Disclaimer

This specification as released by the AUTOSAR Development Partnership is intended for the purpose of information only. The use of material contained in this specification requires membership within the AUTOSAR Development Partnership or an agreement with the AUTOSAR Development Partnership. The AUTOSAR Development Partnership will not be liable for any use of this Specification.

Following the completion of the development of the AUTOSAR Specifications commercial exploitation licenses will be made available to end users by way of written License Agreement only.

No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Copyright © 2004-2006 AUTOSAR Development Partnership. All rights reserved.

Advice to users of AUTOSAR Specification Documents:

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 AUTOSAR compliance certification 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

Abbreviations .............................................................................................................. 7

1 Introduction .......................................................................................................... 9

2 How to read this document ................................................................................ 10

2.1 <Definition> ................................................................................................ 10

3 Definitions .......................................................................................................... 11

3.1 Application .................................................................................................. 11
3.2 Application Programming Interface (API) ................................................... 11
3.3 Application Software Component ............................................................... 11
3.4 Architecture ................................................................................................. 11
3.5 Asserted Property ....................................................................................... 12
3.6 Asset .......................................................................................................... 12
3.7 Asynchronous Communication ................................................................... 12
3.8 Atomic Software Component ...................................................................... 13
3.9 AUTOSAR Authoring Tool ......................................................................... 13
3.10 AUTOSAR Interface ................................................................................... 13
3.11 AUTOSAR Metamodel ............................................................................... 13
3.12 AUTOSAR Model ....................................................................................... 14
3.13 AUTOSAR Service ..................................................................................... 14
3.14 AUTOSAR Software-Component (SW-C) .................................................. 14
3.15 AUTOSAR Virtual Functional Bus (VFB) .................................................... 15
3.16 AUTOSAR XML description ....................................................................... 15
3.17 AUTOSAR XML Schema ........................................................................... 15
3.18 Availability .................................................................................................. 16
3.19 Basic Software (BSW) ................................................................................ 16
3.20 Basic Software Module (BSWM) ................................................................ 16
3.21 Bulk Data .................................................................................................... 16
3.22 Calibration .................................................................................................. 17
3.23 Call Point .................................................................................................... 17
3.24 Causality of Transmission .......................................................................... 17
3.25 Client.......................................................................................................... 18
3.26 Client-Server Communication .................................................................... 18
3.27 Client-Server Interface ............................................................................... 18
3.28 Cluster Signal ............................................................................................. 19
3.29 Code Variant Coding .................................................................................. 19
3.30 Communication Attribute ......................................................................... 19
3.31 Complex Device Driver (CDD) ................................................................... 20
3.32 Component .................................................................................................. 20
3.33 Composition ............................................................................................... 20
3.34 Compositionality ....................................................................................... 21
3.35 Conditioned Signal ..................................................................................... 21
3.36 Configuration ............................................................................................. 21
3.37 Confirmation ............................................................................................... 21
3.38 Conformance Test Suite (CTS) .................................................................. 22
3.39 Connector ................................................................................................... 22
3.40 Control Flow ............................................................................................... 22
<p>| 3.41 | Data........................................................................................................... 23 |
| 3.42 | Data Element ........................................................................................... 23 |
| 3.43 | Data Flow .................................................................................................. 23 |
| 3.44 | Data Variant Coding ................................................................................ 23 |
| 3.45 | Deadline .................................................................................................... 24 |
| 3.46 | Dependability ........................................................................................... 24 |
| 3.47 | Dynamic Routing ....................................................................................... 24 |
| 3.48 | ECU Abstraction Layer ............................................................................ 25 |
| 3.49 | ECU Firmware .......................................................................................... 25 |
| 3.50 | Electronic Control Unit (ECU) .................................................................. 25 |
| 3.51 | Electrical Signal ....................................................................................... 26 |
| 3.52 | Entry Point ................................................................................................ 26 |
| 3.53 | Error .......................................................................................................... 26 |
| 3.54 | Event ......................................................................................................... 26 |
| 3.55 | Execution Time ........................................................................................ 27 |
| 3.56 | Exit Point .................................................................................................. 27 |
| 3.57 | Fail-degraded ............................................................................................ 27 |
| 3.58 | Fail-operational ......................................................................................... 27 |
| 3.59 | Fail-safe ...................................................................................................... 28 |
| 3.60 | Fail-silent .................................................................................................. 28 |
| 3.61 | Failure ........................................................................................................ 28 |
| 3.62 | Fault .......................................................................................................... 29 |
| 3.63 | Fault Detection ........................................................................................... 29 |
| 3.64 | Fault Reaction .......................................................................................... 29 |
| 3.65 | Fault Tolerance ........................................................................................ 30 |
| 3.66 | Feature ....................................................................................................... 30 |
| 3.67 | Flag ............................................................................................................ 30 |
| 3.68 | Frame ........................................................................................................ 30 |
| 3.69 | Frame PDU ................................................................................................ 31 |
| 3.70 | Function .................................................................................................... 31 |
| 3.71 | Functional Network .................................................................................. 31 |
| 3.72 | Functional Unit ........................................................................................ 32 |
| 3.73 | Functionality ............................................................................................. 32 |
| 3.74 | Gateway ..................................................................................................... 32 |
| 3.75 | Gateway ECU ............................................................................................ 32 |
| 3.76 | Hardware Connection ............................................................................... 33 |
| 3.77 | Hardware Element ................................................................................... 33 |
| 3.78 | Hardware Interrupt .................................................................................. 33 |
| 3.79 | Hardware Port ........................................................................................... 33 |
| 3.80 | I-PDU ......................................................................................................... 34 |
| 3.81 | Indication .................................................................................................. 34 |
| 3.82 | Integration ................................................................................................ 34 |
| 3.83 | Interface .................................................................................................... 35 |
| 3.84 | Interrupt ................................................................................................... 35 |
| 3.85 | Interrupt Service Routine (ISR) ............................................................... 35 |
| 3.86 | Invalid Flag ............................................................................................... 35 |
| 3.87 | Invalid Value of Signal ........................................................................... 36 |
| 3.88 | Link time configuration .......................................................................... 36 |
| 3.89 | Mapping .................................................................................................... 36 |
| 3.90 | MCAL Signal ............................................................................................. 36 |</p>
<table>
<thead>
<tr>
<th>Number</th>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.91</td>
<td>Metadata</td>
<td>37</td>
</tr>
<tr>
<td>3.92</td>
<td>Microcontroller Abstraction Layer (MCAL)</td>
<td>37</td>
</tr>
<tr>
<td>3.93</td>
<td>Mistake</td>
<td>38</td>
</tr>
<tr>
<td>3.94</td>
<td>Multimedia Stream</td>
<td>38</td>
</tr>
<tr>
<td>3.95</td>
<td>Multiple Configuration Sets</td>
<td>38</td>
</tr>
<tr>
<td>3.96</td>
<td>Multiplexed PDU</td>
<td>39</td>
</tr>
<tr>
<td>3.97</td>
<td>Non-AUTOSAR Component</td>
<td>39</td>
</tr>
<tr>
<td>3.98</td>
<td>Notification</td>
<td>39</td>
</tr>
<tr>
<td>3.99</td>
<td>OS-Application</td>
<td>40</td>
</tr>
<tr>
<td>3.100</td>
<td>Partitioning</td>
<td>40</td>
</tr>
<tr>
<td>3.101</td>
<td>PCI</td>
<td>40</td>
</tr>
<tr>
<td>3.102</td>
<td>PDU</td>
<td>41</td>
</tr>
<tr>
<td>3.103</td>
<td>PDU Timeout</td>
<td>41</td>
</tr>
<tr>
<td>3.104</td>
<td>Peripheral Hardware</td>
<td>42</td>
</tr>
<tr>
<td>3.105</td>
<td>Personalization</td>
<td>42</td>
</tr>
<tr>
<td>3.106</td>
<td>Port</td>
<td>42</td>
</tr>
<tr>
<td>3.107</td>
<td>Port Interface</td>
<td>42</td>
</tr>
<tr>
<td>3.108</td>
<td>Post-build time configuration</td>
<td>43</td>
</tr>
<tr>
<td>3.109</td>
<td>Pre-Compile-Time configuration</td>
<td>43</td>
</tr>
<tr>
<td>3.110</td>
<td>Private Interface (API 3)</td>
<td>43</td>
</tr>
<tr>
<td>3.111</td>
<td>Probability of failure</td>
<td>44</td>
</tr>
<tr>
<td>3.112</td>
<td>Procedure Call</td>
<td>44</td>
</tr>
<tr>
<td>3.113</td>
<td>Process</td>
<td>44</td>
</tr>
<tr>
<td>3.114</td>
<td>Provide Port</td>
<td>44</td>
</tr>
<tr>
<td>3.115</td>
<td>Redundancy</td>
<td>45</td>
</tr>
<tr>
<td>3.116</td>
<td>Reliability</td>
<td>45</td>
</tr>
<tr>
<td>3.117</td>
<td>Relocatability</td>
<td>45</td>
</tr>
<tr>
<td>3.118</td>
<td>Require Port</td>
<td>45</td>
</tr>
<tr>
<td>3.119</td>
<td>Required property</td>
<td>46</td>
</tr>
<tr>
<td>3.120</td>
<td>Resource</td>
<td>46</td>
</tr>
<tr>
<td>3.121</td>
<td>Resource-Management</td>
<td>46</td>
</tr>
<tr>
<td>3.122</td>
<td>Response Time</td>
<td>46</td>
</tr>
<tr>
<td>3.123</td>
<td>Risk</td>
<td>47</td>
</tr>
<tr>
<td>3.124</td>
<td>Robustness</td>
<td>47</td>
</tr>
<tr>
<td>3.125</td>
<td>RTE Event</td>
<td>47</td>
</tr>
<tr>
<td>3.126</td>
<td>Runnable Entity</td>
<td>48</td>
</tr>
<tr>
<td>3.127</td>
<td>Safety</td>
<td>48</td>
</tr>
<tr>
<td>3.128</td>
<td>Sample Application</td>
<td>48</td>
</tr>
<tr>
<td>3.129</td>
<td>Scalability</td>
<td>48</td>
</tr>
<tr>
<td>3.130</td>
<td>Scheduler</td>
<td>49</td>
</tr>
<tr>
<td>3.131</td>
<td>SDU</td>
<td>49</td>
</tr>
<tr>
<td>3.132</td>
<td>Security</td>
<td>49</td>
</tr>
<tr>
<td>3.133</td>
<td>Sender-Receiver Communication</td>
<td>49</td>
</tr>
<tr>
<td>3.134</td>
<td>Sender-Receiver Interface</td>
<td>50</td>
</tr>
<tr>
<td>3.135</td>
<td>Sensor/Actuator SW-Component</td>
<td>50</td>
</tr>
<tr>
<td>3.136</td>
<td>Server</td>
<td>50</td>
</tr>
<tr>
<td>3.137</td>
<td>Service</td>
<td>51</td>
</tr>
<tr>
<td>3.138</td>
<td>Service Port</td>
<td>51</td>
</tr>
<tr>
<td>3.139</td>
<td>Services Layer</td>
<td>51</td>
</tr>
<tr>
<td>3.140</td>
<td>Shipping</td>
<td>52</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Software Configuration</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Software Interrupt</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Software Module</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Software Signal</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Special Periphery Access</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Standard Periphery Access</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Standardized AUTOSAR Interface</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Standardized Interface</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Standard Software</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Static Configuration</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Synchronous Communication</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>System Constraint</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>System Signal</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Task</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>Technical Signal</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>Template</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>Timeout</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>Use Case</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>Validation</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>Variant Coding</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>VFB View</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Virtual Integration</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Worst Case Execution Time</td>
<td>59</td>
<td></td>
</tr>
<tr>
<td>Worst Case Response Time</td>
<td>59</td>
<td></td>
</tr>
</tbody>
</table>

