Document Title	Requirements on Standardization Template
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	536
Document Classification	Auxiliary

Document Status	Final
Part of AUTOSAR Release	4.2.2

Document Change History

Release	Changed by	Description
4.2.2	AUTOSAR Release Management	• editorial changes
4.2.1	AUTOSAR Release Management	• extend traceability to new document artefacts
4.1.3	AUTOSAR Release Management	• editorial changes
4.1.2	AUTOSAR Release Management	• editorial changes • improvement of document traceability
4.1.1	AUTOSAR Administration	• editorial changes • improvement of document traceability
4.0.3	AUTOSAR Administration	Initial Release

Disclaimer

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

Bibliography

[1] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate

[2] Standardization Template
AUTOSAR_TPS_StandardizationTemplate

[3] Requirements on AUTOSAR Features
AUTOSAR_RS_Features

[4] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration

[5] Specification of Platform Types
AUTOSAR_SWS_PlatformTypes

[6] Generic Structure Template
AUTOSAR_TPS_GenericStructureTemplate

[7] SW-C and System Modeling Guide
AUTOSAR_TR_SWCModelingGuide

1 Introduction

AUTOSAR models are in many cases not created from scratch but existing content is taken as the basis. The existing content could be contributed by the AUTOSAR initiative itself in form of standardized model elements.

This document specifies the requirements for the Standardization Template. This template is intended to support the delivery of standardized model elements by AUTOSAR.

AUTOSAR 4.0 already specifies the blueprint approach for standardization. This approach is continued and refined by the Standardization Template. It thereby will replace Appendix A in Software Component Template [1].

As an particular example, let us consider the standardization of application interfaces. That is, in terms of the AUTOSAR meta-model the standardization mainly applies to the definition of `PortPrototype`s for specific purposes.

Due to the structure of the AUTOSAR meta-model it is not possible to merely express a standardized `PortPrototype` because for good reasons the latter does not exist on its own but is always owned by a `SwComponentType`.

The Standardization Template specifies the approach to overcome this situation.

1.1 Document Conventions

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

1.2 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.

1.3 Use Case Tracing

Following table references the use cases specified and links to the related requirements.

Use Case	Description	Satisfied by
[UC_STDT_00001]	Support of Application Interfaces	[RS_STDT_00001] [RS_STDT_00003] [RS_STDT_00005] [RS_STDT_00006] [RS_STDT_00007] [RS_STDT_00016] [RS_STDT_00019] [RS_STDT_00020] [RS_STDT_00021] [RS_STDT_00022] [RS_STDT_00026] [RS_STDT_00035]
[UC_STDT_00002]	Express parts of SWS	[RS_STDT_00001] [RS_STDT_00002] [RS_STDT_00018] [RS_STDT_00041]
[UC_STDT_00003]	Standardize ECUC Parameters	[RS_STDT_00001] [RS_STDT_00010] [RS_STDT_00029]
[UC_STDT_00004]	Express predefined Paths	[RS_STDT_00001] [RS_STDT_00013] [RS_STDT_00030]
[UC_STDT_00005]	Express Platform Types	[RS_STDT_00001]
[UC_STDT_00006]	Express Examples of applied Standards	[RS_STDT_00001]
[UC_STDT_00007]	Support Verification if an Implementation adheres to defined Standard	[RS_STDT_00001] [RS_STDT_00008] [RS_STDT_00009] [RS_STDT_00015] [RS_STDT_00017]
[UC_STDT_00008]	Support reusable Documentation	[RS_STDT_00001] [RS_STDT_00002] [RS_STDT_00003] [RS_STDT_00023] [RS_STDT_00026] [RS_STDT_00041]
[UC_STDT_00009]	Define name Conventions	[RS_STDT_00001] [RS_STDT_00004] [RS_STDT_00014] [RS_STDT_00023] [RS_STDT_00024] [RS_STDT_00025] [RS_STDT_00042]
[UC_STDT_00010]	STDT shall be applicable beyond the Scope of AUTOSAR	[RS_STDT_00011] [RS_STDT_00012] [RS_STDT_00024] [RS_STDT_00025] [RS_STDT_00032] [RS_STDT_00033]

Use Case	Description	Satisfied by
[UC_STDT_00011]	Derive objects from Blueprints by adding missing Information	[RS_STDT_00015] [RS_STDT_00029] [RS_STDT_00040]
[UC_STDT_00012]	Derive Objects from Blueprints in a completely standardized Way	[RS_STDT_00015] [RS_STDT_00029] [RS_STDT_00040]
[UC_STDT_00013]	Integrate compile test	[RS_STDT_00027]
[UC_STDT_00014]	Generate BSW "Standard AUTOSAR Interface" description from model	[RS_STDT_00028]
[UC_STDT_00015]	Handle General Specification Items	[RS_STDT_00031] [RS_STDT_00041]
[UC_STDT_00016]	Manage requirements in AUTOSAR	[RS_STDT_00036]
[UC_STDT_00017]	Manage specification items in AUTOSAR	[RS_STDT_00037]
[UC_STDT_00018]	Manage constraint items in AUTOSAR	[RS_STDT_00038]
[UC_STDT_00019]	Manage test items in AUTOSAR	[RS_STDT_00039]

