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

Document Version	2.1.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Description
24.08.2011	2.1.0	AUTOSAR Administration	Added requirements for: • Record Type subsetting • Partial networking
30.11.2009	2.0.0	AUTOSAR Administration	Added requirements for: • Variant handling • End-to-end communication protection • Documentation • Triggered events • Integrity and scaling at ports
23.06.2008	1.0.4	AUTOSAR Administration	Legal disclaimer revised
31.10.2007	1.0.3	AUTOSAR Administration	• Document meta information extended • Small layout adaptations made
24.01.2007	1.0.2	AUTOSAR Administration	• "Advice for users" revised • "Revision Information" added

28.11.2006	1.0.1	AUTOSAR Administration	Legal disclaimer revised
16.08.2006	1.0.0	AUTOSAR Administration	Initial release

Table of Contents

References

[1] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate.pdf

[2] Requirements on Standardization Template
AUTOSAR_RS_StandardizationTemplate.pdf

[3] Main Requirements
AUTOSAR_RS_MainRequirements.pdf

[4] Feature Definition
AUTOSAR_FeatureDefinition.pdf

[5] Specification of SW-C End-to-End Communication Protection Library
AUTOSAR_SWS_E2ELibrary.pdf

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 Conventions used

Each requirement is defined as a table. The structure of the tables is as follows:

Initiator:	Initiator (e.g. WP Methodology and Configuration)
Date:	Date of last change
Requirement:	Short description (same as above)
Description:	Detailed description
Rationale:	Why is this requirement important, what its omission could cause?
Use Case:	A scenario that makes the requirement necessary or useful
Dependencies:	References to other requirements which this requirement depends on
Conflicts:	References to other requirements which this requirement is in conflict with
Supporting Material:	References to other documents, models etc.
Comment:	Comments

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. Note that the requirement level of the document in which they are used modifies the force of these words.

- MUST: This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.
- MUST NOT: This phrase, or the phrase "SHALL NOT", means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item.

An implementation, which does not include a particular option, MUST be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, MUST be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides.)

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 [2] and links to the fulfillments of these.

Requirement	Description	Satisfied by
[Main110]	No description	[RS_SWCT_0050]
[Main130]	No description	[RS_SWCT_0070]
[Main140]	No description	[RS_SWCT_0080]
[Main141]	No description	[RS_SWCT_0090]
[Main160]	No description	[RS_SWCT_0110]
[Main180]	No description	[RS_SWCT_0120]
[Main190]	No description	[RS_SWCT_0130]
[Main240]	No description	[RS_SWCT_0150]
[Main250]	No description	[RS_SWCT_0160]
[Main260]	No description	[RS_SWCT_0170]
[Main310]	No description	[RS_SWCT_0190]
[Main320]	No description	[RS_SWCT_0200]
[Main330]	No description	[RS_SWCT_0210]
[Main360]	No description	[RS_SWCT_0220]
[Main50]	No description	[RS_SWCT_0010]
[Main60]	No description	[RS_SWCT_0020]
[Main70]	No description	[RS_SWCT_0030]
[Main80]	No description	[RS_SWCT_0040]

2 Requirements

This chapter describes all requirements driving the work to define the `Software Component Template` specification [1].

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_0010] AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main50].
Rationale:	See requirement [Main50].
Use Case:	See requirement [Main50].
Dependencies:	[Main50]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main50)*

[RS_SWCT_0020] AUTOSAR shall provide open and standardized software interfaces for intra-ECU and inter-ECU communication ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main60].
Rationale:	See requirement [Main60].
Use Case:	See requirement [Main60].
Dependencies:	[Main60]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main60)*

2.1.2 Interfaces to application software and basic software modules

[RS_SWCT_0030] AUTOSAR shall provide complete interfaces to application software and basic software modules ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main70].
Rationale:	See requirement [Main70]
Use Case:	See requirement [Main70].
Dependencies:	[Main70]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main70)*

2.1.3 Re-usability

[RS_SWCT_0040] AUTOSAR shall ease the re-usability of software and its concepts and implementations ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main80].
Rationale:	See requirement [Main80].
Use Case:	See requirement [Main80].
Dependencies:	[Main80]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main80)*