Annex 1: Literature ................................................................. 60
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>Analog Digital Converter</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ASAM</td>
<td>Association for Standardization of Automation and Measuring Systems</td>
</tr>
<tr>
<td>AUTOSAR</td>
<td>Automotive Open System Architecture</td>
</tr>
<tr>
<td>BSW</td>
<td>Basic Software</td>
</tr>
<tr>
<td>BSWM</td>
<td>Basic Software Modul</td>
</tr>
<tr>
<td>CAN</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>CDD</td>
<td>Complex Device Driver</td>
</tr>
<tr>
<td>COM</td>
<td>Communication</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CTA</td>
<td>Conformance Test Agency</td>
</tr>
<tr>
<td>CTS</td>
<td>Conformance Test Suite</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital to Analog Converter</td>
</tr>
<tr>
<td>DIO</td>
<td>Digital Input/Output</td>
</tr>
<tr>
<td>DTD</td>
<td>Document Type Definition</td>
</tr>
<tr>
<td>ECU</td>
<td>Electronic Control Unit</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In First Out</td>
</tr>
<tr>
<td>FPU</td>
<td>Floating Point Unit</td>
</tr>
<tr>
<td>FW</td>
<td>Fire Wire</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communication</td>
</tr>
<tr>
<td>HIS</td>
<td>Hersteller Initiative Software</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electronical Commission</td>
</tr>
<tr>
<td>I-PDU</td>
<td>Interaction Layer Protocol Data Unit</td>
</tr>
<tr>
<td>ISR</td>
<td>Interrupt Service Routine</td>
</tr>
<tr>
<td>LIFO</td>
<td>Last In First Out</td>
</tr>
<tr>
<td>LIN</td>
<td>Local Interconnected network</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Bit</td>
</tr>
<tr>
<td>μC</td>
<td>MicroController</td>
</tr>
<tr>
<td>MCAL</td>
<td>MicroController Abstraction Layer</td>
</tr>
<tr>
<td>MIPS</td>
<td>Million Instructions Per Second</td>
</tr>
<tr>
<td>MMU</td>
<td>Memory Management Unit</td>
</tr>
<tr>
<td>MMI</td>
<td>Man Machine Interface</td>
</tr>
<tr>
<td>MOST</td>
<td>Media Oriented Systems Transport</td>
</tr>
<tr>
<td>μP</td>
<td>MicroProcessor</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>MPU</td>
<td>Memory Protection Unit</td>
</tr>
<tr>
<td>MSB</td>
<td>Most Significant Bit</td>
</tr>
<tr>
<td>NVRAM</td>
<td>Non-Volatile Random Access Memory</td>
</tr>
<tr>
<td>OEM</td>
<td>Original Equipment Manufacturer</td>
</tr>
<tr>
<td>OIL</td>
<td>OSEK Implementation Language</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>OSEK</td>
<td>Open Systems and the Corresponding Interfaces for Automotive Electronics</td>
</tr>
<tr>
<td>PCI</td>
<td>Protocol Control Information</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>PS</td>
<td>Product Supplier</td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse Width Modulation</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Change</td>
</tr>
<tr>
<td>RTE</td>
<td>Runtime Environment</td>
</tr>
<tr>
<td>SDU</td>
<td>Service Data Unit</td>
</tr>
<tr>
<td>SIL</td>
<td>Safety Integrity Level</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>SW-C</td>
<td>Software Component</td>
</tr>
<tr>
<td>TTP</td>
<td>Time Triggered Protocol</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>VFB</td>
<td>Virtual Functional Bus</td>
</tr>
<tr>
<td>WCET</td>
<td>Worst Case Execution Time</td>
</tr>
<tr>
<td>WCRT</td>
<td>Worst Case Response time</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
1 Introduction

This document is the overall glossary of AUTOSAR. It contains definitions of all major terms and notions used within AUTOSAR. It does not claim to be complete and please keep in mind that some WPs have more specific terms defined within their domain specific glossary.

The document is written in English, but to guarantee consistency in all languages AUTOSAR aims for translating all terms to different languages. Currently the translation has partly been done in the following languages:

- German
- French

In future more languages might follow.
2 How to read this document

The title of the subchapters is identical to the term to be defined.

2.1 <Definition>

<table>
<thead>
<tr>
<th>Definition</th>
<th>tbd - term to be defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>tbd – person and/or group who introduced the term</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>tbd – further explanation of the definition</td>
</tr>
<tr>
<td>Comment</td>
<td>tbd – comment or hints</td>
</tr>
<tr>
<td>Example</td>
<td>tbd – example of the term</td>
</tr>
<tr>
<td>Reference</td>
<td>tbd – reference of definition</td>
</tr>
<tr>
<td>German Term</td>
<td>tbd – German translation of the term</td>
</tr>
<tr>
<td>French Term</td>
<td>tbd – French translation of the term</td>
</tr>
</tbody>
</table>
3 Definitions

3.1 Application

**Definition**
A software (or program) that is specified to the solution of a problem of an end user requiring information processing for its solution. The software configuration (→ definition 3.141) of a software entity.

**Initiator**
WP 1.1.1

**Further Explanations**
To 1. of Definition:
In AUTOSAR Application software is located above the AUTOSAR RTE (RunTimeEnvironment).

**Comment**
Definition 1 is the “by default” meaning for application in AUTOSAR. When definition 2 is meant, it has to be explicitly mentioned.

**Example**

**Reference**
[ISO 2382-20]

**German Term**
Applikation

**French Term**
--

3.2 Application Programming Interface (API)

**Definition**
An Application Programming Interface (API) is the prescribed method of a specific software part by which a programmer writing a program can make requests to that software part.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
OSEK OS API

**Reference**
--

**German Term**
--

**French Term**
--

3.3 Application Software Component

**Definition**
An Application Software Component is a specific AUTOSAR Software Component (→ definition 3.14) realizing a defined functionality of a set (one or more) of features.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
Applikations Software Komponente

**French Term**
--

3.4 Architecture

**Definition**
The fundamental organization of a system embodied in its components, their static and dynamic relationships to each other, and to the environment, and the principles guiding its design and evolution.

**Initiator**
WP 1.1.1

**Further**
--
### 3.5 Asserted Property

**Definition**
A property or quality of a design entity (e.g. SW component or system) is asserted, if the design entity guarantees that this property or quality is fulfilled.

**Initiator**
WP 10.1

**Further Explanations**
A property or quality of a design unit can be asserted by the design unit itself or in combination with another design unit.

**Comment**

**Example**
If the worst case execution time of a task (w.r.t. a certain CPU etc.) is asserted to be 3 ms, the execution time of this task will under any circumstances be less than or equal to 3 ms.

**Reference**
Compare required property (→ definition 3.119)

**German Term**
Zugesichert

**French Term**

---

### 3.6 Asset

**Definition**
An item that has been designed for use in multiple contexts.

**Initiator**
WP 1.1.1

**Further Explanations**

**Comment**

**Example**
An asset can be design, specifications, source code, documentation, test suits, manual procedures, etc..

**Reference**
[IEEE 1517], [EAST-Glossary]

**German Term**

**French Term**

---

### 3.7 Asynchronous Communication

**Definition**
Asynchronous communication does not block the sending software entity. The sending software entity continues its operation without getting a response from the communication partner(s).

**Initiator**
WP 1.1.1

**Further Explanations**
There could be an acknowledgement by the communication system about the sending of the information. A later response to the sending software entity is possible.

**Comment**

**Example**

**Reference**

**German Term**

**French Term**

---

---
### 3.8 Atomic Software Component

<table>
<thead>
<tr>
<th>Definition</th>
<th>Smallest non-dividable software entity connected to the AUTOSAR Virtual Functional Bus (definition 3.15).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>An Atomic Software Component might access HW or not, therefore not all Atomic SW-Cs are relocatable.</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>AUTOSAR Software Component, Complex Device Driver</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Atomare Software Komponente</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.9 AUTOSAR Authoring Tool

<table>
<thead>
<tr>
<th>Definition</th>
<th>An AUTOSAR authoring tool is a software tool which supports interpreting, processing and creating of AUTOSAR XML descriptions (definition 3.16).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.2</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.10 AUTOSAR Interface

<table>
<thead>
<tr>
<th>Definition</th>
<th>The AUTOSAR Interface of a component (definition 3.32) refers to the collection of all ports (definition 3.106) of that component through which it interacts with other components.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>Note that an AUTOSAR Interface is different from a Port Interface (definition 3.107). The latter characterizes one specific port of a component.</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[AUTOSAR Specification of Virtual Functional Bus], Chapter “Modeling of Communication, Graphical Notation”</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.11 AUTOSAR Metamodel

<table>
<thead>
<tr>
<th>Definition</th>
<th>The AUTOSAR metamodel is a UML2.0 model that defines the language for describing AUTOSAR systems and related artifacts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>MMT</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>The AUTOSAR metamodel is a graphical representation of a template (definition 3.157). 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.</td>
</tr>
<tr>
<td>Comment</td>
<td>The AUTOSAR XML Schema (definition 3.17) is derived from the AUTOSAR</td>
</tr>
</tbody>
</table>
### 3.12 AUTOSAR Model

**Definition**
An AUTOSAR model is an instance of the AUTOSAR metamodel (definition 3.11). The information contained in the AUTOSAR model (definition 3.12) 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.

**Initiator**
WP 1.1.1

### 3.13 AUTOSAR Service

**Definition**
An AUTOSAR Service is a logical entity of the basic software (definition 3.19) offering general functionality to be used by various AUTOSAR software components. The functionality is accessed via standardized AUTOSAR Interfaces (definition 3.10).

**Initiator**
WP 1.1.1

### 3.14 AUTOSAR Software-Component (SW-C)

**Definition**
The AUTOSAR Software Components encapsulate an application which runs on the AUTOSAR infrastructure. The AUTOSAR Software Components have well-defined interfaces, which are described and standardized within AUTOSAR.

**Initiator**
WP 1.1.1
### 3.15 AUTOSAR Virtual Functional Bus (VFB)

**Definition**
The AUTOSAR Virtual Functional Bus is an abstraction of the communication between Atomic Software Components (definition 3.8) and AUTOSAR Services (definition 3.13). This abstraction is such that specification of the communication mechanisms is independent from the concrete technology chosen to realize the communication.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 1.1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Further Explanations</strong></td>
<td>After compilation and linking of software for a dedicated ECU (definition 3.50) the AUTOSAR Virtual Functional Bus interfaces are realized by the AUTOSAR Runtime Environment.</td>
</tr>
</tbody>
</table>

### 3.16 AUTOSAR XML description

**Definition**
An AUTOSAR XML description describes the XML representation of an AUTOSAR model (definition 3.12). The AUTOSAR XML description can consist of several fragments (e.g. files). Each individual fragment must validate successfully against the AUTOSAR XML schema.

### 3.17 AUTOSAR XML Schema

**Definition**
The AUTOSAR XML Schema is an XML language definition for exchanging AUTOSAR models (definition 3.12) and descriptions.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Further Explanations</strong></td>
<td>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 (definition 3.11). The AUTOSAR XML Schema defines the AUTOSAR data exchange format.</td>
</tr>
</tbody>
</table>

**Reference**

### 3.18 Availability

**Definition**
1. Probability that a system or functional unit is able to perform its normal operation under specified conditions at a specific time.
2. The property of data or resources being accessible and usable on demand by an authorized entity.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 1.1.3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Further Explanations</strong></td>
<td>The time can be used to model a decrease of the availability over time due to e.g. aging of components.</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>1. Degraded modes are covered by this definition (see example)</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>1. Power Steering: if the support function fails it is not available while the steering as a base function has full availability.</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>based on [ISO 2382-14], [ISO 2382-8], Reliability (→ definition 3.116)</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>Verfügbarkeit</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.19 Basic Software (BSW)

**Definition**
The Basic Software provides the infrastructural (schematic dependent and schematic independent) functionalities of an ECU (→ definition 3.50). It consists of ECU Firmware (→ definition 3.49) and Standard Software (→ definition 3.149).