1.4 Requirements Tracing

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

Requirement	Description	Satisfied by
[RS_BRF_01024]	AUTOSAR shall provide naming rules for public symbols	[RS_STDT_00042]
[RS_BRF_01056]	AUTOSAR BSW modules shall provide standardized interfaces	[RS_STDT_00028]
[RS_BRF_01064]	AUTOSAR BSW shall provide callback functions in order to access upper layer modules	[RS_STDT_00018]
[RS_BRF_04000]	AUTOSAR documentation shall support traceability	[RS_STDT_00013] [RS_STDT_00030] [RS_STDT_00041]
[RS_BRF_04008]	AUTOSAR documentation shall support consistency and quality assurance	[RS_STDT_00040]
[RS_BRF_04016]	AUTOSAR shall support modeling and documentation guidelines	[RS_STDT_00002] [RS_STDT_00004] [RS_STDT_00005] [RS_STDT_00006] [RS_STDT_00007] [RS_STDT_00008] [RS_STDT_00009] [RS_STDT_00011] [RS_STDT_00012] [RS_STDT_00014] [RS_STDT_00016] [RS_STDT_00020] [RS_STDT_00023] [RS_STDT_00024] [RS_STDT_00025] [RS_STDT_00031] [RS_STDT_00036] [RS_STDT_00037] [RS_STDT_00038] [RS_STDT_00039]
[RS_BRF_04024]	AUTOSAR shall support guidance for applying the specifications	[RS_STDT_00001] [RS_STDT_00003] [RS_STDT_00010] [RS_STDT_00015] [RS_STDT_00017] [RS_STDT_00019] [RS_STDT_00021] [RS_STDT_00022] [RS_STDT_00026] [RS_STDT_00027] [RS_STDT_00029] [RS_STDT_00032] [RS_STDT_00033] [RS_STDT_00034] [RS_STDT_00035]

2 Use Cases

This chapter describes use-cases for the Standardization Template. The intention of these uses cases is to point out the potential applications of the Standardization Template. In general, the use case of the Standardization Template is to express items standardized by AUTOSAR as AUTOSAR XML artifact. This artifact can subsequently be used to support the development of AUTOSAR compliant products.

Each use-case defined in this document has its unique identifier starting with the prefix "UC_STDT_" (meaning Standardization Template Use Case).

[UC_STDT_00001] Support Application Interfaces ⌈ AUTOSAR : provides standardized, openly disclosed interfaces for different domains such as chassis, powertrain, body etc. The definition of these interfaces can be handled on various levels of deepness:

L8 Complete description of the SW-Cs including behavioral model and ports

L7 Complete description of all ports including interface behavior and data qualities

L6 Complete description of all ports including textual description of interface behavior and the data qualities of the interfaces

L5 Complete description of all ports including data qualities of the interfaces

L4 Complete description of all ports including within-AUTOSAR-agreed data qualities

L3 Partial description of ports/interfaces of a SWC including within-AUTOSAR-agreed data qualities

Note that this partial description includes the fact that only some of the ports are described, as well as the fact that this description of a port is incomplete and is also separated from the applicable component. This is also known as PortBluePrint.

L2 Dictionary of interfaces including a set of within-AUTOSAR-agreed data qualities

L1 Dictionary of data elements including types and ranges.

L0 Dictionary of names

As of Release 4.0, AUTOSAR standardization covers the level L0 ... L3. Nevertheless vendor internal applications might use Standardization Template for higher levels too.

This use case mainly covers the application software aspects.

Applying this formal description will improve consistency and usability of the AUTOSAR Application Interfaces and empower formal checks e.g. of backward compatibility of application interfaces. ⌋*()*

[UC_STDT_00002] Express Parts of SWS ⌈ The Standardization Template shall allow to express parts of SWS for basic software modules formally using the AUTOSAR schema. This includes (but is not restricted to):

- Standardized interfaces (i.e. C-APIs)
- Standardized AUTOSAR Interfaces (Ports, PortInterfaces, ...)
- Definition of ECUC-Parameters (see [UC_STDT_00003])

Applying this formal description will improve consistency and usability of the AUTOSAR SWS and empower formal checks e.g. of backward compatibility of interfaces. ⌋()

[UC_STDT_00003] Standardize ECUCParamdefs ⌈ Part of the AUTOSAR SWS is also the set of ECU configuration parameter definitions. These parameter definitions are the basis of the so called vendor specific parameters which are used in particular AUTOSAR implementations.

Even if this is specified in great detail in [4] it is also in the scope of Standardization Template. ⌋()

[UC_STDT_00004] Express predefined Paths ⌈ Development partners may mutually agree on a particular package layout and thus share AUTOSAR artifacts in a later phase of the development. For this use case it is helpful to initially express a set of predefined resp. partly predefined reference paths respective `referenceBase` which can be loaded in individual AUTOSAR authoring tools.

This use case covers the beginning or the end of a reference path. For example the usecase is to standardize the substructure after a variable path: `<My-Path>/PortInterfaces`. In this case only `PortInterfaces` is standardized. ⌋()

[UC_STDT_00005] Express PlatformTypes ⌈ The platform types defined in [5] need to be available in processable format for AUTOSAR development tools. This approach improves consistency and quality of AUTOSAR products.

The details of [6] chapter 3.1 apply. ⌋()

[UC_STDT_00006] Express Examples of applied Standards ⌈ In addition to the application interfaces [**?**] AUTOSAR provides examples how to apply the standardized elements, in particular blueprints.