2.1.4 Different functional domains

[RS_SWCT_0050] AUTOSAR shall provide a software architecture that is applicable across different functional domains ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main110].
Rationale:	See requirement [Main110].
Use Case:	See requirement [Main110].
Dependencies:	[Main110]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋(*Main110*)

2.1.5 Abstraction and Independency of the application software

[RS_SWCT_0070] AUTOSAR shall provide an abstraction of the application software from hardware ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main130].
Rationale:	See requirement [Main130].
Use Case:	See requirement [Main130].
Dependencies:	[Main130]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋(*Main130*)

[RS_SWCT_0080] AUTOSAR shall provide an independency of application software from in-vehicle communication technologies ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main140].
Rationale:	See requirement [Main140].
Use Case:	See requirement [Main140].
Dependencies:	[Main140]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋(*Main140*)

[RS_SWCT_0090] AUTOSAR should provide an independency of application software from operating systems ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main141].
Rationale:	See requirement [Main141].
Use Case:	See requirement [Main141].
Dependencies:	[Main141]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋(*Main141*)

2.1.6 Functional interface view

[RS_SWCT_0110] AUTOSAR shall provide a functional interface view of the entire system ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main160].
Rationale:	See requirement [Main160].
Use Case:	See requirement [Main160].
Dependencies:	[Main160]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋(*Main160*)

2.1.7 Protection/unlock mechanisms for software

[RS_SWCT_0120] AUTOSAR shall provide protection/unlock mechanisms for software through appropriate services in the infrastructure ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main180].
Rationale:	See requirement [Main180].
Use Case:	See requirement [Main180].
Dependencies:	[Main180]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋(*Main180*)

2.1.8 Interoperability with legacy software

[RS_SWCT_0130] AUTOSAR shall provide interoperability with legacy software ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main190].
Rationale:	See requirement [Main190].
Use Case:	See requirement [Main190].
Dependencies:	[Main190]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main190)*

2.1.9 Protection of SW-Components from malicious SW-Components

[RS_SWCT_0150] AUTOSAR shall provide means to protect SW-Components from malicious SW-Components ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main240].
Rationale:	See requirement [Main240].
Use Case:	See requirement [Main240].
Dependencies:	[Main240]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main240)*

2.1.10 Compositionality

[RS_SWCT_0160] AUTOSAR shall provide means to achieve compositionality ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main250].
Rationale:	See requirement [Main250].
Use Case:	See requirement [Main250].
Dependencies:	[Main250]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main250)*

2.1.11 Diagnostics during runtime, for production and services purposes

[RS_SWCT_0170] AUTOSAR shall provide diagnostics means during runtime, for production and services purposes ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main260].
Rationale:	See requirement [Main260].
Use Case:	See requirement [Main260].
Dependencies:	[Main260]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main260)*

2.1.12 Hierarchical design methods

[RS_SWCT_0190] AUTOSAR shall support hierarchical design methods ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main310].
Rationale:	See requirement [Main310].
Use Case:	See requirement [Main310].
Dependencies:	[Main310]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main310)*

2.1.13 Relations between SW components

[RS_SWCT_0200] Definitions of relations between SW components are exhaustive and formal ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main320].
Rationale:	See requirement [Main320].
Use Case:	See requirement [Main320].
Dependencies:	[Main320]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋*(Main320)*

2.1.14 Protection from illegal access

[RS_SWCT_0210] SW components are protected from illegal access ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main330].
Rationale:	See requirement [Main330].
Use Case:	See requirement [Main330].
Dependencies:	[Main330]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main330)*

2.1.15 Management of vehicle diversity

[RS_SWCT_0220] Management of vehicle diversity is supported by AUTOSAR ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See requirement [Main360].
Rationale:	See requirement [Main360].
Use Case:	See requirement [Main360].
Dependencies:	[Main360]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋*(Main360)*