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 1.1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Further Explanations</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>MCAL, AUTOSAR services, communication layer</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>Basis Software</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.20 Basic Software Module (BSWM)

**Definition**
A collection of software files (code and description) that define a certain basic software functionality present on an ECU.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 4.1.1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Further Explanations</strong></td>
<td>Standard software (→ definition 3.149) may be composed of several software modules (→ definition 3.143) that are developed independently. A software module may consist of ECU firmware (→ definition 3.49), and/or standard software (→ definition 3.149).</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>A Digital IO Driver, Complex Device Driver, OS are examples of basic software modules.</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>Basis-Software-Modul</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.21 Bulk Data

**Definition**
“Bulk Data” is a set of data such big in size, that standard mechanisms used to handle smaller data sets become inconvenient. This implies that bulk data in a software system are modeled, stored, accessed and transported by different
### Glossary

| AUTOSAR confidential |

#### 3.22 Calibration

**Definition**
Calibration is the adjustment of parameters of SW-Components realizing the control functionality (namely parameters of AUTOSAR SWCs, ECU abstraction or Complex Device Drivers).

**Initiator**
WP 1.1.1

**Further Explanations**
Only those software modules can be calibrated, which are above RTE and ECU Abstraction and CDD. Calibration is always done at post-build-time. Used techniques to set calibration data include end-of-line programming, garage programming and adaptive calibration (e.g. in the case of anti-pinch protection for power window).

**Comment**
--

**Example**
The calibration of the engine control will take into account the production differences of the individual motor this system will control.

**Reference**
--

**German Term**
Kalibrierung

**French Term**
--

#### 3.23 Call Point

**Definition**
A point in a AUTOSAR Software-Component (→ definition 3.14) where the SW-C enforce an execution entity (Entry point → definition 0) in an other SW-C.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
Request Service
Send Information

**Reference**
--

**German Term**
Aufrufpunkt

**French Term**
--

#### 3.24 Causality of Transmission

**Definition**
Transmit order of PDUs with the same identifier (instances of PDUs) from a source network is preserved in the destination network.

**Initiator**
WP4.2.2.1.6 Gateway
### Further Explanations
Transmission of PDUs (\(\rightarrow\) definition 3.102) with the same identifier has a particular temporal order in a given source network. After routing over a gateway the temporal order of transmission of PDUs in a destination network may be changed. Only in case that the temporal order is the same, causality is given. Otherwise causality is violated. Causality can be in contradiction to prioritization of PDUs.

| Comment | -- |
| Example | -- |
| Reference | -- |

### German Term
Einhaltung der Übertragungsreihenfolge bei Botschaften mit demselben Identifier

### French Term
--

### 3.25 Client

**Definition**
Software entity which uses services of a server (\(\rightarrow\) definition 3.136).

**Initiator**
WP 1.1.1

**Further Explanations**
The client and the server might be located on one ECU (\(\rightarrow\) definition 3.50) or distributed on different calculation units (e.g. ECU, external diagnostic tester).

**Comment**
Adapted from Balzert.

**Example**
--

**Reference**
[Balzert99]

| German Term | -- |
| French Term | -- |

### 3.26 Client-Server Communication

**Definition**
A specific form of communication in a possibly distributed system in which software entities act as clients (\(\rightarrow\) definition 3.25), servers (\(\rightarrow\) definition 3.136) or both, where 1...n clients are requesting services via a specific protocol from typically one server.

**Initiator**
WP 1.1.1

**Further Explanations**
Client-server communication can be realized by synchronous or asynchronous communication.
- Client takes initiative: requesting that the server performs a service, e.g. client triggers action within server (server does not start action on its own)
- Client is after service request blocked / non-blocked
- Client expects response from server: data flow (+ control flow, if blocked)

One example for 1 client to n server communication (currently not supported) is a functional request by diagnosis. This has to be treated as a specific exception.

**Comment**
Adapted from Hyper Dictionary

**Example**
Internet (TCP/IP)

**Reference**
[Hyper Dictionary]

| German Term | -- |
| French Term | -- |

### 3.27 Client-Server Interface

**Definition**
The client-server interface is a special kind of port-interface (\(\rightarrow\) definition 3.107) used for the case of client-server communication (\(\rightarrow\) definition 3.26). The client-server interface defines the operations that are provided by the server (\(\rightarrow\) definition 3.136) and that can be used by the client (\(\rightarrow\) definition 3.25).

**Initiator**
System Team
3.28 Cluster Signal

**Definition**
A cluster signal represents the aggregating system signal on one specific communication cluster. Cluster signals can be defined independently of frames. This allows a development methodology where the signals are defined first, and are assigned to frames in a later stage.

**Initiator**
WP 2.1.1.3

**Further Explanations**

**Comment**

**Example**

**Reference**
[AUTOSAR Specification of Virtual Functional Bus]

**German Term**

**French Term**

3.29 Code Variant Coding

**Definition**
Adaptation of SW by selection of functional alternatives according to external requirements

**Initiator**
WP 1.1.1

**Further Explanations**
Code Variant Coding might influences RTE (RuntimeEnvironment) and BSW modules (definition 3.20), not only the application software modules. Code Variant Coding is always done at pre-compile-time or at link-time. Code Variant Coding also includes vehicle-specific (not user-specific) SW adaptation due to end-customer wishes (e.g. deactivation of speed dependent automatic locking).

**Comment**
In case of the C language the #if or #ifdef directive can be used for creating code variants. Code Variant Coding is a design time concept.

**Example**
The same window lifter ECU is used for cars with 2 and 4 doors, however different code segments have to be used in both cases.

**Reference**
[AUTOSAR_WP2.1.1.1_SW-C-Attributes]

**German Term**
Variantenkodierung

**French Term**

3.30 Communication Attribute

**Definition**
Communication attributes define, according to the development phase, behavioral as well as implementation aspects of the AUTOSAR communication patterns.

**Initiator**
WP 1.1.1

**Further Explanations**
The exact characteristics of the communication patterns provided by AUTOSAR (client-server and sender-receiver) can be specified more precisely by communication attributes.

**Comment**
See chapter 4.1.6 in Specification of the Virtual Functional Bus

**Example**

**Reference**
[AUTOSAR Specification of Virtual Functional Bus]

**German Term**
Kommunikationsattribut

**French Term**
3.31 Complex Device Driver (CDD)

**Definition**
An Atomic Software Component (→ definition 3.8) that on one side interfaces with the AUTOSAR VFB (→ definition 3.15) and on the other side directly accesses Peripheral Hardware (→ definition 3.104) and/or ECU-Abstraction (→ definition 3.48) and/or MCAL (→ definition 3.92). Complex Device Driver can be accessed via AUTOSAR Interfaces and/or directly by Basic Software Modules.

**Initiator**
WP 1.1.1

**Further Explanations**
SW situated below the AUTOSAR RTE. This software in general is not relocatable.

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
--

**French Term**
Attribut de communication

3.32 Component

**Definition**
A Component encapsulates a complete (automotive) functionality or a piece of it. It is described on the basis of a formal specification, that allows the seamless integration within the VFB (→ definition 3.15). A Component has well defined Ports (→ definition 3.106), through which the Component can interact with other Components.

**Initiator**
Bertrand Delord

**Further Explanations**
When modeling a system with AUTOSAR, a logical interconnection of components can be packaged as a component. Such a component is called a "composition". In contrast to the Atomic Software Components (→ definition 3.8), the components inside a composition can be distributed over several ECUs.

**Comment**
--

**Example**
--

**Reference**
[AUTOSAR Technical Overview]

**German Term**
--

**French Term**
--

3.33 Composition

**Definition**
An AUTOSAR Composition encapsulates a collaboration of Components (→ definition 3.32), thereby hiding detail and allowing the creation of higher abstraction levels. Through Delegation Connectors (→ definition 3.39) a Composition (→ definition 3.33) explicitly specifies, which Ports (→ definition 3.106) of the internal components are visible from the outside. AUTOSAR Compositions are a type of Components, e.g. they can be part of further compositions.

**Initiator**
WP 2.1.1.1

**Further Explanations**
--

**Comment**
See Virtual Functional Bus Specification, Chapter "VFB View, Meta-Model"

**Example**
--

**Reference**
[AUTOSAR Specification of Virtual Functional Bus]

**German Term**
Komposition

**French Term**
--
3.34 Compositionality

**Definition**
Compositionality is given when the behavior of a software component or subsystem of a system is independent of the overall system load and configuration.

**Initiator**
BMW

**Further Explanations**
Compositionality is an important property of deterministic systems. This property leads to a complete decoupling of systems. Smooth subsystem integration without backlashes is then easily achievable.

**Comment**
--

**Example**
A new component or a subsystem can be added to a system without changing the behavior of the original components.

**Reference**
--

**German Term**
Zusammensetbarkeit

**French Term**
--

3.35 Conditioned Signal

**Definition**
The conditioned signal is the internal electrical representation of the electrical signal within the ECU. It is delivered to the processor and represented in voltage and time (or, in case of logical signals, by high or low level).

**Initiator**
--

**Further Explanations**
The Electrical Signal (→ definition 3.51) usually can not be processed by the peripherals directly, but has to be adopted. This includes amplification and limitation, conversion from a current into a voltage and so on. This conversion is performed by some electronical devices in the ECU and the result of the conversion is called the Conditioned Signal.

The description means for the Conditioned Signal can also be the same as for Technical Signals (→ definition 3.156) and Electrical Signals, but limited to electrical voltage.

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
--

**French Term**
--

3.36 Configuration

**Definition**
The arrangement of hardware and/or software elements in a system.

**Initiator**
WP 1.1.1

**Further Explanations**
A configuration in general takes place before runtime.

**Comment**
--

**Example**
--

**Reference**
[AST-Glossary], [SO 61511-1]

**German Term**
Konfiguration

**French Term**
--

3.37 Confirmation

**Definition**
Service primitive defined in the ISO/OSI Reference model (ISO 7498). With the 'confirmation' service primitive a service provider informs a service user about the result of a preceding service request of the service user [OSEK BD]

**Initiator**
WP 1.1.1
Further Explanations: A confirmation is e.g. a specific notification generated by the OSEK underlying layer to inform about a Message Transmission Error.

Example: OSEK Com notification class 2 and 4

Reference: [SEK BD], [OSEK Com]

German Term: -

French Term: -

### 3.38 Conformance Test Suite (CTS)

**Definition:** is a test implementation used in the context of Conformance Testing. Typically, multiple test implementations from different vendors (e.g. CTA) will exist, each of which implements the standardized Conformance Test specifications.

**Initiator:** WP 20

**Further Explanations:** ISO 9646 distinguishes between Abstract Test Suites and Executable Test Suites. For AUTOSAR the earlier relates to the Conformance Test Specifications, whereas the latter to the test implementations or Conformance Test Suites.

**Comment:** -

**Example:** -

**Reference:** [ISO 9646, Parts 1,2 and 4]

**German Term:** -

**French Term:** -

### 3.39 Connector

**Definition:** A connector connects ports (definition 3.106) of components (definition 3.32) and represents the flow of information between those ports.

**Initiator:** WP 1.1.1

**Further Explanations:** -

**Comment:** For more information see AUTOSAR Specification of VFB

**Example:** AssemblyConnector, DelegationConnector

**Reference:** [AUTOSAR Specification of Virtual Function Bus]

**German Term:** Konnektor

**French Term:** -

### 3.40 Control Flow

**Definition:** The directed transmission of information between multiple entities, directly resulting in a state change of the receiving entity.

**Initiator:** WP 1.1.1

**Further Explanations:** A state change could result in an activation of a schedulable entity.

**Comment:** -

**Example:** -

**Reference:** -

**German Term:** Kontrollfluss

**French Term:** -
### 3.41 Data

**Definition**
A reinterpretable representation of information in a formalized manner suitable for communication, interpretation or processing.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
Flag, Notification, etc.

**Reference**
[ISO 2382-1]

**German Term**
Daten

**French Term**
--

### 3.42 Data Element

**Definition**
Data elements are declared within the context of a "Sender-Receiver Interface" (definition 3.134). They serve as the data units that are exchanged between sender and receiver.