The details of [6] chapter 3.1 apply. ⌋()

[UC_STDT_00007] Support Verification if an implementation adheres to defined Standard ⌈ When an AUTOSAR product is developed an initial verification can be performed by verifying the product against the formalized standard. This includes

- the check of compatibility rules between blueprints and derived model elements. These compatibility rules are to be defined for each meta class eligible to blueprints.
- tracing between model elements and SWS respectively blueprints in order to check if all blueprints were implemented.

Note that this use case is a very initial verification and does not compete or even replace conformance test specification. It rather contributes to conformance test.

The compatibility rules need only to be described in the document, e.g. in form of constraints. There is no formal representation of the compatibility rules in the meta-model.

The compatibility rules are specified individually for each meta-class eligible for blueprinting. For example all port blueprints follow the same compatibility rules. ⌋()

[UC_STDT_00008] Support reusable Documentation ⌈ Parts of the AUTOSAR SWS may be published such that it can be reused for the actual product documentation. The vendor of an implementation then takes such parts out of an Instance of a Standardization Template and incorporates it in his own software documentation.

The same approach may apply to the explanation of Application Interfaces. ⌋()

[UC_STDT_00009] Define name conventions ⌈ AUTOSAR has modeling guides and naming conventions. If these conventions are published as instance of the Standardization Template, they can be utilized to configure e.g. modeling tools.

The use case also covers the ability to express various levels of obligation. This may for example be expressed similar to the keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL". ⌋()

[UC_STDT_00010] Perform Standardization on Levels beyond the AUTOSAR Scope ⌈ The Standardization Template shall be applicable for company internal standardization respectively for mutual agreements which go beyond the AUTOSAR standardization on meta level[1] M1 (see [UC_STDT_00001]). ⌋()

[UC_STDT_00011] Derive Objects from Blueprints by manually changing properties ⌈ The user makes a kind of copy of the blueprint and is allowed to add his own things (e.g. adding own field to a standardized enum-type). Example is the usage of ApplicationInterfaces and ConfigurationParamaters.

In case of Standardized PortInterfaces of AUTOSAR Services the Standardization template shall mention the use case that in this case a PortInteface based on the "Standardized PortInterface Blueprint" might contain a subset of ClientServerOperations. ⌋()

[UC_STDT_00012] Derive Objects from Blueprints in a completely standardized Way ⌈ The user can only configure or otherwise influence the content of the copied "blueprint" in a completely standardized way, (e.g. configuring the fields of an enum-type according to the "needs" of the software) but he cannot add own things.

This could even go so far, that only the rules of configuration are standardized (like in the case of DCM-PortInterfaces) - but this nonetheless completely determines the outcome in a concrete project. ⌋()

[UC_STDT_00013] Integrate compile test ⌈ Until Release 4.0 all APIs of the BSW are modeled and chapter 8 of the SWS is mainly generated out of the model. Additionally we propose to generate empty C functions (and data structures/consts/...) out of the

[1] For more details of Meta levels see Chapter 2.2 in [6]

model and link all these functions together. If the compile or link process fails the consistency (e.g. between different SWS) is violated and needs to be fixed. ⌋()

[UC_STDT_00014] Generate BSW ”Standard AUTOSAR Interface” description from model ⌈ Until Release 4.0 the "Standard AUTOSAR Interface" is part of each SWS which offers this interface (Typically contained in an own chapter of sub chapter of 7 or 8). The description is mostly plain text with some pseudo language to show the usage of the interface (including constants, etc.). Furthermore the description of the services often uses "elements" from the meta-model which are not up-to-date or their meaning has changed.

It is intended to standardize this part of the SWS, e.g. via an own model (and then the generated descriptions can be imported into the SWS like chapter 8) OR via a -standardized- language to clarify the understanding of the interface and allow an automatic conversation for RTE purposes. ⌋()

[UC_STDT_00015] Handle General Specification Items ⌈ There might be a set of general requirements which need to fulfilled by all SWS document. On the other hand there might be a general specification which summarizes all common specification items. This situation shall be handled by tracing:

- Tracing shall use both requirements documents, the general one as well as the individual one
- General Requirements may be satisfied by the general specification or by the individual specifications.
- General Requirements may not be applicable for a particular specification.
- General Requirements may be fully satisfied only by both the general together with an individual specification.

⌋()

[UC_STDT_00016] Manage requirements in AUTOSAR ⌈ In AUTOSAR all requirements are formally captured in requirement documents (RS/Feature/SRS) with a unique id. Specification documents (SWS) contain specification items that formally trace to requirements. Dependencies between requirements on the same level are expressed in the requirement block itself by providing references to the requirements. ⌋()

[UC_STDT_00017] Manage specification items in AUTOSAR ⌈ In AUTOSAR all specification items are formally captured in specification documents (TPS, SWS) with a unique id. Specification items formally trace to requirements. ⌋()

[UC_STDT_00018] Manage constraint items in AUTOSAR ⌈ In AUTOSAR all constraint items are formally captured in specification documents (TPS, SWS) with a unique id. ⌋()

[UC_STDT_00019] Manage test items in AUTOSAR ⌈ In AUTOSAR all test items are formally captured in specification documents with a unique id. ⌋()

3 Requirements

This chapter describes all requirements driving the work to define the *Standardization Template* specification [2].

Each requirement in this document has its unique identifier starting with the prefix "RS_STDT_" (meaning Requirement Specification for Standardization Template).

