostics over IP ⌈

Description:	The Software Component Template shall provide means to define service needs for diagnostics over IP.
Rationale:	–
Use Case:	–
Dependencies:	–
Supporting Material:	–

⌋

2.3.47 Optional Elements

[RS_SWCT_03320] The SW-Component template shall support the definition of optional elements for communication

Upstream requirements: RS_Main_00280

⌈

Description:	The Software Component Template shall provide means to define optional elements in data structures used for communication on the VFB
Rationale:	The existence of optional elements in composite data structures is semantically different from the receiver simply taking an initial value if the received information does not contain a respective sub-element of a composite data structure. The receiver is supposed to actively acknowledge the fact that certain information is missing and still be able to deal with this fact in a meaningful way.
Use Case:	A composite data structure is defined and used for communication on the VFB. The design of the data structure already foresees the possibility that, at any given time, the data structure may only be partially filled with meaningful information.
Dependencies:	–
Supporting Material:	–

⌋

3 Change history of AUTOSAR traceable items

3.1 Traceable item history of this document according to AUTOSAR Release R4.0.1

3.1.1 Added Specification Items

Number	Heading
[RS_SWCT_00010]	AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability
[RS_SWCT_00020]	AUTOSAR shall provide open and standardized software interfaces for intra-ECU and inter-ECU communication
[RS_SWCT_00030]	AUTOSAR shall provide complete interfaces to application software and basic software modules
[RS_SWCT_00040]	AUTOSAR shall ease the re-usability of software and its concepts and implementations
[RS_SWCT_00050]	AUTOSAR shall provide a software architecture that is applicable across different functional domains
[RS_SWCT_00070]	AUTOSAR shall provide an abstraction of the application software from hardware
[RS_SWCT_00080]	AUTOSAR shall provide an independency of application software from in-vehicle communication technologies
[RS_SWCT_00090]	AUTOSAR should provide an independency of application software from operating systems
[RS_SWCT_00110]	AUTOSAR shall provide a functional interface view of the entire system
[RS_SWCT_00120]	AUTOSAR shall provide protection/unlock mechanisms for software through appropriate services in the infrastructure
[RS_SWCT_00130]	AUTOSAR shall provide interoperability with legacy software
[RS_SWCT_00150]	AUTOSAR shall provide means to protect SW-Components from malicious SW-Components
[RS_SWCT_00160]	AUTOSAR shall provide means to achieve compositionality
[RS_SWCT_00170]	AUTOSAR shall provide diagnostics means during runtime, for production and services purposes
[RS_SWCT_00190]	AUTOSAR shall support hierarchical design methods
[RS_SWCT_00200]	Definitions of relations between SW components are exhaustive and formal
[RS_SWCT_00210]	SW components are protected from illegal access
[RS_SWCT_00220]	Management of vehicle diversity is supported by AUTOSAR
[RS_SWCT_02000]	Top-down hierarchical design
[RS_SWCT_02010]	Interfaces of atomic software-components
[RS_SWCT_02020]	Bottom-up design of CompositionTypes
[RS_SWCT_02030]	Specification of Communications
[RS_SWCT_02050]	Specification of timing resources for software-component description
[RS_SWCT_02060]	Consider interaction with basic software
[RS_SWCT_02080]	Designing a Sensor Actuator Component
[RS_SWCT_02090]	Data-consistency for communication among RunnableEntities
[RS_SWCT_02100]	Definition of physical units
[RS_SWCT_02110]	Definition of comments
[RS_SWCT_03130]	Connections between PortInterfaces
[RS_SWCT_03140]	Conditional existence of PortPrototypes
[RS_SWCT_03141]	Conditional existence of data element prototypes, operation prototypes, parameter prototypes in an interface
[RS_SWCT_03142]	Conditional existence of ComponentPrototypes