**Initiator**
Stefaan Sonck Thiebaut

**Further Explanations**
--

**Comment**
--

**Example**
--

**Reference**
[AUTOSAR SoftwareComponentTemplate]

**German Term**
Datenelement

**French Term**
--

### 3.43 Data Flow

**Definition**
The directed transmission of data (definition 3.41) between multiple entities. The transmitted data are not directly related to a state change at the receiver side.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
Asynchronous communication.

**Reference**
--

**German Term**
Datenfluss

**French Term**
--

### 3.44 Data Variant Coding

**Definition**
Adaptation of SW by setup of certain characteristic data according to external requirements.

**Initiator**
WP 1.1.1

**Further Explanations**
Data Variant Coding might influence RTE (RunTimeEnvironment) and BSW modules (definition 3.20) not only the application software modules. (Multiple configuration parameter sets are needed.) Variant Coding is always done at post-build-time. Variant Coding also includes vehicle-specific (not user-specific) SW adaptation due to end-customer wishes (e.g. deactivation of speed dependent automatic locking). Used techniques to select variants include end-of-line programming and garage programming.
### 3.45 Deadline

**Definition**
The point in time when an execution of an entity must be finished.

**Initiator**
WP 1.1.1

**Further Explanations**
A deadline is calculated dependent on its local reference system.

**Comment**
--

**Example**
--

**Reference**
[OSEKtime, OS Specification]

**German Term**
--

**French Term**
--

### 3.46 Dependability

**Definition**
Dependability is defined as the trustworthiness of a computer system such that reliance can justifiable be placed on the service it delivers.

**Initiator**
WP 1.1.1

**Further Explanations**

**Comment**
--

**Example**
--

**Reference**
[EAST-Glossary]

**German term**
Verlässlichkeit

**French term**
--

### 3.47 Dynamic Routing

**Definition**
The routing of signals or PDUs (definition 3.102) in a gateway can be changed throughout operation without change of the operation mode of the gateway.

**Initiator**
WP4.2.2.1.6 Gateway

**Further Explanations**
Dynamic routing requires the change of routing tables during operation. It is not intended to use dynamic routing in the gateway.

**Comment**
--

**Example**
--

**Reference**
[EAST-Glossary]

**German term**
Verlässlichkeit

**French term**
--
### 3.48 ECU Abstraction Layer

| Definition | The ECU Abstraction Layer is located above the Microcontroller Abstraction Layer (→ definition 3.92) and abstracts from the ECU schematic. It is implemented for a specific ECU and offers an API for access to peripherals and devices regardless of their location (onchip/offchip) and their connection to the microcontroller (port pins, type of interface). Task: make higher software layers independent of the ECU hardware layout. |
| Initator | WP 1.1.2 |
| Further Explanations | The ECU Abstraction Layer consists of the following parts:  
• I/O Hardware Abstraction  
• Communication Hardware Abstraction  
• Memory Hardware Abstraction  
• Onboard Device Abstraction  
Properties:  
• Implementation: µC independent, ECU hardware dependent  
• Upper Interface (API): µC and ECU hardware independent, dependent on signal type |
| Comment | -- |
| Example | See WP1.1.2, Layered Software Architecture |
| Reference | [AUTOSAR Software Architecture] |
| German Term | Steuergeräte-Abstraktions-Schicht |
| French Term | -- |

### 3.49 ECU Firmware

| Definition | ECU firmware is ECU schematic dependent software located below the AUTOSAR RTE (RunTimeEnvironment). |
| Initator | WP 1.1.1 |
| Further Explanations | -- |
| Comment | -- |
| Example | ECU Abstraction, Complex Device Driver |
| Reference | -- |
| German Term | -- |
| French Term | -- |

### 3.50 Electronic Control Unit (ECU)

| Definition | Embedded computer system consisting out of at least one CPU and corresponding periphery which is placed in one housing. |
| Initator | WP 1.1.1 |
| Further Explanations | -- |
| Comment | “Small” deleted from EAST definition  
The term ECU is problematic using LIN busses: An ECU can carry at least one AUTOSAR-SW-Component. Sensors and Actuators which are according to this definition not an ECU are related to an ECU which capsules the sensors or actuators functionality -new |
| Example | Head Unit (telematics domain). |
| Reference | [EAST-Glossary] |
| German Term | Steuergerät |
| French Term | -- |
3.51 Electrical Signal

**Definition**
The electrical signal is the electrical representation of technical signals (→ definition 3.156). Electrical signals can only be represented in voltage, current and time.

**Initiator**
WP 2.1.1.2

**Further Explanations**
When a sensor processes the Technical Signal it is converted into an Electrical Signal. The information can be provided in the current, the voltage or in the timely change of the signal (e.g. a pulse width modulation).

**Comment**
To describe the Electrical Signal the same means as for the Technical Signal can be used, limited to electrical current and voltage.

**Example**
--

**Reference**
--

**German Term**
Elektrisches Signal

**French Term**
--

3.52 Entry Point

**Definition**
A point in a AUTOSAR Software-Component (→ definition 3.14) where an execution entity of the SW-C begins.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
- Service of the Server in Client/Server Communication
- Reaction after receive Information (Notification)

**Reference**
--

**German Term**
Einsprungpunkt

**French Term**
--

3.53 Error

**Definition**
Discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition.

**Initiator**
WP 1.1.3

**Further Explanations**
--

**Comment**
--

**Example**
--

**Reference**
[IEC 61508, Part 4]

**German Term**
Abweichung

**French Term**
--

3.54 Event

**Definition**
State change of a hardware and/or software entity.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
Ereignis
### 3.55 Execution Time

**Definition**
The time during which a program is actually executing, or more precisely during which a certain thread of execution is active.

**Initiator**
WP 2.1.1.1

**Further Explanations**
The execution time of software is the time during which the CPU is executing its instructions. The time the CPU spends on task switches or on the execution of other pieces of software is not considered here. See also: response time, worst case execution time, worst case response time.

**Example**

**Reference**

**German Term**
Ausführungszeit

**French Term**

### 3.56 Exit Point

**Definition**
A point in an AUTOSAR Software-Component (definition 3.14) where an execution entity of the SW-C ends.

**Initiator**
WP 1.1.1

**Further Explanations**

**Comment**

**Example**
Return point.

**Reference**

**German Term**
Ausstiegspunkt

**French Term**

### 3.57 Fail-degraded

**Definition**
Property of a system or functional unit. Describes the ability of a system to continue with intended degraded operation at its output interfaces despite the presence of hardware or software faults.

**Initiator**
WP 1.1.3

**Further Explanations**

**Comment**
1. Safety means are not regarded as a part of the normal functionality respectively operation.
2. Also known as: Fail-reduced, Fail-soft

**Example**
“Limp home” functionality for ECU (reduce torque to assure an arrival at home or service station)

**Reference**

**German Term**

**French Term**

### 3.58 Fail-operational

**Definition**
Property of a system or functional unit. Describes the ability of a system or functional unit to continue normal operation at its output interfaces despite the presence of hardware or software faults.
### 3.59 Fail-safe

<table>
<thead>
<tr>
<th>Definition</th>
<th>Property of a system or functional unit. In case of a fault the system or functional unit transits to a safe state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
</tbody>
</table>
| Comment | 1. Typically, a fail-operational system or functional unit has no safe state.  
2. Safety means are not regarded as a part of the normal functionality respectively operation. |
| Example | Braking system                                                                                              |
| Reference |                                                                                                           |
| German Term |                                                                                                          |
| French Term |                                                                                                           |

### 3.60 Fail-silent

<table>
<thead>
<tr>
<th>Definition</th>
<th>Property of a system or functional unit. In case of a fault the output interfaces are disabled in a way that no further outputs are made.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
<tr>
<td>Comment</td>
<td>Fail-silent is a special case of the fail-safe property.</td>
</tr>
<tr>
<td>Example</td>
<td>The fail-silent property can be used to avoid that “babbling idiots” disturb the overall communication.</td>
</tr>
<tr>
<td>Reference</td>
<td></td>
</tr>
<tr>
<td>German Term</td>
<td></td>
</tr>
<tr>
<td>French Term</td>
<td></td>
</tr>
</tbody>
</table>

### 3.61 Failure

<table>
<thead>
<tr>
<th>Definition</th>
<th>Termination of the ability of a system or functional unit to perform a required function.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
<tr>
<td>Comment</td>
<td>The cause of a failure is a fault.</td>
</tr>
<tr>
<td>Example</td>
<td></td>
</tr>
<tr>
<td>Reference</td>
<td>[IEC 61508, Part 4]</td>
</tr>
<tr>
<td>German Term</td>
<td>Ausfall, Fehlfunktion</td>
</tr>
<tr>
<td>French Term</td>
<td></td>
</tr>
</tbody>
</table>
### 3.62 Fault

<table>
<thead>
<tr>
<th>Definition</th>
<th>Abnormal condition that may cause a reduction in, or loss of, the capability of a system or functional unit to perform a required function.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>A fault of a functional unit may cause a failure of this unit.</td>
</tr>
<tr>
<td>Example</td>
<td>Physical defect</td>
</tr>
<tr>
<td>Reference</td>
<td>[IEC 61508, Part 4]</td>
</tr>
<tr>
<td>German Term</td>
<td>Störung</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.63 Fault Detection

<table>
<thead>
<tr>
<th>Definition</th>
<th>The action of monitoring errors and setting fault states to specific values is called fault detection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
</tbody>
</table>
| Further Explanations                                                      | The different states are called “not detected” / “present” / “intermittent or maturing” /…  
The names of the fault states are following the ISO/SAE norms; however there is a coordination step in between the states of the DTCs (Diagnostic Trouble Code → see definition in ISO 15765/ ISO14229) and the states of the faults.  
The SW-C’s Fault Detection is executed decentralized, e.g. each SW-C sets the state of a fault according to the defined fault qualification (SW-C Template). Therefore the Fault Detection is implemented in the SW-C (SW-C could be either Application SW Component or Basic SW Component). There are exceptions; these will be pointed out individually for each fault. The SW-C’s developer will define the conditions (=fault qualification), when these conditions are fulfilled the SW-C notifies a fault to the Diagnostic Memory Management. |
| Comment                                                                  | --                                                                                                                                |
| Example                                                                  | --                                                                                                                                |
| Reference                                                                 | [ISO 15765], [ISO14229]  
[AUTOSAR Specification of Virtual Functional Bus]                                                                 |
| German Term                                                               | Fehlerdetektion                                                                                                                  |
| French Term                                                               | Détection de fautes                                                                                                             |

### 3.64 Fault Reaction

<table>
<thead>
<tr>
<th>Definition</th>
<th>In case of a Failure of a SW-C there is a specific action to be carried out. This action is called “Fault Reaction”.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>Fault Reactions can be implemented decentralized in the SW-C. There might also be the need of coordinating the fault reactions since there are reactions excluding each other. This will be done by a central fault reaction manager.</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Fehlerbehandlung</td>
</tr>
<tr>
<td>French Term</td>
<td>Traitement des fautes</td>
</tr>
</tbody>
</table>
### 3.65 Fault Tolerance

**Definition**  
Property of a system or functional unit. In case of n faults the system or functional unit continues with full functionality (n>0).

**Initiator**  
WP 1.1.3

**Further Explanations**  
--

**Comment**  
--

**Example**  
--

**Reference**  
--

**German Term**  
Fehlertoleranz

**French Term**  
--

### 3.66 Feature

**Definition**  
The term feature is commonly used in the software tool community to describe characteristics (functionality) of the software.

**Initiator**  
WP 1.2

**Further Explanations**  
In AUTOSAR a feature is represented by one or many metaclasses and their attributes in the AUTOSAR meta-model. Features are used to implement use cases such that a single use case requires one or more features for implementation.

**Comment**  
--

**Example**  
Automatic windshield wiper

**Reference**  
[EAST-Glossary]

**German Term**  
--

**French Term**  
--

### 3.67 Flag

**Definition**  
A piece of data that can take on one of two values indicating whether a logical condition is true or false.