[RS_STDT_00001] Shall support and explain Blueprints in general ⌈

Type:	valid
Description:	The standardization template shall support blueprints. Blueprint is a kind of "incomplete" model" which is copied and refined lateron. The principles of blueprints shall be defined: • "Instantiation" is done by copy rather than referenced. Downstream processing excludes the usage of blueprints. • Define proper terminology for blueprints and blueprinted model elements. • How are the elements named that are created out of blueprints? • Shall clearly define which parts of the meta-model are eligible for blueprinting. Blueprinting non-eligible parts of the meta-model shall count as a "validation error". • define the rules how to derive objects from blueprints, in particular the strategy, which properties may be added / removed / redefined. These rules shall be defined individually for each meta-class eligible for blueprinting. Necessary facilities of the Meta Model shall be defined: • specific blueprints • mapping blueprints and derived objects
Rationale:	This helps to understand the concept and application of blueprints as blueprints are the main mean of standardization.
Use Case:	[UC_STDT_00001],[UC_STDT_00002],[UC_STDT_00003],[UC_STDT_00004] [UC_STDT_00005],[UC_STDT_00006],[UC_STDT_00007],[UC_STDT_00008] [UC_STDT_00009]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04024)*

[RS_STDT_00002] Formalized description of BSW SWS ⌈

Type:	valid

Description:	The standardization Template shall be able to publish formalized parts of a SWS which then acts as a blueprint of the specified Module. The Standardization Template shall provide means to describe standardized Interfaces (C-APIs). The Standardization Template shall allow the description standardized AUTOSAR Interfaces. The Standardization Template must support the specification of variants of the interfaces.
Rationale:	Especially the "Standard AUTOSAR Interface" is part of each SWS which offers this interface (Typically contained in an own chapter of subchapter of 7 or 8). The description is mostly plain text with some pseudo language to show the usage of the interface (including constants, etc.). Furthermore the description of the services often uses "elements" from the meta-model which are not up-to-date or their meaning has changed. The current state causes several problems when the RTE "routes" calls between the BSWs and SWC. The pseudo language must be manually transferred into some sort of "SWC-Description". If people try to mix modules from different vendors it is not clear how this can work. In our understanding the format needs to be standardized. We propose to standardize this part of the SWS, e.g. via an own model (and then the generated descriptions can be imported into the SWS like chapter 8) OR via a -standardized- language to clarify the understanding of the interface and allow an automatic conversation for RTE purposes.
Use Case:	[UC_STDT_00002],[UC_STDT_00008]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00003] Shall allow to represent port blueprints ⌈

Type:	valid
Description:	AUTOSAR standardizes so called "Application Interfaces". These Interfaces in fact result in port blueprints.
Rationale:	AUTOSAR publishes standardized Models as ARXML.
Use Case:	[UC_STDT_00001],[UC_STDT_00008]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00004] Shall allow to represent `shortName` patterns ⌈

Type:	valid
Description:	–
Rationale:	AUTOSAR publishes the Application Interfaces Modeling guide
Use Case:	[UC_STDT_00009]
Dependencies:	TR_SWCModelingGuide [7] might need to be adapted.

Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00005] Shall support keywords and keyword abbreviations ⌈

Type:	valid
Description:	In [7] AUTOSAR publishes building rules for `ShortName` as sequence of Keywords. These keywords need to be expressed by Standardization Template. The existing Keyword identifiable should be extended with a ShortLabel. Semantics of SHORT-LABEL: used instead of ShortName. Necessary in case there are several nampe parts that are to be abbreviated by the same keyword abbreviation. As Keywords are identifiables, this would lead to a conflict.
Rationale:	Support AUTOSAR publication
Use Case:	[UC_STDT_00001]
Dependencies:	TR_SWCModelingGuide [7] might need to be adapted.
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00006] Shall be implemented without compatibility problems to existing template ⌈

Type:	valid
Description:	–
Rationale:	Maintenance of backwards compatibility of the Schema as requested for 4.0
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00007] Shall be based on the AUTOSAR schema ⌈

Type:	valid
Description:	–
Rationale:	General approach of having one Meta Model for all templates
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00008] Shall provide means to support analyzing the conformity of implementations with the AUTOSAR standards ⌈

Type:	valid
Description:	The Standardization Template should enable the implementer to check if its description (e.g. BSWM) is in conformance with AUTOSAR standards specified on M1 level (SWS).
Rationale:	This establishes traceability between AUTOSAR Implementations and defined standard. And is also a precondition to check the application compatibility between different releases.
Use Case:	[UC_STDT_00007]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04016)*

[RS_STDT_00009] Shall be able to represent requirements stated in SWS ⌈

Type:	valid
Description:	To improve requirements traceability a formalized description of a SWS shall contain textual representatives of each requirement contained in the respective SWS document (specification items of SWSs). The statements shall be categorized as one of {Requirement, Specification, Implementation, Constraint}
Rationale:	This feature enables a automated tracking of changes of requirements, which is the basis for a improved change management and compatibility assessment.
Use Case:	[UC_STDT_00007]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04016)*

[RS_STDT_00010] Shall refer to ECUC parameter definition ⌈

Type:	valid
Description:	ECUC parameter definitions are also standardized by AUTOSAR. Therefore it is in the scope of the Standardization Template. Strictly speaking, these standardizes ECUC Parameter Definitions act as blueprints for the vendor specific parameters, even if these are not mapped using the BluePrintMappingSet. Standardization Template shall not change the approaches at least for AUTOSAR 4.0, but reflect the relationships.
Rationale:	This maintains the overall scope and the applied patterns.
Use Case:	[UC_STDT_00003]
Dependencies:	–
Supporting Material:	[4]

