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

Document Version	1.2.0
Document Status	Final
Part of Release	3.2
Revision	2

Document Change History

Date	Version	Changed by	Change Description
17.05.2012	1.2.0	AUTOSAR Administration	• Added requirement to extend non-volatile memory support
27.04.2011	1.1.0	AUTOSAR Administration	• Added requirements to support partial networking • Legal disclaimer revised
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

Disclaimer

Advice for users

AUTOSAR Specification Documents 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 Specification Documents for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such Specification Documents, 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

1 Introduction

This document defines requirements on tool interoperability and compliance.

1.1 Terminology

- The **AUTOSAR metamodel** is a UML2.0 model that defines the language for describing AUTOSAR systems. The AUTOSAR metamodel is a graphical representation of a template. UML2.0 class diagrams are used to describe the attributes and their interrelationships. Stereotypes and OCL (object constraint language) are used for defining specific semantics and constraints.

- An **AUTOSAR model** is an instance of the AUTOSAR metamodel. The information contained in the AUTOSAR model can be anything that is representable according to the AUTOSAR metamodel. The AUTOSAR model can be stored in many different ways: it might be a set of files in a file system, an XML stream, a database or memory used by some running software tools, etc.

- The **AUTOSAR XML Schema** is a W3C XML schema that defines the language for exchanging AUTOSAR models. This Schema is derived from the AUTOSAR metamodel and defines the **AUTOSAR data exchange format**.

- An **AUTOSAR XML description** describes the XML representation of an AUTOSAR model. The AUTOSAR XML description can consist of several fragments (e.g. files). Each individual fragment must validate successfully against the AUTOSAR XML schema.

1.2 About Requirements

Each requirement has its unique identifier starting with the prefix "AR_CONTENT_" (meaning AUTOSAR Software Component Template Content).

1.2.1 Structure

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

Initiator:	< number of originating work package, company, etc >
Date:	< date of last change >
Requirement:	< the normative text of the requirement >
Description:	< detailed description of the requirement >
Rationale:	< why is this necessary, what its omission could cause >
Use Case:	< example to a scenario that makes the requirement necessary or useful >
Dependencies:	< reference to depending and depended-on requirements >
Conflicts:	< reference to conflicting requirement >
Supporting Material:	< links to other documents >
Comment:	< additional remarks >

1.2.2 Conventions used

In requirements, the following specific semantics are used (taken from Request for Comment RFC 2119 from the Internet Engineering Task Force IETF):

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.)

1.2.3 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.

2 Requirements

This chapter provides a definition of the relevant requirements

2.1 Requirements that will be taken into account

2.1.1 [AR_CONTENT_0010] Compositions

Initiator:	WP 2.1.1.1
Date:	14.05.2004
Requirement:	Compositions
Description:	The SW-Component template must allow the aggregation of existing SW-Component(s) as a component, which may also be aggregated.
Rationale:	--
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.2 [AR_CONTENT_0020] Interfaces

Initiator:	WP 1.1.1
Date:	06.08.2003
Requirement:	Interfaces
Description:	The SW-Component template must allow specifying interfaces independently from the SW Components (a definition might exist even if no component uses it)
Rationale:	Reuse of interface description WP 10.x
Use Case:	• Variables to observe • Parameters/ maps to calibrate • Error-Handling
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.3 [AR_CONTENT_0030] Libraries

Initiator:	WP 2.1.1.1
Date:	14.05.2004
Requirement:	Needed libraries
Description:	The SW-Component Template must allow specifying the libraries used by the SW-Component.
Rationale:	--
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.4 [AR_CONTENT_0035] Integration on object code level

Initiator:	Martin Lunt
Date:	03.08.2004
Requirement:	Integration on object code level
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:	--

2.1.5 [AR_CONTENT_0040] Behaviour

Initiator:	WP 2.1.1.1
Date:	14.05.2004
Requirement:	Behaviour
Description:	The SW-Component Template must allow linking of the SWComponent 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:	--

2.1.6 [AR_CONTENT_0050] Schedulability

Initiator:	WP 2.1.1.1
Date:	14.05.2004
Requirement:	Schedulability
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:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.7 [AR_CONTENT_0060] Ordering

Initiator:	WP 1.1.1 (1st Workshop)
Date:	03.06.2003
Requirement:	Sequence of execution of runnable entities
Description:	The template must allow specifying: Constraint on the order of execution of different runnables entities
Rationale:	--
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.8 [AR_CONTENT_0070] Resource

Initiator:	WP2.1.1.1
Date:	13.05.2004
Requirement:	Needed resource for SW Components
Description:	The SW-Component template must provide mechanisms to describe the complete resource requirements of an AUTOSAR SWcomponent
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)

Use Case:	The SW-Component template must allow specifying the memoryconsumption of a SW Component: • ROM • NVRAM • RAM • Stack • Heap The SW-Component template must allow describing its executiontime (time during which the CPU is executing its instructions)
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.9 [AR_CONTENT_0080] Timing