2.2 Category: AUTOSAR Feature Definition Requirements

This section defines the requirements from use cases in chapter 3.2 in `Feature Definition` [4]. 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_2000] Top-down hierarchical design ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.1
Rationale:	See use case in [4], chapter 3.2.1
Use Case:	See use case in [4], chapter 3.2.1
Dependencies:	[FDUC 3.2.1], [RS_SWCT_0190]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.2 Interfaces of atomic software-components

[RS_SWCT_2010] Interfaces of atomic software-components ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.2
Rationale:	See use case in [4], chapter 3.2.2
Use Case:	See use case in [4], chapter 3.2.2
Dependencies:	[FDUC 3.2.2], [RS_SWCT_0020]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.3 Bottom-up design of CompositionTypes

[RS_SWCT_2020] Bottom-up design of CompositionTypes ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23

Description:	See use case in [4], chapter 3.2.3
Rationale:	See use case in [4], chapter 3.2.3
Use Case:	See use case in [4], chapter 3.2.3
Dependencies:	[FDUC 3.2.3], [RS_SWCT_0160], [RS_SWCT_0200]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.4 Specification of Communications

[RS_SWCT_2030] Specification of Communications ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.5
Rationale:	See use case in [4], chapter 3.2.5
Use Case:	See use case in [4], chapter 3.2.5
Dependencies:	[FDUC 3.2.5], [RS_SWCT_0010], [RS_SWCT_0020]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.5 Specification of timing resources

[RS_SWCT_2050] Specification of timing resources for software-component description ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.8
Rationale:	See use case in [4], chapter 3.2.8
Use Case:	See use case in [4], chapter 3.2.8
Dependencies:	[FDUC 3.2.8]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.6 Interaction with basic software

[RS_SWCT_2060] Consider interaction with basic software ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.9
Rationale:	See use case in [4], chapter 3.2.9
Use Case:	See use case in [4], chapter 3.2.9
Dependencies:	[FDUC 3.2.9], [RS_SWCT_0030]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.7 Sensor Actuator Components

[RS_SWCT_2080] Designing a Sensor Actuator Component ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Requirement:	Designing a Sensor Actuator Component
Description:	See use case in [4], chapter 3.2.14
Rationale:	See use case in [4], chapter 3.2.14
Use Case:	See use case in [4], chapter 3.2.14
Dependencies:	[FDUC 3.2.14]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.8 Data-consistency for communication among RunnableEntities

[RS_SWCT_2090] Data-consistency for communication among RunnableEntities ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.20
Rationale:	See use case in [4], chapter 3.2.20
Use Case:	See use case in [4], chapter 3.2.20
Dependencies:	[FDUC 3.2.20], [RS_SWCT_0010]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋

2.2.9 Physical units

[RS_SWCT_2100] Definition of physical units ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.21
Rationale:	See use case in [4], chapter 3.2.21
Use Case:	See use case in [4], chapter 3.2.21
Dependencies:	[FDUC 3.2.21]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.2.10 Comments

[RS_SWCT_2110] Definition of comments ⌈

Initiator:	AUTOSAR PL Team
Date:	2008-04-23
Description:	See use case in [4], chapter 3.2.22
Rationale:	See use case in [4], chapter 3.2.22
Use Case:	See use case in [4], chapter 3.2.22
Dependencies:	[FDUC 3.2.22]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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_3000] Compositions ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
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_2020], [RS_SWCT_0160]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.2 Interfaces

[RS_SWCT_3010] Interfaces ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
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_2010], [RS_SWCT_0010], [RS_SWCT_0020]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.3 Libraries

[RS_SWCT_3020] Libraries ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The SW-Component Template must allow specifying the libraries used by the SW-Component.
Rationale:	–
Use Case:	It should be possible to specify which C-libraries that are used by the implementation of the component, e.g. math.h, string.h, etc.