[RS_SWCT_03143]	Conditional existence of ConnectorPrototypes
[RS_SWCT_03144]	Configurable size of Arrays
[RS_SWCT_03145]	Describe supported combinations of System Constant Value of an Software Component Type
[RS_SWCT_03146]	Describe supported combinations of System Constant Value of an InternalBehavior
[RS_SWCT_03147]	Describe supported combinations of System Constant Value of an Implementation
[RS_SWCT_03148]	Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by an System Constant Definition
[RS_SWCT_03149]	Conditional existence of RunnableEntitys
[RS_SWCT_03150]	Conditional existence of RTEEvents
[RS_SWCT_03151]	Conditional existence of InterRunnableVariables
[RS_SWCT_03152]	Conditional accessibility for measurement
[RS_SWCT_03153]	Conditional existence of parameter prototypes
[RS_SWCT_03154]	Support of conditional ports for SW-C
[RS_SWCT_03155]	Support of Interfaces with different resolutions
[RS_SWCT_03170]	Fixed data exchange
[RS_SWCT_03175]	M2 support for definition of calibration datasets
[RS_SWCT_03180]	Support of SAE J1939 Protocol Features
[RS_SWCT_03181]	Need data type and access support for arrays of variable number of elements within the maximum size
[RS_SWCT_03182]	Need data type and access support for byte arrays of variable number of elements
[RS_SWCT_03190]	Ability to publish/specify the diagnostic capabilities and its resources of an SWC
[RS_SWCT_03200]	Add support for Vehicle and Application Mode Management Concept
[RS_SWCT_03201]	Add support for Portgroups
[RS_SWCT_03202]	Enable SWCs to request dedicated Modes
[RS_SWCT_03203]	Propagation of Mode Information
[RS_SWCT_03210]	Integrity and Scaling at Ports
[RS_SWCT_03215]	Need to add application data type on top of implementation data type
[RS_SWCT_03216]	Application data type
[RS_SWCT_03217]	Implementation data type
[RS_SWCT_03220]	Allow Communication Attributes on Compositions
[RS_SWCT_03225]	Enhancing the Non-Volatile (NV) memory interface
[RS_SWCT_03230]	Documentation of M1 artifacts
[RS_SWCT_03240]	Support end-to-end communication protection

Table 3.1: Added Specification Items in 4.0.1

3.1.2 Changed Specification Items

N/A

3.1.3 Deleted Specification Items

N/A

3.2 Traceable item history of this document according to AUTOSAR Release R4.0.2

3.2.1 Added Specification Items

N/A

3.2.2 Changed Specification Items

N/A

3.2.3 Deleted Specification Items

N/A

3.3 Traceable item history of this document according to AUTOSAR Release R4.0.3

3.3.1 Added Specification Items

Number	Heading
[RS_SWCT_03135]	Record Type Subsetting
[RS_SWCT_03136]	Record Type Subsetting with Primitive Types
[RS_SWCT_03241]	Support for partial networking

Table 3.2: Added Specification Items in 4.0.3

3.3.2 Changed Specification Items

N/A

3.3.3 Deleted Specification Items

N/A

3.4 Traceable item history of this document according to AUTOSAR Release R4.1.1

3.4.1 Added Specification Items

Number	Heading
[RS_SWCT_03045]	The SW-Component template shall allow enabling of RTE-Feature to get the activating RTE-Event of Runnable Entity
[RS_SWCT_03046]	The SW-Component template shall support instance specific RTE-Events
[RS_SWCT_03055]	The SW-Component template shall support optional configuration of ExclusiveArea usage within RunnableEntities
[RS_SWCT_03065]	The SW-Component template shall support the definition of implicit communication behavior
[RS_SWCT_03115]	The SW-Component template shall support mapping of mode declarations
[RS_SWCT_03218]	The SW-Component template shall support data types for primitive data mapping
[RS_SWCT_03250]	The SW-Component template shall support bidirectional communication
[RS_SWCT_03260]	The SW-Component template shall support rule-based initialization of arrays
[RS_SWCT_03270]	The SW-Component template shall support overriding the activation period time on instance level
[RS_SWCT_03280]	The SW-Component template shall support the description of bypass points and bypass scenarios
[RS_SWCT_03290]	The SW-Component template shall support the initialization of runnables without usage of mode management
[RS_SWCT_03310]	The SW-Component template shall support Diagnostics over IP

Table 3.3: Added Specification Items in 4.1.1

3.4.2 Changed Specification Items

N/A

3.4.3 Deleted Specification Items

Number	Heading
[RS_SWCT_00040]	AUTOSAR shall ease the re-usability of software and its concepts and implementations
[RS_SWCT_00050]	AUTOSAR shall provide a software architecture that is applicable across different functional domains
[RS_SWCT_00130]	AUTOSAR shall provide interoperability with legacy software
[RS_SWCT_02050]	Specification of timing resources for software-component description
[RS_SWCT_03020]	Libraries
[RS_SWCT_03030]	Integration on object code level
[RS_SWCT_03060]	Sequence of execution of runnable entities
[RS_SWCT_03070]	Needed resources for SW Components
[RS_SWCT_03080]	Timing-requirements of SW-Components
[RS_SWCT_03145]	Describe supported combinations of System Constant Value of an Software Component Type
[RS_SWCT_03146]	Describe supported combinations of System Constant Value of an InternalBehavior
[RS_SWCT_03147]	Describe supported combinations of System Constant Value of an Implementation

Table 3.4: Deleted Specification Items in 4.1.1

3.5 Traceable item history of this document according to AUTOSAR Release R4.1.2

3.5.1 Added Specification Items

N/A

3.5.2 Changed Specification Items

N/A

3.5.3 Deleted Specification Items

N/A

3.6 Traceable item history of this document according to AUTOSAR Release R4.2.1

3.6.1 Added Specification Items

Number	Heading
[RS_SWCT_00230]	The Software Component Template shall provide the ability to define naming conventions for public symbols
[RS_SWCT_03221]	The SW-Component template shall allow port specific configuration of data transformation properties
[RS_SWCT_03222]	The SW-Component template shall support error notification for transformed data commmunication