⌋*(RS_BRF_04024)*

[RS_STDT_00011] Shall be able to standardize components ⌈

Type:	valid
Description:	STDT shall be able to express standardization of components, even if AUTOSAR does not standardize components. This requirement covers a set of individual components. No compatibility rules shall hardwired in STDT. Support is provided by the fact that it only allows to specify SwComponentType as eligible for blueprinting.
Rationale:	This allows to leverage AUTOSAR standardization principles inside a company.
Use Case:	[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04016)*

[RS_STDT_00012] Shall be able to standardize architecture ⌈

Type:	valid
Description:	–
Rationale:	This allows to leverage AUTOSAR standardization principles inside a company.
Use Case:	[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04016)*

[RS_STDT_00013] Shall be able to express parts of reference paths resp. package hierarchies ⌈

Type:	valid
Description:	–
Rationale:	This allows to leverage AUTOSAR standardization principles inside a company.
Use Case:	[UC_STDT_00004]
Dependencies:	–
Supporting Material:	–

⌋*(RS_BRF_04000)*

[RS_STDT_00014] Shall be able to express levels of obligation ⌈

Type:	valid
Description:	–
Rationale:	This allows to use Standardization Models to evaluate conformity of an implementation
Use Case:	[UC_STDT_00009]
Dependencies:	–

Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00015] Shall support different Approaches to derive from Blueprints ⌈

Type:	valid
Description:	–
Rationale:	This allows to use Standardization Models to evaluate conformity of an implementation. Conformity depends on the approach to derive objects
Use Case:	[UC_STDT_00007],[UC_STDT_00011],[UC_STDT_00012]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00016] Shall be able to express information about the state of model elements ⌈

Type:	valid
Description:	–
Rationale:	This supports a continues evolution of a standard
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00017] Shall cover the compatibility of blueprints and derived objects ⌈

Type:	valid
Description:	–
Rationale:	This supports a continues evolution of a standard
Use Case:	[UC_STDT_00007]
Dependencies:	[RS_STDT_00001]
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00018] Shall allow to describe the dependencies of APIs (e.g. invocation and callback/polling interfaces) ⌈

Type:	valid
Description:	–
Rationale:	Standardized interfaces consist in many cases of invocation interfaces (C-APIs) and callback or polling interfaces. In many cases it is configurable, which communication pattern is used. This configurable dependency and the parameters shall be described via blueprints.
Use Case:	[UC_STDT_00002]
Dependencies:	[RS_STDT_00002]
Supporting Material:	–

⌋(*RS_BRF_01064*)

[RS_STDT_00019] Shall define the mandatory semantics for a Blueprint ⌈

Type:	valid
Description:	For a given model element,the template must define which attributes of the model element must be standardized to be entitled as a blueprint. For e.g which information of a PortInterface must be present to be called as blueprint ? In case of Standardized PortInterfaces of AUTOSAR Services the Standardization template shall mention the use case that in this case a PortInteface based on the "Standardized PortInterface Blueprint" might contain a subset of ClientServeOperations.
Rationale:	Helps to have a common understanding on blueprint model element.
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00020] Shall support variants of a VariableDataprototype ⌈

Type:	valid
Description:	The PortBlueprint should be able to map to different VariableDataprototype of the same instance of PortInterface.
Rationale:	Variant handling for WP10.3 is mostly intended for reusing the definition between passenger cars and trucks. Therefore it's probably useful to have variants at data type level instead of creating new blueprints.
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00021] Shall support multiple instantiation for an example SWC with PortBlueprint ⌈

Type:	valid
Description:	It should be possible for the PortBlueprint to support multiple instantiation.
Rationale:	–
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00022] Means of exchange format between stakeholders for blueprints ⌈

Type:	valid
Description:	AUTOSAR methodology shall define the exchange of the PortInterfaceMapping for a given SWCdescription file, i.e The RTE and the VFB in principle ignores the blueprints but how should the exchange be established between the stakeholders with the blueprint information, while creating the PortPrototypes out of it?.
Rationale:	–
Use Case:	[UC_STDT_00001]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00023] Shall be able to standardize Alias Names ⌈

Type:	valid
Description:	STDT shall be able to standardize alias names.
Rationale:	e.g. used for system constants in measurement and calibration system Necessary if system constants will be standardized in future (what is not yet decided)
Use Case:	[UC_STDT_00008],[UC_STDT_00009]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00024] Shall be able to standardize Unique Names and Display Names ⌈

Type:	valid
Description:	STDT shall be able to standardize Unique Names and Display Names e.g. in documentation (the complete instance reference would not be readable) and in measurement and calibration systems (standardized measurement and calibration formats like A2L require unique names for sw signals).