**Initiator**  
WP 1.1.1

**Further Explanations**  
--

**Comment**  
--

**Example**  
Notification flag

**Reference**  
--

**German Term**  
--

**French Term**  
--

### 3.68 Frame

**Definition**  
Data unit according to the data link protocol specifying the arrangement and meaning of bits or bit fields in the sequence of transfer across the transfer medium.

**Initiator**  
WP1.1.2

**Further Explanations**  
--

**Comment**  
--

**Example**  
A CAN frame consists of up to 8 bytes of payload data and additional protocol specific bits / bit fields (e.g. CAN-Identifier).
### 3.69 Frame PDU

**Definition**
A PDU that fits into 1 frame instance. e.g. it does not need to be fragmented across more than 1 frame for transmission over a network.

**Initiator** WP1.1.2

| Further Explanations | -- |
| Comment               | -- |
| Example               | -- |
| Reference             | -- |
| German Term           | -- |
| French Term           | -- |

### 3.70 Function

**Definition**
1. A task, action or activity that must be accomplished to achieve a desired outcome.
2. A part of programming code that is invoked by other parts of the program to fulfill a desired purpose.
3. In mathematics, a function is an association between two sets of values in which each element of one set has one assigned element in the other set so that any element selected becomes the independent variable and its associated element is the dependent variable.

**Initiator** WP 1.1.1

| Further Explanations | -- |
| Comment              | Due to the different meanings in texts using the term application the appropriate meaning should be explained in detail or referenced. |
| Example              | 2. C-Code Function  
|                      | 3. Y=f(x) |
| Reference            | [IEEE12331], [EAST-Glossary] |
| German Term          | Funktion |
| French Term          | -- |

### 3.71 Functional Network

**Definition**
A logical structure of interconnections between defined functional parts of features ( definition 3.66).

**Initiator** WP 1.1.1

| Further Explanations | -- |
| Comment              | -- |
| Example              | -- |
| Reference            | -- |
| German Term          | Funktionales Netzwerk |
| French Term          | -- |
### 3.72 Functional Unit

<table>
<thead>
<tr>
<th>Definition</th>
<th>An entity of software or hardware, or both, capable of accomplishing a specified purpose.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>ECU, AUTOSAR Software Component, ...</td>
</tr>
<tr>
<td>Reference</td>
<td>[ISO 2382-1]</td>
</tr>
<tr>
<td>German Term</td>
<td>Funktionale Einheit</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.73 Functionality

<table>
<thead>
<tr>
<th>Definition</th>
<th>Functionality comprises User-visible and User-non-visible functional aspects of a system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>EAST glossary not applicable, due to use of function.</td>
</tr>
<tr>
<td>Example</td>
<td>Functionality of a communication system is a user-non-visible aspect.</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Funktionalität</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.74 Gateway

<table>
<thead>
<tr>
<th>Definition</th>
<th>A gateway is functionality within an ECU that performs a frame or signal mapping function between two communication systems. Communication system in this context means e.g. a CAN system or one channel of a FlexRay system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 2.1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>Gateway ECU 3.75</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.75 Gateway ECU

<table>
<thead>
<tr>
<th>Definition</th>
<th>A gateway ECU is an ECU (definition 3.50) that is connected to two or more communication channels, and performs gateway functionality.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 2.1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>Gateway 3.74</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>
### 3.76 Hardware Connection

**Definition**

HW Connections are used to describe the connection of HW elements (→ definition 3.77) among each other. It defines/characterizes the interrelationship among HW Elements (for abstract modelling). The HW Ports (→ definition 3.79) of the HW Elements serve as connection points for this purpose.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 2.1.1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Further Explanations</td>
<td>In AUTOSAR are 2 kinds of HW Connections defined: Assembly HW Connection Delegation HW Connection</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[AUTOSAR Specification of ECU Resource Template]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.77 Hardware Element

**Definition**

The HW Element is the main describing element of an ECU (→ definition 3.50). It provide HW ports (→ definition 3.79) for being interconnected among each others. A generic HW Element specifies definitions valid for all specific HW Elements.

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 2.1.1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Further Explanations</td>
<td>A HW Element is the piece or a part of the piece to be described with the ECU Resource Template. It uses other elements as primitive: This means HW elements can be nested (through HW Containers, a hierarchical structur of HW Elements). At the lowest level a HW Element only uses primitives</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[AUTOSAR Specification of ECU Resource Template]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.78 Hardware Interrupt

**Definition**

Interrupt triggered by HW event

<table>
<thead>
<tr>
<th>Initiator</th>
<th>WP 1.1.1</th>
</tr>
</thead>
</table>
| Further Explanations | 2 sorts of HW events
- Processor-intern: events as for example division by zero, arithmetical overflow, non-implemented instruction
- Processor-extern: events as for example response of peripheral device (e.g. PWM), memory error, timer |
| Comment | -- |
| Example | -- |
| Reference | Translation/Adaptation from [VDI Lexikon] |
| German Term | -- |
| French Term | -- |

### 3.79 Hardware Port

**Definition**

The HW port exposes functionality to the exterior of the HW element (→ definition 3.77). HW elements can be connected via HW Connections (→ definition 3.76). It defines a connection Endpoint for the HW Element.

| Initiator | SystemTeam |
### 3.80 I-PDU

**Definition**  
Interaction Layer Protocol Data Unit  
Collection of messages for transfer between nodes in a network. At the sending node the Interaction Layer (IL) is responsible for packing messages into an I-PDU and then sending it to the Data Link Layer (DLL) for transmission. At the receiving node the DLL passes each I-PDU to the IL which then unpacks the messages sending their contents to the application.

**Initiator**  
WP1.1.2

**Further Explanations**  

**Comment**  

**Example**  
OSEK COM specifies an Interaction Layer and works on I-PDUs

**Reference**  
[ISO OSEK, Glossary]

**German Term**  

**French Term**  

### 3.81 Indication

**Definition**  
Service primitive defined in the ISO/OSI Reference Model (ISO 7498). With the service primitive 'indication' a service provider informs a service user about the occurrence of either an internal event or a service request issued by another service user. [OSEK BD]

**Initiator**  
WP 1.1.1

**Further Explanations**  
An indication is e.g. a specific notification generated by the OSEK underlying layer to inform about a Message Reception Error.

**Comment**  

**Example**  
OSEK Com notification class 1 and 3.

**Reference**  
[OSEK BD], [OSEK Com]

**German Term**  

**French Term**  

### 3.82 Integration

**Definition**  
The progressive assembling of system components into the whole system.

**Initiator**  
WP 1.1.1

**Further Explanations**  

**Comment**  

**Example**  

**Reference**  
[ISO 2382-20]

**German Term**  

**French Term**  

---

HW elements provide HW ports for being interconnected among each others. Each HW port has a name which is unique within the HW element it is located in.

---
3.83 Interface

**Definition**
A shared boundary between two functional units (definition 3.72) defined by various characteristics pertaining to the functions, physical interconnections, signal exchanges, and other characteristics, as appropriate.

**Initiator**
WP 1.1.1

**Further Explanations**
In AUTOSAR the interface has specific meanings: See Standardized AUTOSAR Interface (definition 3.147) and Standardized Interface (definition 3.148).

**Comment**
--

**Example**
Diagnosis Service

**Reference**
[ISO 2382-1]

**German Term**
Schnittstelle

**French Term**
--

3.84 Interrupt

**Definition**
Event that enforces the processor to change its state. This interruption causes the normal sequence of instructions to be stopped. Once an interrupt occurred, the running software entity is suspended and an interrupt service routine (definition 3.85) (the one dedicated to this interrupt) is called.

**Initiator**
WP 1.1.1

**Further Explanations**
Two sorts of interrupts exists: HW and SW interrupts (definition 3.78 and definition 3.142)

**Comment**
--

**Example**
--

**Reference**
Translation/Adaptation from [VDI Lexikon]

**German Term**
--

**French Term**
--

3.85 Interrupt Service Routine (ISR)

**Definition**
A software routine called in case of an interrupt (definition 3.84)

**Initiator**
WP 1.1.1

**Further Explanations**
ISRs have normally higher priority than normal processes and can only be suspended by another ISR which presents a higher priority than the one running.

**Comment**
--

**Example**
--

**Reference**
[VDI Lexikon]

**German Term**
--

**French Term**
--

3.86 Invalid Flag

**Definition**
For a signal in a PDU an optional invalid flag can be added to the PDU payload layout. This flag indicates the validity of other signals in the payload. In case the invalid flag of a signal is set to true in a PDU instance, the respective signal in the payload of the PDU instance does not contain a valid signal value.

**Initiator**
WP 4.2.2.1.6 Gateway

**Further Explanations**
This mechanism may be used in gateways to indicate that parts of an PDU do not contain valid data.

**Comment**
--

**Example**
--
### 3.87 Invalid Value of Signal

**Definition**
For a signal in a PDU an optional invalid value can be defined.

**Initiator**
WP4.2.2.1.6 Gateway

**Further Explanations**
The invalid value is element of the signal value range that can be represented and transported by the signal. The invalid value is the value that is used in all situations where the receiver should be notified that the value in a signal is not valid.

**Comment**

**Example**
In case a PDU for a destination network of a gateway is composed from two PDUs of two different source networks, the failure to receive one PDU can be indicated as invalid values in the respective signals of the transmitted PDU in the destination network.

### 3.88 Link time configuration

**Definition**
The configuration of the SW module is done during link time.

**Initiator**
WP4.1.1.2

**Further Explanations**
The object code of the SW modules receives parts of its configuration from another object code file or it is defined by linker options.

**Comment**

**Example**
Initial value of a signal.

### 3.89 Mapping

**Definition**
Mapping designates the distribution of elements in the logical view to elements in the physical view.

**Initiator**
WP 1.1.1

**Further Explanations**
In general several entities may be allocated to one container but an entity may be allocated to only one container.

**Comment**

**Example**
a) Mapping of AUTOSAR Signals onto Frames (for inter-ECU communication).
b) Mapping of SW-C onto ECUs (Distribution of the SW-Components to the ECUs).

### 3.90 MCAL Signal

**Definition**
The MCAL signal is the software representation of the conditioned signal (definition 3.35). It is provided by the microcontroller abstraction layer (MCAL) and
### 3.91 Metadata

<table>
<thead>
<tr>
<th>Definition</th>
<th>Metadata is data about data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.2</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>Metadata includes pertinent information about data, including information about the authorship, versioning, access-rights, timestamps etc.</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.92 Microcontroller Abstraction Layer (MCAL)

<table>
<thead>
<tr>
<th>Definition</th>
<th>Software layer containing drivers to enable the access of onchip peripheral devices of a microcontroller and offchip memory mapped peripheral devices by a defined API (→ definition 3.2). Task: make higher software layers independent of the microcontroller.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.2</td>
</tr>
</tbody>
</table>
| Further Explanations | The Microcontroller Abstraction Layer is the lowest software layer of the Basic Software. The Microcontroller Abstraction Layer consists of the following parts:  
• I/O Drivers  
• Communication Drivers  
• Memory Drivers  
• Microcontroller Drivers  
Properties:  
• Implementation: µC dependent  
• Upper Interface (API): standardizable and µC independent |
| Comment    | --                                                                        |
| Example    | Examples of drivers located in the Microcontroller Abstraction Layer are:  
• onchip eeprom driver  
• onchip adc driver  
• offchip flash driver |
| Reference  | [AUTOSAR Software Architecture] |
| German Term | Mikrocontroller-Abstraktions-Schicht |
| French Term | Couche d'abstraction du microcontrôleur |
3.93 Mistake

**Definition**
Human error

**Initiator**
WP 1.1.1

**Further Explanations**

**Comment**

**Example**

**Reference**
[DIN 40041]

**German Term**
menschliches Versagen

**French Term**
erreur humaine

3.94 Multimedia Stream

**Definition**
A consistent sequence of digital data versus time which is suited as input for devices which transfer these data into a continuous visible or audible impression to humans. When transferred over a physical link, multimedia stream data typically are produced at the same rate (by the data source), as they are consumed (by the data sinks).