Dependencies:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.4 Integration on object code level

[RS_SWCT_3030] Integration on object code level ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The Software Component Template must contain the right information to allow the integration of the corresponding runnable entity on object code level.
Rationale:	This requirement is important for all AUTOSAR members providing code to others, which shall be protected against unauthorized modification.
Use Case:	
Dependencies:	–
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.5 Behavior

[RS_SWCT_3040] Behavior ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
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:	
Conflicts:	–
Supporting Material:	
Comment:	–

⌋

2.3.6 Schedulability

[RS_SWCT_3050] Schedulability ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The SW-Component Template must content 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_0090]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.7 Sequence of execution of runnable entities

[RS_SWCT_3060] Sequence of execution of runnable entities ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The template must allow specifying: Constraint on the order of execution of different runnable entities
Rationale:	–
Use Case:	
Dependencies:	[RS_SWCT_0090]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.8 Needed resources for SW Components

[RS_SWCT_3070] Needed resources for SW Components ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The SW-Component template must provide mechanisms to describe the complete resource requirements of an AUTOSAR SW-component

Rationale:	The AUTOSAR Software-Components ("above" the RTE) will be described in such detail that it is possible to analyze whether the software-component can run on a given ECU (for example: the software-component description specifies how many "resources" the component requires). The SW-Component template must allow specifying the memory consumption of a SW Component: • ROM • NVRAM • RAM • Stack • Heap The SW-Component template must allow describing its execution time (time during which the CPU is executing its instructions)
Use Case:	
Dependencies:	–
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.9 Timing-requirements of SW-Components

[RS_SWCT_3080] Timing-requirements of SW-Components ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The SW-Component template must allow specifying the timing-requirements of each runnable entity of a SW-Component (e.g.): • Period • Reaction time
Rationale:	The SW-Component template must allow describing: • how often it has to be run • the time between a stimulus like e.g. the state change of a hardware or software entity and the expected reaction of the system (e.g. response, actuator activation)
Use Case:	
Dependencies:	[RS_SWCT_2050]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.10 Needed and usable sensors and actuators

[RS_SWCT_3090] Needed and usable sensors and actuators ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.11 Variants

[RS_SWCT_3100] Support of variant handling ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
Description:	The SW-Component template must allow specifying the different variants of a SW-Component e.g. SW Configuration.
Rationale:	–
Use Case:	
Dependencies:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.12 Modes

[RS_SWCT_3110] Support of modes ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23

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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3120] Dependency on modes ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-04-23
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:	–
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.13 Connections between PortInterfaces

[RS_SWCT_3130] Connections between PortInterfaces ⌈

Initiator:	WP Methodology and Configuration
Date:	2008-05-20
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_0200]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3135] Record Type Subsetting ⌈

Initiator:	WP 1.2
Date:	2011-08-23
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3136] Record Type Subsetting with Primitive Types ⌈

Initiator:	WP 1.2
Date:	2011-08-23
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_3135]
Use Case:	see [RS_SWCT_3135]
Dependencies:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.14 Conditional existence of Prototypes due to variant handling

[RS_SWCT_3140] Conditional existence of PortPrototypes ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3141] Conditional existence of data element prototypes, operation prototypes, parameter prototypes in an interface. ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3142] Conditional existence of ComponentPrototypes ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3143] Conditional existence of ConnectorPrototypes ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of conditional existence ConnectorPrototypes.
Rationale:	Conditional data flow depended from system configuration.
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.15 Configurable size of Arrays

[RS_SWCT_3144] Configurable size of Arrays ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.16 Supported combinations of System Constant Value

[RS_SWCT_3145] Describe supported combinations of System Constant Value of an Software Component Type ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of the allowed combinations of System Constant Values of a SoftwareComponentType.
Rationale:	Avoid illegal configuration of a software components, Enable selection of an appropriate SoftwareComponentType.
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–

Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3146] Describe supported combinations of System Constant Value of an InternalBehavior ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of the allowed combinations of System Constant Values of an InternalBehavior.
Rationale:	Avoid illegal configuration of a Software Components Enable selection of an appropriate InternalBehavior.
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

[RS_SWCT_3147] Describe supported combinations of System Constant Value of an Implementation ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of the allowed combinations of System Constant Values of an Implementation.
Rationale:	Avoid illegal configuration of a Software Components Enable selection of an appropriate Implementation.
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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

[RS_SWCT_3148] Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by an System Constant Definition ⌈

Initiator:	Continental
Date:	2008-11-14

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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.18 Conditional existence of RunnableEntitys