Rationale:	support standardization of unique names, e.g. for • documentation • calibration and measurement tools
Use Case:	[UC_STDT_00009],[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00025] Shall be able to standardize life cycle states ⌈

Type:	valid
Description:	STDT shall be able to standardize life the states of a particular lifecycle.
Rationale:	Since AUTOSAR has the goal of being backward compatible it is not possible to just delete a standardized model element and add a new one or - even worse - to change the model element without notification.
Use Case:	[UC_STDT_00009],[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00026] Shall allow to represent port interface blueprints ⌈

Type:	valid
Description:	AUTOSAR standardizes so called "Application Interfaces". These Interfaces in fact result in port blueprints and appropriate port interface blueprints
Rationale:	AUTOSAR publishes standardized Models as ARXML.
Use Case:	[UC_STDT_00001],[UC_STDT_00008]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00027] Shall allow to evaluate the integrity of Blueprints ⌈

Type:	valid
Description:	Until Release 4.0 all APIs of the BSW are modeled and chapter 8 of the SWS is mainly generated out of the model. Additionally we propose to generate empty C functions (and data structures/consts/...) out of the model and link all these functions together. If the compile or link process fails the consistency (e.g. between different SWS) is violated and needs to be fixed.
Rationale:	In the past we had often problems that some SWS assume specific services (of structs/consts/...) from other modules and the interface did not match (or even did not exists at all). In the compile test such errors will be found.

Use Case:	[UC_STDT_00013]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00028] Shall allow to generate BSW ”Standard AUTOSAR Interface” description from model ⌈

Type:	valid
Description:	"Standard AUTOSAR Interface" is part of each SWS which offers this interface (Typically contained in an own chapter of subchapter of 7 or 8). The description is mostly plain text with some pseudo language to show the usage of the interface (including constants, etc.). Furthermore the description of the services often uses "elements" from the meta-model which are not up-to-date or their meaning has changed. STDT shall provide support to standardize this part of the SWS, e.g. via an own model (and then the generated descriptions can be imported into the SWS like chapter 8) OR via a -standardized- language to clarify the understanding of the interface and allow an automatic conversation for RTE purposes.
Rationale:	–
Use Case:	[UC_STDT_00014]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_01056*)

[RS_STDT_00029] Shall be able to represent further Blueprints ⌈

Type:	valid

Description:	STDT shall be able to represent blueprints for the following elements: • AliasNameSet (see RS_STDT_00023) • ApplicationDatatype • BswModuleEntry • BaseType • BswModuleDescription • CompuMethod - to enhance enumerators • DataConstr - to widen ranges as long as it fits in the BaseType not allowed to restrict : Wp: Not allowed to change the range • DatatypeMappingSet - mapped types in derived Mappings set must be the derived ones from blueprint • EcucModuleDef • EcucDefintionCollection • ImplementationDatatype • ModeDeclarationGroup - to add addtional modes • PortInterfaces (for sender receiver and client server interfaces) (see RS_STDT_00026) • PortPrototypeBlueprints (see RS_STDT_00003) • SwComponentType
Rationale:	–
Use Case:	[UC_STDT_00003],[UC_STDT_00011],[UC_STDT_00012]
Dependencies:	See also [RS_STDT_00003], [RS_STDT_00026]
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00030] Shall allow to standardize package structures ⌈

Type:	valid
Description:	STDT shall be able to represent blueprints of package structures in particular to predefine access paths.
Rationale:	–
Use Case:	[UC_STDT_00004]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04000*)

[RS_STDT_00031] Shall support general specification items ⌈

Type:	valid

Description:	support the explicit indication of non applicable requirements and complementary specification items. In particular allow the specifc trace-Indexes "NA" and "SPEC".
Rationale:	–
Use Case:	[UC_STDT_00002],[UC_STDT_00015]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00032] Shall be able to provide Blueprints for Roles and Rights ⌈

Type:	valid
Description:	Standardization Template shall support blueprinting of roles and rights
Rationale:	–
Use Case:	[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00033] Shall be able to provide Blueprints for Build Action Manifest ⌈

Type:	valid
Description:	Standardization Template shall support blueprinting of Processor Manfest.
Rationale:	–
Use Case:	[UC_STDT_00010]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00034] Blueprinting of Implicit Communication Behavior ⌈

Type:	valid
Description:	The AUTOSAR Templates and Methodology shall support blueprinting of the Implicit Communication Behavior descriptions. Grouping of data shall be possible before the `RunnableEntity`s with all the details (data access points) are known. In a top down approach the grouping of `DataPrototype`s can already be used to design the system in a way that consistency properties are guaranteed and that consistency is not required for unrelated `DataPrototype`s.
Rationale:	Define Implicit Communication Behavior requirements in a top down design approach
Use Case:	–
Dependencies:	–

Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00035] Shall support blueprinting of keywords ⌈

Type:	valid
Description:	Keywords shall be blueprintable in order to support vendor specific extensions.
Rationale:	Support AUTOSAR publication
Use Case:	[UC_STDT_00001]
Dependencies:	TR_SWCModelingGuide [7] might need to be adapted.
Supporting Material:	–

⌋(*RS_BRF_04024*)

[RS_STDT_00036] StandardizationTemplate shall specify the representation of requirements in AUTOSAR documents ⌈

Type:	valid
Description:	Specify content and prefered graphical representation of structured requirements in AUTOSAR documents and AUTOSAR meta model.
Rationale:	Consistent specification and representation of requirements and specifcation items in AUTOSAR.
Use Case:	[UC_STDT_00016]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00037] StandardizationTemplate shall specify the representation of specification items in AUTOSAR documents ⌈

Type:	valid
Description:	Specify content of specification items in AUTOSAR documents and AUTOSAR meta model.
Rationale:	Consistent specification and representation of specification items in AUTOSAR.
Use Case:	[UC_STDT_00017]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00038] StandardizationTemplate shall specify the representation of constraint items in AUTOSAR documents ⌈