**Initiator**
--

**Further Explanations**
A multimedia stream usually follows a certain standard (e.g. MPEG-x). When transferred over a physical link, a multimedia stream needs a certain minimum bandwidth (in terms of bits/second) in order to allow continuous impressions.

A multimedia stream in a car typically exists for several seconds (a warning signal, a navigation hint) up to several hours (a video film, a phone call, playing a radio program). Resources (e.g. bus system channels) needed by the stream have to be allocated continuously over this lifetime (this is a difference to e.g. file transfer, which may be split into several chunks of data).

The source of a multimedia stream typically is a specialized device and/or software program (a tuner, a microphone, a text-to-speech engine, etc.). The same holds for the sinks (an audio amplifier or mixer, a voice recognition software, an MPEG decoder, etc.).

**Comment**
The term “visible or audible impression to humans” should not be taken too literally, because streams can also be used to transfer machine readable data (e.g. modem, encrypted signals). But it is this condition, which defines the standards and technology used in multimedia streams.

**Example**
Audio stream as output of or input to a telephone (mono, low bandwidth)
Audio stream as output of a radio tuner (stereo, high bandwidth)
Video stream as output of a television tuner
An example for the physical implementation on a multimedia bus is the Firewire isochronous stream. see reference

**Reference**
[IEEE 1394]

**German Term**
--

**French Term**
--

3.95 Multiple Configuration Sets

**Definition**
A SW module has more than one alternative configuration (parameter) set, which can be selected according to external requirements. The set can ONLY be selected during start-up and it is not allowed to switch the set during runtime.

**Initiator**
WP4.1.1.2

**Further Explanations**
Multiple configuration sets reside in the ECU non-volatile memory at the same time, the active configuration is selected at the start-up of the ECU. Only BSW modules can have multiple configuration sets

**Comment**
Multiple configuration is a kind of data variant coding
### 3.96 Multiplexed PDU

**Definition**
A multiplexed PDU is a PDU with a configurable number of different payload layouts.

**Initiator**
WP4.2.2.1.6 Gateway

**Further Explanations**
Each instance of a multiplexed PDU has a distinct layout. The set of possible layouts is statically defined. A selector signal defines which layout is used in a PDU instance. The selector signal must reside at the same position in all layouts. Each layout is identified by a unique selector value. The length of each instance of a multiplexed PDU is fixed.

**Comment**

**Example**

**Reference**

**German Term**

**French Term**

### 3.97 Non-AUTOSAR Component

**Definition**
A Non-AUTOSAR Component is a Component (→ definition 3.3 and definition 3.32) not implemented according to AUTOSAR guidelines and processes and which is not using the AUTOSAR Runtime Environment to communicate with its environment. A Non-AUTOSAR Component is located on a separate (non AUTOSAR) hardware entity which communicates with at least one Atomic Software Component.

**Initiator**
WP 1.1.1

**Further Explanations**
All interfaces have to be adapted to AUTOSAR Interfaces (→ definition 3.10) by the AUTOSAR Runtime Environment and/or Basic Software (→ definition 3.19) of ECUs containing Atomic Software Components (→ definition 3.8) communicating with Non-AUTOSAR Components.

**Comment**
A Non-AUTOSAR Component is a representation on the VFB of a non-AUTOSAR entity, such as a non-AUTOSAR ECU.

**Example**
ECUs developed not according to the AUTOSAR process and with no AUTOSAR Software inside. Smart Sensors

**Reference**

**German Term**

**French Term**

### 3.98 Notification

**Definition**
Informing a software entity about a state change of a hardware and/or software entity which has occurred.

**Initiator**
WP 1.1.1

**Further Explanations**
The informing about a state change can be done by an activation of a software part or by setting a flag (→ definition 3.67).

**Comment**

**Example**

**Reference**

**German Term**

**French Term**
### 3.99 OS-Application

<table>
<thead>
<tr>
<th>Definition</th>
<th>A block of software including tasks, interrupts, hooks and user services that form a cohesive functional unit.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>Robert Rimkus</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
<tr>
<td>Trust:</td>
<td>An OS-Application that is executed in privileged mode and has unrestricted access to the API and hardware resources.</td>
</tr>
<tr>
<td>Non-trusted:</td>
<td>An OS-Application that is executed in non-privileged mode has restricted access to the API and hardware resources.</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[AUTOSAR Specification of OS]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.100 Partitioning

<table>
<thead>
<tr>
<th>Definition</th>
<th>Decomposition, the separation of the whole system into functional units and further into software components.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
<tr>
<td>Partitioning has to be done before the mapping in order to identify the components that are to be mapped.</td>
<td></td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>Based on [IEEE Std.610.12-1990]</td>
</tr>
<tr>
<td>German Term</td>
<td>Partitionierung</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.101 PCI

<table>
<thead>
<tr>
<th>Definition</th>
<th>PCI is the abbreviation of “Protocol Control Information”. This Information is needed to pass a SDU (definition 3.131) from one instance of a specific protocol layer to another instance. E.g. it contains source and target information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.2</td>
</tr>
<tr>
<td>Further Explanations</td>
<td></td>
</tr>
<tr>
<td>The PCI is added by a protocol layer on the transmission side and is removed again on the receiving side.</td>
<td></td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>
### 3.102 PDU

**Definition**

PDU is the abbreviation of “Protocol Data Unit”. The PDU contains SDU (definition 3.131) and PCI (definition 3.101).

**Initiator**

WP1.1.2

**Further Explanations**

On the transmission side the PDU is passed from the upper layer to the lower layer, which interprets this PDU as its SDU.

**Example**

OSEK COM specifies an Interaction Layer

**Reference**

[ISO OSEK Glossary]

**German Term**

--

**French Term**

--

### 3.103 PDU Timeout

**Definition**

Maximum time between the receptions of two instances of one PDU is exceeded.

**Initiator**

WP4.2.2.1.6 Gateway

**Further Explanations**

This timeout indicates that the last reception of a PDU instance is too long in the past. As a consequence it can be concluded that the data in the last PDU instance is outdated.

**Comment**

--

**Example**

--

**Reference**

--

**German Term**

--

**French Term**

--
### 3.104 Peripheral Hardware

<table>
<thead>
<tr>
<th><strong>Definition</strong></th>
<th>Hardware devices integrated in micro-controller architecture to interact with the environment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initiator</strong></td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td><strong>Further Explanations</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Memory, CAN-Controller, ADC, DIO, etc.</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>Peripherie</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.105 Personalization

<table>
<thead>
<tr>
<th><strong>Definition</strong></th>
<th>User-specific and memorized adjustment of SW data or selection of functional alternatives.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initiator</strong></td>
<td>WP 10.1</td>
</tr>
<tr>
<td><strong>Further Explanations</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Seat parameters (position, activation status of drive-dynamic seat) can be stored in correlation to a user ID. For a given user ID the seat can be adjusted according to the stored position parameters and the drive-dynamic seat can be activated or deactivated.</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>Personalisierung</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.106 Port

<table>
<thead>
<tr>
<th><strong>Definition</strong></th>
<th>A port belongs to a component (→ definition 3.32) and is the interaction point between the component and other components. The interaction between specific ports of specific components is modeled using connectors (→ definition 3.39). A port can either be a p-port (→ definition 3.114) or an r-port (→ definition 3.118).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initiator</strong></td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td><strong>Further Explanations</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Comment</strong></td>
<td>For more information see AUTOSAR Specification of VFB</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Reference</strong></td>
<td>[AUTOSAR Specification of Virtual Functional Bus]</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.107 Port Interface

<table>
<thead>
<tr>
<th><strong>Definition</strong></th>
<th>A Port Interface characterizes the information provided or required by a port (→ definition 3.106) of a component (→ definition 3.32).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initiator</strong></td>
<td>Wp 1.1.1</td>
</tr>
<tr>
<td><strong>Further Explanations</strong></td>
<td>A Port Interface is either a Client-Server Interface (→ definition 3.27) in case client-server communication (→ definition 3.26) is chosen or a sender-receiver Interface (→ definition 3.134) in case sender-receiver communication (→ definition 3.133) is used.</td>
</tr>
<tr>
<td><strong>German Term</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>French Term</strong></td>
<td>--</td>
</tr>
</tbody>
</table>
3.108 Post-build time configuration

**Definition**  
The configuration of the SW module is possible after building the SW module.

**Initiator**  
WP4.1.1.2

**Further Explanations**  
The SW may either receive elements of its configuration during the download of the complete ECU software resulting from the linkage of the code, or it may receive its configuration file that can be downloaded to the ECU separately, avoiding a re-compilation and re-build of the ECU SW modules. In order to make the post-build time re-configuration possible, the re-configurable elements shall be stored at a known position in the ECU storage area.

**Example**  
Identifiers of the CAN frames

**Reference**

**German Term**  
--

**French Term**  
--

3.109 Pre-Compile-Time configuration

**Definition**  
The configuration of the SW module is done at source code level and will be effective after compile time.

**Initiator**  
WP4.1.1.2

**Further Explanations**  
The source code contains all the ECU configuration data and when compiled together, it produces the given SW.

**Comment**  
--

**Example**  
Preprocessor switch for enabling the development error detection and reporting

**Reference**

**German Term**  
--

**French Term**  
--

3.110 Private Interface (API 3)

**Definition**  
A private interface is an interface within the Basic Software (→ definition 3.19) of AUTOSAR which is neither standardized nor defined within AUTOSAR.

**Initiator**  
WP 1.1.2

**Further Explanations**  
The goal of the private interface is to enable a more efficient implementation of basic software modules. Basic software modules sharing a private interface have to be distributed as one package. This package has to behave exactly the same as separate modules would. It must provide the same standardized interfaces to the rest of the basic software and/or RTE as separate modules would. It has to be configured exactly the same as separate modules would be configured.

**Comment**  
Private interfaces contradict the goal of exchangeability of standard software modules and should be avoided.

**Example**  
--

**Reference**

**German Term**  
--

**French Term**  
--
### 3.111 Probability of failure

<table>
<thead>
<tr>
<th>Definition</th>
<th>Probability of the occurrence of a failure in a system or functional unit.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Ausfallwahrscheinlichkeit</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.112 Procedure Call

<table>
<thead>
<tr>
<th>Definition</th>
<th>A simple statement that provides the actual parameters for and invokes the execution of a procedure (software function).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>A synchronous communication mechanism can be implemented by a procedure call.</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[ISO 2382-15]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.113 Process

<table>
<thead>
<tr>
<th>Definition</th>
<th>An executable unit managed by an operating system scheduler that has its own name space and resources (including memory) protected against use from other processes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>A process consists of n Task (n&gt;=1)</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Prozess</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.114 Provide Port

<table>
<thead>
<tr>
<th>Definition</th>
<th>Specific Port (定义 3.106) providing data (定义 3.41) or providing a service of a server (定义 3.136).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>The Provide Port is sometimes abbreviated as PPort or P-Port.</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>• Server Port</td>
</tr>
<tr>
<td></td>
<td>• Sender Port</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>
### 3.115 Redundancy

**Definition**
Existence of means, in addition to the means which would be sufficient for a system or functional unit to perform a required function.

**Initiator**
WP 1.1.3

<table>
<thead>
<tr>
<th>Further Explanations</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[IEC 61508, Part 4]</td>
</tr>
<tr>
<td>German Term</td>
<td>Redundanz</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.116 Reliability

**Definition**
Probability of a system or functional unit to perform as expected under specified conditions within a time interval.

**Initiator**
WP 1.1.3

<table>
<thead>
<tr>
<th>Further Explanations</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Zuverlässigkeit</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.117 Relocatability

**Definition**
Capability of a software part being executed on different hardware environments without changing the code of the software part.