Table 3.5: Added Specification Items in 4.2.1

3.6.2 Changed Specification Items

N/A

3.6.3 Deleted Specification Items

N/A

3.7 Traceable item history of this document according to AUTOSAR Release R4.2.2

N/A

3.8 Traceable item history of this document according to AUTOSAR Release R4.3.0

3.8.1 Added Specification Items

Number	Heading
[RS_SWCT_03281]	The SW-Component template shall support post-build hooking tools for rapid prototyping
[RS_SWCT_03282]	The SW-Component template shall support the description of service points and rapid prototyping scenarios

Table 3.6: Added Specification Items in 4.3.0

3.8.2 Changed Specification Items

N/A

3.8.3 Deleted Specification Items

N/A

3.9 Traceable item history of this document according to AUTOSAR Release R4.3.1

3.9.1 Added Specification Items

N/A

3.9.2 Changed Specification Items

N/A

3.9.3 Deleted Specification Items

N/A

3.10 Traceable item history of this document according to AUTOSAR Release R4.4.0

3.10.1 Added Specification Items

Number	Heading
[RS_SWCT_03320]	The SW-Component template shall support the definition of optional elements for communication

Table 3.7: Added Specification Items in 4.4.0

3.10.2 Changed Specification Items

N/A

3.10.3 Deleted Specification Items

N/A

3.11 Traceable item history of this document according to AUTOSAR Release R19-11

3.11.1 Added Specification Items

N/A

3.11.2 Changed Specification Items

Number	Heading
[RS_SWCT_03320]	The SW-Component template shall support the definition of optional elements for communication

Table 3.8: Changed Specification Items in 19-11

3.11.3 Deleted Specification Items

N/A

3.12 Traceable item history of this document according to AUTOSAR Release R20-11

3.12.1 Added Requirements in R20-11

none

3.12.2 Changed Requirements in R20-11

none

3.12.3 Deleted Requirements in R20-11

none

3.13 Traceable item history of this document according to AUTOSAR Release R21-11

3.13.1 Added Requirements in R21-11

none

3.13.2 Changed Requirements in R21-11

Number	Heading
[RS_SWCT_00010]	AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability
[RS_SWCT_03065]	The SW-Component template shall support the definition of implicit communication behavior

Table 3.9: Changed Requirements in R21-11

3.13.3 Deleted Requirements in R21-11

none

3.14 Traceable item history of this document according to AUTOSAR Release R22-11

3.14.1 Added Requirements in R22-11

none

3.14.2 Changed Requirements in R22-11

Number	Heading
[RS_SWCT_00010]	The Software Component Template shall support inter- and intra-ECU-communication mechanisms with high reliability
[RS_SWCT_00020]	The Software Component Template shall provide open and standardized software interfaces for intra-ECU and inter-ECU communication
[RS_SWCT_00030]	The Software Component Template shall provide complete interfaces to application software and basic software modules
[RS_SWCT_00070]	The Software Component Template shall provide an abstraction of the application software from hardware
[RS_SWCT_00080]	The Software Component Template shall provide an independence of application software from in-vehicle communication technologies
[RS_SWCT_00090]	The Software Component Template should provide an independence of application software from operating systems
[RS_SWCT_00110]	The Software Component Template shall provide a functional interface view of the entire system
[RS_SWCT_00120]	The Software Component Template shall provide protection/unlock mechanisms for software through appropriate services in the infrastructure
[RS_SWCT_00150]	The Software Component Template shall provide means to protect SW-Components from malicious SW-Components
[RS_SWCT_00160]	The Software Component Template shall provide means to achieve compositionality
[RS_SWCT_00170]	The Software Component Template shall provide diagnostics means during runtime, for production and services purposes
[RS_SWCT_00190]	The Software Component Template shall support hierarchical design methods
[RS_SWCT_00220]	The Software Component Template shall support management of vehicle diversity
[RS_SWCT_02000]	The Software Component Template shall support a top-down hierarchical design
[RS_SWCT_03065]	The Software Component Template shall support the definition of implicit communication behavior

Table 3.10: Changed Requirements in R22-11

3.14.3 Deleted Requirements in R22-11

none

3.15 Traceable item history of this document according to AUTOSAR Release R23-11

3.15.1 Added Requirements in R23-11

none

3.15.2 Changed Requirements in R23-11

none

3.15.3 Deleted Requirements in R23-11

none

3.16 Traceable item history of this document according to AUTOSAR Release R24-11

3.16.1 Added Requirements in R24-11

none

3.16.2 Changed Requirements in R24-11

none

3.16.3 Deleted Requirements in R24-11

none