Type:	valid
Description:	Specify content of constraint items in AUTOSAR documents and AUTOSAR meta model.
Rationale:	Consistent specification and representation of constraint items in AUTOSAR.
Use Case:	[UC_STDT_00018]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00039] StandardizationTemplate shall specify the representation of test items in AUTOSAR documents ⌈

Type:	valid
Description:	Specify content of test items in AUTOSAR documents and AUTOSAR meta model.
Rationale:	Consistent specification and representation of test items in AUTOSAR.
Use Case:	[UC_STDT_00019]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04016*)

[RS_STDT_00040] Multiplicity of elements in derived objects ⌈

Type:	valid
Description:	The standardization template shall support for elements with upper multiplicity > 1 the description of the expected number of derived objects.
Rationale:	This supports the task Derive From Blueprint.
Use Case:	[UC_STDT_00011],[UC_STDT_00012]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_04008*)

[RS_STDT_00041] Formalized description of BSW Abstract SWS ⌈

Type:	valid
Description:	The standardization template shall be able to publish formalized parts of a general SWS which then are inherited by derived specific SWSs.
Rationale:	Groups of BSW SWSs often share a common set of specification elements. These general specification elements should not be stated multiple times in order to avoid redundancy. Therefore a specific SWS should be able to inherit the general parts from an abstract SWS and focus on those elements that have to be specified in particular.
Use Case:	[UC_STDT_00002],[UC_STDT_00008],[UC_STDT_00015]
Dependencies:	–

Supporting Material:	–

⌋(*RS_BRF_04000*)

[RS_STDT_00042] Shall provide the ability to define naming conventions for public symbols ⌈

Type:	valid
Description:	The standardization 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:	[UC_STDT_00009]
Dependencies:	–
Supporting Material:	–

⌋(*RS_BRF_01024*)

A Change History

A.1 Change History R4.0.3

A.1.1 Added Use Cases

Number	Heading
[UC_STDT_00001]	Support Application Interfaces
[UC_STDT_00002]	Express Parts of SWS
[UC_STDT_00003]	Standardize ECUCParamdefs
[UC_STDT_00004]	Express predefined Paths
[UC_STDT_00005]	Express PlatformTypes
[UC_STDT_00006]	Express Examples of applied Standards
[UC_STDT_00007]	Support Verification if an implementation adheres to defined Standard
[UC_STDT_00008]	Support reusable Documentation
[UC_STDT_00009]	Define name conventions
[UC_STDT_00010]	Perform Standardization on Levels beyond the AUTOSAR Scope
[UC_STDT_00011]	Derive Objects from Blueprints by manually changing properties
[UC_STDT_00012]	Derive Objects from Blueprints in a completly standardized Way
[UC_STDT_00013]	Integrate compile test
[UC_STDT_00014]	Generate BSW ”Standard AUTOSAR Interface” description from model

Table A.1: Added Use Cases in 4.0.3

A.1.2 Added Requirements

Number	Heading
[RS_STDT_00001]	Shall support and explain Blueprints in general
[RS_STDT_00002]	Formalized description of BSW SWS
[RS_STDT_00003]	Shall allow to represent port blueprints
[RS_STDT_00004]	Shall allow to represent `shortName` patterns
[RS_STDT_00005]	Shall support keywords and keyword abbreviations
[RS_STDT_00006]	Shall be implemented without compatibility problems to existing template
[RS_STDT_00007]	Shall be based on the AUTOSAR schema
[RS_STDT_00008]	Shall provide means to support analyzing the conformity of implementations with the AUTOSAR standards
[RS_STDT_00009]	Shall be able to represent requirements stated in SWS
[RS_STDT_00010]	Shall refer to ECUC parameter definition
[RS_STDT_00011]	Shall be able to standardize components
[RS_STDT_00012]	Shall be able to standardize architecture
[RS_STDT_00013]	Shall be able to express parts of reference paths resp. package hierarchies
[RS_STDT_00014]	Shall be able to express levels of obligation
[RS_STDT_00015]	Shall support different Approaches to derive from Blueprints
[RS_STDT_00016]	Shall be able to express information about the state of model elements
[RS_STDT_00017]	Shall cover the compatibility of blueprints and derived objects
[RS_STDT_00018]	Shall allow to describe the dependencies of APIs (e.g. invocation and callback/polling interfaces)
[RS_STDT_00019]	Shall define the mandatory semantics for a Blueprint
[RS_STDT_00020]	Shall support variants of a VariableDataprototype
[RS_STDT_00021]	Shall support multiple instantiation for an example SWC with PortBlueprint

[RS_STDT_00022]	Means of exchange format between stakeholders for blueprints
[RS_STDT_00023]	Shall be able to standardize Alias Names
[RS_STDT_00024]	Shall be able to standardize Unique Names and Display Names
[RS_STDT_00025]	Shall be able to standardize life cycle states
[RS_STDT_00026]	Shall allow to represent port interface blueprints
[RS_STDT_00027]	Shall allow to evaluate the integrity of Blueprints
[RS_STDT_00028]	Shall allow to generate BSW ”Standard AUTOSAR Interface” description from model
[RS_STDT_00029]	Shall be able to represent further Blueprints
[RS_STDT_00030]	Shall allow to standardize package structures