[RS_SWCT_3149] Conditional existence of RunnableEntitys ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of conditional existence RunnableEntitys.
Rationale:	Adaptation of algorithms depended from System Configuration
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.19 Conditional existence of RTEEvents

[RS_SWCT_3150] Conditional existence of RTEEvents ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.20 Conditional existence of InterRunnableVariables

[RS_SWCT_3151] Conditional existence of InterRunnableVariables ⌈

Initiator:	Continental
Date:	2008-11-14
Description:	The SWCT shall support the description of conditional existent InterRunnableVariables.
Rationale:	Adaptation of algorithms depended from System Configuration.
Use Case:	
Dependencies:	[RS_SWCT_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.21 Conditional accessibility for measurement

[RS_SWCT_3152] Conditional accessibility for measurement ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.22 Conditional existence of parameter prototypes

[RS_SWCT_3153] Conditional existence of parameter prototypes ⌈

Initiator:	Continental
Date:	2008-11-14
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_3100]

Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.23 Support of conditional ports for SW-C

[RS_SWCT_3154] Support of conditional ports for SW-C ⌈

Initiator:	WP Chassis Control
Date:	2008-11-14
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_3100]
Conflicts:	–
Supporting Material:	
Comment:	–

⌋

2.3.24 Support of Interfaces with different resolutions

[RS_SWCT_3155] Support of Interfaces with different resolutions ⌈

Initiator:	WP Chassis Control
Date:	2008-11-14

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_3100]
Conflicts:	
Supporting Material:	
Comment:	–

⌋

2.3.25 Fixed data exchange

[RS_SWCT_3170] Fixed data exchange ⌈

Initiator:	Continental
Date:	2008-12-03
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 want to fix the value of an input data of a SWC (BRF00157) • UC2: a SWC produces a fixed data that is set in the SWC template • UC3: the integrator want to definitively set the value of a calibration parameter
Dependencies:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.26 M2 support for definition of calibration datasets

[RS_SWCT_3175] M2 support for definition of calibration datasets ⌈

Initiator:	Bosch
Date:	2008-12-03
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.27 Support of SAE J1939 Protocol Features

[RS_SWCT_3180] Support of SAE J1939 Protocol Features ⌈

Initiator:	Vector
Date:	2008-12-03
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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

[RS_SWCT_3181] Need data type and access support for arrays of variable number of elements within the maximum size ⌈

Initiator:	Vector
Date:	2008-12-03
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:	Support of SAE J1939 Protocol Features, [RS_SWCT_3180]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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

[RS_SWCT_3182] Need data type and access support for byte arrays of variable number of elements ⌈

Initiator:	Vector
Date:	2008-12-03
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:	Support of SAE J1939 Protocol Features, [RS_SWCT_3180]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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

[RS_SWCT_3190] Ability to publish/specify the diagnostic capabilities and its resources of an SWC ⌈

Initiator:	AB Volvo/Vector
Date:	2008-12-03
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 • DiagtnosticRoutine (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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.31 Add support for Vehicle and Application Mode Management Concept

[RS_SWCT_3200] Add support for Vehicle and Application Mode Management Concept ⌈

Initiator:	Denso
Date:	2008-12-03
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.32 Add support for Portgroups

[RS_SWCT_3201] Add support for Portgroups ⌈

Initiator:	Denso
Date:	2008-12-03
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:	VMM/AMM Top Req, [RS_SWCT_3200]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.33 Enable SWCs to request dedicated Modes

[RS_SWCT_3202] Enable SWCs to request dedicated Modes ⌈

Initiator:	Denso
Date:	2008-12-03
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:	VMM/AMM Top Req, [RS_SWCT_3200]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.34 Propagation of Mode Information

[RS_SWCT_3203] Propagation of Mode Information ⌈

Initiator:	Denso
Date:	2008-12-03
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:	VMM/AMM Top Req, [RS_SWCT_3200]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.35 Integrity and Scaling at Ports

[RS_SWCT_3210] Integrity and Scaling at Ports ⌈

Initiator:	AB Volvo/DC/Valeo
Date:	2008-12-03
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:	
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