**Initiator**
WP 1.1.1

<table>
<thead>
<tr>
<th>Further Explanations</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>Verschiebbarkeit</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.118 Require Port

**Definition**
Specific Port (definition 3.106) requiring data (definition 3.41) or requiring a service of a server.

**Initiator**
WP 1.1.1

<table>
<thead>
<tr>
<th>Further Explanations</th>
<th>The Require Port is sometimes abbreviated as RPort or R-Port.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>Client Port, Receiver Port</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>
### 3.119 Required property

| Definition | A required property or quality of a design entity (e.g. SW component or system) is a property or quality which has to be fulfilled by the environment of this design entity. |
| Initiator | WP10.1 |
| Further Explanations | A property or quality can be required by a stakeholder (e.g. customer) or another design entity. |
| Comment | -- |
| Example | 1) In order to meet its functionality, a SW component A requires a minimum temporal resolution of a signal (information on a required port) which has to be fulfilled by SW component B.  
2) SW component requires to be activated by the runtime environment every 100ms with a jitter of 10ms. |
| Reference | Compare term asserted property (definition 3.5) |
| German Term | Erforderliche Eigenschaft |
| French Term | -- |

### 3.120 Resource

| Definition | A resource is a required but limited hardware entity of an ECU (definition 3.50), which in general can be accessed concurrently, but not simultaneously, by multiple software entities. |
| Initiator | WP 1.1.1 |
| Further Explanations | -- |
| Comment | The OSEK definition [OSEK BD] cannot be used, due to the specific usage in OSEK OS. |
| Example | CPU-load, interrupts (mechanism itself and the resulting CPU-load), memory, peripheral hardware, communication, … |
| Reference | -- |
| German Term | -- |
| French Term | -- |

### 3.121 Resource-Management

| Definition | Entity which controls the use of resources (definition 3.120). |
| Initiator | WP 1.1.1 |
| Further Explanations | The main functionality of resource management is the control of simultaneous use of a single resource by several entities, e.g. scheduling of requests, multiple access protection. |
| Comment | -- |
| Example | OS-scheduler (CPU-load management) |
| Reference | -- |
| German Term | -- |
| French Term | -- |

### 3.122 Response Time

| Definition | Time between receiving a stimulus and delivering an appropriate response or reaction. |
| Initiator | WP 2.1.1.1 |
| Further Explanations | The response time describes the time between a stimulus like e.g. the state change of hardware or software entity and the expected reaction of the system. |
(e.g. response, actuator activation).
Synonym: reaction time
See also: execution time, worst case execution time and worst case response time.

Comment --
Example --
Reference --

German Term Antwortzeit
French Term --

3.123 Risk

**Definition** The product of the probability of failure and the severity of outcome.

**Initiator** WP 1.1.3

**Further Explanations** --

**Comment** --

**Example** --

**Reference** --

**German Term** Risiko
**French Term** --

3.124 Robustness

**Definition** Ability of a system or functional unit to perform as expected also under unexpected conditions.

**Initiator** WP 1.1.3

**Further Explanations** --

**Comment** --

**Example** --

**Reference** --

**German Term** Robustheit
**French Term** --

3.125 RTE Event

**Definition** An RTEEvent encompasses all possible situations that can trigger execution of a runnable entity (definition 3.126) by the RTE. Thus they can address timing, data sending and receiving, invoking operations, call server returning, mode switching, or external events. RTE-Events can either activate a runnable entity or wakeup a runnable entity at its waitpoints.

**Initiator** Stefaan Sonck Thiebaut

**Further Explanations** Note ‘event’ in this context is not necessarily synonymous with ‘RTEEvent’ as defined in the VFB specification. In particular, RTEEvents that result from communication are handled by communication-triggered runnable entities.

**Comment** Events can have a variety of sources including time.

**Example** Scheduling of runnable entities from angular position, e.g. a crankshaft, that are used to trigger an interrupt and hence an RTE notification.
A software component needs to perform a regular interval, e.g. flash an LED, reset a watchdog, etc.

**Reference** --
**German Term** --
**French Term** --
### 3.126 Runnable Entity

**Definition**
A Runnable Entity is a part of an Atomic Software-Component (definition 3.8) which can be executed and scheduled independently from the other Runnable Entities of this Atomic Software-Component. It is described by a sequence of instructions that can be started by the RTE (definition 3.xx). Each runnable entity is associated with exactly one EntryPoint (definition 3.xx).

**Initiator**
WP 1.1.1

**Further Explanations**
A Runnable Entity contains at least two points for the Scheduler (definition 3.130):
1 Entry Point (definition 3.52) and 1 Exit Point (definition 3.56).
Due to the reason that an Atomic Software Component is not dividable, all its Runnable Entities are executed on the same ECU.

**Comment**
In general a task in the runtime system consists out of n Runnable Entities of m Atomic Software-Components.

**Example**
Server function of an AUTOSAR-Software Component.

**Reference**
Based on [IEC 61508, Part 4]

**German Term**
Ausführbare Einheit

**French Term**
entité executable

### 3.127 Safety

**Definition**
Freedom from unacceptable risk for personal/goods

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
--

**Reference**
Based on [IEC 61508, Part 4]

**German Term**
Sicherheit

**French Term**
--

### 3.128 Sample Application

**Definition**
Defined system used for evaluation purposes.

**Initiator**
WP 1.1.1

**Further Explanations**
The application may be simplified for better understanding within the evaluation phase.

**Comment**
--

**Example**
Diagnosis Application
Exterior Light Management

**Reference**
--

**German Term**
Beispielapplikation

**French Term**
--

### 3.129 Scalability

**Definition**
The degree to which assets can be adapted to specific target environments for various defined measures.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
Target environment introduced compared to EAST-Glossary.
### 3.130 Scheduler

**Definition**
The scheduler handles the scheduling of the tasks/runnable entities (definition 3.155 / 3.126) according to the priority and scheduling policy (pre-defined or configurable). It has the responsibility to decide during run-time when which task can run on the CPU of the ECU.

**Initiator**
System Team

**Further Explanations**
There are many strategies (priority-based, time-triggered, round-robin, …) a scheduler can use, depending of the selected and/or implemented algorithms.

**Comment**
--

**Example**
--

**Reference**
[EAST-Glossary]

**German Term**
Skalierbarkeit

**French Term**
--

### 3.131 SDU

**Definition**
SDU is the abbreviation of “Service Data Unit”. It is the data passed by an upper layer, with the request to transmit the data. It is as well the data, which is extracted after reception by the lower layer and passed to the upper layer.

**Initiator**
WP 1.1.2

**Further Explanations**
A SDU is part of a PDU (definition 3.102).

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
--

**French Term**
--

### 3.132 Security

**Definition**
Protection of data, software entities or resources from accidental or malicious acts.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
Slightly adapted norm.

**Example**
--

**Reference**
[ISO 2382-8]

**German Term**
Sicherheit

**French Term**
--

### 3.133 Sender-Receiver Communication

**Definition**
A communication pattern which offers asynchronous distribution of information where a sender communicates information to one or more receivers, or a receiver receives information from one or several senders.
### 3.134 Sender-Receiver Interface

**Definition**: A sender-receiver interface is a special kind of port-interface (⇒ definition 3.107) used for the case of sender-receiver communication (⇒ definition 3.133). The sender-receiver interface defines the data-elements which are sent by a sending component (which has a p-port providing the sender-receiver interface) or received by a receiving component (which has an r-port requiring the sender-receiver interface).

**Initiator**: Stefaan Sonck Thiebaut

**Further Explanations**

**Comment**: A special kind of Port-Interface

**Example**

**Reference**: [AUTOSAR Specification of Virtual Functional Bus]

**German Term**: Sender-Empfänger Schnittstelle

**French Term**: --

### 3.135 Sensor/Actuator SW-Component

**Definition**: AUTOSAR SW-Component (⇒ definition 3.14) dedicated to the control of a sensor or actuator.

**Initiator**: WP 1.1.1

**Further Explanations**: There will be several Sensor/Actuator SW-Cs in each ECU. In general there will be one Sensor/Actuator SW-C for each sensor and one for each actuator (⇒ number of Sensor/Actuator SW-C = number of sensors + number of actuators).

**Comment**

**Example**

**Reference**: [AUTOSAR Specification of Virtual Functional Bus]

**German Term**: Sensor/Aktuator Software Komponente

**French Term**: Composant Logiciel de traitement des Capteurs/Actionneurs

### 3.136 Server

**Definition**: Software entity which provides services for clients (⇒ definition 3.25).

**Initiator**: WP 1.1.1

**Further Explanations**: The server (⇒ definition 3.136) and the clients using its service might be located on one ECU or distributed on different calculation units (e.g. ECU).

**Comment**: Adapted from Balzert.

**Example**

**Reference**: [Balzert99]

**German Term**: --

**French Term**: --
3.137 Service

**Definition**  
A service is a type of operation that has a published specification of interface and behavior, involving a contract between the provider of the capability and the potential clients.

**Initiator**  
WP 1.1.1

**Further Explanations**  
--

**Comment**  
--

**Example**  
Diagnosis service, ...

**Reference**  
[EAST-Glossary]

**German Term**  
--

**French Term**  
--

3.138 Service Port

**Definition**  
A Service Port is a Port (→ definition 3.106) of an AUTOSAR SW-C (→ definition 3.14), Complex Device Driver (→ definition 3.31) and/or ECU Abstraction (→ definition 3.48) connected to an AUTOSAR Service (→ definition 3.13).

**Initiator**  
WP 1.1.1

**Further Explanations**  
The interface of a Service Port has to be a Standardized AUTOSAR Interface (→ definition 3.10 and 3.147).

A Service Port does not need to be connected to another Port in the VFB View (→ definition 3.163).

**Comment**  
If a service is provided by the ECU where a specific Atomic Software Component is located the VFB View is sufficient.

If a service is provided by another ECU the connection of the service call to the service has to be done explicitly during the mapping step.

**Example**  
Write data to non volatile memory.

**Reference**  
--

**German Term**  
--

**French Term**  
--

3.139 Services Layer

**Definition**  
The Services Layer is the highest layer of the Basic Software which also applies for its relevance for the application software: while access to I/O signals is covered by the Hardware Abstraction Layer, the Services Layer offers Operating system services

Vehicle network communication and management services

Memory services (NVRAM management)

Diagnosis Services (including KWP2000 interface and error memory)

ECU state management

Task: Provide basic services for application and basic software modules

**Initiator**  
WP 1.1.2

**Further Explanations**  
The Services Layer consists of the following parts:

Communication Services

Memory Services

System Services

**Comment**  
--

**Example**  
Network Management, NVRAM Manager, ECU State Manager

**Reference**  
[AUTOSAR Software Architecture]

**German Term**  
Dienste-Schicht

**French Term**  
--
3.140 Shipping

**Definition**
Component shipment refers to the action of a supplier releasing a software component (also a composition) to the system integrator. The integrator will gather all shipments of the components that make up the whole system and then map them to ECUs.

**Initiator**
WP 2.1.1.1

**Further Explanations**

**Comment**

**Example**

**Reference**

**German Term**
Auslieferung

**French Term**

---

3.141 Software Configuration

**Definition**
The arrangement of software elements in a SW system.

**Initiator**
WP 1.1.1

**Further Explanations**
A software element is a clearly definable software part. A software configuration is a selection version of software modules, AUTOSAR software components, parameters and generator configurations. Calibration and Variant Coding (→ definition 3.161) can be regarded as subset of Software Configuration.

**Comment**

**Example**

**Reference**
[EAST-Glossary]

**German Term**
Software Konfiguration

**French Term**

---

3.142 Software Interrupt

**Definition**
Interrupt triggered by SW event.

**Initiator**
WP 1.1.1

**Further Explanations**
SW events are for example calling an operating system service, starting a process with higher priority.

**Comment**

**Example**

**Reference**
Translation/Adaptation from [VDI Lexikon]

**German Term**

**French Term**

---

3.143 Software Module

**Definition**
A collection of software files (code and description) that define a certain software functionality present on an ECU.

**Initiator**
WP 4.1.1.2

**Further Explanations**
A software module may be an Atomic SW-C (→ definition 3.8), or a Basic Software module (→ definition 3.20), or the RTE (RunTimeEnvironment).