Table A.2: Added Requirements in 4.0.3

A.2 Change History R4.1.1

A.2.1 Added Use Cases

Number	Heading
[UC_STDT_00016]	Manage requirements in AUTOSAR

Table A.3: Added Use Cases in 4.1.1

A.2.2 Added Requirements

Number	Heading
[RS_STDT_00031]	Shall support general specification items
[RS_STDT_00032]	Shall be able to provide Blueprints for Roles and Rights
[RS_STDT_00033]	Shall be able to provide Blueprints for Build Action Manifest
[RS_STDT_00034]	Blueprinting of Implicit Communication Behavior
[RS_STDT_00035]	Shall support blueprinting of keywords
[RS_STDT_00036]	StandardizationTemplate shall specify the representation of requirements in AUTOSAR documents

Table A.4: Added Requirements in 4.1.1

A.3 Change History R4.1.2

A.3.1 Added Use Cases

Number	Heading
[UC_STDT_00017]	Manage specification items in AUTOSAR
[UC_STDT_00018]	Manage constraint items in AUTOSAR

Table A.5: Added Use Cases in 4.1.2

A.3.2 Added Requirements

Number	Heading
[RS_STDT_00037]	StandardizationTemplate shall specify the representation of specification items in AUTOSAR documents
[RS_STDT_00038]	StandardizationTemplate shall specify the representation of constraint items in AUTOSAR documents

Table A.6: Added Requirements in 4.1.2

A.4 Change History R4.1.3

A.4.1 Added Use Cases

none

A.4.2 Added Requirements

none

A.5 Change History R4.2.1

A.5.1 Added Traceables in 4.2.1

Id	Heading
[RS_STDT_00039]	StandardizationTemplate shall specify the representation of test items in AUTOSAR documents
[RS_STDT_00040]	Multiplicity of elements in derived objects
[RS_STDT_00041]	Formalized description of BSW Abstract SWS
[RS_STDT_00042]	Shall provide the ability to define naming conventions for public symbols
[UC_STDT_00019]	Manage test items in AUTOSAR

Table A.7: Added Traceables in 4.2.1

A.5.2 Changed Traceables in 4.2.1

Id	Heading
[RS_STDT_00001]	Shall support and explain Blueprints in general
[RS_STDT_00002]	Formalized description of BSW SWS
[RS_STDT_00003]	Shall allow to represent port blueprints
[RS_STDT_00004]	Shall allow to represent `shortName` patterns
[RS_STDT_00005]	Shall support keywords and keyword abbreviations
[RS_STDT_00006]	Shall be implemented without compatibility problems to existing template
[RS_STDT_00007]	Shall be based on the AUTOSAR schema
[RS_STDT_00008]	Shall provide means to support analyzing the conformity of implementations with the AUTOSAR standards
[RS_STDT_00009]	Shall be able to represent requirements stated in SWS
[RS_STDT_00010]	Shall refer to ECUC parameter definition

[RS_STDT_00011]	Shall be able to standardize components
[RS_STDT_00012]	Shall be able to standardize architecture
[RS_STDT_00013]	Shall be able to express parts of reference paths resp. package hierarchies
[RS_STDT_00014]	Shall be able to express levels of obligation
[RS_STDT_00015]	Shall support different Approaches to derive from Blueprints
[RS_STDT_00016]	Shall be able to express information about the state of model elements
[RS_STDT_00017]	Shall cover the compatibility of blueprints and derived objects
[RS_STDT_00018]	Shall allow to describe the dependencies of APIs (e.g. invocation and call-back/polling interfaces)
[RS_STDT_00019]	Shall define the mandatory semantics for a Blueprint
[RS_STDT_00020]	Shall support variants of a VariableDataprototype
[RS_STDT_00021]	Shall support multiple instantiation for an example SWC with PortBlueprint
[RS_STDT_00022]	Means of exchange format between stakeholders for blueprints
[RS_STDT_00023]	Shall be able to standardize Alias Names
[RS_STDT_00024]	Shall be able to standardize Unique Names and Display Names
[RS_STDT_00025]	Shall be able to standardize life cycle states
[RS_STDT_00026]	Shall allow to represent port interface blueprints
[RS_STDT_00027]	Shall allow to evaluate the integrity of Blueprints
[RS_STDT_00028]	Shall allow to generate BSW ”Standard AUTOSAR Interface” description from model
[RS_STDT_00029]	Shall be able to represent further Blueprints
[RS_STDT_00030]	Shall allow to standardize package structures
[RS_STDT_00031]	Shall support general specification items
[RS_STDT_00032]	Shall be able to provide Blueprints for Roles and Rights
[RS_STDT_00033]	Shall be able to provide Blueprints for Build Action Manifest
[RS_STDT_00034]	Blueprinting of Implicit Communication Behavior
[RS_STDT_00035]	Shall support blueprinting of keywords
[RS_STDT_00036]	StandardizationTemplate shall specify the representation of requirements in AUTOSAR documents
[RS_STDT_00037]	StandardizationTemplate shall specify the representation of specification items in AUTOSAR documents
[RS_STDT_00038]	StandardizationTemplate shall specify the representation of constraint items in AUTOSAR documents
[UC_STDT_00006]	Express Examples of applied Standards

Table A.8: Changed Traceables in 4.2.1

A.5.3 Deleted Traceables in 4.2.1

none

A.6 Change History R4.2.2

A.6.1 Added Traceables in 4.2.2

none

A.6.2 Changed Traceables in 4.2.2

none

A.6.3 Deleted Traceables in 4.2.2

none