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

[RS_SWCT_3215] Need to add application data type on top of implementation data type ⌈

Initiator:	Vector/Bosch
Date:	2008-09-24
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_0070], [RS_SWCT_0080]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.37 Application data type

[RS_SWCT_3216] Application data type ⌈

Initiator:	Vector/Bosch
Date:	2008-09-24
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_3215]
Use Case:	See [RS_SWCT_3215]
Dependencies:	[RS_SWCT_3215]
Conflicts:	–
Supporting Material:	See [RS_SWCT_3215]
Comment:	–

⌋

2.3.38 Implementation data type

[RS_SWCT_3217] Implementation data type ⌈

Initiator:	Vector/Bosch
Date:	2008-09-24
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_3215]
Use Case:	See [RS_SWCT_3215]
Dependencies:	[RS_SWCT_3215]
Conflicts:	–
Supporting Material:	See [RS_SWCT_3215]
Comment:	–

⌋

2.3.39 Allow Communication Attributes on Compositions

[RS_SWCT_3220] Allow Communication Attributes on Compositions ⌈

Initiator:	Daimler
Date:	2009-01-23
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_0160]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.40 Enhancing the Non-Volatile (NV) memory interface

[RS_SWCT_3225] Enhancing the Non-Volatile (NV) memory interface ⌈

Initiator:	Renault/Continental
Date:	2009-04-23
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_0030], [RS_SWCT_0070]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.41 Documentation of M1 artifacts

[RS_SWCT_3230] Documentation of M1 artifacts ⌈

Initiator:	Bosch/MAN
Date:	2009-04-23
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_2110], [RS_SWCT_2060]
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

2.3.42 Support end-to-end communication protection

[RS_SWCT_3240] Support end-to-end communication protection ⌈

Initiator:	BMW
Date:	2009-06-30
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_0010]
Conflicts:	–
Supporting Material:	The details of the end-to-end protection in AUTOSAR are described in [5]
Comment:	–

⌋

2.3.43 Partial Networking

[RS_SWCT_3241] Support for partial networking ⌈

Initiator:	WPII-1.2
Date:	2011-08-05
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-behaviour and other ports read out and react on the current status of the VFC.
Dependencies:	–
Conflicts:	–
Supporting Material:	–
Comment:	–

⌋

3 Change History

3.1 Change History for AUTOSAR 4.0.1 against 3.1.5

3.1.1 Removed SRS Items

N/A

3.1.2 Changed SRS Items

N/A

3.1.3 Added SRS Items

Number	Heading
[RS_SWCT_0010]	AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability
[RS_SWCT_0020]	AUTOSAR shall provide open and standardized software interfaces for intra-ECU and inter-ECU communication
[RS_SWCT_0030]	AUTOSAR shall provide complete interfaces to application software and basic software modules
[RS_SWCT_0040]	AUTOSAR shall ease the re-usability of software and its concepts and implementations
[RS_SWCT_0050]	AUTOSAR shall provide a software architecture that is applicable across different functional domains
[RS_SWCT_0070]	AUTOSAR shall provide an abstraction of the application software from hardware
[RS_SWCT_0080]	AUTOSAR shall provide an independency of application software from in-vehicle communication technologies
[RS_SWCT_0090]	AUTOSAR should provide an independency of application software from operating systems
[RS_SWCT_0110]	AUTOSAR shall provide a functional interface view of the entire system
[RS_SWCT_0120]	AUTOSAR shall provide protection/unlock mechanisms for software through appropriate services in the infrastructure
[RS_SWCT_0130]	AUTOSAR shall provide interoperability with legacy software
[RS_SWCT_0150]	AUTOSAR shall provide means to protect SW-Components from malicious SW-Components
[RS_SWCT_0160]	AUTOSAR shall provide means to achieve compositionality
[RS_SWCT_0170]	AUTOSAR shall provide diagnostics means during runtime, for production and services purposes
[RS_SWCT_0190]	AUTOSAR shall support hierarchical design methods
[RS_SWCT_0200]	Definitions of relations between SW components are exhaustive and formal
[RS_SWCT_0210]	SW components are protected from illegal access
[RS_SWCT_0220]	Management of vehicle diversity is supported by AUTOSAR
[RS_SWCT_2000]	Top-down hierarchical design
[RS_SWCT_2010]	Interfaces of atomic software-components
[RS_SWCT_2020]	Bottom-up design of CompositionTypes
[RS_SWCT_2030]	Specification of Communications