Initiator:	WP 1.1.1 (1st Workshop)
Date:	03.06.2003
Requirement:	Timing-requirements of SW-Components
Description:	The SW-Component template must allow specifying the timing requirements of each runnable entity of a SW-Component (eg.): • 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:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.10 [AR_CONTENT_0090] Sensors / Actuators

Initiator:	WP2.1.1.1
Date:	14.05.2004
Requirement:	Needed and usable sensors and actuators
Description:	The SW-Component Template must allow: • listing the required sensors / actuators • enumerating the sensors / actuators that can be used • referencing external descriptions of sensors / actuators for a SW-Component
Rationale:	--
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.11 [AR_CONTENT_0100] Variant

Initiator:	WP 1.1.1 (1st Workshop)
Date:	20.08.2003
Requirement:	Variant
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:	--

2.1.12 [AR_CONTENT_0110] Modes

Initiator:	WP 1.1.1 (1st Workshop)
Date:	16.06.2004
Requirement:	Modes
Description:	The SW-Component template must provide some simple means to define modes. The name of the mode is the most important attribute that has to be provided.
Rationale:	The assumption from the SW-Component point of view is that State Managers are using a Standardized AUTOSAR Interface to influence the SW-Component and also provide an interface to get requests and confirmations from the SW-Component.
Use Case:	Each SW-Component can be relying on some modes that have to be provided on the ECU it will run later on.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.13 [AR_CONTENT_0120] Dependency on Modes

Initiator:	WP2.1.1.1
Date:	16.06.2004
Requirement:	Dependency on Modes
Description:	SW-Component must provide a matrix where the runnable entities and ports are enabled or disabled depending on the modes coming from the State Managers.
Rationale:	A SW-Component can change its active interface due to the different modes provided by the State Managers.
Use Case:	There could be a different communication behavior in the diagnostic mode than in the normal operational mode.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.14 [AR_CONTENT_0130] Support for Partial Networking

Initiator:	WP1.1.1
Date:	14.03.2011
Requirement:	Support for Partial Networking
Description:	Software Component Template shall support the definition of Virtual Function Clusters (VFC).
Rationale:	Partial networking is supported by means of a Virtual Function Cluster (VFC) on VFB Level.
Use Case:	A VFC defines all ports, which participate in the communication of a partial network. Some ports control the VFC-behaviour and other ports read out and react on the current status of the VFC.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.15 [AR_CONTENT_0140] Support for Port Groups

Initiator:	WP1.1.1
Date:	14.03.2011
Requirement:	Support for port groups
Description:	A software component can aggregate one or more port groups, which define a logical grouping of ports sharing a common functionality.
Rationale:	Port Groups are used to configure the implementation of partial networking on VFB level.
Use Case:	Virtual function clusters (VFC) are defined by the usage of port groups. A port is the smallest atomic member of a VFC, SWCs can be part of several VFCs or one VFC can contain ports of one or more SWCs.
Dependencies:	[AR_CONTENT_130] Support for partial networking
Conflicts:	--
Supporting Material:	--
Comment:	--

2.1.16 [AR_CONTENT_0150] Enable SWCs to request dedicated modes

Initiator:	WP1.1.1
Date:	14.03.2011
Requirement:	Enable SWCs to request dedicated modes
Description:	On an ECU one or more 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:	VFCs are requested or released by means of mode requests from dedicated SWCs.
Use Case:	A SWC shall control the behaviour of a partial network. It requests a mode through a certain port, which is part of the VFC, by communicating with a BSW mode manager, i.e. the BswM or ComM.
Dependencies:	[AR_CONTENT_130] Support for partial networking
Conflicts:	--

Supporting Material:	--
Comment:	--

2.1.17 [AR_CONTENT_0160] Enhancing the Non-Volatile (NV) memory interface

Initiator:	WP1.1.1
Date:	19.02.2012
Requirement:	Enhancing the Non-Volatile (NV) memory interface
Description:	Specify means to define and configure one NvBlockComponent that all SW Components in one ECU can use to access NvData.
Rationale:	Ensure data consistency by defining an NvBlockComponent with sender-receiver interfaces that will be provided to the SW-Components. By using one common NvBlockComponent the amount of RAM needed for keeping RAM copies of the NvData in each SW Component decreases.
Use Case:	One use case is to provide a really efficient (both memory and calculation) for NV memory accesses from only one SW Component in the ECU. The other use case concerns providing a NvBlockComponent that can be used as NV memory interface from many SW Components. This common NV memory interface can ensure consistency and configuration means for all SW components in the ECU.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Comment:	--

3 References

In this section the reference used in this specification are listed

3.1 References to AUTOSAR documents

[1] Glossary
AUTOSAR_Glossary.pdf

[2] Metamodel
AUTOSAR_Metamodel.eap

[3] Template Modeling Patterns
AUTOSAR_TemplateModelingPatterns.pdf

[4] Template UML Profile and Modeling Guide
AUTOSAR_TemplateModelingGuide.pdf

3.2 References to external documents

[5] Key words for use in RFCs to Indicate Requirement Levels. Network Working
Group, S. Brandner, Harvard University, 1997.
(http://www.ietf.org/rfc/rfc2119.txt).