**Comment**
This term shall be used if both basic software modules and Application software Components are addressed.

**Example**
A Digital IO Driver, Complex Device Driver, OS are examples of software modules.

**Reference**

### 3.144 Software Signal

**Definition**
A Software Signal is an asynchronous event transmitted between one process and another.

**Initiator**
WP 1.1.1

**Further Explanations**
A SW Signal is the software implementation of an (control-) information. Additionally it may have attributes (e.g. freshness, data type, …). It is exchanged between SW-Components.

**Comment**
--

**Example**
--

**Reference**
--

**German Term**
Software-Modul

**French Term**
--

### 3.145 Special Periphery Access

**Definition**
Special functions to standard peripheral devices or special peripherals.

**Initiator**
WP 1.1.1

**Further Explanations**
Is only used when, because of technical issues, no standard periphery access can be used

**Comment**
--

**Example**
--

**Reference**
[HIS API IO Driver version 2.1] Interaction with [Hardware v0.3]

**German Term**
Spezielle Peripherie

**French Term**
--

### 3.146 Standard Periphery Access

**Definition**
Standard functions to typical standard peripheral devices that is available on an ECU (most microcontroller integrated) used in automotive embedded applications.

**Initiator**
WP 1.1.1

**Further Explanations**
--

**Comment**
--

**Example**
Digital Input/Output, Analog/Digital Converter, Pulse Width (De)Modulator, EEPROM, FLASH, Capture Compare Unit, Watchdog Timer

**Reference**
[HIS API IO Driver version 2.1] Interaction with Hardware v0.3

**German Term**
Standard Peripherie

**French Term**
--

### 3.147 Standardized AUTOSAR Interface

**Definition**
This is an AUTOSAR Interface which is standardized within the AUTOSAR project.

**Initiator**
WP 1.1.1

**Further Explanations**
AUTOSAR Services interact with other components through a Standardized AUTOSAR Interface.

**Comment**
--

**Example**
--
### 3.148 Standardized Interface

**Definition**
A software interface is called Standardized Interface if a concrete standardized API exists.

**Initiator**
WP 1.1.1

**Further Explanations**
Modules in the Basic Software interact which each other through Standardized Interfaces.

**Comment**

**Example**
OSEK COM Interface

**Reference**

**German Term**

**French Term**

### 3.149 Standard Software

**Definition**
Standard Software is software which provides schematic independent infrastructural functionalities on an ECU. It contains only Standardized Interfaces (⇒ definition 3.148), Standardized AUTOSAR Interfaces (⇒ definition 3.147) and/or Private Interfaces (⇒ definition 3.110).

**Initiator**
WP 1.1.1

**Further Explanations**

**Comment**

**Example**
OSEK COM, MCAL, Services, OSEK OS

**Reference**

**German Term**

**French Term**

### 3.150 Static Configuration

**Definition**
A setup where the routing configuration cannot be changed during normal operation of the gateway.

**Initiator**
WP 4.2.1.6 Gateway

**Further Explanations**
Static configuration doesn't allow reconfiguration of the routing during normal operation e.g. during driving.
Static configuration does not restrict the update of the configuration in specific maintenance operation modes (e.g. programming mode).

**Comment**

**Example**
A software update may change a routing configuration such that a PDU is routed into two instead of one destination networks.

**Reference**

**German Term**
Statische Konfiguration

**French Term**

### 3.151 Synchronous Communication

<table>
<thead>
<tr>
<th>Definition</th>
<th>A communication is synchronous when the calling software entity is blocked until the called operation is evaluated. The calling software entity continues its operation by getting the result.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>Synchronous communication between distributed functional units has to be implemented as remote procedure call.</td>
</tr>
<tr>
<td>Comment</td>
<td>Are further mechanisms possible?</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.152 System

<table>
<thead>
<tr>
<th>Definition</th>
<th>An integrated composite that consists of one or more of the processes, hardware, software, facilities and people, that provides a capability to satisfy a stated need or objective.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>ITEA EAST uses IEEE 14407 standard. Here not applicable because of problem with the definition of function. One correct interpretation is: it might be a composition of one or more ECUs</td>
</tr>
<tr>
<td>Example</td>
<td>Braking system</td>
</tr>
<tr>
<td>Reference</td>
<td>[ISO 12207]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.153 System Constraint

<table>
<thead>
<tr>
<th>Definition</th>
<th>Boundary conditions that restrict the Design-Freedom of the (cars E/E-) System.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 2.1.1.3</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>The design of ECU Networks and the distribution of functionalities to ECUs are limited by several constrains. These constraints result mostly by the communication matrix and safety requirements</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>An existing communication matrix that restricts the distribution of signals to frames is a system constraint. Another system constraint is a safety requirement that does not allow to map a specified Software component to specific ECU.</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.154 System Signal

<table>
<thead>
<tr>
<th>Definition</th>
<th>The system signal represents the communication system’s view of data exchanged between SW components which reside on different ECUs. The system signals allow to represent this communication in a flattened structure, with (at least) one system signal defined for each data element sent or received by a SW component instance. If data has to be sent over gateways, there is still only one system signal representing this data. The representation of the data on the</th>
</tr>
</thead>
</table>

**Note:** The document contains sensitive information marked as AUTOSAR confidential.
3.155 Task

**Definition**
A Task is the smallest scheduleable unit managed by the OS. The OS decides when which task can run on the CPU of the ECU.

**Further Explanations**
A runnable entity (definition 3.126) of a software component runs in the context of a task. Also the Basic Software Modules runs in the context of a task.

**Reference**
[AUTOSAR Specification of Virtual Functional Bus]

3.156 Technical Signal

**Definition**
The technical signal is the physical value of an external event coupled to an AUTOSAR system. Technical signals are represented in SI units (e.g. pressure in PA).

**Further Explanations**
The term Technical Signal is used when we are referring to the "real world" signal that is under consideration. So typical Technical Signals are temperature, velocity, torque, force, electrical current and voltage, etc.

3.157 Template

**Definition**
A template is a structured collection of attributes that are required to formally describe AUTOSAR artifacts like e.g. software components or configurations of ECUs.

**Further Explanations**
The term "Template" stresses the fact that the collected attributes still need to have actual values assigned in order to describe a particular artifact. Those values are collected in a Description. Templates are independent of the technology used for serialization of their respective descriptions. Possible serializations include XML, databases tables and so on.

**Example**
The templates defined by AUTOSAR are represented as an UML2.0 model (definition 3.11) and an W3C XML Schema (definition 3.17). Models and descriptions created according to the templates can be exchanged.
### 3.158 Timeout

<table>
<thead>
<tr>
<th>Definition</th>
<th>Notification with respect to deadline violation of an event or task (e.g., while working on/with information: receiving, sending, processing, ...).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.159 Use Case

<table>
<thead>
<tr>
<th>Definition</th>
<th>A model of the usage by the user of a system in order to realize a certain functional feature of the system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>Added certain compared to EAST-glossary.</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[EAST-Glossary]</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.160 Validation

<table>
<thead>
<tr>
<th>Definition</th>
<th>Confirmation by examination and provision of objective evidence that the particular requirements of a specific intended use are fulfilled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>System Team</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>--</td>
</tr>
<tr>
<td>Comment</td>
<td>--</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>[IEC 61508]</td>
</tr>
<tr>
<td>German Term</td>
<td>Validierung</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.161 Variant Coding

<table>
<thead>
<tr>
<th>Definition</th>
<th>Adaptation of SW by selection of functional alternatives according to external requirements (e.g., country-dependent or legal restrictions).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanations</td>
<td>The major difference with calibration is that this later doesn’t aim to adapt the SW functionality itself but only aims to adjust the SW to the HW/SW environment, e.g. the calibration of engine control SW that is adjusted to the physical parameters of</td>
</tr>
</tbody>
</table>
### 3.162 Verification

**Definition**

Confirmation by examination and provision of objective evidence that the requirements have been fulfilled.

**Initiator**

System Team

**Further Explanations**

- 

**Comment**

--

**Example**

--

**Reference**

[IEC 61508]

**German Term**

Verifikation / Überprüfung

**French Term**

--

### 3.163 VFB View

**Definition**

The VFB View describes systems or subsystems in the car independently of these resources; in other words, independently of:

- what kind of and how many ECUs are present in the car
- on what ECUs the entities in the VFB-View run
- how the ECUs are interconnected: what kind of network technology (CAN, LIN,...) and what kind of topology (presence of gateways) is used

**Initiator**

WP 1.1.1

**Further Explanations**

In the VFB-View, the system or subsystem under consideration is a Composition which consists out of Connectors and Components.

**Comment**

--

**Example**

--

**Reference**

[AUTOSAR Specification of Virtual Functional Bus]

**German Term**

--

**French Term**

--

### 3.164 Virtual Integration

**Definition**

The simulated, modeled and/or calculated (not real) combination of software entities forming a system (definition 3.152).

**Initiator**

WP 1.1.1

**Further Explanations**

By virtual integration several constraints and/or requirements are checked without the need of real hardware units, like needed CPU load, needed memory, completeness of interfaces, fulfillment of timing requirements etc.).

**Comment**

--

**Example**

--

**Reference**

--

**German Term**

Virtuelle Integration
### 3.165 Worst Case Execution Time

<table>
<thead>
<tr>
<th>Definition</th>
<th>Maximum possible time during which a program is actually executing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanation</td>
<td>The worst case execution time of a piece of software is the maximum possible time during which the CPU is executing instructions which belong to this piece. The worst case execution time is often identified by analytical methods. It is required to determine if a schedule meets the overall timing requirements. Abbreviation: WCET</td>
</tr>
<tr>
<td>Comment</td>
<td>This definition has been extended by 2.1.1.1</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>--</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>

### 3.166 Worst Case Response Time

<table>
<thead>
<tr>
<th>Definition</th>
<th>Maximum possible time between receiving a stimulus and delivering an appropriate response or reaction.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiator</td>
<td>WP 1.1.1</td>
</tr>
<tr>
<td>Further Explanation</td>
<td>The worst case response time describes the maximum possible time between a stimulus like e.g. the state change of hardware or software entity and the expected reaction of the system (e.g. response, actuator activation). Typically: worst-case execution-time + infrastructure-overhead + scheduling-policy = worst-case reaction time Synonym: worst case reaction time</td>
</tr>
<tr>
<td>Comment</td>
<td>Worst case reaction time was renamed to worst case response time because response time is the more common terminology. This definition has been extended by 2.1.1.1.</td>
</tr>
<tr>
<td>Example</td>
<td>--</td>
</tr>
<tr>
<td>Reference</td>
<td>--</td>
</tr>
<tr>
<td>German Term</td>
<td>n/a</td>
</tr>
<tr>
<td>French Term</td>
<td>--</td>
</tr>
</tbody>
</table>
Annex 1: Literature

[Balzert99], Balzert, H. “Lehrbuch Grundlagen der Informatik” Spektrum Verlag, Heidelberg, 1999

[DCE-IDL], Remote Procedure Call http://www.opengroup.org/onlinepubs/9629399/chap1.htm


[EN 50126], European Norm “EN 50126 – part 103, March 2000 Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)”


[ISO 61511-1], International Standardization Organization “ISO/IEC 61511 Part 1

[OMG-IDL],
http://www.omg.org/technology/documents/formal/corba_2.htm


[OSEK Com], OSEK/VDX “Communication”, Version 3.0.1, January 2003

[IEC 61508, Part 4], Functional safety of electrical/electronic/programmable
electronic safety-related systems; Part 4: Definitions and abbreviations

[ISO 7498], Information processing systems -- Open Systems Interconnection --
Basic Reference Model

[DIN 40041], DIN 40041 Ausgabe:1990-12 Zuverlässigkeit; Begriffe
Deutsche Industrie Norm

[VDI Lexikon], Translation/Adaptation from VDI Lexikon Informatik und
Kommunikationstechnik,
Springer Verlag, Berlin 1999,

[HIS API IO Driver V 2.1],
http://www.automotive-his.de/download
API_IODriver_2_1_3.pdf

ISBN 1-55937-067-X, SH13748

[UML 2.0] Unified Modeling Language

[IEEE 1394], Firewire, see “isochronous stream”
http://www.1394ta.org/Technology/Specifications/specifications.htm