[RS_SWCT_2050]	Specification of timing resources for software-component description
[RS_SWCT_2060]	Consider interaction with basic software
[RS_SWCT_2080]	Designing a Sensor Actuator Component
[RS_SWCT_2090]	Data-consistency for communication among RunnableEntities
[RS_SWCT_2100]	Definition of physical units
[RS_SWCT_2110]	Definition of comments
[RS_SWCT_3130]	Connections between PortInterfaces
[RS_SWCT_3140]	Conditional existence of PortPrototypes
[RS_SWCT_3141]	Conditional existence of data element prototypes, operation prototypes, parameter prototypes in an interface
[RS_SWCT_3142]	Conditional existence of ComponentPrototypes
[RS_SWCT_3143]	Conditional existence of ConnectorPrototypes
[RS_SWCT_3144]	Configurable size of Arrays
[RS_SWCT_3145]	Describe supported combinations of System Constant Value of an Software Component Type
[RS_SWCT_3146]	Describe supported combinations of System Constant Value of an InternalBehavior
[RS_SWCT_3147]	Describe supported combinations of System Constant Value of an Implementation
[RS_SWCT_3148]	Attributes swMinAxisPoints and swMaxAxisPoints shall be adjustable by an System Constant Definition
[RS_SWCT_3149]	Conditional existence of RunnableEntitys
[RS_SWCT_3150]	Conditional existence of RTEEvents
[RS_SWCT_3151]	Conditional existence of InterRunnableVariables
[RS_SWCT_3152]	Conditional accessibility for measurement
[RS_SWCT_3153]	Conditional existence of parameter prototypes
[RS_SWCT_3154]	Support of conditional ports for SW-C
[RS_SWCT_3155]	Support of Interfaces with different resolutions
[RS_SWCT_3170]	Fixed data exchange
[RS_SWCT_3175]	M2 support for definition of calibration datasets
[RS_SWCT_3180]	Support of SAE J1939 Protocol Features
[RS_SWCT_3181]	Need data type and access support for arrays of variable number of elements within the maximum size
[RS_SWCT_3182]	Need data type and access support for byte arrays of variable number of elements
[RS_SWCT_3190]	Ability to publish/specify the diagnostic capabilities and its resources of an SWC
[RS_SWCT_3200]	Add support for Vehicle and Application Mode Management Concept
[RS_SWCT_3201]	Add support for Portgroups
[RS_SWCT_3202]	Enable SWCs to request dedicated Modes
[RS_SWCT_3203]	Propagation of Mode Information
[RS_SWCT_3210]	Integrity and Scaling at Ports
[RS_SWCT_3215]	Need to add application data type on top of implementation data type
[RS_SWCT_3216]	Application data type
[RS_SWCT_3217]	Implementation data type
[RS_SWCT_3220]	Allow Communication Attributes on Compositions
[RS_SWCT_3225]	Enhancing the Non-Volatile (NV) memory interface
[RS_SWCT_3230]	Documentation of M1 artifacts
[RS_SWCT_3240]	Support end-to-end communication protection

Table 3.1: Added Specification Items in 4.0.1

3.2 Change History for AUTOSAR 4.0.2 against 4.0.1

3.2.1 Removed SRS Items

N/A

3.2.2 Changed SRS Items

N/A

3.2.3 Added SRS Items

N/A

3.3 Change History for AUTOSAR 4.0.3 against 4.0.2

3.3.1 Removed SRS Items

N/A

3.3.2 Changed SRS Items

N/A

3.3.3 Added SRS Items

Number	Heading
[RS_SWCT_3135]	Record Type Subsetting
[RS_SWCT_3136]	Record Type Subsetting with Primitive Types
[RS_SWCT_3241]	Support for partial networking

Table 3.2: Added Specification Items in 4.0.3